Wei Wei

Mining Online Text Data for Sentiment and News Impact Analysis

Thesis for the degree of Philosophiae Doctor

Trondheim, September 2013

Norwegian University of Science and Technology
Faculty of Information Technology, Mathematics
and Electrical Engineering
Department of Computer and Information Science

NTNU – Trondheim
Norwegian University of
Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the degree of Philosophiae Doctor

Faculty of Information Technology, Mathematics and Electrical Engineering
Department of Computer and Information Science

ISBN 978-82-471-4636-1 (printed ver.)
ISBN 978-82-471-4637-8 (electronic ver.)
ISSN 1503-8181

Doctoral theses at NTNU, 2013:256

Printed by NTNU-trykk

To my family

Abstract

As continuous growth of Internet, an ever increasing amount of information becomes available on the World Wide Web (WWW). Information on the WWW has never been so exploded that search engines using traditional keyword-based searching strategies hardly meet people's needs to retrieve knowledge from online massive text data. The motivation of this thesis comes from the great demands on discovering implicit knowledge and rich semantics from online documents.

This thesis focuses on analyzing online business news, a representative of objective information, and online customer reviews, a representative of subjective information. For online business news, a topic driven impact analysis model is proposed that quantifies the impact of topic of a news article. With the proposed topic driven impact analysis model, an explorative visual analysis system called ImpactWheel is developed to help users better navigate and understand topic-specific companies' impact relationships through mining rich information source of online business news.

For online customer reviews, both document overall sentiment classification and attributed-based sentiment analysis are performed. In the regard of document overall sentiment classification, taking advantages of high frequency of Co-occurring Term (CoT) patterns in customer reviews, a frequency-based algorithm is proposed to generate complex features which benefits sentiment classifiers. In order to search for effective features and ignore useless ones produced by the frequency-based complex feature generation algorithm, an Effective Feature Search (EFS) framework is proposed, which makes a novel connection between feature candidate generation and a Stochastic Local Search process. In the regard of attributed-based sentiment analysis, the concept of Sentiment Ontology Tree is proposed, which organizes a product's domain specific knowledge as well as sentiments in a tree-like ontology structure. With the concept of SOT, a Hierarchial Learning via Sentiment Ontology Tree (HL-SOT) approach is proposed to solve the sentiment analysis tasks in a hierarchical classification process. To enhance the classification performance and computational efficiency of the HL-SOT approach which encodes texts using a globally unified index term space, a Localized Feature Selection (LFS) framework is developed which generates the customized index term space for each node of SOT. Since that the HL-SOT approach was estimated by a RLS estimator which is not competent enough to find max class separation and that the statistical linear classifier has been evidently proven its fallibility on classifying sentiment, a more pragmatic Hybrid Hierarchical Classification Process (HHCP) is proposed. The HHCP approach employs a linear classifier that is capable of maximizing the class separation while minimizing the within-class variance for attribute detection and turns to a rule-based solution for sentiment orientation.

Preface

This thesis is submitted to the Norwegian University of Science and Technology (NTNU) for partial fulfilment of the requirements for the degree of philosophiae doctor.

This doctoral work has been performed at the Department of Computer and Information Science (IDI), NTNU, Trondheim, with Professor Jon Atle Gulla as main supervisor and with Professor Kjetil Nørvåg and Dr. Terje Brasethvik as co-supervisors.

The thesis has been part of the research project Cooperative Mining of Independent Document Repositories (COMIDOR), funded by the Norwegian Research Council under the VERDIKT research programme, grant No. 183337.

Acknowledgements

Foremost, I would like to express my sincere gratitude to my supervisor Professor Jon Atle Gulla for his fruitful discussions, guidance and help throughout my PhD work. I would also like to thank Professor Kjetil Nøvåg and Dr. Terje Brasethvik for being my co-supervisors.

I would like to thank Associate Professor Ole J. Mengshoel for his mentoring and valuable research discussions during my research stay at Silicon Valley campus of Carnegie Mellon University.

In addition, I would like to thank Geir Solskinnsbakk and Stein L. Tomassen. We have interesting discussions on research and good cooperation on teaching. I would also like to thank Rune Sætre for his patience on teaching me Norwegian and interesting discussions we had on various topics. Thanks to the administrative and technical staff at IDI for providing me kindly assistance. I would also like to thank all my colleagues at IDI for their help and support for providing a pleasant working environment.

