Mining on-line user reviews

Bin, Shi

2007

Bin, S. (2007). Mining on-line user reviews. Master's thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/2494>

<https://doi.org/10.32657/10356/2494>

Nanyang Technological University

Downloaded on 26 Apr 2023 22:11:55 SGT

NANYANG TECHNOLOGICAL UNIVERSITY

Mining On-line User Reviews

by

BIN SHI

A Dissertation

submitted in fulfillment of the requirements

for the degree of Master of Engineering

in

School of Computer Engineering

Nanyang Technological University

Jan 02, 2007

Abstract

In this report, we propose a system for mining on-line user product reviews. Each sentence in a review is analyzed to extract the user's opinion/orientation of a product feature. The sentiment/orientation towards each product feature is then tallied across all users to generate an overall scoring for the product and its individual features. For testing purposes, we extract reviews from the Chinese cellphone on-line community.

On-line product review mining falls under the realm of web structured data mining. In order to unify the different variations describing a particular feature like screen/display, we manually created an ontology of unique product features specific to cellphone with a list of commonly used synonyms. Later on, this ontology is enhanced by statistic analysis results on on-line cellphone reviews. After surveying the information extraction research landscape, we decided on using a template-based hybrid extraction method. We proposed using a novel method to rid Chinese content from noise such as extraneous punctuation frequently used in on-line posts.

Sentiment classification has attracted quite an interest in the community; we therefore first surveyed existing sentiment classification techniques. Next, an improved lexicon based sentiment classification method and an machine learning technique based sentiment classification method, both operating at the sentence segment resolution, are proposed. In order to present intuitive review results to the end user, we have created an interactive graphical web interface which allows the user to show/hide/aggregate selective review feature scores.

A fully-functioning prototype has been created to demonstrate the end-to-end system. Experiments were conducted to evaluate our suite of proposed methods. Based on analysis

of the experimental results, we discussed drawbacks of our current approaches and explored possible improvements and future directions of research.

Acknowledgments

I would like to first thank my supervisor, Professor Kuiyu CHANG. Without his great guide and patience, I would not have foraged into the amazing field of data mining. Before I met him, I knew nothing about data mining, not even the basic concepts. His encouragement and kindness inspired me to explore this completely new field. In addition to growing academically, my English oral and written communication skills have also improved a lot. Many novel ideas in this report came out of our discussions.

Special thanks go to the CAISians for settling me down in a conducive research environment and providing technical support. I am also grateful to Mr. Qiankun ZHAO and Mr. Jianshu WENG from CAIS for their assistance on my research and experiments.

The persistent support and encouragement given to me by my family deserves my deepest appreciation. Last but not least, I would like to say a big thank you to my wife, Mdm. Bo WANG, for all the great sacrifices she has made so far. Without her constant support and encouragement, I would not have had the opportunity to pursue my graduate studies at NTU.

Contents

Abstract	i
Acknowledgments	iii
List of Figures	viii
List of Tables	x
 1 Introduction	 1
1.1 Motivation	1
1.2 Objectives	2
1.3 Organization of this Dissertation	3
 2 Literature Review	 4
2.1 Extracting Product Review Content	4
2.2 Chinese Word Segmentation and POS Tagging	7
2.3 Concept Hierarchy Construction	7
2.3.1 Automatic Product Feature Extraction	8
2.3.2 Feature Word Clustering	8
2.3.3 Concept Hierarchy Construction	10
2.4 Sentiment Orientation Identification	13
2.4.1 Sentence Classification and Retrieval	14
2.4.2 Sentiment Orientation Classification	16
2.5 Summarization and Visualization	18
2.6 Remarks	18
 3 System Architecture	 19

4	Crawling Review Pages and Extracting Reviews	21
4.1	Selection of Review Sites	21
4.2	Matching Product Models	22
4.3	Crawling Review Pages	22
4.4	Extracting Individual Reviews from a Page	23
4.4.1	Storing a Crawled Page as a Local File	23
4.4.2	Converting Malformed HTML Pages to XML	23
4.4.3	Storage of Extracted Reviews	24
4.4.4	Review Information Extraction Template	25
4.5	Experiments and Results	26
4.5.1	Experiment Setup	27
4.5.2	Experiment Results	27
4.5.3	Discussion	27
5	Pre-processing Chinese On-line Reviews	29
5.1	Punctuation Filtering	29
5.2	Justification for Analyzing Sentence Segment	31
5.3	Eliminating Questioning Segments	32
5.4	Segmentation and POS Tagging via ICTCLAS	33
5.5	Experiments and Results	34
5.5.1	Punctuation Filtering Experiment	34
5.5.2	Pruning Questioning Segments Experiment	37
5.5.3	ICTCLAS Utilization Experiment	38
6	Learning the Product Concept Hierarchy	41
6.1	Manual Construction of the Concept Hierarchy	41
6.2	Learning the Concept Hierarchy	43
6.2.1	Methodology	44
6.2.2	Experiments and Results	50
6.2.3	Discussions	60

7	Sentiment Identification	62
7.1	Lexicon Based Sentiment Identification	63
7.1.1	Associating Opinion Word with Feature	63
7.1.2	Orientation Identification and FO pair Composing	64
7.2	Adjacency Matrix Representation for Sentence	
	Classification	67
7.2.1	Matrix Representation Methodology	67
7.2.2	Experiment Setup	70
7.2.3	Experiment Results	73
7.2.4	Discussion	77
7.3	Matrix SVM Based Sentiment Classification	79
7.4	Matrix versus Concept Experiments and Results	81
7.4.1	Experiment Setup	81
7.4.2	Evaluation	82
7.4.3	Experiment Results	84
7.4.4	Discussion	85
8	Summarization and Presentation of On-line Reviews	88
8.1	Aggregating Review Segments	88
8.1.1	Minimizing Fraudulent Reviews and Outliers	88
8.1.2	XML Summary File	90
8.1.3	Sample Update to the XML Summary Tree	90
8.2	Interactive Web Interface to Present Summarization	91
9	Conclusion and Future Work	97
9.1	Conclusion	97
9.2	Future Work	99
9.2.1	Automatic Review Extraction	99
9.2.2	Improving Chinese Preprocessing	100
9.2.3	Improving Concept Hierarchy	101

9.2.4	Enhancement of Opinion Orientation Identification	101
9.3	Concluding Remarks	102
A	Review Information Extraction Template (RIET)	103
A.1	RIET Elements	103
A.2	RIET Example	103
B	Sample Mined Concept Hierarchies	106
B.1	Mined Cellphone Concept Hierarchy	106
B.2	Mined Hotel Concept Hierarchy	106
B.3	Mined Mp3player Concept Hierarchy	106
	References	110

List of Figures

3.1	Level 0 DFD of the Review Mining System.	20
3.2	Level 1 DFD of the Review Mining System.	20
4.1	Level 2 DFD for Crawling and Extracting Reviews from a Site.	23
5.1	Level 2 DFD for Pre-processing Extracted Chinese On-line Review	30
5.2	Strategies of Chinese Questioning Pattern Identification.	33
5.3	ICTCLAS Demonstration.	34
5.4	Examples of Wrongly POS Tagged Punctuations (Underlined).	35
6.1	Hierarchical Concept Model of Cellphone.	42
6.2	Procedure of Clustering Vectors and Constructing Concept Hierarchy.	48
6.3	Projection Based Hypernym Selection. Here the density vector captures only features with more than 50% overlap.	49
6.4	Sub-hierarchies of Hotel.	58
6.5	Sub-hierarchies of Cellphone.	58
6.6	Sub-hierarchies of Mp3player.	59
7.1	Activity diagram for identifying feature orientation pairs from a sentence segment, which is an elaboration of the “Extract Feature Orientation” pro- cess in Figure 3.2.	66
7.2	Representing a Four-word Sentence with a 3 Order Adjacency Matrix.	69
7.3	Atomic Unit Denoting a Term.	72
7.4	Cross-validated Accuracy.	74
7.5	Cross-validated F-Measure of Major Categories.	74

7.6	Cross-validated F-Measure of Minor Categories.	76
7.7	Cross-validated Precision of Minor Categories.	77
7.8	Cross-validated Recall of Minor Categories.	78
7.9	Activity diagram for identifying feature orientation pairs from a sentence segment using a <i>Matrix</i> SVM classifier. This graph provides more details about the “Extract Feature Orientation” process in Figure 3.2.	80
8.1	Level 2 DFD for Generating a Per Product XML Summary File.	89
8.2	Demonstration of Inputting Searching Terms for Brand and Model.	92
8.3	Pull-down Menus for Brand/Model Selection.	93
8.4	The Nokia 3230 Opinion Tree.	94
8.5	The Nokia 3230 Opinion Tree with Some Nodes Collapsed.	95
8.6	Hovering over the green bar under Appearance node shows that 136 or 77% of the users like the appearance of the Nokia 3230.	95
8.7	Sample Positive Reviews about Appearance of the Nokia 3230.	95
8.8	Sample Negative Reviews about (lack of a) Camera of the Nokia 3230.	95
8.9	Sample Comparison of Two Cellphones.	96
A.1	RIET for www.163.com	105
B.1	Mined Cellphone Concept Hierarchy	107
B.2	Mined Hotel Concept Hierarchy	108
B.3	Mined Mp3player Concept Hierarchy	109

List of Tables

4.1	Extraction results via RIET. All numbers are relative to the 100 sample URLs. Unique indicates number of unique products; Empty refers to number of review pages with no reviews; Crawled refers to number of product directories created. Raw refers to count of all reviews; Duplicates refer to number of identical postings; Extracted refers to number of uniquely extracted individual reviews.	27
5.1	Precision of ICTCLAS segmentation & POS tagging on the three data sets: cleaned, raw, and <i>Punctuation Filtered</i> reviews. Under the Terms column, “+n” refers to the number of invalid terms (e.g., “misspelled” Chinese phrases).	36
5.2	Result of Pruning Questioning Sentence Segments.	38
5.3	Word Segmentation Results.	39
6.1	Distribution of the Three Review Sets.	51
6.2	SF Thresholds for Selecting Candidate Nouns from the 3 Review Sets. . . .	51
6.3	Total Number of Non-zero Entries over All Context Vectors.	52
6.4	Clusters’ Count of Different <i>WS</i> Size and Intra-similarity Threshold <i>T</i> . . .	52
6.5	Comparison of Clustering Results	54
6.6	Subjective Evaluation of <i>Projection</i> and <i>Z-score</i> Extracted Hypernyms for Each Node in the Hierarchy.	55
6.7	Statistics for <i>Projection</i> and Sanderson’s Graphs.	56
6.8	Results of Noun Pair Evaluation.	57
6.9	Count of Nouns in <i>Projection</i> and Sanderson’s Graphs.	58

7.1	Basketball Match Data Set Distribution.	70
7.2	Cellphone Review Data Set Distribution.	71
7.3	Divided Cellphone Review Data Sets Distribution.	81
7.4	Sentiment Identification Results for 1000 Test Reviews.	85
A.1	Specifications for Review Information Extraction Template (RIET).	104

Chapter 1

Introduction

1.1 Motivation

The Internet has become the universal information source, allowing people from all over the world to share their personal experiences through participation in on-line forums, discussion groups, wiki pages, and blogs. This sharing invariably includes user opinions about products, services, persons, organizations, among many other things. The sharing of personal experiences regarding a product or service has become the epitome of user empowerment. Internet sites such as Amazon.com (pioneer in on-line book reviews) and eBay.com (first to popularize buyer/seller ratings) have propelled into the leagues of major on-line shopping brands simply because they understand how important user reviews/opinions affect people's buying decision. Smaller sites specializing in a particular product genre has also blossomed. For example, instead of walking into a retail store to make an impulsive digital camera purchase, people have learned to educate themselves by visiting camera review sites such as dpreview.com, steves-digicams.com, and others, before making the plunge to actually buy a digital camera. More generic review sites like epinions.com have also been gaining in popularity, but at a slower pace, simply because as a generic review site for all products, it possesses no special know-how to attract users.

The Internet has thus created a new class of informed consumers, who regularly consult on-line sources for information eventually leading to a purchase decision. For electronic products, especially cellphones with a 3-6 month model revamp time-frame, the informed consumer is typically faced with the problem of deciding among tens or hundreds of models

(including brandless economical alternatives, for example, in China alone, there are hundreds of no-name cellphones and Mp3player manufacturers). The smart consumer would of course like to compare similar products, but this usually entails the painstaking process of manually sieving through pages of product specification and review sites.

Naturally, first hand evaluations or comments from customers who have used a product are extremely valuable. More so if a large number of customer opinions/reviews can be assembled and summarized in a neat fashion. However, it is not easy to compare similar products by manually reading on-line posts/comments that could add up to the hundreds for some popular brand/model. In addition, some of the comments may not be relevant, or could simply be advertisements. Lastly, most product review web sites consolidate reviews on each brand/model in one sub branch, making comparison of user reviews across different models difficult at best and impractical at worst.

1.2 Objectives

Instead of having each prospective buyer gathering the on-line product reviews over and over again, wouldn't it be great if he/she can consult a product "expert" who aggregates and summarizes all major on-line reviews for him/her in an easy-to-read format? Consequently, our goal is to build a system that does exactly that. Our system is designed to be simple and user-friendly, which it achieved by hiding the unnecessary details and making use of a hierarchical graphical interface to show increasingly refined product review information at different feature resolutions.

With the Chinese economy continually growing annually at 8-9% for the foreseeable future, it is a matter of time before China eventually replaces US as the largest consumer market in the world. In fact, Chinese Internet users now number at just over 100 million, currently the second largest in the world, and growing at a rate of 5% annually. Both factors together speaks volumes about the demand for a Chinese on-line product review system.

Consequently, our current system has a major focus and eye on the Chinese Internet community. Although Chinese is quite different from English in a large number of aspects, the fundamental language-independent ideas of our proposed solution can be easily ported to English and other languages.

1.3 Organization of this Dissertation

The organization of this dissertation is as follows. Chapter 2 surveys related work. Chapter 3 presents the system architecture. Chapter 4 describes the method of extracting on-line reviews, including experiments and discussion. Chapter 5 lists the strategies and techniques proposed to pre-process Chinese on-line reviews, where the feasibility and efficiency of each method are evaluated experimentally. Chapter 6 is about our novel method to build a product concept hierarchy. Chapter 7 presents and compares the two novel sentiment identification approaches that we have proposed. Chapter 8 describes the web interface prototype that will allow a user to search and retrieve reviews and summaries pertaining to a particular brand and model of cellphone. Chapter 9 concludes our work by summarizing our achievements, and proposing directions for future work.

Chapter 2

Literature Review

Our review mining approach can be performed roughly in four steps. First, individual product reviews/comments are extracted from web pages; second, the ontology, product concept hierarchy, is constructed; third, the orientation of each reviewer, be it positive or negative, with respect to a product feature is extracted; finally a summary of all product feature orientation/review is computed. In this chapter, we review existing work in each of the above steps. In addition, we also look at current state-of-the-art Chinese word segmentation and part of speech (POS) tagging tools.

2.1 Extracting Product Review Content

Most on-line product reviews are well structured since they are usually generated from back-end database records via a presentation (e.g., HTML and CSS) template. Product review extraction can thus be viewed as an Information Extraction (IE) task, with the goal of identifying and extracting individual reviews from a web page.

Liu et al. [LZ05] classifies IE techniques into two groups, namely 1) *wrapper based extraction* (supervised), and 2) *automatic extraction* (unsupervised). *Wrapper based extraction* aims to recover the underlying templates used to create the structured content by finding the corresponding extraction rules. For this purpose, a training data set of human annotated HTML pages is usually required. Wrapper approaches require significant human input, which can be error prone, time consuming, and laborious. Further, the learnt wrapper must be re-learnt whenever the underlying generating template changes. On the other

CHAPTER 2. LITERATURE REVIEW

hand, *automatic extraction* attempts to extract records from a web page just by examining its local layout. It does not involve too much human effort, which is needed only to validate the extracted information. The advantage of *automatic extraction* is that it theoretically does not involve humans. However, in practice, a lot of manual intervention is needed to tune the various parameters such as similarity threshold of the automatic extraction process parameters.

An automatic string similarity comparison based method has been previously proposed [LGZ03][ZL05b] to extract structured on-line reviews by first building a document tree using HTML tags from the review page and ignoring content completely. It follows by comparing the similarity of HTML tags within each subtree to identify frequently occurring patterns which could constitute individual reviews. One obvious drawback is that the method cannot extract from pages containing only a single review, albeit a rare occurrence. Another problem lies in selecting an appropriate string similarity threshold. Further, the same comparisons must be performed on every page, which can be time-consuming for large collections. Our experience with this approach showed it to be overly sensitive to the threshold, which must be adjusted individually for pages from different web sites.

It is well-known that for historical reasons, the HTML source of many web pages are loosely organized and may contain erroneous/missing tags. These HTML errors will lead to a lot of problems for strict string comparison based methods. To resolve the problem of non-compliant HTML tags, a partial tree alignment [Yan91] based method to organize and align extracted reviews and their content was proposed [ZL05b].

Another approach to automatic structured data extraction is to first derive an extraction template from an initial set of HTML pages, which is then applied to extract data from the remaining pages. Arasu and Garcia-Molina [AGM03] proposed one such approach which first identifies the constructors of structured data by finding the associated tokens (HTML tag and content) from a small set of sample pages. The discovered tokens set collectively form the extraction template. Moreover, it assumes that all pages containing the desired structured data shares the same rendering template.

CHAPTER 2. LITERATURE REVIEW

One of the more recent wrapper induction approaches is [ZL05a], which is interesting because it takes into consideration the fact that more than one template may be adopted to generate the various pages in a web site. This method requires a human to manually annotate a random page by noting the prefix and suffix HTML tags surrounding a desired data record (containing the review). The highlighted HTML tags are then used to identify candidate data records in an unseen/new page. If extraction fails for a page, relabeling is considered, which can be tedious if it happens too often.

Browsers have traditionally been extremely tolerant towards non-compliant HTML pages. Consequently, a visual (browser) rendering based approach was proposed for extracting on-line reviews [LZ05]. By utilizing the fact that each nested HTML element in a web page is typically rendered as a rectangle, a robust visual tree is first built via detecting the boundaries of rectangles and their mutually inclusive relationships. Next, a bottom-up traversal is used to compare and extract reviews, similar to the approach of [LGZ03]. We note that their approach only considered the inclusive relationship among the rectangles, although in practice the positions of the rectangles may also provide valuable hints or information. One visual IE approach that uses the positions of rendered HTML rectangles is [CYWM03], which visually segments a web page into semantically related content blocks using the positions of the various rectangular content regions. In addition, it also relies on the information extracted from HTML source, e.g., tag, color, size, etc., in the IE process.

In our system, HTML string similarity comparisons [LGZ03] are used to generate an approximate extraction template for identifying the data record candidates. The template is then manually refined with additional elements and structures. Our algorithm thus combines the best of breed approaches from both automatic and wrapper induction methods; it requires less human involvement (no training/labeling), produces highly accurate ¹ and robust extraction results (human validated extraction rules) in a relatively short time (no compulsory string comparisons). We further note that our manual wrapper approach is able to extract reviews from pages containing only a single review. As our approach is still

¹in fact 100% accuracy was achieved in our tests as shown later in Chapter 4

largely based on HTML source code, human intervention is still needed to tackle sites that employ several different rendering templates.

2.2 Chinese Word Segmentation and POS Tagging

Although not our primary focus, POS (Part Of Speech) tagging plays a vital role in text sentiment analysis, and therefore will be briefly described here. Prior to POS tagging, word segmentation must be applied to preprocess and segment Chinese sentences into known vocabulary phrases. The vast majority of current Chinese word segmentation systems rely on a word lexicon [Zho05]. As a matter of fact, unknown word detection is one major bottleneck for Chinese word segmentation. A novel unknown Chinese word detection and chunking method has been previously proposed, which uses a combination of Mutual Information Independence Model, maximum entropy, and error-driven learning [Zho05].

Based on the benchmarks reported by Zhou [Zho05], it appears that **ICTCLAS** (**I**nstitute of **C**omputing **T**echnology, **C**hinese **L**exical **A**nalysis **S**ystem) [ZLC⁺03b], a software package based on Hierarchical Hidden Markov Model (HHMM) [ZLC⁺03a], is the right tool to use. HHMM is adopted in **ICTCLAS** to perform segmentation of both known and unknown words, and also to perform POS tagging. **ICTCLAS** processes text in four steps as follows: (1) word atom segmentation, (2) recursive recognition of unknown words, (3) class-based word chunking, and (4) POS tagging. In addition to extracting known words, nine different word class tags are defined, including name entity, time, numeric, etc.

2.3 Concept Hierarchy Construction

In order to generate a highly specialized and useful review summary, a hierarchical concept model of the review product is indispensable. The hierarchy encourages exploration by grouping similar features together. That is, extracting features of the product and organizing these mined features organically are two essential roles of the product concept hierarchy. Here, we review existing work related to the construction of a product concept hierarchy.

2.3.1 Automatic Product Feature Extraction

Liu et al. proposed an automatic way to extract product features from reviews [HL04]. They classify a product’s features into two types, frequent and infrequent. *Association Rule Mining* is then used to automatically discover frequent product features. Any noun or noun phrase whose *Minimum Support* is larger than 1% in the whole review set is made a candidate feature. To reduce this huge candidate set, they then employ two pruning procedures, (1) compactness pruning to remove phrases whose words do not appear together, and (2) redundancy pruning to remove single words which are also contained in another candidate noun phrase. To unearth infrequent features, they regard the “nearest” noun or noun phrase to a known opinion to be one. Clearly, this could lead to a significant number of irrelevant nouns mistakenly identified as infrequent product features.

Ultimately, there is no guarantee that the approach can identify all relevant product features, and a human is always needed in the loop to filter the list of candidate product features. We feel that this automatic technique is useful, not as a primary method for finding a comprehensive list of product features, but as a secondary tool to help a human expert discover new trends of describing a product feature, e.g., 5 meg (resolution). Thus, in the end we feel that it is most cost-effective to have a human filter extracted candidate features and then merge them with the basic list of product features come up with by a human.

2.3.2 Feature Word Clustering

In fact, not all extracted features are unique. Some may be synonyms or conceptually related. To reduce the complexity of the concept hierarchy, the first thing that should be done is to group the synonyms or conceptually related terms together.

An automatical Japanese noun grouping approach was presented in [Ino91]. Inoue’s approach builds on the linguistic hypothesis that similar nouns tend to appear together with the same set of verbs, which is a stronger hypothesis than ours². A Japanese-English

²we assume that similar nouns will appear together with a set of common words, be it verbs, adjectives, or other nouns

CHAPTER 2. LITERATURE REVIEW

bilingual database is used to distinguish the different senses of a verb, probably due to the incompleteness of the Japanese semantic lexicon. The similarity between two Japanese nouns are then calculated based on the co-occurrence frequencies of common verbs with the same sense.

Wu et al. [WZ03] studied bilingual concept extraction using an English synonym extraction approach. They compute three similarity values for a word such as “paris” from an English dictionary, an English-Chinese corpus, and a large English corpus, respectively. The overall similarity between the two words is a linear weighted sum of the three individual similarity values, after tuning the weights globally, with the individual similarities computed as follows:

- (1) The similarity value from the English dictionary is obtained by computing the cosine similarity of the two word vectors, each comprised of other words used to define it or words defined by it.
- (2) The similarity value from the English-Chinese corpus is computed also as the cosine similarity of the two representative word vectors, with each vector made up of corresponding Chinese candidate translations weighted by their likelihood.
- (3) To get the third similarity from the large English corpus, the authors first create an attribute set for each word comprising dependency triplets describing the word’s roles and other related words in the corpus. The similarity value is computed as between the two attribute sets.

In comparison, our approach require neither a large monolingual corpus nor a bilingual resource.

An English noun classification approach based on linguistic hypothesis of [Ino91] is presented in [Hin90]. Hindle et al. used predicate-argument (*subject-predicate and predicate-object*) structures to mine noun-verb co-occurrence frequency, on which English noun are classified or grouped.

For on-line reviews, most sentences tend to be opinion sentence. And the nouns to be grouped are most likely to be about a product’s features. In other words, the nouns in on-line reviews are usually subjects of a sentence and their frequently co-occurring verbs are typically link verbs like “is”, “are”, “be”, which makes noun-verb co-occurrence based approach too specific for on-line reviews. Thus in our case, we adopted a free-for-all approach, by counting all co-occurring words.

A similar idea to [Ino91] was adopted in [FS97] to cluster verbs, adjectives and adjectival verbs in Japanese, based on the hypothesis that “similar verbs connect similar nouns”. The similarity between two verbs are computed between their noun sets, which is comprised of common nouns that connect the two verbs. Adjectives and adjectival verbs are clustered in the same fashion.

In [MH93][HM93], clustering semantically related adjectives is implemented based on the assumption that adjectives describing the same property are likely to modify the same set of nouns, and co-occurring adjectives that are premodifiers of the same nouns are likely to belong to different groups. This approach relies heavily on the specific co-occurrences of noun-adjective and adjective-adjective patterns, which may be too specific for the colloquial nature of on-line reviews.

In [CM02a], all semantically related terms such as verbs (nouns being their subject or object), nouns, adjectives and prepositional phrases, are treated as contextual information and used to group similar nouns. This approach is an improvement over [CM02b], which only treats adjacent words of nouns as context. It is also the most related approach to ours, which treat all neighboring words as contextual elements.

2.3.3 Concept Hierarchy Construction

There are two types of approaches to derive a concept hierarchy, lexicon based and statistically based. Since summary statistics of a corpus are usually examined before any advanced operation is performed, we describe related work for the statistically based approach first.

2.3.3.1 Statistic Based Approaches

In [CC04], search engine returned snippets are used as representative vectors of the query keyword. The vectors are then clustered using bottom-up agglomerative clustering to generate a binary tree resembling a concept hierarchy. Since agglomerative clustering starts by treating each vector as a cluster, each leaf node in the generated binary tree actually corresponds to a keyword. Clearly, this makes the hierarchy deep and leaf nodes meaningless. To avoid this, the authors attempted to enhance the hierarchy's structure by manually specifying two parameter to limit the cluster size and tree depth, respectively. Moreover, the two parameters are hard to tune. Further, they simply select the most frequent co-occurring term as the node label, which is not a reliable way to find a hypernym of a group. As shown in chapter 6, the most frequently co-occurring terms may be something less representative of the group.

In [Car99], Caraballo et al. proposed a novel method to automatically construct a hypernym labeled noun hierarchy from the Wall Street corpus. They hypothesize that “nouns in conjunction or appositives tend to be semantically related”. Each noun is represented by a weighted vector of other nouns, where each weight denotes the number of times another noun appears in conjunction with or in an appositive relationship to it. The similarity between two nouns is simply their cosine angle in vector space. Likewise, the similarity between two groups of nouns is computed as the average similarity between all pairs of nouns from distinct groups. Based on this representation, a hierarchical tree structure is obtained by merging similar nouns or noun groups, which is another bottom-up agglomerative approach. Observing that Z is a hypernym of X and Y if the pattern “ X , Y and other Z s” can be found in corpus, a hypernym candidate vector is created for each noun. Each entry records the number of times that other noun (Z) satisfies the pattern. To choose the representative hypernym of a noun group, hypernym vectors in the group are summed to obtain an average hypernym vector for the group, and the entry with the largest count is selected as the representative hypernym.

CHAPTER 2. LITERATURE REVIEW

Like [Car99], Jikels et al. [JK06] make use of subject and appositive dependency relationship among nouns to build noun clusters, labeling them with hypernyms extracted from each cluster.

In [DH02], noun-noun relationships are classified into 4 types, “is a”, “of or ’s”, acronym, and co-occurring with verbs. To automatically extract the hyponymys of noun group,

Cederberg et al. [CW03] search for several patterns like “x such as y”, “y and other x”, and “x, including y” where y is treated as a hyponym of x. They then apply *Latent Semantic Analysis* to further improve the extracted hypernyms.

In other related work, Rydin et al. [Ryd02] proposed five hypernym-hypernym constructs to mine hypernym pairs and create the hierarchy, Nanas et al. [NUR03] construct the hierarchy using three types of term-dependent relations.

All of the above approaches rely on some sort of known language pattern to mine hypernyms and create the hierarchy. There are other statistical approaches that are language agnostic, such as the seminal work by Sanderson [SC99], which uses co-occurrence of words and phrases to create a concept hierarchy, a process termed “subsumption”. The relationship between words or phrases is determined by their mutual conditional probabilities estimated from the corpus. Lawrie et al. [LCR01] also utilized term co-occurrence and subsumption to extract nouns and construct hierarchy.

Improving upon Sanderson’s approach, [DIW05] and [CfBW05] also use term mutual conditional probabilities to build a concept hierarchy. In particular, [DIW05] added two type classifications, *subsumption* or *equivalence*, to the mutually inclusive probabilities, with each type handled differently in the subsumption process. In [CfBW05], an additional threshold is used to extract the subsumption term pairs precisely.

Glover et al. [GPLK02] proposed using a noun’s category-specific document frequency (DF) *vis-a-vis* the corpus DF to predict their roles, which could be one of either *parent*, *child*, or *self*. In this approach, the category or group membership must be known.

One major disadvantage of most existing approaches is the reliance on language-specific special grammatical or syntactic patterns like *conjunction* and *appositive*. On-line reviews

are typically very colloquial, short, and mostly not grammar-conforming. Thus most of the classical methods will not work well.

2.3.3.2 Lexicon Based Concept Hierarchy Construction

Zhai et al. [GYE02] built a semantic hierarchy of Chinese nouns from an existing Chinese noun lexicon, which associate with each noun relationships like “class”, “synonym” and “partof” attributes.

Calvo et al. [CG04] used the Spanish equivalent of “*is a*” pattern to create a semantic hierarchy of Spanish nouns. By manually defining 25 noun categories and assigning several “top concepts” to each category. By iterating through the “*is a*” noun chain, starting from the noun itself, and terminating with a “top concept” noun, each noun will be classified into one of the “top concepts”.

Due to the rapid advancement of products, especially for the IT field, using a lexicon based approach to automatically construct a concept hierarchy is no longer viable. Thus we propose a hybrid approach that includes a completely automatic hierarchy construction and manual refinement. The automatic concept hierarchy can be generated right at the beginning to let a human user know what is out there for a particular product domain, from which he or she can create a manual product hierarchy. After the manual product hierarchy is created, the automatic approach can be run periodically to alert to the human user any new terms or features out there. For example, cellphones did not come with camera functionality in the past, but the automatic approach would discover the new camera feature as it becomes mentioned more often in phone reviews.

2.4 Sentiment Orientation Identification

Machine learning techniques such as Support Vector Machines, Decision Trees, and Naive Bayes classifier have been applied to review sentiment orientation classification at the document resolution [NSK⁺04][SKC03][PLV02]. However, since our goal is to identify and extract sentiments about specific product features, we have to analyze the review at least at the sentence resolution.

As sentiment orientation identification at the sentence level can be treated as sentence classification, besides sentiment identification, we also review related approaches about sentence classification and Retrieval.

2.4.1 Sentence Classification and Retrieval

As stated in [MS03], analysis at the sentence level will result in an extremely sparse vector representation compared to a document. Several sentence level analysis approaches involving the use of additional context information have been proposed.

Tamura et al. [TTO05] proposed a method to classify multi-sentence questions for Question-Answering (QA) systems. They reformulated the problem as that of selecting the “core” sentence from a multi-sentence question, followed by classification of the “core” sentence. SVM was used to select the “core” sentence. To improve the context, each sentence is represented by a vector that includes features from itself and sentences appearing before and after. While the feasibility of this method has been demonstrated for QA systems, it cannot be apply to our problem, where each sentence segment may contain different opinions. Intuitively, to determine if a sentence in a review is an opinion sentence, involving adjacent sentences in the review can only interfere with this goal.

Another approach to add contextual information that includes the sentence position was proposed by [MS03]. The authors defined four sentence types for medical abstracts, “introduction”, “method”, “result” and “conclusion”. The sentence type is classified according to its content and relative position in the abstract. Experimental results show that the approach helped improved SVM’s performances on classifying partial sentence. We have yet to apply position information in our classification of sentence types, it could be studied in future work.

Improving on McKnight’s work, Yamamoto et al. [YT05] included additional syntactic features. Sentence types in [MS03] such as “introduction” was splitted into “background” and “purpose”. The relative position of a sentence was reformulated as two numbers, one that counts from the beginning, and one Other syntactic attributes such as a term’s affinity

CHAPTER 2. LITERATURE REVIEW

to certain sentence types, co-occurrences, and the presence of special auxiliary verbs were added to enhance the classification. We feel that the approach has become too specialized with the addition of these syntactic features, some of which specific to English. We might eventually incorporate some of these syntactic rules to go the last mile of improvement, but for now that will be part of the future work.

Unlike methods that use external information of a sentence to enhance sentence classification [MS03][YT05], Zaanen et al. [MvZM05] extracts implicit information within the sentence itself using an learning Information Extraction (IE) approach. Sentence structure patterns are extracted from the training set, and used to form a sentence classification regular expression. In particular, Alignment-Based Learning (ABL) is applied to extract sentence structure patterns and their respective frequencies for each sentence category. Regular expressions are constructed from these patterns and applied to a new sentence to look for matches. The category that yields the highest number of matches is then assigned to the sentence. Despite the sophistication, it appears that the classical trie-based approach [CFV98] would easily beat this method, as shown by Zaanen et al. The good thing about this approach is that the extracted rules can be validated by humans, and applied very easily to a new sentence.

Also related to the question sentence classification domain, several benchmark experiments were conducted by Augusto et al. [AP04]. In that paper, they compared SVM to the trie-based [CFV98] approach and found SVM to be as good as or better in most cases.

In a different domain, the structural information of a sentence has been shown to be helpful for sentence classification and retrieval using machine learning techniques by Hachey and Grover [HG04]. They compared the classification performances on rhetorical sentences from the legal domain of five machine learning methods. Each sentence included contextual and lingual features such as length, location, thematic words, etc. The comparison indicated that SVM (a bag of words) combined with a maximum entropy sequence tagger achieved the best results.

2.4.2 Sentiment Orientation Classification

Many sentiment classification approaches at the document level have been proposed. Roughly, they can be divided into two groups, ML (machine learning) and non-ML approaches.

One typical non-ML approach [Tur02][YSL06] computes a document’s SO (Semantic Orientation) value using several heuristics. First, a pair of extremely positive and negative opinion words are selected as a reference, e.g., “classical” and “disappoint”. Next, five categories of two-word opinion phrases are manually defined. For each found opinion phrase in a movie review, the number of search engine returned pages containing both the opinion phrase and “classical” or “disappoint” are noted, respectively. The SO value of the opinion phrase is then computed as a weighted sum of the number of positive and negative hits. After the SO values of all opinion phrases in the movie review have been computed, an average SO value is tallied for the entire movie review and compared to a pre-defined threshold to determine whether the movie review is positive or negative.

This approach, while interesting, can only determine if a review is overall positive or negative, with no sentiment indication about specific aspects/features (e.g., length, story, music, etc.) of the movie/product. Moreover, Ye et al. [YLjL05] showed that the heuristic based method cannot compete with a machine learning approach (SVM) using a bag of words.

Another kind of non-ML sentiment classification approach [SC06][HL04][LHC05] that focuses on sentence level classification is the concept lexicon based method. They all work more or less the same, as described below. Two polarized (positive and negative) sets of “seed” opinion words are manually collected. Each set of words is then automatically expanded with synonyms and antonyms from a semantic lexicon. A sentence orientation is then determined by a simple tally of the number of matching words from the two sets.

A sentence level opinion orientation identification and feature mining method was proposed in [HL04], which finds Association Rules [AS94] of frequent product features. A concept dictionary such as WordNet [MBF⁺90] was used to generate positive and negative opinion/sentiment word sets. To find the orientation of a sentence, opinion sentences are

CHAPTER 2. LITERATURE REVIEW

first extracted from the review, after which their orientations are determined. An opinion sentence is a sentence containing at least one known feature word and one known opinion word. From the dictionary of generated positive and negative opinion words, an opinion sentence’s orientation is assumed to be the orientation of the adjective “nearest” to the product feature word, while taking into account the reversal effect of nearby negation words such as “not”. From a grammatical perspective, any nearby negation word will invert a word’s orientation. Sometimes, a conjunction such as *but* may completely reverse a sentiment. Word distance and a simple threshold is used to quantify “nearness”.

There are several drawbacks in the above approach. First, opinion orientation should not be classified into only two categories, positive and negative. Sometimes, neutral sentiments are also expressed. In Liu’s approach, only adjectives were considered opinion words. Moreover, experiments [NSK⁺04] indicated that most sentiments are expressed through verbs, adjectives and adverbs. For example, the verb “like” can be construed as expressing a positive opinion. Another disadvantage of this approach is that it lacks the ability to extract implicit opinion about a feature since it only processes sentences containing at least one feature word. An improvement over [HL04] was presented in [LHC05], which considered short sentences, expanded opinion words to include verbs and adjectives, and also made an effort to extract implicit features.

With regard to ML techniques for sentiment classification, Pang et al. [PLV02] benchmarked three classifiers, Naive Bayes, maximum entropy, and SVM on document level sentiment classification and found SVM to give the best overall performance. They notice that some of the errors are due to SVM treating text as a bag of words.

Based on these related work, the SVM based approach appears to be the most flexible and dynamic choice for sentence classification. It can be improved slightly with the addition of contextual information. Thus in our research, we will use a SVM classifier whenever machine learning is needed.

2.5 Summarization and Visualization

We present a brief review of related summarization and visualization techniques since our system uses such methods to display the sentiment results. A plain text based summary method was used in [HL04], which summarizes each product by features with corresponding orientation counts and sample review sentences. The drawback of a simple approach like this is that it is not intuitive and comparison between products is not visually contrasting. In [LHC05], a graphical, comparison-enabled summary method was used where all available product features are listed as vertical bar-graphs scaled by the relative number of positive and negative review counts. Clearly, this layout allows two or more products to be compared feature by feature. A disadvantage is that *all* product features are listed, which can get messy, and there is also no interactive component. Our approach aims to combine the best ideas from all of the above methods.

2.6 Remarks

Review mining is a very complex area, involving several different areas of research spanning the linguistic and computer science spectrum. In this chapter, we reviewed the current state-of-the-art developments in review mining, and found that there are still a lot of work that can be done. In the next few chapters, we describe our contribution to this interesting field.

Chapter 3

System Architecture

Our review miner is a complete end-to-end system that takes care of everything: from downloading on-line user reviews, identifying each review sentence segment's orientation, summarizing and aggregating each product feature's sentiment, to presenting the end user with an intuitive and graphical interactive interface. In particular, there are five main components in our system, each performing a task as follows:

- (1) Crawl and extract user reviews from web pages.
- (2) Perform Natural Language Processing (NLP) of each review like Chinese word segmentation, POS tagging and punctuation filtering etc.
- (3) Extract and identify feature orientations of each sentence segment.
- (4) Summarize and aggregate feature orientations based on a conceptual model.
- (5) Present results to user via a searchable and interactive web interface.

Figure 3.1 summarizes our system as a Level 0 Data Flow Diagram (DFD). Figure 3.2 shows the five main components in our review mining system as a Level 1 DFD.

In addition to the five components, there is an important behind-the-scenes component that is indispensable to our system. The hierarchical concept model of product features is generated to aid in sentiment analysis and result summarization. Each of the five components plus the hierarchical concept model will be discussed in detail in the following chapters.

CHAPTER 3. SYSTEM ARCHITECTURE

Figure 3.1: Level 0 DFD of the Review Mining System.

Figure 3.2: Level 1 DFD of the Review Mining System.

Chapter 4

Crawling Review Pages and Extracting Reviews

4.1 Selection of Review Sites

Through our casual browsing of numerous on-line Chinese resources, we found many Chinese web sites featuring cellphone reviews, with several well-established sites (e.g., 3533.com) whose only focus are cellphone reviews and on-line shopping. This is good news for us because it means that there is no shortage of reviews for us to mine.

In general, we have observed two major types of web sites featuring user reviews. The first type is a generic on-line shopping and review site providing user reviews on all kinds of products, including cellphones. An example of such a site is www.it.com.cn. The second type offers on-line shopping and review information dedicated to a specific type of product such as cellphones, like www.3533.com. In our research, we extract cellphone reviews from four of the most popular sites from each of the above two categories. The four sites are shown as follows.

- (1) www.3533.com
- (2) www.163.com
- (3) www.it.com.cn
- (4) www.pcpop.com

4.2 Matching Product Models

While crawling some sites for cellphone reviews, we found review pages of other products like digital camera and laptops. Clearly, the crawler cannot tell the difference between a cellphone and some other product, unless there is a master list of product models. To generate this list, we crawled one dedicated cellphone review site, www.3533.com, which uses a regular template for displaying cellphone brand and model. The final list was quite comprehensive, listing all available models and brands to date.

Since some web sites only use partial product names, e.g., “3230” for “Nokia 3230”, a candidate product name is used to search the master list for all partial matches. For example, the candidate product name “3230” will be partially matched to “Nokia 3230”, and its downloaded review filename will include the fully qualified brand/model, i.e. “Nokia 3230”. By doing so, we can aggregate reviews from different site about the same model. Note that, we still download all review pages, cellphones and what not, from the generic review sites, although we currently only focus on processing the cellphone reviews.

4.3 Crawling Review Pages

Once the target review sites have been identified, the next step is to crawl and download the relevant review pages into local storage. Consequently, there are two steps involved: (1) focused crawling of web pages containing reviews, (2) extracting individual user reviews from each downloaded review page. The detailed level 2 DFD is shown in Figure 4.1.

We used the open-source GNU web crawler `wget` [WGE] to crawl our target sites, discarding all non-text content such as images, videos, animations, etc. To further reduce storage space and increase throughput, we manually specified the web domain and subdirectory containing the reviews. Review pages crawled from a web site are saved in a local directory named after the site.

CHAPTER 4. CRAWLING REVIEW PAGES AND EXTRACTING REVIEWS

Figure 4.1: Level 2 DFD for Crawling and Extracting Reviews from a Site.

4.4 Extracting Individual Reviews from a Page

4.4.1 Storing a Crawled Page as a Local File

For simplicity, we store each crawled page locally as a HTML text file named after its URL (Universal Resource Locator) [URL]. However, the URLs of some review sites contain Chinese characters encoded in the archaic GB2312 [GB2] format, which is incompatible with the modern operating system's UTF-8 [UTF] format. Thus, the open source command line tool `convmv` [CON] was used to convert the GB2312 encoded URLs to UTF-8 format, after these pages have been saved locally. Notice that here we only change the encoding of the filename, and not the file content itself.