I am immensely grateful to my father Nanxiang Wei and my mother Yuqi Zhou who made a bad boy turn to a right track with their great love and patience. Thanks to my grandma Xianming Zhu who lead the hyperactive boy to be quiet and to get used to reading and thinking so that he can find his way to research for a PhD. I would also like to thank my parents-in-law Dapeng Ge and Xiaoli Wang for bringing up my wife and their understanding, encouragement and help when we are in Norway. Finally, I would like to give a special thank to my loving wife Wei Ge and our wonderful daughter Zhihan Wei for the love, support and happiness I have from them.

Contents

Part I

Introduction

Chapter 1

Introduction

In this chapter, an overview of research work conducted during my PhD study is presented. In Section 1.1, the background and motivation of the work is discussed. The problem outline for the thesis is described in Section 1.2. A brief description of research context is presented in Section 1.3, followed by research goal and questions discussed in Section 1.4. Research approach and our research contributions are respectively presented in Section 1.5 and Section 1.6. Our papers that are included in this thesis are listed in Section 1.7. Finally, an overview of the structure of the rest of the thesis is given in Section 1.8.

1.1 Background and Motivation

As the internet reaches almost every corner of the world, more and more people get used to accessing information on the World Wide Web (WWW). Information on the WWW has never been so exploded. Search engine, e.g., Google[1], has become an important tool for users to look for information they need. Traditional searching technologies have been studied in a relatively mature research area. However, documents retrieved by traditional searching technologies only capture the explicit information and knowledge of documents. Document collections usually contain implicit knowledge and rich semantics, e.g., topic trends hidden behind large scales of news and sentiment expressed within customer reviews, where technologies with traditional keyword-based searching strategy do not work very well.

There are basically two kinds of information on the WWW, i.e., objective information and subjective information. The representative of objective information is online news. As the continuous growth of the online medias such as New York Times(NYT)[2], an ever increasing amount of information is becoming available through collections of news articles.

[1] http://www.google.com
[2] http://www.nytimes.com

These news collections contain rich context information and complex inter connections. There is a great need and challenging to help people to understand and navigate through the online news with rich context in nature. With the rapid expansion of Web 2.0 technologies that facilitates people to write reviews and share opinions online, a large amount of review texts are generated and available on the WWW. The online user-generated reviews are the representative of subjective information and are deemed to be rich in opinions and can be very useful information for potential customers, online advertisers as well as product manufacturers. As the amount of opinion information grows rapidly, it becomes impossible for humans to manually collect and digest these opinion-rich texts exhaustively. However, ranked lists of web contents retrieved by traditional search engines are insufficient for more complex data exploration and analytical tasks discussed above.

The motivation for this thesis comes from the demands on discovering knowledge and semantics from online text data. The thesis focuses on analyzing both objective information, e.g., online business news, and subjective information, e.g., online customer reviews, of online text data. For online business news, we analyze the impact of news of companies so as to better understand the affection of a specified event and its epidemic through mining news collections. For online customer reviews, we perform sentiment analysis on them so that we can recommend products to new customers using opinions from previous customers.

1.2 Problem Outline

The problems that are investigated in this thesis belong to the field of web intelligence[3], which is the area of study and research of the application of artificial intelligence and information technology on the web in order to create the next generation of products, services and frameworks based on the internet. Technologies such as machine learning, data mining, and semantic web, etc. are usually involved in solving web intelligence tasks.

The first main problem studied in this thesis is analyzing the impact of news of companies and extracting the affection of a specified event and its epidemic through mining news collections. As we know, online business news collections may cover various topics of companies, their products, events and related people. Intuitively, each news of a company represents one topic the company is involved. With its development, a topic once it comes forth will usually impact more other companies. Information retrieval technologies help to find related news pages on a certain topic. However it is still difficult when we are trying to find innate relations of the content of multiple topic contexts. When considering underlying contexts, there are no clear answers to the questions such as “Are company A and company B are related? how are they related? and why are they related?".