4.4.2 Converting Malformed HTML Pages to XML

There are a considerable number of (mostly human generated) malformed HTML documents on the web, which are tolerated by most modern web-browsers but not inter-

pretable by the vast majority of XML parsers. Examples of malformed HTML include missing/mismatched starting or end HTML tags. In order to build a document tree model of a HTML page for further analysis, these irregularities must be corrected. For this, we use the open source `htmlparser` [HTM] to parse a HTML document and generate the corresponding XML Document Object Model (DOM) tree representation. Since `htmlparser` refuses to accept a non RFC3986 [URL] compliant URL as its input, the locally stored URL-named file must be made RFC3986 compliant. As stated in the RFC3986 [URL] standard, a list of special characters have to be “escaped” if they are to appear in the URL. The list includes, for example, the set of control ASCII characters and Chinese characters encoded in traditional two-byte schemes (the first byte of which is usually an ASCII control character).

Content-wise, some ASCII characters within the HTML page are reserved under XML, and thus should be escaped with an equivalent sequence of characters as detailed in the XML specification [BPSMM03]. For example, if the character “<” appears within the content of a node tag in an HTML document, it must be replaced by the equivalent character sequence “<”. Otherwise, an exception will occur during the HTML to XML conversion.

Since we cannot be 100% sure that all downloaded pages contain reviews, converting all crawled HTML documents to XML may not be the most efficient way. Fortunately, we noticed that review pages typically share a common URL root. We thus wrote Java regular expressions to convert only those pages whose URL matches our known set of URLs with reviews.

4.4.3 Storage of Extracted Reviews

Generally, apart from the review content, each individual review can include several additional fields such as user ID, date/time stamp, author’s IP address, user ratings, etc. To facilitate easy retrieval and organization, we store each individual product reviews using a unique filename and path on the file system as follows:

- (1) Create a top-level directory named after the web site hosting the review, e.g., `163.com`.

- (2) Create a brand-model sub-directory named after a product brand and model corresponding to each review, e.g., `Nokia 3230`.
- (3) Store each individual review under the corresponding product brand and model sub-directory. The review is a plain ASCII text file whose filename is a concatenation of time stamp, IP (optional), userID. For example, the full pathname of a review on Nokia 3230 phone from the 3533.com site is

`3533.com/Nokia 3230/@2005-08-25 00:25:12@58.82.6.*@yala299.txt`

4.4.4 Review Information Extraction Template

As mentioned in the previous section, various types of information pertaining to each review must be extracted from a review page. To this end, a Review Information Extraction Template (RIET), which is basically a pattern matcher, can be defined to extract the individual reviews (subtrees) from the XML document tree.

Based on our collection of review pages and observations in [LGZ03], we realized that individual reviews in a review page with multiple reviews usually share the same HTML parent node, forming a contiguous HTML code region with information organized by similar HTML tags. Our extraction approach thus takes advantage of this key observation.

Unfortunately, our implementation of the MDR [LGZ03] unsupervised IE algorithm was overly sensitive to the string similarity threshold. By setting the string similarity threshold to be from 0.1 to 0.9 with 0.05 as the increment pace, the appropriate thresholds we got for the four Chinese review web sites are ranging from 0.4 to 0.8. This observation repeats the fact that the underlying review page templates from different web sites vary. Therefore, in order to extract all available reviews from a review page, it is better to customize the similarity threshold for each web site. Even so, there were still some web sites that MDR could not handle. Nevertheless, we found MDR to be good at detecting the general subtree containing the reviews, if not finding the exact reviews.

In the end, we decided to use our MDR to first derive a rough RIET for detecting the review region in the DOM tree, after which manual rules were added so as to extract all

available reviews from a page. Thus, most of the information in the RIET for one web site could be automatically obtained by the program we implemented, which only requires as input a single typical review page. We note that our approach only works for sites that use a common rendering template, and will fail for sites that use different rendering templates. For those site, a RIET will have to be generated for each type of review pages, and additional logic will have to be created to classify a page as matching one of the known RIETs.

Appendix A provides a list of the XML elements used to specify a RIET, and also includes an example RIET. Note that the major portion of the RIET is automatically generated, with the remaining portion comprising manually added elements and regex logic.

4.5 Experiments and Results

We observed that most review sites display a fixed number of review postings per page. For example, a review site that displays at most 20 reviews per page will show 22 reviews in two pages, 20 in one and 2 in another. In this example, we call the page with 20 reviews a “full” page, and the one with 2 reviews a “partial” page. In addition, some sites (most notably www.epinions.com) create placeholder review pages for products which have not received any review, which we call “empty” pages.

Recall that our system creates a product directory under the site directory to store the reviews of each product for that site. In addition to verifying that our RIET can extract all reviews from the “full” and “partial” pages, we would like to make sure that no product directory is created for products with no reviews, i.e. “empty” page. Lastly, we would like to ignore duplicate posts, i.e. reviews by the same author with identical date/time stamp, which are likely caused by network connection delays or client software problems. We call these type of reviews duplicates.

4.5.1 Experiment Setup

To evaluate the reliability of our RIET for extracting individual reviews from full, empty or partial review pages of different sites, We select 100 review pages from 4 web sites, 25 pages from each site. All three types of review page were present in the 25 pages from each site. Sites which allow identical individual reviews also include pages containing duplicates.

4.5.2 Experiment Results

The extraction results using our RIET (Review Information Extraction Template) are shown in Table 4.1. From the table, we see that our system achieves a 100% precision and recall for both identifying the correct number of products and extracting unique individual reviews.

Table 4.1: Extraction results via RIET. All numbers are relative to the 100 sample URLs. Unique indicates number of unique products; Empty refers to number of review pages with no reviews; Crawled refers to number of product directories created. Raw refers to count of all reviews; Duplicates refer to number of identical postings; Extracted refers to number of uniquely extracted individual reviews.

Sites	# Products			# Reviews		
	Unique	Empty	Crawled	Raw	Duplicates	Extracted
www.3533.com	25	1	24	526	0	526
www.163.com	25	2	23	460	0	460
www.it.com.cn	25	10	15	300	2	298
www.pcpop.com	17	3	14	285	114	171
Total	92	16	76	1571	116	1455
Recall			100%			100%
Precision			100%			100%

4.5.3 Discussion

Clearly, our RIET can extract 100% of unique reviews and handle the various types of review page very well, fully meeting our requirements. For example, it will not create a directory for a product if no review has been posted. It was also verified to collect all available reviews on a certain product from a web site and store them in one directory.

CHAPTER 4. CRAWLING REVIEW PAGES AND EXTRACTING REVIEWS

Later redundant reviews are also discarded by virtue of our file naming convention, i.e. later reviews by the same author with the same time stamp as older reviews will be discarded.

The reason our review extraction is so accurate is because it uses a manually generated combination of HTML tree parsing logic and text pattern regular expressions. Thus, there is little surprise that it works so reliably, compared to automatic review extraction approaches, as it is *supposed* to be 100% correct.

In fact, our RIET has crawled 313102 Chinese on-line reviews on 2687 cellphones by 76 manufacturers from the four cellphone review sites and 14929 Chinese on-line reviews on 423 hotels in Shanghai from www.ctrip.com, stably and robustly, regardless the complexity of web pages and difference among review web sites. With sampling check on those crawled reviews, referring to the original ones residing in web pages, our RIET works as it is supposed, organizing crawled reviews neatly and ensuring the content of crawled reviews is identical to the original. These two review corpora are widely used in our later experiments. Due to RIET's hierarchical organization of reviews and excellent extracting accuracy, the two Chinese on-line review corpora above are shared by our U.S. and Hong Kong research cooperators.

Chapter 5

Pre-processing Chinese On-line Reviews

It is well-known that one of the most significant differences between Chinese and Latin-based languages is the lack of a whitespace separator in the former [Zho05]. Thus, meaningful phrases have to be segmented/extracted based on context. After segmentation, the next step is to assign a word sense, i.e. verb, noun, adjective to each of the segmented phrases/words, a process commonly known as Part-Of-Speech (POS) tagging. POS tagging is extremely important for a review system like ours, which relies on identifying sentimental words and associating them with potential product features. Figure 5.1 shows the data flow diagram of the Chinese Pre-processing steps we used. Notice that prior to word segmentation and POS tagging, there are two extra steps for cleaning punctuations and segments. These two procedures were created to deal with the informal nature of on-line text postings, and their benefits will be quantified later in the experiment section of this chapter. And as some special characters have been escaped into XML format when extracting reviews from HTML web pages, the first step of pre-processing procedure is to recover those escaped XML characters, as shown in Figure 5.1.

5.1 Punctuation Filtering

Unlike formal text documents, the writing style in on-line reviews can be extremely colloquial. For example, the latest trend is to use punctuations and characters liberally, e.g., “~”

CHAPTER 5. PRE-PROCESSING CHINESE ON-LINE REVIEWS

Figure 5.1: Level 2 DFD for Pre-processing Extracted Chinese On-line Review .

or white spaces are used to break sentences instead of the traditional comma “,”. Hence, in order not to affect ICTCLAS’s segmentation performance, we proposed the following set of *Punctuation Filtering* rules to eliminate extraneous punctuations:

- (1) Remove all punctuations except for comma, period, question mark, exclamatory mark, and whitespace.
- (2) Replace all white spaces with a period. As the space bar remains the largest and most reachable key, some authors used it to replace all or some hard-to-reach punctuation characters.
- (3) For the remaining five punctuations, delete all consecutive occurrences of a punctuation, e.g., for “,. ?!!!,” or “,,,,,,,,,”, only the first character of each block will be kept. This is fairly reasonable since the first punctuation in such instances is usually what the reviewer originally wished to express.

Results showing the merit of applying *Punctuation Filtering* will be listed later in Section 5.5.1 of this chapter.

5.2 Justification for Analyzing Sentence Segment

We notice that in web reviews, not only are punctuations abused as discussed, but the sentences themselves are not well formed in terms of grammar or syntax. For example, some sentences may be missing a subject, predicate, or even object, and in some extreme cases, all sentences end with commas. The last case clearly highlights the difficulty in identifying proper sentence boundaries using traditional punctuation characters; in more formal writings, we could use full-stops, question/exclamation marks, etc. to delineate sentence boundaries.

Consider a typical example where an incomplete sentence like “*Great, but the price is too high.*” was used instead of the proper “*This is great. But the price is too high.*”. If we analyze the whole of the first sentence, the positive effect of “*Great*” would be neutralized by the negative effect of “*too high*”. On the other hand, if we consider the first sentence as two separate sentence segments, we capture the positive general sentiment (may not even be product related, but that is another matter) and the negative price sentiment.

Another consideration is about the later essential procedure, eliminating questioning sentence in on-line reviews. If on sentence level, for those pure questioning sentences, pruning them as a whole is absolutely correct. However, some review sentences are the combination of opinion statement segments and non-opinion questioning segments. Deleting such sentences entirely obviously violates the purpose of our preprocessing procedure, filtering useless information as much as possible and keeping maximum available meaningful information at the same time. Hence, it is reasonable to switch to segment level for pruning questioning sentence, only questioning segments or clauses are filtered while useful sentence segments are kept.

In view of the above considerations, the sentence segment is clearly a more viable and practical candidate for our system. Formally, we define a sentence segment as the shortest string between two reserved punctuation characters (after punctuation filtering). Thus, to extract a sentence segment, we simply cut a sentence at the remaining set of reserved punctuation characters, which currently includes “.,?!”. One drawback of using

sentence segment is its isolated context and reduced semantics. Fortunately, most on-line review sentences are not long and complicated, and their main ideas can still be effectively captured in sentence segments.

5.3 Eliminating Questioning Segments

Opinions are generally expressed either as a statement or exclamatory segment. Occasionally, people posts questions in reviews, e.g., “*How good is the camera functionality?*”, which should not be construed as an opinion but unfortunately would interfere our sentiment identification procedure. Another problem about on-line reviews is that there is no guarantee that all sentences in reviews are related to the product. Based on our manual observation on crawled reviews, irrelevant chatting or questioning-answering sentences are not rare in them. It is hard to filter all these irrelevant sentences just based on their types of sentences, statement, questioning or exclamatory sentence, as both opinion sentences and irrelevant sentences in reviews can be of statement or exclamatory. However, as the ground truth, most of questioning sentences are not about opinion on the corresponding product or the product itself. We therefore try to remove questioning segments before the actual concept hierarchy construction and opinion analysis stage to reduce the amount of noisy information being involved.

To identify questioning segments, the simplest approach is to check if a segment is terminated by a question mark. However, this does not guarantee that all remaining statements are non-questioning as there are people who use only white spaces, periods, or commas to end their sentences, including questioning ones. And it is usual that some sentences contain the clauses of question sentences. To handle all of such situations, we deploy several strategies to identify question patterns as shown in Figure 5.2.

The “English equivalent” parts in Figure 5.2 are about English translation of corresponding “Chinese” parts, not the real English question patterns. The “Chinese” parts list all detailed patterns utilized in our approach. In fact, there are more questioning patterns

- (1) Sentence ended by question mark.
? ?
- (2) Sentence of typical question structure.
Chinese : ... 有没有 ...; ... 是 ... 还是 ...; ... 是否 ...; ... 能否 ...; ... 可否 ...
English equivalent : ...has/have...or not...; ...is/are...or...; ...if/whether...
- (3) Sentence containing question words.
Chinese : 为什么 ; 为啥 ; 谁 ; 哪 ; 多少 ; 怎么 ; 怎样 ; 咋 ; 难道 ;
English equivalent : why; who; which; where; how many/much; what; how
- (4) Sentence ended by questioning stop word.
Chinese : " 吗 " as the end of sentence
English equivalent : "do/does" as the beginning of sentence

Figure 5.2: Strategies of Chinese Questioning Pattern Identification.

available in Chinese. As those patterns not involved can also exist in statement or exclamation sentences, to ensure not pruning useful information, they are not adopted in this procedure.

Recall that as it has been adjusted to analyze sentence segment, what being pruned in our approach is questioning segments instead of questioning sentence. And because ICTCLAS can not guarantee 100% accuracy in word segmentation, in practice, this procedure is implemented in the way of regular expression based string search instead of word pattern matching relying on Chinese word segmentation result. Therefore, this procedure is put ahead of Chinese word segmentation and POS tagging procedure in Figure 5.1. The feasibility of the questioning sentence pruning strategies we proposed will be demonstrated by the experiment results in Section 5.5.2 of this chapter.

5.4 Segmentation and POS Tagging via ICTCLAS

There are several existing approaches for Chinese word segmentation. Based on Zhou's benchmark evaluations [Zho05], we picked the best performing ICTCLAS open source tool for segmentation and POS tagging. ICTCLAS is an open source Hidden-Markov-Model based Chinese lexical analyzer [ICT] with word segmentation and POS tagging capabilities, available for Linux and Windows platforms [ZLC⁺03b]. Figure 5.3 shows an example of a review sentence tagged and segmented by ICTCLAS.

```

Original Sentence:
机型有点土，但是性能可以。

ICTCLAS Processed Sentence
机型/n 有点/d 土/a ，/w 但是/c 性能/n 可以/v 。/w

Notation:
n: Noun
d: adverb
a: adjective
w: punctuation
c: conjunction
v: verb

```

Figure 5.3: ICTCLAS Demonstration.

As ICTCLAS is a probability estimating based Chinese word segmenting approach and it plays an important role in our system, it is necessary to find out in which kind of situation it performs best, feeding entire review or just one segment each time.

5.5 Experiments and Results

5.5.1 Punctuation Filtering Experiment

In this experiment, we evaluate how well ICTCLAS does segmentation and POS tagging of Chinese reviews with and without our *Punctuation Filter* procedure. To recap, since our sentiment identification approach works on sentence segments, we need to be able to reliably detect the sentence and sentence segment boundaries in reviews. The simplest approach is to cut the text at the punctuation character. However, as previously mentioned, on-line user reviews are not formal documents but instead resembles colloquial speech. Punctuations are frequently abused to stress emotions, or simply for convenience (as in the space bar example). If we do not handle these punctuation abuses, they could affect subsequent sentence boundary detection and orientation identification.

Since Chinese word segmentation and POS tagging will greatly affect feature orientation identification, we expect in the least that our *Punctuation Filter* should not make the results worse. Therefore, we compare the accuracies of word segmentation and POS tagging using ICTCLAS with and without *Punctuation Filtering*.

CHAPTER 5. PRE-PROCESSING CHINESE ON-LINE REVIEWS

5.5.1.1 Experiment Setup

Our test¹ data set comprises 100 randomly selected Chinese reviews, 25 each from the four review sites. We deliberately included a larger than normal proportion of noisy reviews. Notice that when we tally the wrongly segmented or POS tagged words (for the non filtered set), we also count the wrongly segmented or POS tagged punctuations (including abused ones). In other words, if a proper punctuation is not segmented/labeled as punctuation, we count it as one wrong POS tag. However, if abused punctuations are not grouped with a proper word or tagged as a punctuation, we ignore them, i.e. we don't care if they are wrong or right. Figure 5.4 shows some examples of wrong POS tags of punctuations. Only by doing so will comparisons make sense. As all we care is the accuracy, the parameter, “recall”, is not evaluated in this experiment.

1. Proper punctuation incorrectly POS tagged:
 Original Sentence:
 这款机样式很不错的,各功能也较好,但是有时会死机.如果像素再大一点就更好了.
 Segmented Sentence:
 这/r 款/n 机/ng 样式/n 很/d 不错/a 的/u ./nr
 各/r 功能/n 也/d 较/d 好/a ./w 但是/c 有时/d 会/v 死/v 机/ng ./w
 如果/c 像/v 索/dg 再/d 大/a 一点/t 就/d 更/d 好/a 了/u ./nr
2. Abused punctuation ignored:
 Original Sentence:
 我喜欢啊~~~~~!!!!!!
 Segmented Sentence:
 我/r 喜欢/v 啊/y ~~~~~/nx !/w !/w !/w
!/w !/w !/w !/w !/w !/w !/w !/w !/w !/w
3. Abused punctuation incorrectly POS tagged:
 Original Sentence:
 还算可以啦~~~~只是我不知道是不是买到水货啦只可以收彩信
 却不能发彩信出去啊气人啊~~~~如果有人知道什么原因请告诉我好吗????
 谢谢 李雄 (QQ156355109)
 Segmented Sentence:
 还/d 算/v 可以/v 啦/y ~~~~/a ~~~~/a 只是/c 我/r 不/a 知道/v 是/a
 不/a 是/a 买/v 到/c 水货/n 啦/y 只/c 可/c 以/c 收/j 彩/ng 信/ag
 却/c 不能/d 发/ng 彩/ng 信/ag 出去/v 啊/e 气人/a 啊/e ~/w ~/w
~/w ~/w ~/w ~/w ~/w ~/w ~/w ~/w ~/w ~/w
 好/a 吗/r ~/w ~/w ~/w ~/w ~/w ~/w ~/w ~/w ~/w ~/w
 李/nr 雄/nr (/w QQ156355109/nx

Figure 5.4: Examples of Wrongly POS Tagged Punctuations (Underlined).

For comparison, we generated a set of “cleaned” reviews by manually correcting any punctuation errors, including eliminating extraneous punctuation and white spaces, and adding missing punctuations at appropriate positions. Altogether, ICTCLAS was applied to

¹There is no training set as no machine learning is involved.

CHAPTER 5. PRE-PROCESSING CHINESE ON-LINE REVIEWS

three sets of reviews, cleaned reviews, raw reviews, and raw reviews processed with *Punctuation Filter*. Because the purpose of this experiment is to evaluate our punctuation filtering procedure, pruning questioning patterns in review is not involved and what being fed into ICTCLAS is whole review. As ground truth, the 100 reviews were manually segmented and tagged to identify 1855 terms, of which 10 were erroneous (“misspelled” Chinese words).

5.5.1.2 Experiment Results

For each of the three sets of input reviews, we compute the number of correctly segmented terms and POS tagged terms with respect to our ground truth set of segmented and POS tagged terms. The precision results are shown in Table 5.1. Note that for segmentation and POS, the incorrect results could be due to either erroneous punctuations *or* inherent limitations of ICTCLAS (i.e. a cleaned version of the review would not have been correctly segmented or tagged by ICTCLAS anyway).

Table 5.1: Precision of ICTCLAS segmentation & POS tagging on the three data sets: cleaned, raw, and *Punctuation Filtered* reviews. Under the Terms column, “+n” refers to the number of invalid terms (e.g., “misspelled” Chinese phrases).

Site	# Terms	Cleaned		Raw		Filtered	
		Seg.	POS	Seg.	POS	Seg.	POS
www.3533.com	588+1	558	522	556	486	557	496
www.163.com	514+4	496	469	477	431	493	453
www.it.com.cn	379+3	356	344	352	333	353	335
www.pcpop.com	364+2	340	331	331	309	337	316
Total	1845+10	1750	1666	1716	1559	1740	1600
Precision		94.34%	89.81%	92.51%	84.04%	93.80%	86.25%

5.5.1.3 Discussion

From the results, we see that ICTCLAS performance is robust to punctuation noise; results on the raw data set was less than 2% worse for segmentation and shy of 6% worse for POS tagging, compared to the manually cleaned data set. In fact, the *Punctuation Filtered* data set only improved ICTCLAS segmentation and POS performance on the raw data set by 1.29% and 2.21%, respectively.

At first glance, it appears that our *Punctuation Filtering* is superfluous. However, upon further examination, we found that ICTCLAS was unable to process 2 out of the 100 raw reviews whilst the same 2 reviews in the cleaned and filtered data set could be processed by ICTCLAS. Thus, we believe that *Punctuation Filtering* remains a relevant and useful preprocessing step. At worst it will not decrease performance, and could potentially help ICTCLAS process more reviews at best.

Obviously, there is still a gap in performance (0.54% for segmentation and 3.56% for POS tagging) between the manually corrected cleaned reviews and the *Punctuation Filtered* reviews. After observing and comparing the wrongly segmented or tagged words from both sets, we found that except for those wrongly processed words present in both sets, most of the wrongly processed words in the filtered reviews were caused by the absence of proper punctuations (to separate sentences or sentence segments). These words were correctly processed in the manually corrected set because missing punctuations were manually inserted. This observation thus provided useful suggestions for improving our “preprocessing” procedure, while at the same time highlighted the important role played by punctuations in Chinese word segmentation and POS tagging.

Among the 10 misspelled words, most of them contained one or more Chinese characters with the same pronunciation but different written form. This is quite reasonable, since most Chinese users enter Chinese characters with the phonetic, i.e. PinYin, input method. The mapping from phonetic to character is one-to-many, thus most spelling mistakes are of the type we call “same sound different shape”. Recall that our “preprocess” procedure does not try to correct these category of errors, but doing so in the future can clearly further improve word segmentation and POS tagging accuracies.

5.5.2 Pruning Questioning Segments Experiment

500 reviews are randomly selected from our on-line cellphone review corpus crawled from the four web sites. After noisy punctuations of them have been filtered, the 500 reviews are divided into sentence segments by the four reserved punctuations. Experiments result

is shown in Table 5.2, where the 2nd column is about the actual count of segments being pruned by each strategy, the 3rd column lists the amount of non-questioning segments of the corresponding strategy's pruned segments and the 4th column shows how many segments of those pruned wrongly are pruned because of abused punctuation or spelling error. There

Table 5.2: Result of Pruning Questioning Sentence Segments.

	QS Pruned	Wrongly Pruned	Abused P. Spell Error
Question Mark	134	11	11
Question Structure	26	4	2
Question Word	113	8	7
Question Stop Word	4	0	0

are 292 question sentence segments in the 500 reviews. 254 of them are pruned and only 38 of them are missed because of the strategies themselves, abused punctuations or not properly used words in segments. That is, nearly 87% of real questioning segments in the 500 reviews are successfully filtered. By looking at the 2nd and 3rd column in Table 5.2, it is easy to find that more than 91% of pruned segments are real questioning segments. And as demonstrated in the 4th column of Table 5.2, most of pruning errors are caused by abused punctuation and spelling error, which are the genius of on-line reviews and can not be avoided. Based on the two observations on the result, most real questioning segments having been pruned and most pruned segments being real questioning segments, it can be said that the questioning pattern eliminating strategies we deploy is quite applicable.

5.5.3 ICTCLAS Utilization Experiment

The purpose of this experiment is to find the way to make full use of ICTCLAS, segmenting and POS tagging Chinese words in higher accuracy. There are two approaches to compare the segmentation results. One is to feed each whole review to ICTCLAS, then get the segmentation results and divide the segmented review into sentence segments by the four reserved punctuations. Another approach is on the contrary, which first divides review into sentence segments and then feed the segments to ICTCLAS one by one to execute word segmentation task.

5.5.3.1 Experiment Setup

1000 reviews are randomly selected from our cellphone review corpus. Prior to segmentation and POS tagging, these reviews are preprocessed such as recovering escaped XML characters, eliminating punctuation noise and question sentence segments. After that, only 961 reviews are not empty, which means 39 of the 1000 reviews are entirely consisted of questioning segments. The rest processed 961 reviews are the experiment data set of this experiment. As our purpose is to find the better way to utilize ICTCLAS, we only check those sentence segments which are segmented differently in the two approaches.

5.5.3.2 Experiment Results and Discussion

There are 3402 sentence segments in the processed 961 reviews. The word segmentation and POS tagging results are listed in Table 5.3.

Table 5.3: Word Segmentation Results.

# Segment	# Different	# Not Matter
3402	319	212
	# Wrongly Segmented	# Wrongly POS Tagged
Review	38	14
Segment	27+17	11

It seems the approach of feeding entire review to ICTCLAS is better. But if the fact that 17 of 44 errors from the segment approach are identical, “buhao” (“not good” in Chinese) being segmented as “bu/d hao/a”, which is not wrong, are considered, the segment approach obviously perform better. Another thing to be noticed is that only 319 sentence segments are segmented differently, which is less than 10% of the 3402 segments. And in the 319 differently segmented segments, 212 of them (“# Not Matter” column in Table 5.3), more than 66% of 319, are processed in the almost identical way, most of which are about stop words. In another word, only less than 2% of total segments are wrongly processed in both approaches. That indicates performances of both approaches are close. Since the segment approach, dividing review into segments first and then performing word segmentation and POS tagging is slightly better, it is deployed in our system.

CHAPTER 5. PRE-PROCESSING CHINESE ON-LINE REVIEWS

Based on these experiment results, the pre-processing procedures on Chinese on-line reviews are organized as Figure 5.1. Such flow can promise to generate most reliable processed Chinese on-line reviews for the sequential procedures in our system. Notice that after this pre-processing procedure, processed segments of a review are still saved together in a plain text document in local file system for the convenience of latter sentiment summarization procedure.

Chapter 6

Learning the Product Concept Hierarchy

The product concept hierarchy used in our system is comprised of product features loosely organized by their meronymic (part of) and hyponymic (kind of) associations. At the topmost level, we define the product type, which is a holonym of three feature types, visible, invisible, and general. Beneath each feature type are the hyponymic feature classes such as Appearance, BasicHardware, AccessoryHardware. Lastly, the leaf nodes are made up of individual features.

The product concept hierarchy plays an important part in two key areas of our system, namely, 1) extraction of Feature/Opinion pairs and, 2) summarization of Feature/Opinion pairs. Since the summarization of Feature/Opinion pairs is to be shown to web users, our product concept hierarchy is naturally organized into a simple tree structure.

In our approach, a product concept hierarchy is initially built by a human using domain knowledge, independent of actual product reviews. The product concept hierarchy is then continuously updated with new concepts or words derived from on-line reviews using machine learning analysis.

6.1 Manual Construction of the Concept Hierarchy

Using domain knowledge, a hierarchical concept model of product features can be created semi-automatically with relative ease. First, we create the hierarchical concept model in

CHAPTER 6. LEARNING THE PRODUCT CONCEPT HIERARCHY

the format of a XML tree with each feature as the leaf node. Then, hierarchical layers are manually added to group similar features together, as shown in the concept model for a cellphone product in Figure 6.1.

Figure 6.1: Hierarchical Concept Model of Cellphone.

At each feature leaf node, a list of similar or related nouns are stored. For example, “wai guan” (appearance) and “wai xing” (shape) are both stored as synonyms of the “Appearance” feature leaf node as shown in Figure 6.1.

We used the Chinese equivalent of WordNet, HowNet [HOW], to look for synonyms of each feature node. The set of synonyms are also known as the feature *synset* [MBF⁺90]. Initially, the feature name is used to search for similar nouns, and if the first or second sense of HowNet returned noun matches any of the search noun’s first two senses, it will be added to the feature synset. The newly added terms are also used to search for synonyms, until there are no more matches. Finally, a manual check and pruning is done to remove irrelevant nouns. Our approach is quite similar to that of Hu [HL04] and Liu [LHC05], with the minor difference of us searching for nouns instead of adjectives and adverbs.

We noticed that this approach of enlarging the feature synset does not work well in practice, primarily due to the limited concept vocabulary and lack of timely updates. Technological products bear the brunt from this limitation because they often include *advante garde* words not modeled by HOWNET. Further, this approach was also unable to extract a substantial number of conceptually related nouns.

It may seem that the most straightforward solution is to continuously update the lexicon with the latest relationships and words. However, this solution is almost impossible in practice due to the complex nature of concepts, limited syntax of the lexicon, and varying user interpretation, all of which collectively belongs to an entirely different research field.

On the other hand, if we can somehow extract the knowledge of a product from the hundreds if not thousands of on-line reviews about it, that would be a more viable alternative. In this chapter, we look at ways of automatically deriving concepts from product reviews using machine learning techniques. The derived concepts can then be systematically reviewed and incorporated into the product concept hierarchy by a human.

6.2 Learning the Concept Hierarchy

To refine the manually created product concept hierarchy, we apply top-down bisection clustering to group similar or concept related nouns together. To this end, we propose a

novel way to measure the conceptual similarity between two nouns, by counting the co-occurrence of neighbor words, i.e., a noun is represented solely by its neighboring words; two nouns are more similar if they tend to occur in sentences sharing a lot of common words. Thus this new similarity measure estimates the conceptual distance between two words based on word usage. To control the clustering procedure, an intra cluster similarity threshold must be experimentally determined. Besides that, no other manual intervention is required.

After clustering at each level, we propose a unique projection method to identify the best hypernym to label each cluster at that level. The final clustering hierarchy thus resemble a tree structure, and could be considered as an automatic way to construct a (rough and initial) concept hierarchy. Moreover, the quality of this automatically created hierarchy can never approach that of a manually created hierarchy, given the limitations in our way of approximating concepts.

Our overall approach can be summarized into three steps, as follows:

- (1) Identify nouns from the review data set.
- (2) Create a corpus-wide contextual representation vector for each noun.
- (3) Cluster from top-down the set of nouns to create a concept hierarchy.

6.2.1 Methodology

6.2.1.1 Collecting Candidate Nouns

Low level semantic word analysis is typically performed at the sentence resolution as the context usually changes across sentence boundaries. This means that we have to be able to segment text into proper sentences ending with a terminating punctuation such as a period or colon. Unfortunately, owing to the colloquial nature of on-line user reviews, punctuations are widely abused, making automatic sentence boundary detection non-trivial. We therefore have to take a step further and perform analysis at the sentence segment level. A sentence segment is a continuous string of words between two adjacent punctuations (end of line

inclusive). The disadvantage of resorting to sentence segment resolution is the possible loss of higher level contextual information. However, for the vast majority of sentences, the affect is minimal. For example, a sentence “This product is OK, except for the high price.” will likely result in one positive one negative sentiment extracted from each of the two segments, and one negative sentiment extracted for the whole sentence, in which case the sentence segment analysis results is more accurate!

The candidate noun set is comprised of all POS-tagged nouns. Here we adopt the simple *Segment Frequency* (SF) criterion, an analogy of document frequency, to pick important nouns from the candidate noun set for analysis. A noun’s SF refers to the number of segments containing it in the corpus, thus a noun with a low SF is rare and not representative. Assuming that frequently mentioned nouns have a higher probability of being a product feature, we pick nouns with a SF value of at least 7%-10%. Keep in mind that this simple assumption will inadvertently prune infrequent feature nouns and include a whole lot of frequent non-feature nouns.

6.2.1.2 Context Vector for Representing a Noun

Unlike the vast majority of existing approaches, a noun is represented as a vector space made up completely of its common neighbors, except for itself. In effect, the noun is defined by all of its typical neighbors, constructed in the following two step procedure:

- (1) Extract a set of N term vectors $S = \{\mathbf{s}_n\}_{n=1}^N$ to represent each review segment in the corpus of on-line reviews. Assuming there are D terms in the lexicon, then $\mathbf{s}_n \in \mathbb{R}^D$. Binary representation is used here since there is not much difference between binary and Term Frequency representation for segments, which are typically short(10-20 words).
- (2) For a candidate feature noun $w_d \in V$, where $V = \{w_1, \dots, w_D\}$ is the set of all words in the lexicon. Let T be the subset of binary term vectors from S containing a non zero weight for candidate noun w_d , i.e., $T = \{\mathbf{t} : (\mathbf{t} \in S) \wedge (t_d = 1)\}$. Define the

context vector $\mathbf{u}_d$ for noun w_d as follows:

$$\mathbf{u}_d = \mathbf{U} \sum_{t \in T} \mathbf{t} \quad (\text{Eq. 6.1})$$

where $\mathbf{U}$ is a diagonal matrix of all ones except for one zero at the d -th position:

$$\mathbf{U} = [u_{ij}]_{D \times D} = \begin{cases} 1, & \text{if } i = j \neq d \\ 0, & \text{otherwise} \end{cases} \quad (\text{Eq. 6.2})$$

Intuitively, each context vector $\mathbf{u}_d$ is simply an average of all words used in the context of w_d . As with standard term weighting schemes, $\mathbf{u}_d$ is normalized to unit length to facilitate cosine similarity computation.

Enumerating all possible neighbors of a noun over all sentence segments containing the noun ensure that each noun is represented by a unique context vector, which has several advantages:

- (1) A rich representation of the noun is created, which also approximates the relative occurrence frequency of each of the neighboring words used in the context of that noun.
- (2) We avoided the problem of a noun having different context representations from various sentence segments, which might end up in different clusters, making interpretability difficult.
- (3) The influence of frequently occurring neighbors are enhanced while at the same time decreasing the effect of outlier neighbors. For example, consider the following hotel review sentence segment, “Both the environment and services are excellent”, from which the extracted context representation for “environment” and “services” would have 80% overlapping terms. This similarity will be significantly reduced after we average all the segment context vectors for “environment” and “services”, respectively.

Considering every word in a segment as a noun’s neighbors might inadvertently add noise to its context representation. Intuitively, words lying nearer to the noun carry more context. We thus use a *Sliding Window* scheme to determine the optimal range of neighboring words to include. For example, a *Sliding Window* of size 3 will only use the 3 words immediately before and after (total of 6 words) the candidate noun as its context representation.

6.2.1.3 Top Down Clustering of Feature Context Vectors

To find the hypernymic relationships between different feature nouns, we apply a top down binary clustering of all feature context vectors. We used the cosine similarity to measure the degree of similarity between two feature context vectors $\mathbf{u}$ and $\mathbf{v}$, which is defined as follows:

$$\text{sim}(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u}^T \mathbf{v}}{\|\mathbf{u}\| \|\mathbf{v}\|} \quad (\text{Eq. 6.3})$$

We use CLUTO [CLU] perform top-down bisection clustering. In general, determining the optimal number of clusters k for a dataset remains an open research problem. Thus, instead of specifying k , an intra-similarity cluster threshold is used to determine when to stop splitting a cluster, defined as:

$$IS = \frac{\sum_{\mathbf{u} \neq \mathbf{v}} \text{sim}(\mathbf{u}, \mathbf{v})}{n(n-1)} \quad (\text{Eq. 6.4})$$

where n is the number of vectors in the cluster.

The algorithm starts by splitting all feature context vectors into two clusters. The intra-similarity metric of data in each of the two clusters is then computed. If a cluster’s metric exceeds the threshold, it is labeled as a final cluster and no more clustering will be done. Otherwise, bisection clustering is applied to the sub-cluster and the procedure continues iteratively, until all clusters have intra-similarities exceeding the preset threshold (i.e., final). Figure 6.2 shows a detailed flow chart of our clustering approach.

As shown in Figure 6.2, bisection clustering creates a binary tree structure with the final clusters as leaf nodes. Meaningful words could be assigned to label each node in the

CHAPTER 6. LEARNING THE PRODUCT CONCEPT HIERARCHY

Figure 6.2: Procedure of Clustering Vectors and Constructing Concept Hierarchy.

tree, thereby transforming it into a useful concept hierarchy. Since we are clustering feature nouns, we could naturally select the most representative feature context vector (hypernym) under each node as its text label.

We tried various approaches of finding the most representative vector, including the vector nearest to the mean. In the process, we found two methods seem to work well for our specific task. The first method assumes that a noun that shares the largest number of words with other nouns is most likely a broader term. The second method computes the *z-score*.

In the first method, for each node we compute the feature overlap vector $\mathbf{o}$ for all members underneath it as follows:

$$\mathbf{o} = \begin{cases} 1, & \text{if there are at least a fraction } p \text{ vectors } \mathbf{u} \text{ where } u_d > 0 \\ 0, & \text{otherwise} \end{cases}$$

The overlap vector can be visualized as a binary projection of all features onto an empty vector, as shown in 6.3 where $p = 0.5$. Then we pick the vector that has the highest amount of binary overlap with the overlap vector as the most representative noun. In case of a tie,

CHAPTER 6. LEARNING THE PRODUCT CONCEPT HIERARCHY

we pick the vector with a higher total weight. For example, in Figure 6.3, although both vectors 1 and 2 yield 3 non-zero entries with the overlap vector, vector 2 is picked over vector 1 as vector 2 has a larger sum weight of its overlapping terms, 7 versus 4.

Figure 6.3: Projection Based Hypernym Selection. Here the density vector captures only features with more than 50% overlap.

The second method of selecting the best representative is based on the internal (z_k^I) and external (z_k^E) z -scores for a vector $\mathbf{u}_i$ in the k -th cluster, defined as follows [CLU]:

$$z_k^I(\mathbf{u}_i) = \frac{s_{ik}^I - \mu_k^I}{\sigma_k^I} \quad (\text{Eq. 6.5})$$

$$z_k^E(\mathbf{u}_i) = \frac{s_{ik}^E - \mu_k^E}{\sigma_k^E} \quad (\text{Eq. 6.6})$$

where for the N_k vectors in cluster set C_k we have,

$$\begin{aligned}
 s_{ik}^I &= \frac{1}{N_k} \sum_{\mathbf{v} \in C_k} \text{sim}(\mathbf{u}_i, \mathbf{v}) \\
 \mu_k^I &= \frac{1}{N_k} \sum_i s_{ik}^I \\
 \sigma_k^I &= \frac{1}{N_k} \sum_i (s_{ik}^I - \mu_k^I)^2 \\
 s_{ik}^E &= \frac{1}{N_k} \sum_{\mathbf{v} \notin C_k} \text{sim}(\mathbf{u}_i, \mathbf{v}) \\
 \mu_k^E &= \frac{1}{N_k} \sum_i s_{ik}^E \\
 \sigma_k^E &= \frac{1}{N_k} \sum_i (s_{ik}^E - \mu_k^E)^2
 \end{aligned} \quad (\text{Eq. 6.7})$$

Intuitively, vectors with large internal z -scores and low external z -scores tend to be the centroid of a cluster. Thus, the most representative vector is the vector with the largest

internal and smallest external z -scores. In case of a multiple candidates, the vector with the largest internal z -score and/or smallest external z -score is preferred.

We implemented both the *projection* and z -score hypernym selection methods, and detailed experiment results will be given in next section.

6.2.2 Experiments and Results

6.2.2.1 Data Sets

Three on-line review sets were collected:

- (1) **Hotel.** This data set contains 2000 Chinese hotel reviews randomly selected from 14929 reviews on 423 different hotels in Shanghai. The 14929 reviews are crawled from *www.ctrip.com*, which is the largest on-line travel portal in China.
- (2) **Cellphone.** This data set contains 2000 Chinese Cellphone reviews randomly selected from the 313102 Chinese cellphone reviews we have crawled from four major Chinese websites.
- (3) **Mp3player.** This is a smaller public dataset comprising 95 English reviews on Creative’s Nomad Jukebox Zen Xtra 40GB Mp3player collected from *www.amazon.com*. This dataset has been previously evaluated and made available by Hu [HL04].

The three datasets cover different domains and languages, each with their own characteristics. For example, both the hotel and cellphone reviews are short as they largely include real personal experiences. On the other hand, the Mp3player reviews are longer, and contain less noise (spam reviews).

The only natural language processing tool required for our approach is the Part-Of-Speech (POS) tagger, which is needed to identify nouns. For Chinese, we use ICTCLAS [ICT] to segment words and perform POS tagging. For English, Brill’s tagger [RBT] was used. Preprocessing such as stemming for English, punctuation filtering, question segmentremoval, etc., are also performed. Table 6.1 shows the distribution of the three review sets after processing.

Table 6.1: Distribution of the Three Review Sets.

	Hotel	Cellphone	Mp3player
# reviews	2000	2000	95
# segments	15628	6703	3081
average # words per segment	5.27	4.96	8.9
# unique nouns	1813	695	1117
# unique words	5986	3496	2431

6.2.2.2 Selecting Candidate Feature Nouns

Although the ideal Segment Frequency (SF) cut-off point should be 7-10%, we found that in practice, simply selecting the top 200 most frequent (in terms of highest SF) nouns as feature candidates suffices. Note that single character nouns are a rarity in Chinese, and are therefore discarded. Different SF thresholds were set to extract a comparable number of candidate nouns as shown in Table 6.2.

Table 6.2: SF Thresholds for Selecting Candidate Nouns from the 3 Review Sets.

	Hotel	Cellphone	Mp3player
SF Threshold	10	3	8
# Candidate Nouns	175	161	163

6.2.2.3 Varying the Amount of Context

Each noun context vector can include varying amount of neighboring words, up to the total number of words in the segment. Naturally, including all words in the segment may result in too much irrelevant context, while including too few neighboring words may be too specific. Let WS be a *Sliding Window* parameter that specifies WS words to include before and WS words after the noun in the segment. Noting from Table 6.1 that on average there are less than 6 words per segment, we vary WS from 1, 2, 3, and ∞ .

Table 6.3 lists the number of non-zero entries over all context vectors in the corpus for different WS . Intuitively, as WS is increased, more context information is added, as seen in the larger number of non-zero entries in the context vector.

Four different intra-similarity threshold values, $T = 0.15, 0.2, 0.25, 0.3$ are used to evaluate the relationship of WS to final cluster count, as shown in Table 6.4.

CHAPTER 6. LEARNING THE PRODUCT CONCEPT HIERARCHY

Table 6.3: Total Number of Non-zero Entries over All Context Vectors.

	Hotel	Cellphone	Mp3player
$WS = \infty$	23198	7576	23969
$WS = 3$	15398	5104	10423
$WS = 2$	11625	3931	7299
$WS = 1$	6289	2159	3809

Table 6.4: Clusters' Count of Different WS Size and Intra-similarity Threshold T .

	Hotel			
	T=0.15	T=0.20	T=0.25	T=0.30
$WS = \infty$	19	28	44	55
$WS = 3$	20	32	46	60
$WS = 2$	24	34	44	60
$WS = 1$	25	34	48	60
	Cellphone			
	T=0.15	T=0.20	T=0.25	T=0.30
$WS = \infty$	17	29	42	51
$WS = 3$	20	32	43	53
$WS = 2$	21	32	42	55
$WS = 1$	19	32	43	56
	Mp3player			
	T=0.15	T=0.20	T=0.25	T=0.30
$WS = \infty$	10	22	33	45
$WS = 3$	19	31	40	52
$WS = 2$	18	30	39	55
$WS = 1$	20	28	44	52

CHAPTER 6. LEARNING THE PRODUCT CONCEPT HIERARCHY

From Table 6.4, it is obvious that at the same threshold T , the number of clusters decreases with increasing WS , reaching the minimum value at $WS = \infty$. This is expected, since a smaller WS would correspond to a feature context vector with smaller vocabulary, making it less similar to other context vectors, thereby resulting in more clusters.

In practice, we found that restricting WS actually result in the creation of more *irrelevant* clusters. This is a natural result of not having enough data; by restricting WS , we are effectively requiring the noun similarity comparisons to be more specific, thereby decreasing the number of available data, which can be seen in Table 6.3. Therefore, in the later experiments, we will use $WS = \infty$.

As our concept hierarchy construction relies heavily on the clustering quality, we evaluate the top-down bisecting clustering performance against the default flat CLUTO clustering method at the same number of clusters, k , which is determined by the bisecting clustering approach at a given stopping criterion. We use the average Intra-cluster similarity (Eq. 6.4) as the evaluation metric:

$$IS_{avg} = \frac{1}{k} \sum_k IS_k \quad (\text{Eq. 6.8})$$

We compute the change in IS_{avg} as follows:

$$\Delta IS_{avg} = \frac{IS_{avg}^{bisecting} - IS_{avg}^{flat}}{IS_{avg}^{default}} \quad (\text{Eq. 6.9})$$

Table 6.5 shows the comparison of the two clustering. It can be seen that the overall IS_{avg} of our bisecting approach is about 12% worse than the pre-specified k approach. This is because if k is known, as in the default CLUTO clustering method, some adjustments are made to reassign vectors to other clusters, thereby achieving a better IS_{avg} . On the other hand, the bisecting clustering approach we use is a top-down clustering approach that does not back-track.

Moreover, despite the slight decrease of 12% in IS_{avg} , we believe the principal advantage of using a bisecting approach lies in its generated hierarchy, and is a necessary trade-off.

Table 6.5: Comparison of Clustering Results

	T=0.15		T=0.20	
	k	ΔIS_{avg}	k	ΔIS_{avg}
Hotel	19	11.30%	28	9.22%
Cellphone	17	17.39%	29	8.80%
Mp3player	10	-0.51%	22	15.24%
	T=0.25		T=0.30	
	k	ΔIS_{avg}	k	ΔIS_{avg}
Hotel	44	12.94%	55	10.78%
Cellphone	42	12.16%	51	12.00%
Mp3player	33	12.99%	45	16.21%

6.2.2.4 Extracting Hypernyms

Now that we have experimentally validated the best feature context representation parameter $WS = \infty$ and evaluated our bisecting clustering approach against the flat clustering approach, what remains is determining the best approach to label each node in the hierarchy.

To evaluate the hypernyms extracted for each node in the hierarchy, first and foremost, the number of clusters should be reasonable. Having too many clusters may reduce the number of segments per cluster resulting in poor generalization of the extracted hypernyms over specific, too few clusters would group unrelated nouns together and resulting in over generalized hypernyms. We have evaluated various similarity thresholds and found $T = 0.20 - 0.25$ to be a reasonable compromise.

After the candidate nouns are clustered, we apply the two hypernym extraction methods, *projection* and *z-score*, to evaluate each node in the cluster hierarchy. A human will evaluate the suitability of each extracted hypernym for the same node. Table 6.6 shows the result of the human evaluation.

From Table 6.6 we see that both hypernym extraction methods agree approximately 67.52% of the time irregardless of correctness. They disagree 15.46% of the time, at which both produced reasonable hypernyms. For the remaining 17% of the nodes, our *projection* method was subjectively found to be more accurate 60% (10.31/17) of the time.

CHAPTER 6. LEARNING THE PRODUCT CONCEPT HIERARCHY

Table 6.6: Subjective Evaluation of *Projection* and *Z-score* Extracted Hypernyms for Each Node in the Hierarchy.

	<i>Projection</i> better	<i>Z-score</i> better	Different (both OK)	Same Hypernyms	Total Extracted
Hotel (T=0.20)	3	2	6	44	55
Hotel (T=0.25)	4	4	11	68	87
CellP. (T=0.20)	11	5	12	29	57
CellP. (T=0.25)	11	9	20	43	83
Mp3P. (T=0.25)	4	3	4	32	43
Mp3P. (T=0.25)	7	3	7	46	63
Sum	40	26	60	262	388
Percent	10.31%	6.70%	15.46%	67.52%	100%

One minor disadvantage of the *z-score* method is that it can only be computed on a partitioning/clustering, which means that it cannot be used to compute the hypernym representing the root node (before any clustering).

We observed that the hypernym extracted using our *projection* method does not always corresponding to the noun with the largest Segment Frequency (SF). For example, the root hypernym for the Chinese Hotel review dataset is “jiudian” (“Hotel”, SF=1118) instead of “fangjian” (“room”, SF=1409) which has a higher SF. In fact, “room” is a child node of “hotel”, which makes perfect sense. Likewise, the root hypernym for the Cellphone and Mp3player datasets are “shouji” (“handphone”, SF=344) and “player” (SF=270), respectively.

We thus conclude our *projection* method to be effective in labeling a node, as it will attempt to find the most suitable noun regardless of the SF.

6.2.2.5 Evaluating a Concept Hierarchy

Evaluating a concept hierarchy is an extremely difficult and subjective task that depends heavily on the evaluator’s domain knowledge, personal understanding, mood, and bias. Comparing two hierarchies is even harder. Intuitively, an ideal product concept hierarchy should group similar nouns or concepts at the leaf nodes, with higher level nodes describing broader concepts.

CHAPTER 6. LEARNING THE PRODUCT CONCEPT HIERARCHY

Naturally, one way of evaluating a hierarchy is by determining the correctness of each parent-child link, but how should one quantify or qualify “correctness”? To this end, we adopt the five relationship definitions of Sanderson [SC99].

To evaluate our hierarchy, we benchmark it against Sanderson’s subsumption approach [SC99], which is focused on deriving non-hierarchical disconnected relationships. As the latter’s approach is quite different from our *projection* method of deriving a single hierarchy, to ensure a fair comparison, we ensure that both graphs have an approximately equal number of unique edges/links.

Unique edges are links that connect two non-identical nouns, which make up the majority of the edges in both methods. Note since we do not enforce each node to have a unique hypernym in our *projection* method, as a result, non-unique edges may be present.

We selected 3 clustering results with the lowest reduction in IS (emphasized in bold in Table 6.5) to benchmark our *projection* hypernym extraction. Note that the three selected hierarchies may be locally optimal in terms of noun intra-similarity, but could otherwise be sub-optimal with respect to the best hierarchy. Sanderson’s method, on the other hand, is not restricted to finding a strict hierarchy.

Table 6.7 lists the number of unique edges and nouns for the graphs/trees obtained by both methods. Both methods were given the same set of candidate nouns and reviews. Sanderson’s mutual inclusive threshold value was adjusted so that it resulted in an approximate number of unique edges as our *projection* results. From Table 6.7, Sanderson’s result include far fewer nouns at the same link density as our *projection* method.

Table 6.7: Statistics for *Projection* and Sanderson’s Graphs.

	# Unique edges		# Nouns captured	
	Projection	Sanderson	Projection	Sanderson
Hotel	176	154	175	120
Cellphone	163	163	161	126
Mp3player	167	169	163	125

We manually classify each unique edge in both graphs to one of 5 types as listed in Table 6.8. Sanderson’s method clearly benefits from this simplistic evaluation methodology. This is because our hierarchies are top-down binary, which restrict hypernyms at each

CHAPTER 6. LEARNING THE PRODUCT CONCEPT HIERARCHY

intermediate node to have only two children, thus limiting the number of edges for each noun. Further, the hierarchical nature of our tree means that only the first two relationships should be considered correct in the strictest sense. On the other hand, each edge in Sanderson’s graph is considered “correct” as long as any of the 4 relationships applies.

Table 6.8: Results of Noun Pair Evaluation.

	Hotel		Cellphone		Mp3player	
	Proj.	San.	Proj.	San.	Proj.	San.
1) Aspect of	30.7%	42.9%	24.5%	27%	38.3%	27.2%
2) Type of	6.8%	0%	4.3%	1.2%	7.2%	4.1%
3) Same	19.3%	3.3%	5.5%	4.9%	7.2%	3.0%
4) Opposite	0%	2.0%	1.8%	4.3%	0%	0%
5) Unknown	43.2%	52%	63.8%	62.6%	47.3%	65.7%
Sum (1-3)	56.8%	46.2%	34.3 %	33.1%	52.7%	34.3%
Sum (1-4)	56.8%	48.2%	36.1%	37.4 %	52.7%	34.3%

As indicated in [SC99], only the first three type of relationships, “Aspect of”, “Type of”, and “Same” should be considered “correct” for our *projection* concept hierarchy. Whereas even if the fourth relationship holds true for some pairs within the hierarchy, it should be considered outright “wrong”. Thus, if we consider the sum of correct type 1 to 3 noun pairs, our *projection* approach was 10.6%, 1.2%, and 18.4% better than Sanderson’s approach for each of the 3 datasets, which averages to about 10% across-the-board improvement. Note that both methods performed the worst for cellphone due to the poor data quality caused by many irrelevant posts.

As the complete tree and graph for each dataset is quite large, we only show part of it here. To facilitate objective comparison of the two graphs, sub-hierarchies consisting of the same nouns are extracted and shown. A sub-tree from the “projection” hierarchy is first selected, then a sub-graph from Sanderson’s hierarchy containing the same nouns is extracted and shown. Note that not all nouns in the “projection” hierarchy may be available in the even the full Sanderson graph, as shown in Table 6.9.

Sub-hierarchies and sub-graphs for the three datasets are shown in Figures 6.4 to 6.6.

The *projection* method of labeling each node goes hand-in-hand with the top-down

CHAPTER 6. LEARNING THE PRODUCT CONCEPT HIERARCHY

Table 6.9: Count of Nouns in *Projection* and Sanderson's Graphs.

	Hotel	Cellphone	Mp3player
Projection	12	9	12
Sanderson	7	7	11

6.4.a: Projection

6.4.b: Sanderson

Figure 6.4: Sub-hierarchies of Hotel.

6.5.a: Projection

6.5.b: Sanderson

Figure 6.5: Sub-hierarchies of Cellphone.

CHAPTER 6. LEARNING THE PRODUCT CONCEPT HIERARCHY

6.6.a: Projection

6.6.b: Sanderson

Figure 6.6: Sub-hierarchies of Mp3player.

clustering procedure, thus it will not remove any candidate nouns from the set even if they were selected as the node label (hypernym). As a result, two anomalies may arise:

- (1) The same node labels may be subsequently assigned to one or more descendants, resulting in edges such as “zhiliang-zhiliang” (“quality-quality”) in Figure 6.5 and “track-track” in Figure 6.6.
- (2) Edges with reversed relationships may be created, e.g., “track-album” in Figure 6.6.

To make the comparison with Sanderson’s method fair, both of these cases are treated as class 5, i.e., “Unknown”, and do not contribute to the scores of our *projection* method in Table 6.8.

From the comparison of sub-hierarchies between our method and Sanderson’s, as shown in Figures 6.4, 6.5, and 6.6, we see that our method has the following qualitative advantages in addition to the quantitative advantages previously presented in Table 6.8:

- (1) Our method can produce a neat hierarchical structure that is intuitive to end users. On the other hand, Sanderson’s method generates fragments of disconnected graphs that has no clear hierarchical ordering.

- (2) Our method can identify synonyms and meronyms effectively as descendants of a node, whereas in Sanderson’s graph, synonyms may be disconnected and shown in different graphs.

6.2.3 Discussions

As demonstrated in the quantitative experimental results and qualitative sub-hierarchy figures, our proposed machine learning approach can automatically group similar and related nouns of a particular product into an intuitive hierarchy that is both user-friendly and up-to-date.

Our original goal was not to replace the manual update of product concept hierarchy completely with this automated procedure, but to use the automatically generated hierarchy as a real-time guide for a human operator to revise and update the existing product concept hierarchy. In detail, by matching nouns both involved in the automatically generated hierarchy and existing hierarchy, newly extracted nouns are put into appropriate position or appended as another child node of certain noun in the existing hierarchy. Meanwhile, the structure of the existing hierarchy will be adjusted if the corresponding part of the automatically generated hierarchy is more reasonable. Both of the procedures are to be done manually.

There are some limitations of our existing approach which will require additional future work, as listed below:

- (1) The current method of collecting candidate nouns for consideration as a node label is quite simple, relying solely on a noun’s *Segment Frequency* (SF). Candidate nouns with high SF may not be relevant to our concept hierarchy, while on the other hand, nouns with low SF that we left out may represent emerging product features.
- (2) Our current construction of the feature context vector for a candidate noun includes all words such as nouns, verbs, adjectives, and stop words. The added noise may produce a lot of false similarity among different feature context vectors, especially

CHAPTER 6. LEARNING THE PRODUCT CONCEPT HIERARCHY

knowing the fact that each feature context vector is a sum of all context vectors containing the same noun.

- (3) Our hierarchy is currently binary at the intermediate nodes, a limitation imposed by the bisecting clustering algorithm that we have chosen. This is not a major problem because at the leaf nodes, we can still have multiple nouns. However, for some concepts, allowing for multi-way split would make more sense. This problem manifests itself in the repeated election of the same node labels along some paths of the tree.
- (4) Currently, we allow a candidate noun to be elected multiple times along the path from the root node to the leaf nodes. This is allowed partly to account for the limitation imposed by our binary split. However, with the incorporation of a multi-way split, we should remove a noun from consideration once it has been selected as a label for a parent node.

In conclusion, we have achieved what we set out to do at the beginning of this chapter, i.e., to come up with a workable and useful system to help a human operator to continually refine an existing product concept hierarchy. Without an automated system such as ours, a product concept hierarchy is destined for obscurity once it leaves the creators' hands.

Chapter 7

Sentiment Identification

Sentiment identification attempts to objectively gauge the opinions of user by modeling and extracting *Feature/Opinion* pairs from sentence segments. The whole process usually entails two stages as follows. First, a sentence segment is analyzed to determine the existence of human sentiment. Second, pending the positive outcome from the first stage, the actual sentiment will be extracted.

During the second stage, the segment orientation (O) is first identified, followed by searching for matching nouns in the segment to the feature nouns of the product concept hierarchy. If there is a matching feature (F), it is paired with the extracted orientation. If no matching nouns can be found, a special feature “General” will be assigned to the orientation. In either case, one or more FO pairs will be produced as the extracted feature orientation pair(s).

As the part on finding a matching feature (F) is trivial and largely depends on the quality of the product concept hierarchy, the focus of this chapter lies on accurately extracting the sentence segment orientation. In particular, we propose two different orientation identification approaches, 1) using an orientation lexicon (*Concept*), and 2) using machine learning to acquire the sentiment from segments modeled with an adjacency matrix representation (*Matrix*).

7.1 Lexicon Based Sentiment Identification

The basic idea of lexicon-based sentiment identification is to simply look for opinion words (adjectives, adverbs, and verbs) with known sentiment orientation in the candidate sentence segment. If multiple opinions are found, the overall opinion is a simple sum of the individual opinions, with negative, neutral, and positive opinions each assigned a weight of -1 , 0 , $+1$, respectively.

We applied two heuristics to enhance this simple approach, 1) account for the reversing effect of negation words such as “but”, “not”, “don’t”, and 2) treat the class of strong opinion words differently. Strong opinion words are words that are strongly linked to a given feature, e.g., “heavy” to weight.

7.1.1 Associating Opinion Word with Feature

Product features typically vary across product types, and the associated opinion words could have entirely opposite orientation depending on the product type. For example, a “large” display for a cellphone is generally a positive thing, whereas a “large” display for a laptop computer may be a minus (bulky). In order to capture the subtle differences between features and sentiments across different products, we need to add a collection of sentimental words and their known orientation to each feature leaf node in the hierarchical concept model. Specifically, under each product feature leaf node, one child node, “StrongOpinionWords”¹ are maintained. The “OpinionWord” node includes three child nodes, “Positive”, “Neutral”, and “Negative”, each containing the relevant verbs, adjectives, and adverbs commonly used to describe the parent feature. For example, the adjective “cheap” (positive) is assigned to the leaf node “Price”, and the adjective “heavy” (negative) is assigned to the leaf node “Weight” in the cellphone concept hierarchy.

To populate the “GeneralOpinionWord” node, we use the method of [HL04][LHC05] to obtain related terms from HowNet [HOW]. First, we create the “seed” opinion word set which includes adjectives, verbs, and adverbs. Next, we add a new term into the

¹Called “GeneralOpinionWord” for the “General” feature node.

corresponding sentiment node’s “seed” set if that term’s first two senses (according to HowNet) match the first two senses of *any* word already in the “seed” set. This procedure is recursively executed until all matching words in HowNet are exhausted.

7.1.2 Orientation Identification and FO pair Composing

We can now easily identify the sentiment of each sentence segment using the modified hierarchical concept model. We found from our experiments that if we only consider sentences containing at least one product feature, we were able to achieve very high precision but low recall for product feature sentiment classification. The low recall is due to the fact that the subject is often omitted in reviews, e.g., the colloquial expression “*looks nice.*” is frequently used instead of the more formal expression “*the exterior looks nice.*”

Thus, if we wish to extract the complete set of user sentiments, we need to consider sentences without explicit product features. In our case, if we detect sentences expressing sentiment (not expressed through “StrongOpinionWords”) with no matching product feature, we will assigned to it a “General” feature. For example, a positive sentiment is assigned to the feature “General” for the the colloquial expression “*looks nice.*”.

Sentiments associated with “StrongOpinionWords” will be assigned to the the corresponding product feature in the product concept hierarchy, regardless of whether the feature noun actually appeared in the sentence.

With the aid of the hierarchical concept model, the orientation of each sentence segment is extracted as follows:

- (1) Search sentence segment for an unprocessed “StrongOpinionWord” as listed in the concept model. If found, determine² orientation O of that opinion word from concept model. Reverse orientation if a negation word exists within a 3-word neighborhood of the opinion word. Assign the implicit feature F to the opinion O , and append the FO pair to the segment. Repeat this step until all “StrongOpinionWords” have been processed.

²Recall that an opinion word orientation varies depending on the product type.

CHAPTER 7. SENTIMENT IDENTIFICATION

- (2) Initialize Segment Count (SC) to 0. The segment count SC tracks the overall orientation of the sentence segment.
- (3) Search sentence segment for an unprocessed “GeneralOpinionWord”. Exit if none can be found, otherwise determine orientation O from concept model. Negate orientation if a negation word exists within a 3-word neighborhood of the opinion word. Increment SC by one if orientation O is positive, and decrement SC by one if negative. Repeat this step until all “GeneralOpinionWords” have been processed.
- (4) Derived overall segment orientation O as positive if $SC > 0$, negative if $SC < 0$, and neutral if $SC = 0$.
- (5) Search sentence segment for product features according to the concept model. If no features are found, assign F to the “General” feature³. Assign the feature F to the segment orientation O and append FO pair to segment. Repeat until all features have been found.

Figure 7.1 highlights the procedure using an activity diagram.

As an example, the sentence segment “good and cheap” will be tagged with the two FO pairs as “good and cheap//Price/+|cheap;General/+|good”, which means that a total of two positive sentiments have been identified, one on price and one general. Note that our approach here does not care whether a feature is “explicit” or “implicit”. As long as there is a known opinion word in the sentence segment, our approach can always extract its orientation and assign a corresponding (or general) feature. This is quite different from the approach of Hu [HL04] and Liu [LHC05], which first finds feature words in a sentence before identifying the corresponding sentiment. The primary advantage of our approach is that it achieves a higher recall (without missing out on the strong opinions).

³The “General” feature is assigned if and only if there is at least one “GeneralOpinionWord” in the segment.

CHAPTER 7. SENTIMENT IDENTIFICATION

Figure 7.1: Activity diagram for identifying feature orientation pairs from a sentence segment, which is an elaboration of the “Extract Feature Orientation” process in Figure 3.2.

7.2 Adjacency Matrix Representation for Sentence Classification

Despite throwing away the sequence information of words, the venerable bag-of-words vector space Model (VSM) text representation and its variants [MN98] have demonstrated remarkable accuracy and robustness in many document retrieval tasks [YP97, YL99]. This is partly due to the fact that what is lost in word sequence information is made up for by the sheer volume of words available in a typical document. Moreover, word sequence and context plays an important role in sentence segment sentiment classification. Further, the number of words/features in a sentence is one to two orders of magnitude less than a document, which means that subtle changes in term sequences would completely alter the sentence semantics. Thus, the vector representing a sentence is extremely sparse [MS03].

As a matter of fact, bag-of-words models have been shown to consistently underperform in sentence retrieval tasks like sentiment classification and event detection [ALJ00] because the limited features and context embedded in the ordering of terms in a sentence are largely lost in a VSM representation.

Hence, we propose using a high-order term adjacency matrix to capture the semantics embedded in the sequences of a text sentence. One nice feature about the matrix representation is its completeness property, i.e., the original sentence can be fully recovered from the adjacency matrix. We evaluate the performance of our matrix representation against the classical unigram, bigram, and trigram binary bag-of-words representation on two sentence classification tasks, namely sentiment classification and news event detection. Preliminary results on both English and Chinese corpus looks very promising.

7.2.1 Matrix Representation Methodology

7.2.1.1 Model

Definition 7.1 (*Dictionary*) Let $\mathbf{w}$ be a set of N distinct terms, $\mathbf{w} = \{w_1, \dots, w_N\}$, where w_i denotes the i -th term.

CHAPTER 7. SENTIMENT IDENTIFICATION

Definition 7.2 (*Sentence*) A sentence $\mathbf{s}$ is a sequence of m terms, $\mathbf{s} = [s_1, \dots, s_m]$, where $s_i \in \mathbf{w}$.

Definition 7.3 (*t-Order Adjacency Matrix*) Given a sentence $\mathbf{s}$ with m terms, its t -Order Adjacency Matrix $\mathbf{M}^t = [m_{ij}^t]_{m \times m}$ where each entry is defined as:

$$m_{ij}^t = \begin{cases} m_{ij}^t + 1, & \text{if } \exists u, v = u + t \text{ such that } s_u = w_i \text{ and } s_v = w_j, \\ 0, & \text{Otherwise.} \end{cases} \quad (\text{Eq. 7.1})$$

The 0-order adjacency matrix values are all zeros except for those terms contained in the sentence, i.e., $\mathbf{M}^0 = \text{diag}[m_1, \dots, m_N]$ where $m_i = 1$ if $w_i \in \mathbf{s}$. In fact, the vector formed by the diagonal entries of $\mathbf{M}^0$ degenerates to the VSM vector.

The 1-order adjacency matrix representation for a sentence simply measures the immediate co-occurrence of two terms w_i and w_j , i.e., $m_{ij}^1 = 1$ if w_j occurs immediately after w_i in the sentence. In fact, $\mathbf{M}^{t+1}$ can be iteratively computed from $\mathbf{M}^t$ using simple graph transitivity property as follows:

$$m_{ij}^{t+1} = \sum_{k \geq i}^N m_{ik}^t m_{kj}^1 \quad \forall i \leq j, t > 0 \quad (\text{Eq. 7.2})$$

For a sentence with m terms, up to the $(m - 1)$ -order adjacency matrix can be computed. We thus propose the following.

Proposition 1 (*Matrix Representation*) A sentence $\mathbf{s}$ containing m terms can be represented by its $m - 1$ order adjacency matrix.

Figure 7.2 shows an example of our matrix representation of the four word sentence, “I like this phone”. For simplicity, only the 4×4 sub-matrix of the full $N \times N$ matrix is displayed, whereas all other terms are zero.

CHAPTER 7. SENTIMENT IDENTIFICATION

Figure 7.2: Representing a Four-word Sentence with a 3 Order Adjacency Matrix.

7.2.1.2 Advantages of Matrix Representation

As mentioned, one serious problem with VSM for sentence is its data sparseness. Given a sentence comprising m unique terms, the density of its VSM vector is $\frac{m}{N}$, whereas the density of its $m - 1$ order adjacency matrix is $\frac{m(m-1)}{2N'}$ where $N' = N^2$ for a full adjacency matrix. Consider the inequality, $\frac{m(m-1)}{2N'} > \frac{m}{N}$, if we can heavily prune the infrequent entries of the adjacency matrix, such that $N' = N$ instead of the full $N' = N^2$, we obtain the limit $m > 3$.

Another implicit advantage of the matrix representation is that in theory, the sentence sequence can be completely recovered from the adjacency matrix of order $m - 1$ provided that there are no duplicate terms. This means that the matrix completely captures the ordering semantics of the sentence. The recovery is straightforward, outlined below:

- (1) Sum all rows of the $m - 1$ -th order adjacency matrix to obtain a $1 \times m$ row vector.
- (2) Count the non-zero entries for each column of the row vector.
- (3) Rank the each word according to the count obtained in the previous step.
- (4) Order the words in ascending order according to their rank in the previous step, this is the original ordering of the words.

CHAPTER 7. SENTIMENT IDENTIFICATION

For example, for the adjacency matrix of Figure 7.2, the count of non-zero entries for “phone” is 4, “this” is 3, “like” is 2, and “I” is 1. By ranking the words in ascending order, the recovered sequence is “I like this phone”.

Notice that this recovery procedure applies only to sentences without duplicate terms. For these type of sentences, the ordering of the duplicate word cannot be uniquely determined.

Although the matrix representation is $N \times N$, in practice we have to scan convert it to a $N^2 \times 1$ vector by converting each row into a column.

7.2.2 Experiment Setup

7.2.2.1 Data Sets

Two sentence classification data sets, basketball (English) and cellphone (Chinese), were used to evaluate the suitability of our matrix representation across language boundaries.

The basketball dataset contains 2340 English sentence segments of 3 classes: “postype” indicating current win, “histype” indicating historical win, and “negtype” for irrelevant sentence, distributed as shown in Table 7.1. Porter stemming was applied and a simple flat SVM classifier was used on varying amounts of training data. The goal is to observe how the performance changes with increasing availability of training data.

Table 7.1: Basketball Match Data Set Distribution.

	postype	histype	negtype
count	189	117	2034

The cellphone data set includes 2000 on-line reviews randomly selected from our crawled Chinese cellphone on-line review corpus. Each review has been pre-processed into sentence segments, with distribution as shown in Table 7.2. These sentence segments are grouped into three categories, “positive” expressing positive opinion, “negative” expressing negative opinion, and “non-opinion” which includes neutral opinion (e.g., “OK”) and non-opinion. Notice that due to the small number of neutral training opinion segments, no decent training can be achieved, thus neutral opinion segments are treated as non-opinion segments.

Even at the sentence segment resolution, some segments may still contain more than one opinion, as listed in Table 7.2. These types of segments can only hurt a SVM classifier, and are therefore discarded for the machine learning approach.

Table 7.2: Cellphone Review Data Set Distribution.

	positive	negative	non-opinion	neutral	multiple	total
count	1141	1051	4479	32	60	6763

Clearly, the two datasets are highly skewed and unbalanced in terms of class distribution, which is nevertheless quite representative of real life classification problems.

7.2.2.2 Sentence Representation Evaluation Schemes

We evaluate a number of sentence representation approaches, listed below as 1) Single (VSM), 2) NGram2 (Bigram), and 3) Matrix.

Trigrams are not evaluated as it will only increase the data sparseness for the already sparse sentence segments. Likewise, we do not restrict the order of the matrix, i.e., we use the full $m - 1$ order adjacency matrix.

7.2.2.3 Atomic Unit Evaluation Schemes

There are three common ways to extract the atomic unit of a sentence, namely *Unigram* (single character or letter), *Word*, and *POS tagged word* (Part-Of-Speech). As it stands, sentence segment vector is already extremely sparse, representing a segment using *POS tagged word* will make it even worse (each word will have several different representations, vastly increasing the vocabulary size). Thus, we will not consider the *POS tagged word* representation.

In English, *Unigram* refers to a single letter/character, which is quite useless, so we will only evaluate the English basketball content at the *Word* representation level. On the other hand, single Chinese characters or Unigram contains enough information, and will be investigated along with *Word* representation for the cellphone review dataset. To make the definition unambiguous, in Figure 7.3, a simple English sentence, "*I like it*", and its

CHAPTER 7. SENTIMENT IDENTIFICATION

Chinese equivalent are used to illustrate the three kinds of atomic units for each language.

Original: I like it. 我喜欢它。

POS Tagged: I/p like/v it/p ./w 我/p 喜欢/v 它/p 。/w

p=pronoun, v=verb, w=punctuation

2 nd Element	Unigram	Word	Word-With-POS
English	I	like	like/v
Chinese	喜	喜欢	喜欢/v

Figure 7.3: Atomic Unit Denoting a Term.

7.2.2.4 Term Weighting Evaluation Schemes

Binary (Term Presence), *TF* (Term Frequency), and *TFIDF* (Term Frequency Inverse Document Frequency) [SKC03] are three common weighting schemes that are evaluated in our experiments. For most problems, the TFIDF weighting seems to work very well. In fact, there is no big difference between the three when it comes to the sentence segment because the probability of a word repeating itself is very low.

7.2.2.5 Evaluation Metrics

We look at a number of different metrics for assessing the quality of a classifier. The *Accuracy* metric evaluates overall performances as:

$$Accuracy = \frac{\# \text{correctly classified samples}}{\# \text{samples}} \quad (\text{Eq. 7.3})$$

To analyze the detailed performance of specific categories, we computed their *Precision*, *Recall*, and *F-Measure*, defined as follows.

$$Precision = \frac{\# \text{classified true positive}}{\# \text{classified positive}} \quad (\text{Eq. 7.4})$$

$$Recall = \frac{\# \text{classified true positive}}{\# \text{true positive}} \quad (\text{Eq. 7.5})$$

$$F-Measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (\text{Eq. 7.6})$$

7.2.3 Experiment Results

For each of the two atomic unit representation schemes, three choices of sentence representations are possible, *Matrix*, *NGram2*, and *Single*, which can be weighted using three schemes, *Binary*, *TF*, *TFIDF*. So in theory, there are a $2 \times 3 \times 3 = 18$ possible permutations. Moreover, since our goal is to find the best sentence representation, only the best results from each of the 3 sentence representations will be shown.

For each representation scheme, we plot the metric value versus an increasing percentage of cross-validated training data. For example, at 10% of cross-validated training data, 90% of the whole dataset is used for testing averaged over 10 different partitions of the 90% data. For other percentages, we maximize the amount of non-overlapping data while sampling. For example, at 60% training, we sample two different chunks of 60% training data with 20% overlap.

7.2.3.1 Overall Accuracy

Figure 7.4 shows the cross-validated accuracy metric for various segment representations. In the three sub-diagrams, it is easy to see that *Matrix* outperforms the other two sentence representation methods. Comparing the *Matrix* results of Figure 7.4.b and 7.4.c, it appears that *Unigram* is about 2% better than *Word*, and *Matrix* is about 4% better than *Single*. As shown in Figure 7.4.a, the results for *Matrix* and *Single* experience substantial fluctuations, which highlights the inconsistency of the basketball dataset (the annotations can be quite subjective at times).

The *Accuracy* metric evaluates overall sentence representation classification performance, regardless of the class distribution, and thus could be biased by a large negative class. In practice, minor categories in the data set are usually more interesting, this includes the “postype” and “histype” sentences for the basketball dataset, and “positive” and “negative” opinion segments for the cellphone review dataset. Hence, besides overall performance, we need to further evaluate the performance of the representation schemes on each “major” and “minor” category.

CHAPTER 7. SENTIMENT IDENTIFICATION

Figure 7.4: Cross-validated Accuracy.

7.2.3.2 F-Measure of Major Categories

Figure 7.5 plots the F-measure values for the “major” categories, i.e., “negtype” for basketball and “nonopinion” for cellphone. For the basketball data set, *Matrix* performs better than *NGram2*, especially when training data fraction exceeds 60%. It is just slightly (0.5%) worse than *Single* otherwise. Further, as training data increases to over 70%, *Matrix* performs better and more consistent than *Single*, as shown in Figure 7.5.a.

For the cellphone data set, *Matrix* consistently give the best result that is on average 1.5% better than *Single*. Therefore, it can be said that, overall, *Matrix* beats *Single* and *NGram2* in the major category’s F-Measure for both Chinese and English datasets.

As with the Accuracy plots, *Matrix* and *Single* exhibits substantial instability with respect to increasing training data. We also found that *Unigram* is about 1% better than *Word* for the cellphone dataset, which could be attributed to the low quality of word segmentation.

Figure 7.5: Cross-validated F-Measure of Major Categories.

7.2.3.3 F-Measure of Minor Categories

As demonstrated in Figure 7.6, the F-Measures for minor categories are not stable across the two datasets. In fact, it is hard to tell which one is the overall best, *Matrix* or *Single*. *Matrix* beats *NGram2* in five out of the six sub-figures (except Figure 7.6.a). With the “Unigram” atomic unit scheme, shown in Figure 7.6.c and Figure 7.6.f, *Matrix* is always the best and leads *Single* by 2%.

By comparing the peak F-Measure values of the two datasets using the same representation scheme, such as *Single*’s 57% for “postype” class of basketball (Figure 7.6.a) and *Single*’s 80% for “positive” class of cellphone (Figure 7.6.c), *Single*’s 50% for “histype” class of basketball (Figure 7.6.d) and *Single*’s 72.5% for “negative” class of cellphone (Figure 7.6.e), we observe that the maximum F-Measure values of the basketball dataset is more than 20% less than the ones for the cellphone dataset even using the same representation method. This observation, in some degree, confirms the low quality of our basketball data set.

Another observation is that *Matrix* remains the best representation for *Unigram* (Figure 7.6.c and Figure 7.6.f), outperforming all other representations by 2%.

Thus, we conclude that for Chinese sentence classification, the *Matrix* representation method with *Unigram* atomic unit is the best approach.

Since F-Measure is derived from Precision and Recall, to identify what is causing the inconstant results in the minor categories results, we look at the Precision and Recall curves. To summarize, by sacrificing some Precision, *Matrix* achieves a much larger increase in Recall and thus performs the best.

7.2.3.4 Precision of Minor Categories

The precision values for the minor categories are shown in Figure 7.7. Note that *Matrix* always performs best in minor categories, regardless of the distribution of the minor categories. Also, *NGram2* almost beat *Single* (except Figure 7.7.d). This implies that involving contextual and structural information will benefit sentence classification.

CHAPTER 7. SENTIMENT IDENTIFICATION

Figure 7.6: Cross-validated F-Measure of Minor Categories.

Quantitatively, the precision of *Matrix* is 12% better than *Single*. And for the basketball data set, more than 30% improvement in precision is achieved by using *Matrix* instead of *Single*.

The best sentence representation method for minor category precision, *Matrix*, will perform about 2% better if we switch from *Word* to *Unigram* (comparing Figure 7.7.b, Figure 7.7.c and Figure 7.7.e , Figure 7.7.f respectively).

Performance of all three representations on the basketball dataset is unstable, and is especially bad for *Matrix*. This confirms our previous observation about the basketball data set.

7.2.3.5 Recall of Minor Categories

From Figure 7.8, we can understand why *Matrix* performs badly on the F-Measure of minor categories; *Matrix*'s recall is almost always the worst among the three, off by 25% and 10% compared to *Single* for the basketball and cellphone dataset, respectively. This degradation overshadows *Matrix*'s otherwise excellent precision performances.

CHAPTER 7. SENTIMENT IDENTIFICATION

Figure 7.7: Cross-validated Precision of Minor Categories.

Ideally, a classifier needs to perform well in both Precision and Recall, but for real-life classification tasks where minor categories are more important, precision can take more importance. This is especially true for review mining, where users want the classified reviews they are seeing to be accurate, but not necessarily comprehensive and complete. Hence, for minor category classification, *Matrix* remains the best choice despite its low F-Measure scores.

7.2.4 Discussion

Several observations can be drawn from the experimental results across two different languages. Overall speaking, *Matrix* appears to be the best sentence representation with respect to classification, yielding on average 2% better accuracy than *Single*, as demonstrated in Figure 7.4. *Matrix* also beats *NGram2* and *Single* for the specific category performance. For the major categories, *Matrix*'s performance is stable and ranks amongst the top performers, as illustrated in Figure 7.5. For minor categories, which is the focus of our research, *Matrix* achieves the highest precision, as shown in Figure 7.7. Even on a

CHAPTER 7. SENTIMENT IDENTIFICATION

Figure 7.8: Cross-validated Recall of Minor Categories.

poorly annotated dataset such as the basketball dataset, *Matrix* was able to outperform the other two representations.

Notice that the superiority of *Matrix* holds true for both English and Chinese datasets, and also over different types of writing style. The basketball dataset is extracted from formal reports, whereas the cellphone dataset is comprised of user reviews that are more colloquial and short. We are quite confident that the *Matrix* sentence segment representation will work well for other domains and languages.

As all of our experiments were conducted at the sentence resolution, which results in a very sparse representation. The *Matrix* representation was able to partially overcome this sparseness problem by completely capturing the sequence information of all terms in the sentence, where are otherwise discarded by the classical *Single* representation. For the basketball test data set (1169 test segments, 50% of all), the density of non-zero entries is 0.47% for VSM (*Single*), and 0.043% for *Matrix* representation. Similarly, for the cellphone test data set (1000 test reviews, 50% of all), the densities are 0.19% for *Single* and 0.03% for *Matrix*, respectively. Despite having 10 times sparser representation, the

Matrix approach was able to outperform the vector space representation. This goes a long way to show the superiority of our *Matrix* method. With additional pruning, our *Matrix* approach could perform even better.

Traditionally, word segmentation plays an important part in the preprocessing of Chinese languages. Moreover, current Chinese word segmentation approaches cannot always get the job done right, resulting in errors at the later stages of processing. However, our experiment results indicate that the Chinese word segmentation procedure could be skipped if we use a *Matrix* representation. As shown in Figures 7.4, 7.5, and 7.6, *Matrix* combined with *Unigram* actually outperformed the corresponding *Matrix* with *Word* representation. Clearly, the latter’s segmentation method (based on ICTCLAS) has a lot of room for improvement.

7.3 Matrix SVM Based Sentiment Classification

After validating the effectiveness of the *Matrix* representation with respect to classifying sentence segments, we created a set of training sentence segments encoded with the *Matrix* representation. The data is used to train a SVM to classify segments into non-opinions, negative, and positive sentiment. The detailed flow of *Matrix* sentiment classifying procedure is shown in Figure 7.9.

The *Matrix* approach handles “StrongOpinionWords” identically to the *Concept* approach, as shown in the top part of Figures 7.1 and 7.9. In fact, both *Matrix* and *Concept* assume only one opinion per sentence segment, although in practice there could be more than one orientation per segment as shown in Table 7.2. Further, some multi-opinions do not even share the same orientation (26 out of 60 in Table 7.2).

To deal with sentence segments containing multiple opinions, we implement a brute-force FO extraction procedure which first checks for “StrongOpinionWords” in both the *Matrix* and *Concept* approaches. As this process does not rely on the segment’s sentiment orientation, it could be applied to both methods.

CHAPTER 7. SENTIMENT IDENTIFICATION

Figure 7.9: Activity diagram for identifying feature orientation pairs from a sentence segment using a *Matrix* SVM classifier. This graph provides more details about the “Extract Feature Orientation” process in Figure 3.2.

For the *Matrix* approach, we only consider two kinds of sentiment orientation, “Positive” and “Negative”. “Neutral” is not considered due to the lack of training data. Nevertheless, if a sentence segment contains known neutral sentiment, it would be extracted by our brute-force FO pre-extraction process.

7.4 Matrix versus Concept Experiments and Results

Our lexicon based sentiment identification approach [SC06] has previously beaten another well known lexicon approach [HL04]. We would thus like to pit our new *Matrix* based SVM approach against our previous lexicon based method.

To provide a base line, we also implemented Hu’s method [HL04] and refer it to as the *Strict* approach.

7.4.1 Experiment Setup

We evaluated the 4 methods on the same 2000 cellphone reviews previously used in Section 7.2.2.1. Unlike the review selection strategy of Shi et al. [SC06], these 2000 reviews were randomly selected from our cellphone review corpus spanning multiple models (regardless their popularities). Randomly collecting reviews from popular cellphone review sites ensure that the dataset is representative of the domain, and that the tested method is robust enough to deal with real world data.

The 2000 reviews are randomly partitioned into a training and test set with 1000 reviews each. Training is required only for the *Matrix* approach. Results are evaluated only on the Test set for all approaches. Table 7.3 shows the distribution of the processed sentence segments.

Table 7.3: Divided Cellphone Review Data Sets Distribution.

	Positive	Negative	Non-opinion	Neutral	Multiple	Total
Training Set	550	515	2213	23	29	3330
Testing Set	591	536	2266	9	31	3433

Our results in Section 7.2.3 indicate the best setup to be a combination of *Matrix*, Unigram, and TFIDF. Nevertheless, we will also show the results for TF and Binary.

7.4.2 Evaluation

Let the true sentiment of the n -th review posting be expressed mathematically as a set

$$R_n = \{F_{n1}O_{n1}, \dots, F_{n|R_n|}O_{n|R_n|}\} \quad (\text{Eq. 7.7})$$

with $|R_n|$ feature-orientation (FO) pairs, and the corresponding suggested (machine generated) sentiment be represented by the set

$$R'_n = \{F'_{n1}O'_{n1}, \dots, F'_{n|R'_n|}O'_{n|R'_n|}\} \quad (\text{Eq. 7.8})$$

where $|R'_n|$ can be less than, equal to, or greater than $|R_n|$. Note that by definition, every FO pair in the set is unique. In theory, more than one orientation is allowed for the same feature, but we specifically forbid that in our system. Thus we can assume that all the features in each review are unique, i.e. $F_{ni} \neq F_{nj} \forall i \neq j$ and $F'_{ni} \neq F'_{nj} \forall i \neq j$. Suppose there are N reviews, $R_1, \dots, R_N$, for a particular product. For each review R_n and the whole set of N reviews, we define the following:

- **Total FO.** The total number of true FO pairs is given by

$$\#FO = \sum_{n=1, \dots, N} |R_n| \quad (\text{Eq. 7.9})$$

- **Total F.** The total number of true features, which is the same as $\#FO$, since we assume that all features in a review must be unique:

$$\#F = \sum_{n=1, \dots, N} |R_n| = \#FO \quad (\text{Eq. 7.10})$$

- **Suggested FO.** The total number of suggested FO pairs, regardless of matching or not:

$$\#FO' = \sum_{n=1, \dots, N} |R'_n| \quad (\text{Eq. 7.11})$$

- **Suggested F.** The total number of suggested features, regardless of matching or not:

$$\#F' = \sum_{n=1, \dots, N} |R'_n| = \#FO' \quad (\text{Eq. 7.12})$$

Again, this is the same as $\#FO'$ due to our assumption of unique features in a FO pair for both true and suggested reviews.

CHAPTER 7. SENTIMENT IDENTIFICATION

- **Matched F.** This is simply the intersection of features between the true and machine-generated review set:

$$\#MatchedF_n = |\{F_{n1}, \dots, F_{n|R_n|}\} \cap \{F'_{n1}, \dots, F'_{n|R'_n|}\}| \quad (\text{Eq. 7.13})$$

$$\#MatchedF = \sum_n \#MatchedF_n \quad (\text{Eq. 7.14})$$

- **Added F.** This is the number of incorrectly identified features which can be due to wrong segmentation and/or POS tagging. In other words, the system found this number of features not present in the true review.

$$\#AddedF_n = |R'_n| - \#F_n \quad (\text{Eq. 7.15})$$

$$\#AddedF = \sum_n \#AddedF_n \quad (\text{Eq. 7.16})$$

- **Missed F.** This is the number of missed features:

$$\#MissedF_n = |R_n| - \#MatchedF_n \quad (\text{Eq. 7.17})$$

$$\#MissedF = \sum_n \#MissedF_n \quad (\text{Eq. 7.18})$$

- **Matched FO.** This is the number of correctly identified FO pairs:

$$\#MatchedFO_n = |R_n \cap R'_n| \quad (\text{Eq. 7.19})$$

$$\#MatchedFO = \sum_n \#MatchedFO_n \quad (\text{Eq. 7.20})$$

We can now proceed to evaluate our sentiment identification results using the following metrics:

- **F Recall.** This is defined as the number of features recovered with respect to the true review set:

$$F_{Recall} = \frac{\#MatchedF}{\#F} \quad (\text{Eq. 7.21})$$

- **F Precision.** This is defined as the number of suggested features that is actually correct:

$$F_{Precision} = \frac{\#MatchedF}{\#F'} \quad (\text{Eq. 7.22})$$

- **F F-Measure.** This is the harmonic mean for features.

$$F_{F-Measure} = \frac{2 \times F_{Precision} \times F_{Recall}}{F_{Precision} + F_{Recall}} \quad (\text{Eq. 7.23})$$

- **F Superfluous.** This is the percentage of additional errors/features introduced with respect to all suggested features:

$$F_{Superfluous} = \frac{\#AddedF}{\#F'} \quad (\text{Eq. 7.24})$$

- **FO Recall.** This is defined as the number of FO pairs recovered with respect to the true review set:

$$FO_{Recall} = \frac{\#MatchedFO}{\#FO'} \quad (\text{Eq. 7.25})$$

- **FO Precision.** This is defined as the number of FO pairs suggested that is actually correct:

$$FO_{Precision} = \frac{\#MatchedFO}{\#FO'} \quad (\text{Eq. 7.26})$$

- **FO F-Measure.** This is the harmonic mean for FO pairs.

$$FO_{F-Measure} = \frac{2 \times FO_{Precision} \times FO_{Recall}}{FO_{Precision} + FO_{Recall}} \quad (\text{Eq. 7.27})$$

- **FO Relative.** Since the FO precision is upper-bounded by the number of correctly identified features, we compute the relative FO precision as follows:

$$FO_{Relative} = \frac{\#MatchedFO}{\#MatchedF} \quad (\text{Eq. 7.28})$$

7.4.3 Experiment Results

The experimental results of the five approaches are listed in Table 7.4, where M means Matrix and U stands for Unigram.

7.4.4 Discussion

7.4.4.1 Concept versus Strict

From Table 7.4, we can draw the following observations [SC06]:

- (1) **More than 220% more features (F) found.** Just by looking at the number of matched features, *Concept* approach is way superior to the *Strict* approach. It was able to find 698 features versus 315. This is due to the “O first F second” strategy we adopt, which finds the orientation first and then assigns a feature (which may not be present in the review text) where applicable.
- (2) **Close to 230% more sentiments (FO) found.** With a much better identified feature set, *Concept* easily beat *Strict* in terms of FO_{Recall} .
- (3) **Increase in F-Measure for *Concept*.** Overall, there was about 20% improvement of *Concept* in $F_{F-Measure}$ and $FO_{F-Measure}$, compared with *Strict*. This is because of the large recall and comparable precision of *Concept*.
- (4) **About 12% more superfluous F.** This is due to the strategy, “O first F second”.

Table 7.4: Sentiment Identification Results for 1000 Test Reviews.

	M.U.Binary	M.U.TF	M.U.TFIDF	Concept	Strict
# Segment	3433	3433	3433	3433	3433
# $FO = \#F$	1200	1200	1200	1200	1200
# $SuggestedF'/FO'$	1221	1218	1140	1258	468
# $MatchedF$	747	747	715	698	315
# $MatchedFO$	708	707	679	618	262
# $MissedF$	453	453	485	502	885
# $AddedF$	474	471	425	560	153
F_{Recall}	62.25%	62.25%	59.58%	58.17%	26.25%
$F_{Precision}$	61.18%	61.33%	62.71%	55.49%	67.31%
$F_{F-Measure}$	61.71%	61.79%	61.11%	56.79%	37.77%
$F_{Superfluous}$	38.82%	38.67%	37.28%	44.52%	32.69%
FO_{Recall}	59.00%	58.92%	56.58%	51.50%	21.83%
$FO_{Precision}$	57.99%	58.05%	59.56%	49.13%	55.98%
$FO_{F-Measure}$	58.49%	58.48%	58.03%	50.29%	31.42%
$FO_{Relative}$	94.78%	94.65%	94.97%	88.54%	83.18%

Although it will generate about 12% additional irrelevant FO pairs, considering the 2 to 3-fold improvement in extracting actual features and FO pairs, *Concept* is undoubtedly much better than *Strict*.

7.4.4.2 Matrix Approaches

The following observations can be made by comparing the performances of the three *Matrix* approaches:

- (1) **Binary and TF have almost identical performances.** This is expected, since at the sentence segment resolution, a TF vector is equivalent to a binary vector for the vast majority of sentence segments. As previously shown, the average length of a segment is 5 characters, the probability of a character repeating itself in 5 characters is miniscule. This also means that the matrix representation can recover the vast majority of sentence segments (with no repeating characters).
- (2) **Binary/TF gives the best overall performance.** In this case, Binary/TF enjoys a slight advantage of 0.5% over TFIDF in terms of overall performance ($FO_{F-Measure}$).
- (3) **TFIDF gets highest Precision with least Superfluous.** For both Features and Feature-Orientation pairs, TFIDF is the undisputed winner in terms of Precision. TFIDF also contributes to fewer superfluous errors.

7.4.4.3 Matrix versus Concept

Comparing the best *Matrix* results with *Concept*, we found that the former wins hands-down in every metric, summarized as follows:

- (1) ***MatrixTF* achieved 5% improvement in Feature extraction.** In terms of $F_{F-Measure}$, *MatrixTF* showed a noticeable improvement over *Concept*.
- (2) ***MatrixTFIDF* reduced superfluous features by 7%.** As reflected in the lower $F_{superfluous}$ values.

- (3) *MatrixBinary* achieved 8% improvement in Feature-Orientation pair extraction.

Based on the promising results obtained compared to the *Concept* approach, we conclude that the *Matrix – Unigram – Binary* approach is indeed a reliable way to extract FO pairs for our Review Mining System.

Chapter 8

Summarization and Presentation of On-line Reviews

After the FO (Feature/Opinion) pairs of review segments have been extracted, the next step is to present the results in an intuitive way to the end user.

8.1 Aggregating Review Segments

To aggregate from different web sites review segments about the same product, we need to traverse through the various downloaded URL-named directories containing the processed reviews and generate a single XML summary file for each product brand/model. The XML summary file should be based on the product concept model, as it gives two benefits. First, the XML tree structure facilitates periodical updating and adding of new reviews for each brand/model. Second, the XML summary file can be easily transformed into any graphical/text representation using standard XML rendering templates and tools. Figure 8.1 shows the process of generating a XML summary file for a given product.

8.1.1 Minimizing Fraudulent Reviews and Outliers

One important consideration of review mining is to ensure that it is minimally affected by erroneous or spam posts from users trying to affect the rating statistics. In order to achieve this, we have to be able to uniquely identify a reviewer within a site. A simple way to represent the reviewer is to use the user ID followed by the review site, e.g., userID@reviewsite. The user ID for sites requiring registration before posting is easy to

CHAPTER 8. SUMMARIZATION AND PRESENTATION OF ON-LINE REVIEWS

Figure 8.1: Level 2 DFD for Generating a Per Product XML Summary File.

extract, but for the few sites that allow anonymous posting without IP addresses, all we could extract is a default/anonymous user ID. In the latter situation, we artificially append a unique numeric ID to the extracted default user ID, thereby treating each as a different user, e.g., anonymous123@reviewsite.

Once we have the user ID extracted from each review, we apply the simple rule of thumb to only count each *same* product feature sentiment from the *same* user ID across different postings *once*. In other words, a reviewer sentiment towards a product feature is only tallied once, regardless of the number of postings. This is achieved as shown in process 4.1 of Figure 8.1 by concatenating the model, Feature, Orientation, userID, and site URL as a hashtable key with only a single sentence segment as the value. This guarantees that the same user can only express one opinion on one product feature in the same site at anytime. Further, later postings on the same FO pair will automatically overwrite earlier values.

This simple rule takes care of the majority of fraudulent or erroneous reviews, including the case where one promoter or competitor posts tens of dozens of positive or negative reviews, respectively. It also takes into account cases where people accidentally post the same review multiple times (due to network delay or human error).

8.1.2 XML Summary File

Each XML Summary file contains all the necessary review information pertaining to one product, including product features, raw reviews, processed sentiments, and summary sentiment statistics. The file has a similar general structure as the concept tree, where each feature (leaf and non-leaf) node contains several child nodes, as described below:

- “Name”. Contains feature itself and its similar and concept related nouns.
- “Positive”, “Neutral”, “Negative”. These three sentiment nodes each have two child nodes:
 - (1) “Count”. An integer tracking number of reviews containing the parent sentiment.
 - (2) “Sentence”. This is where the list of relevant opinion sentence segments are stored with the userID and URL. Note that sentences are only stored at the feature leaf nodes and not at any nodes higher up in the hierarchy. For a segment containing more than one feature, the segment will be duplicated at the two correspondingly feature nodes.

8.1.3 Sample Update to the XML Summary Tree

To better illustrate the roles of each nodes in the XML Summary Tree, we provide the following example.

- (1) Suppose the opinion sentence segment “good and cheap phone” is stored in a review file for the Nokia 3230 phone named

`3533.com/Nokia 3230/@2005-09-15 13:11:23@218.64.63.*@samy.predict"`

- (2) After sentiment identification, the tagged sentence segment becomes “good and cheap //Price/++|cheap;General/++|good”, generating two hash table (key, value) pairs as follows:

```
("@Nokia3230@3533.com@samy@Price/++", "good and cheap")
```

```
("@Nokia3230@3533.com@samy@General/++", "good and cheap")
```

- (3) Create the corresponding XML Summary File “Nokia 3230.xml”, if it does not exist, and load the DOM tree into memory.
- (4) Search for the two feature nodes “General” and “Price”, and update their child nodes as follows:

(I) Increment the Price/Positive/Count node by one, and append the first hash value to the Price/Sentence node. Increment the Count node for all parent nodes of Price, all the way to the root node.

(II) Increment the General/Positive/Count node by one, and append the second hash value to the General/Sentence node. Increment the Count node for all parent nodes of General, all the way to the root node.

Initially, the value of each “Count” and “Sentence” node is zero and null, respectively. The detailed update procedure is shown in Figure 8.1.

Note that each XML summary file is stored using a similar directory structure as the raw reviews. For example, the full pathname for the “Nokia 3230” XML summary file is XMLSummaryFile/Cellphone/Nokia 3230.xml.

8.2 Interactive Web Interface to Present Summarization

To test our review system, we have created an Apache Tomcat based web site to allow users to quickly find a phone model and retrieve its corresponding reviews. To make the environment more friendly, corresponding image and price of cellphones are also extracted

from web sites if they are available and presented to end user with the review summary together. Users can either search for the make and model (Figure 8.2), or select it from a dynamic pull down menu (Figure 8.3).

Figure 8.2: Demonstration of Inputting Searching Terms for Brand and Model.

Once a particular model is selected, its concept tree model, called the Opinion Tree, will be displayed. At each feature node, a horizontal bar graph with varying proportions of green, gray, and red depicts respectively, the relative percentages of positive (green), neutral (gray), and negative (red) sentiments aggregated for that feature. Black denotes lack of sentiments for the corresponding feature. Note that each parent node quantitatively summarizes and aggregates all the feature sentiments of its children nodes. Figure 8.4 shows the opinion tree for the selected cellphone model “Nokia 3230”.

Users can interact with this tree in a number of ways:

- (1) By clicking on a parent node label, the node will expand or collapse to show/hide children features (Figure 8.5).
- (2) By hovering over a color bar of a node, the number and percentage of that orientation will be displayed (Figure 8.6).
- (3) By right clicking over a color bar of a leaf node, sample reviews corresponding to the selected sentiment will be shown (Figures 8.7-8.8).

Besides the existing web interfaces that allows a user to interact with a product opinion tree, we have also developed another prototype that allows several products to be compared, as shown in Figure 8.9.

The interactive portion of the page was coded using JavaScript and CSS while the back-end was based on Java, JavaBeans, and JSP technologies running on the Apache Tomcat application server.

CHAPTER 8. SUMMARIZATION AND PRESENTATION OF ON-LINE REVIEWS

Mobile Phone ▼ Now Select Brand ▼ Select Type ▼ search

- 杰特Ztel
- 朗讯
- 华硕
- 京瓷
- ZTC中天
- 南方高科
- 首信
- 乐华
- 萨基姆
- 三洋
- 东信
- 东芝
- 浪潮
- 酷派
- 托普
- 诺基亚**
- 普天
- 步步高
- 阿尔卡特
- 爱立信
- 中兴
- 波导
- 摩托罗拉
- 海信
- 大显
- 大唐
- 明基
- 佳通
- 厦华
- 金立

8.3.a: Pull-down Menu for Brand Selection.

Mobile Phone ▼ 诺基亚 ▼ Now Select Type ▼ search

- 诺基亚 3128
- 诺基亚 3152
- 诺基亚 3155
- 诺基亚 3200
- 诺基亚 3210
- 诺基亚 3220
- 诺基亚 3230**
- 诺基亚 3300
- 诺基亚 3310
- 诺基亚 3315
- 诺基亚 3330
- 诺基亚 3350
- 诺基亚 3410
- 诺基亚 3510
- 诺基亚 3530
- 诺基亚 3560
- 诺基亚 3610
- 诺基亚 3650
- 诺基亚 3660
- 诺基亚 3810
- 诺基亚 5100
- 诺基亚 5110
- 诺基亚 5110i
- 诺基亚 5140
- 诺基亚 5210
- 诺基亚 5510
- 诺基亚 6020
- 诺基亚 6021
- 诺基亚 6030
- 诺基亚 6060

8.3.b: Pull-down Menu for Model Selection.

Figure 8.3: Pull-down Menus for Brand/Model Selection.

CHAPTER 8. SUMMARIZATION AND PRESENTATION OF ON-LINE REVIEWS

Figure 8.4: The Nokia 3230 Opinion Tree.

CHAPTER 8. SUMMARIZATION AND PRESENTATION OF ON-LINE REVIEWS

Figure 8.5: The Nokia 3230 Opinion Tree with Some Nodes Collapsed.

Figure 8.6: Hovering over the green bar under Appearance node shows that 136 or 77% of the users like the appearance of the Nokia 3230.

Figure 8.7: Sample Positive Reviews about Appearance of the Nokia 3230.

Figure 8.8: Sample Negative Reviews about (lack of a) Camera of the Nokia 3230.

CHAPTER 8. SUMMARIZATION AND PRESENTATION OF ON-LINE REVIEWS

Figure 8.9: Sample Comparison of Two Cellphones.

The current search page takes some noticeable time to load because the list of brands and models are created on-the-fly by scanning the directory of all XML summary files using Java Server Pages (JSP), a server side script. A future improvement is to pre-generate the list periodically.

Further, the result page also takes a considerable time to load (close to one minute) because the statistics are computed on-the-fly by parsing the XML Summary File and generating the dynamic HTML code in real time. Specifically, the percentages of the three orientations under every leaf and non-leaf feature node is computed on-the-fly. A simple improvement is to pre-compute and pre-render the page for every make/model.

Chapter 9

Conclusion and Future Work

9.1 Conclusion

In the introductory chapter, we stated the motivation of our project and proposed a Chinese on-line review mining and sentiment identification task. Detailed methods of review extraction and orientation classification were developed in the subsequent chapters. Problems encountered and their solutions were also described. In the following, we summarize our main contributions as follows:

- (1) A template-based on-line review extracting method was introduced. This method only consider HTML tags, ignoring the review content, and can be easily generated semi-automatically. The major skeletal portion of the Review Information Extraction Template (RIET) is generated by a program, with the remaining portions to be manually refined. A single HTML sample review page from a site is all that is needed to generate the initial RIET, to be refined with human testing and additional pattern matching logic. Experiments results show the RIET approach to be 100% accurate. Besides RIET, a novel hierarchical concept model was proposed for storing and summarizing the reviews of each product.
- (2) We proposed a useful *punctuation filtering* method to reduce punctuation noise in on-line reviews. This approach was inspired after observing a number of on-line reviews from different sites. Experimental results shows that our proposed method not only improves the subsequent Chinese word segmentation and POS tagging, but it also

CHAPTER 9. CONCLUSION AND FUTURE WORK

enables the system to be more robust by accepting more reviews (which would have been rejected without punctuation filtering).

- (3) A new concept model based approach of extracting sentiment orientation from reviews is proposed. With the relevant domain knowledge and basic language lexicon, the concept model for any kind of product can be easily created.
- (4) We proposed a novel way to represent text by its neighbors, called the feature context vector. We then propose using top-down bisection clustering to group and extract related nouns into a hierarchy. The automatically created hierarchy is not meant to replace the human created concept hierarchy, but to complement and update it from time to time with human guidance.
- (5) We proposed a new matrix adjacency matrix representation to capture the semantics and ordering information of a sentence segment. We evaluated this representation against classical “a bag of words” and bigram representation, and found it to consistently outperform the latter.
- (6) We then showed how machine learning can be used to train a classifier to learn the orientation from a set of corpus with known sentiments. Combining this with the adjacency matrix representation yielded the highest accuracy for our system.
- (7) A concept hierarchy based summarization and visualization interface was implemented. Once the concept model is built, the whole summarizing procedure does not require any further human input. A set of fraud and spam reduction rules ensures that the review summaries are as objective as possible. As the review summaries for each product are stored in XML files following the concept model’s tree structure, they can be updated very conveniently. A web interface was created which allows users to search or select for a model/make of cellphone. An interactive review opinion tree is then displayed, allowing the user to view different aspects of the product feature reviews. To better meet users’ demand, associated image and price are also displayed with the queried review summary if they are available.

Experiments were conducted to rigorously test and verify the methods and solutions we proposed.

Notice that the major blocks of the system are independent from one another. Thus, the basic ideas, observations, and methods we proposed can be applied to mine reviews in languages other than Chinese, as long as the necessary segmentation, POS tagging tools can be replaced with those of the target language. As a matter of fact, we have demonstrated that this is indeed feasible by performing some of the comparative evaluations on two English datasets, namely the basketball and Mp3player datasets.

9.2 Future Work

In this section, limitations and drawbacks of our system will be presented and possible future research directions are proposed.

9.2.1 Automatic Review Extraction

As discussed in chapter 4, although the extraction method we proposed can achieve a rather high accuracy, there are still several drawbacks, the main one being its requirement for human refinement.

The first improvement is to utilize browser generated visual information to extract reviews instead of solely relying on the HTML tree. In addition to the containment relationship between HTML rectangle elements, their absolute positions should also be taken into account. This is reasonable because most HTML pages are designed for human viewing, more so for pages containing reviews, whose review blocks are designed to cover significant screen real-estate or the most central area, so as to allow easy navigation and reading. Reviews on the same page are often listed contiguously in the vertical or horizontal dimension, i.e. their bounding rectangles could be included by a bigger rectangle. Once this bigger rectangle is located, the review region is thus found.

In a page containing multiple reviews, the rectangles of the reviews are usually aligned, sharing the same length or width. This provides us a valuable clue to segment the parent

bounding rectangle. The rectangle's relative position could also aid in detecting the internal structure of reviews. Since review content is usually rendered onto pages using a template, the relative positions among essential elements of a reviews remain fixed from one page to another. Examples include reviews appearing at the same horizontal level, before and after, below and above some fixed HTML elements. Once the relative positions are confirmed, the review content can be extracted automatically.

To make the extraction task fully automatic, the problem of a site using different templates to generate dynamic review pages must be tackled. Thus, after successfully extracting reviews from several pages of a site, an extraction template should be automatically generated to extract reviews from the remaining pages. The template will not be changed or regenerated unless extraction fails at some point, upon when a new template can be automatically generated by the same automatic procedure, while keeping the remaining templates intact. In short, a new template will be generated if and only if a page cannot be processed by all existing templates. In this way, the extraction can be truly automatic.

9.2.2 Improving Chinese Preprocessing

Based on our analysis of the review errors, misspelled words and unknown words seem to be the major cause. In ICTCLAS, all known words are maintained in separate dictionaries. Therefore, despite that both common and unknown words will be divided into atoms and then chunked by HHMM, common words in the lexicon still have a higher chance to be segmented and tagged correctly simply because words in the lexicon are given higher priority. So as an improvement, we can manually add those frequent wrongly processed words into ICTCLAS lexicon. For misspelled words, as most of these errors are caused by ambiguity of words sharing similar pronunciation, we could build or use a phonetic spelling checker. Obviously, any serious attempt to improve these aspect of our review mining system will involve significant effort.

9.2.3 Improving Concept Hierarchy

In our clustering approach of building a concept hierarchy, the way of collecting candidate feature terms plays an important role. But as stated, the current strategy of collecting candidate noun terms is based on term *SegmentFrequency*. There is no guarantee that all terms collected are features. Those irrelevant terms will not only interfere with the final clusters, but also the structure of concept hierarchy.

We should also develop a method to automatically prune and refine an existing concept hierarchy. For example, new feature leaf nodes could be added to the right place in the tree. Currently, we simply create a brand new tree from the review data instead of appending to the existing tree.

9.2.4 Enhancement of Opinion Orientation Identification

Our system is currently still hovering around 60% in terms of F-Measure for both Feature and Feature-Opinion pair extraction. There is still a whole lot of room for improvement. New paradigms will have to be devised, and perhaps new ways of understanding the semantics of the sentence will have to be investigated.

Not considering sentence structure and word sequences have caused a lot of errors during our lexicon sentiment classification at both the sentence and sentence segment level. One possibility is to parse the sentence segment. ICTPROP (Institute of Computing Technology, **PRO**abilistic **P**arser) [ZLZ⁺03] is a tool that can statistically parse a sentence to derive its corresponding tree structure with embedded context information. Based on our analysis of sentiment identification errors, mis-handling of the modifying objects, features, and negation words, and affecting range are the two major causes of error. However, with ICTPROP's aid, the function of each sentence, such as subject and object can be identified. We can then segment a sentence into several blocks to get the effective modifier's range, and derive a clearer relationship among different words.

Although PCFG (Probabilistic Context Free Grammars) [MS99] have been proposed for a long time and works well in English, there aren't many Chinese PCFG tree banks

and training set available. To tell how well ICTPROP could aid in improving our opinion orientation identification is a bit early at this stage.

9.3 Concluding Remarks

We believe the review mining system will be a useful tool that every user will come to appreciate. As we continue to refine the underlying technologies, we hope that it will one day be good enough to serve the needs of all consumers on the Internet.

Appendix A

Review Information Extraction Template (RIET)

A.1 RIET Elements

Table A.1 lists the elements used in a RIET.

A.2 RIET Example

Figure A.1 shows a RIET example for extracting individual reviews from the `www.163.com` site.

CHAPTER A. REVIEW INFORMATION EXTRACTION TEMPLATE (RIET)

Table A.1: Specifications for Review Information Extraction Template (RIET).

Node Name & Sample Value	Description
<DesireFramePosition> 1	Specifies HTML frame number containing actual reviews. Default value of 1 refers to review page itself (if frameless), higher values possible for frame based review sites.
<AbsoluteURLPart> null	Contains absolute root URL of current review page, useful for frame based HTML review pages to compute the absolute URL from typically relative URLs specifying frames. Default value of null works for non-frame based review page.
<DataRecordStoragePath> ./datarecord	Specify path for storing extracted individual reviews. Default value stores reviews in ./datarecord.
<HtmlPageCharSet> GB2312	Specifies character encoding of review page. Default value of GB2312 works for most Chinese review sites.
<TagNameSimilarity> 0.6	Specifies MDR similarity threshold [LGZ03]. Default value is 0.6.
<ProductNameAtNode> #document/HTML/BODY/table/tr/td/a/#text	Specifies the path of nodes containing raw product brand/model. No defaults, must be specified manually.
<ProductNameRegex> TYPE:(.*)\$	Regex for extracting brand/model from the raw IE result of <ProductNameAtNode>. Default value of null is suitable if previous result already successfully extracted the product brand/model.
<DataRegionAtNode> #document/HTML/BODY	Specifies path of nodes containing reviews. No defaults, must be specified manually.
<DataRegionAppearOrdinal> 1	Specifies the index to the actual review data region out of possibly multiple <DataRegionAtNode> matches in the DOM tree. A value of 1 means the path is unique. No defaults, must be specified manually.
<DataRegionChildnodeStartEndOffset> 1:4	Specifies the number of header and trailer review nodes to discard, as they could contain header and irrelevant information. The sample value of 1:4 refers to discarding 1 header node and 4 trailer nodes. Default value is 0:0.
<SingleDataRecordNodeSize> 1	Specifies the number of child nodes in an individual review. Default value is 1.
<IndividualDataRecordRegex> (\d{4}-\d{1,2}-\d{1,2} \d{1,2}:\d{1,2}:\d{1,2}) Username:(.*) IP:(\d{1,3}\. \d{1,3}\. \d{1,3}\. \d{1,3}) Vote:+(.*?)Reply(.*?)\$	Specifies the Regex for extracting all fields from an individual data record from the output of <SingleDataRecordNodeSize>.
<IndividualDataRecordFilenameRegexGroups> 1:3:2	Specifies the indices (separated by ":") of matched regex fields from <IndividualDataRecordRegex> to use for naming the file containing the individual review. No defaults, must be specified manually.
<IndividualDataRecordContentRegexGroups> 4	Specifies the index of matched regex fields from <IndividualDataRecordRegex> to use for the actual review content. No defaults, must be specified manually.

CHAPTER A. REVIEW INFORMATION EXTRACTION TEMPLATE (RIET)

```

<Config>
  <DesireFramePosition>1</DesireFramePosition>
  <AbsoluteURLPart></AbsoluteURLPart>
  <DataRecordStoringPath>
    ./IndividualReview
  </DataRecordStoringPath>
  <HtmlPageCharSet>GB2312</HtmlPageCharSet>
  <TagNameSimilarity>0.6</TagNameSimilarity>
  <ProductNameAtNode>
    /#document/html/head/title/#text
  </ProductNameAtNode>
  <ProductNameRegex>
    网易论坛-手机产品评论-(.*)$
  </ProductNameRegex>
  <DataRegionAtNode>
    /#document/html/body/div/ul
  </DataRegionAtNode>
  <DataRegionAppearOrdinal>1</DataRegionAppearOrdinal>
  <DataRegionChildnodeStartEndOffset>
    0:0
  </DataRegionChildnodeStartEndOffset>
  <SingleDataRecordNodeSize>1</SingleDataRecordNodeSize>
  <IndividualDataRecordRegex>
    回复: (.*) 作者: (.*) 发表时间: (\d{4}-\d{1,2}-\d{1,2}
      \d{1,2}:\d{1,2}:\d{1,2})
    发表ip: (\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}) 引用
  </IndividualDataRecordRegex>
  <IndividualDataRecordFilenameRegexGroups>
    3:4:2
  </IndividualDataRecordFilenameRegexGroups>
  <IndividualDataRecordContentRegexGroups>
    1
  </IndividualDataRecordContentRegexGroups>
</Config>

```

Figure A.1: RIET for www.163.com.

Appendix B

Sample Mined Concept Hierarchies

B.1 Mined Cellphone Concept Hierarchy

Figure B.1 is the mined cellphone concept hierarchy where IS is 0.2.

B.2 Mined Hotel Concept Hierarchy

Figure B.2 is the mined hotel concept hierarchy where IS is 0.2.

B.3 Mined Mp3player Concept Hierarchy

Figure B.3 is the mined Mp3player concept hierarchy where IS is 0.25.

CHAPTER B. SAMPLE MINED CONCEPT HIERARCHIES

Figure B.1: Mined Cellphone Concept Hierarchy

CHAPTER B. SAMPLE MINED CONCEPT HIERARCHIES

Figure B.2: Mined Hotel Concept Hierarchy

CHAPTER B. SAMPLE MINED CONCEPT HIERARCHIES

Figure B.3: Mined Mp3player Concept Hierarchy

References

- [AGM03] Arvind Arasu and Hector Garcia-Molina. Extracting structured data from web pages. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, 2003.
- [ALJ00] James Allan, Victor Lavrenko, and Hubert Jin. First story detection in tdt is hard. In *9th ACM CIKM Conf.*, pages 374–381, 2000.
- [AP04] Luiz Augusto and Sangoi Pizzato. Using a trie-based structure for question analysis. In *Proceedings of the Australasian Language Technology Workshop*, pages 25–31, Macquarie University, Sydney, Australia, December 2004.
- [AS94] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499, 1994.
- [BPSMM03] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, and Eve Maler. Extensible Markup Language (XML) 1.0 (Third Edition). *W3C Recommendation 4th February 2004*, Francois Yergeau, June, 2003.
- [Car99] Sharon A. Caraballo. Automatic construction of a hypernym-labeled noun hierarchy from text. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, College Park, Maryland, 1999.
- [CC04] Shui-Lung Chuang and Lee-Feng Chien. A practical web-based approach to generating topic hierarchy for text segments. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, Washington, D.C., USA, November 2004.
- [CfBW05] Xin Chen and Yi fang Brook Wu. Web mining from competitors’ websites. In *Proceeding of the 11th ACM SIGKDD international conference on Knowledge discovery in data mining*, Chicago, Illinois, USA, August 2005.

REFERENCES

- [CFV98] J. Clment, P. Flajolet, and B. Valle. The analysis of hybrid trie structures. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 531–539, PA, USA, 1998.
- [CG04] Hiram Calvo and Alexander Gelbukh. Extracting semantic categories of nouns for syntactic disambiguation from human-oriented explanatory. In *Proceeding of the 5th International Conference of Computational Linguistics and Intelligent Text Processing*, volume 2945, Seoul, Korea, 2004.
- [CLU] CLUTO & Manual. <http://glaros.dtc.umn.edu/gkhome/views/cluto/>.
- [CM02a] James R. Curran and Marc Moens. Improvements in automatic thesaurus extraction. In *Proceedings of the ACL-02 workshop on Unsupervised lexical acquisition*, pages 59–66, Philadelphia, Pennsylvania, July 2002.
- [CM02b] James R. Curran and Marc Moens. Scaling context space. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, Philadelphia, Pennsylvania, July 2002.
- [CON] CONVMV. <http://j3e.de/linux/convmv/>.
- [CW03] Scott Cederberg and Dominic Widdows. Using lsa and noun coordination information to improve the precision and recall of automatic hyponymy extraction. In *Conference on Natural Language Learning (CoNLL)*, Edmonton, Canada, 2003.
- [CYWM03] Deng Cai, Shipeng Yu, Ji-Rong Wen, and Wei-Ying Ma. Extracting content structure for web pages based on visual representation. In *The Fifth Asia Pacific Web Conference*, 2003.
- [DH02] Melania Degeratu and Vasileios Hatzivassiloglou. Building automatically a business registration ontology. In *Proceedings of the 2002 annual national conference on Digital government research*, pages 1–7, Los Angeles, California, May 2002.
- [DIW05] Wisam Dakka, Panagiotis G. Ipeirotis, and Kenneth R. Wood. Automatic construction of multifaceted browsing interfaces. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, Bremen, Germany, 2005.
- [FS97] T. Fujimoto and M. Sugeno. Clustering verb, adjective, adjectival-verb concepts using proximity relation. In *Fuzzy Systems, Proceedings of the Sixth IEEE International Conference on*, Barcelona, Spain, 1997.

REFERENCES

- [GB2] GB2312. <http://en.wikipedia.org/wiki/GB2312>.
- [GPLK02] Eric Glover, David M. Pennock, Steve Lawrence, and Robert Krovetz. Inferring hierarchical descriptions. In *Proceedings of the 11th international conference on Information and knowledge management*, McLean, Virginia, USA, November 2002.
- [GYE02] Zhai Gaoshou, Zhang Yongkui, and Yang Erhong. Construction of semantic classification hierarchy of chinese nouns. In *Proceedings of 2002 IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering*, 2002.
- [HG04] Ben Hachey and Claire Grover. Sentence Classification Experiments for Legal Text Summarisation. In *Proceedings of the 17th Annual Conference on Legal Knowledge and Information Systems*, Berlin, Germany, 2004.
- [Hin90] Donald Hindle. Noun classification from predicate-argument structures. In *Proceedings of the 28th annual meeting on Association for Computational Linguistics*, Pittsburgh, Pennsylvania, 1990.
- [HL04] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, Seattle, Washington, USA, August 2004.
- [HM93] Vasileios Hatzivassiloglou and Kathleen R. McKeown. Towards the automatic identification of adjectival scales: clustering adjectives according to meaning. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, Columbus, Ohio, 1993.
- [HOW] HOWNET: Chinese Concept Dictionary. <http://www.keenage.com>.
- [HTM] HTML Parser- A java-based open source html parser. <http://htmlparser.sourceforge.net/>.
- [ICT] ICTCLAS linux version. <http://www.nlp.org.cn>.
- [Ino91] Naomi Inoue. Automatic noun classification by using japanese-english word pairs. In *Proceedings of the 29th annual meeting on Association for Computational Linguistics*, Berkeley, California, 1991.
- [JK06] Theresa Jickels and Grzegorz Kondrak. Unsupervised labeling of noun clusters. In *Proceedings of the 19th Canadian Conference on Artificial Intelligence*, pages 278–287, Quebec City, June 2006.

REFERENCES

- [LCR01] Dawn Lawrie, W. Bruce Croft, and Arnold Rosenberg. Finding topic words for hierarchical summarization. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 349–357, New Orleans, Louisiana, United States, September 2001.
- [LGZ03] Bing Liu, Robert Grossman, and Yanhong Zhai. Mining data records in web pages. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, Washington, DC, USA, August 2003.
- [LHC05] Bing Liu, Minqing Hu, and Junsheng Cheng. Opinion observer: Analyzing and comparing opinions on the web. In *Proceedings of the 14th international World Wide Web conference*, Chiba, Japan, May 2005.
- [LZ05] Bing Liu and Yanhong Zhai. Net - a system for extracting web data from flat and nested data records. In *Proceedings of 6th International Conference on Web Information Systems Engineering*, 2005.
- [MBF⁺90] G.A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller. Introduction to wordnet: An on-line lexical database. *Journal of Lexicography*, 3:235–244, 1990.
- [MH93] Kathleen McKeown and Vasileios Hatzivassiloglou. Augmenting lexicons automatically: clustering semantically related adjectives. In *Human Language Technology Conference, Proceedings of the workshop on Human Language Technology*, Princeton, New Jersey, 1993.
- [MN98] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification, 1998.
- [MS99] Christopher D. Manning and Hinrich Schutze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.
- [MS03] L. McKnight and P. Srinivasan. Categorization of sentence types in medical abstracts. In *AMIA Symposium*, pages 440–444, 2003.
- [MvZM05] L.A. Pizzato M. van Zaanen and D. Moll. Classifying sentences using induced structure. In *Proceedings of the 12th Edition of the Symposium on String Processing and Information Retrieval*, pages 139–150, Buenos Aires, November 2005.

REFERENCES

- [NSK⁺04] J.C. Na, H. Sui, C. Khoo, S. Chan, and Y Zhou. Effectiveness of simple linguistic processing in automatic sentiment classification of product reviews. In *Proceedings of the Eighth International ISKO Conference*, Wurzburg, Germany, 2004.
- [NUR03] Nikolaos Nanas, Victoria Uren, and Anne De Roeck. Building and applying a concept hierarchy representation of a user profile. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, Toronto, Canada, 2003.
- [PLV02] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 79–86, 2002.
- [RBT] RBT. <http://www.cs.jhu.edu/~brill/>.
- [Ryd02] Sara Rydin. Building a hyponymy lexicon with hierarchical structure. In *Proceedings of the ACL-02 workshop on Unsupervised lexical acquisition*, pages 26–33, Philadelphia, Pennsylvania, July 2002.
- [SC99] Mark Sanderson and Bruce Croft. Deriving concept hierarchies from text. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 1999.
- [SC06] Bin Shi and Kuiyu Chang. Mining chinese reviews. In *Sixth IEEE International Conference on Data Mining - Workshops (ICDMW'06)*, pages 585–589, Hong Kong, 2006. IEEE.
- [SKC03] H. Sui, C. Khoo, and S Chan. Sentiment classification of product reviews using svm and decision tree induction. In *Proceedings of State of the Art Implementation of Classification Research in Information Technologies: 14th Annual ASIST SIG CR Workshop*. American Society for Information Science & Technology, 2003.
- [TTO05] Akihiro Tamura, Hiroya Takamura, and Manabu Okumura. Classification of multiple-sentence questions. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing*, Jeju Island, Korea, October 2005.

REFERENCES

- [Tur02] Peter D. Turney. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the Association for Computational Linguistics 40th Anniversary Meeting*, New Brunswick, N.J., 2002.
- [URL] Uniform Resource Identifier (URI) : Generic Syntax. <http://www.ietf.org/rfc/rfc3986.txt?number=3986>.
- [UTF] UTF-8: A transformation format of ISO 10646. <http://www.ietf.org/rfc/rfc2279.txt?number=2279>.
- [WGE] GNU Wget. <http://www.gnu.org/software/wget/>.
- [WZ03] Hua Wu and Ming Zhou. Optimizing synonym extraction using monolingual and bilingual resources. In *Proceedings of the Second International Workshop on Paraphrasing: Paraphrase Acquisition and Applications*, Sapporo, Japan, 2003.
- [Yan91] Wu Yang. Identifying syntactic differences between two programs. *Software - Practice and Experience*, 21(7):739–755, 1991.
- [YL99] Y. Yang and X. Liu. A re-examination of text categorization methods. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 42–49, Berkley, August 1999.
- [YLjL05] Qiang Ye, Bin Lin, and Yi jun Li. Sentiment classification for chinese reviews: A comparison between svm and semantic approaches. In *Proceedings of the 4th International Conference on Machine Learning and Cybernetics*, Guangzhou, August 2005.
- [YP97] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *the 14th International Conference on Machine Learning*, pages 412–420, Nashville, US, 1997.
- [YSL06] Qiang Ye, Wen Shi, and Yijun Li. Sentiment classification for movie reviews in chinese by improved semantic oriented approach. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences*, volume 03, page 53.2, 2006.
- [YT05] Y. Yamamoto and T. Takagi. A sentence classification system for multi biomedical literature summarization. In *Proceedings of the 21st International Conference on Data Engineering Workshops*, 2005.

REFERENCES

- [Zho05] Guo Dong Zhou. A chunking strategy towards unknown word detection in chinese word segmentation. In *The 2nd International Joint Conference on Natural Language Processing*, Jeju Island, Republic of Korea, 2005.
- [ZL05a] Yanhong Zhai and Bing Liu. Extracting web data using instance-based learning. In *Proceedings of 6th International Conference on Web Information Systems Engineering*, 2005.
- [ZL05b] Yanhong Zhai and Bing Liu. Web data extraction based on partial tree alignment. In *Proceedings of the 14th international World Wide Web conference*, Chiba, Japan, May 2005.
- [ZLC⁺03a] Hua-Ping Zhang, Qun Liu, Xue-Qi Cheng, Hao Zhang, and Hong-Kui Yu. Chinese lexical analysis using hierarchical hidden markov model. In *Proceedings of the 2nd SIGHAN Workshop on Chinese Language Processing (ACL03)*, Sapporo, Japan, 2003.
- [ZLC⁺03b] Hua-Ping Zhang, Qun Liu, Xue-Qi Cheng, Hao Zhang, and Hong-Kui Yu. Hhmm-based chinese lexical analyzer ictclas. In *Proceedings of the 2nd SIGHAN Workshop on Chinese Language Processing (ACL03)*, Sapporo, Japan, 2003.
- [ZLZ⁺03] Hao Zhang, Qun Liu, Kevin Zhang, Gang Zou, and Shuo Bai. Statistical Chinese parser ICTPROP. Technical report, Institute of Computing Technology, P.R. China, June 2003.