To answer these questions, users usually have to examine fine-grained local-level relations

[3] http://en.wikipedia.org/wiki/Web_intelligence

over multiple topics. For instance, the news of the bankruptcy of “Lehman Brother" had a great impact on a number of financial institutions. In order to better understand this event, the users need to check a series of news articles to find which companies are most related with Lehman and most affected by Lehman. Although some existing techniques provide valuable insights into solving the similar challenges, none of them offers a complete solution to address the following two challenges: 1). given a news topic or article of a company, how to detect its impact to other companies? 2). how to make the complex analysis approaches in a simple explorative manner that can be used by common users? To bridge this gap, in this thesis we present an approach that enables users to navigate and analyze large news corpora with rich topic contexts.

The other main problem studied in this thesis is performing sentiment analysis on customer reviews. The research of sentiment analysis was proposed to concern not only what topics are talking about in the documents but also what opinions and sentiments are expressed on the related topics. Existing works on sentiment analysis can be divided into two categories. One category of work focuses on analyzing document overall sentiment, i.e. overall sentiment classification (e.g. [1, 2, 3]). The other category of work focuses on analyzing which aspect of the product the sentiment is expressed on, which is usually referred to as attribute-based sentiment analysis (e.g. [4, 5, 6]). This thesis studies both categories of the problems.

As suggested by its name, document overall sentiment classification is concerned with analyzing a document’s overall sentiment, which can be solved by two main approaches. Lexicon-based methods [7] conduct sentiment analysis by inferring a document’s overall sentiment from sentiments of words (e.g. [8]) or phrases (e.g., [9]). Machine learning approaches build classifiers to classify a document’s overall sentiment through a supervised [10] or unsupervised [11] learning process.

Since Pang et al. [10] studied sentiment classification using machine learning techniques, a lot of work has addressed the document overall sentiment classification problem in a supervised text classification process. Within exiting publications there exist various techniques to improve performance of traditional topic-based classifiers on sentiment classifications. These techniques include feature selection approaches, identifying more important subjective portions of texts [12], learning from human-annotator rationale [13] or human interaction [14]. The problem of document overall sentiment classification we dealt in this thesis aims at improving performance of sentiment classifiers from the perspective of feature selection.

Attributes-based sentiment analysis is to analyze sentiment based on each attribute of a product. When we look into the details of each example of product reviews, we find that online product reviews usually constitute domain specific knowledge. The product’s attributes mentioned in reviews might have some relationships between each other. For example, for a digital camera, comments on image quality are usually mentioned. However, a sentence like “40D handles noise very well up to ISO 800", also refers to image quality of the camera 40D. Here we say “noise" is a sub-attribute factor of “image quality". As we know, in computer science and information science, an ontology formally repre-

sents knowledge as a set of concepts within a domain, and the relationships among those concepts[4]. Therefore, in this thesis we aims at studying the problem of ontology-based sentiment analysis, where ontology structure serves as external knowledge to organize a product's attributes.

1.3 Research Context

The research in this PhD thesis has been conducted at the Department of Computer and Information Science (IDI) at Norwegian University of Science and Technology (NTNU) within the project Cooperative Mining of Independent Document Repositories (COMIDOR). The COMIDOR project is funded by the Norwegian Research Council under the VERDIKT research programme with project number 183337. The COMIDOR project started in 2008 and ended in 2012.

The main objective of the COMIDOR project is to understand the form and contents from cooperatively mining independent document collections. Traditional search technologies have been studied in a relatively mature research area. However, documents retrieved by traditional search technologies only capture the explicit information and knowledge of documents. Document collections usually contain implicit rich semantics, where technologies with traditional keyword-based searching strategy do not work very well. In this research context, the focus of this thesis is on mining implicit rich semantics and knowledge from document collections. The targeted document collections used in our research are respectively 1) online business news which represents objective information on the WWW and 2) online customer reviews which represents subjective information on the WWW.

1.4 Research Goal and Questions

The research in this thesis aims at developing approaches to extracting implicit knowledge from mining online text data. Specifically, the knowledge to be extracted depends on which online text data are analyzed. Focus on the online business news (one representative of online objective information) and online customer reviews (one representative of online subjective information), the main research objectives of this thesis are two-fold:

RO1: How can we detect companies' relations through analyzing online business news collections?

RO2: How can we extract peoples' opinions and sentiments through analyzing online customer reviews?

[4] http://en.wikipedia.org/wiki/Ontology_(information_science)

1.4.1 Business News Analysis

As the continuous growth of the online medias such as New York Times(NYT)[5], an ever increasing amount of information is becoming available through collections of news articles. Traditionally, people use search tools to retrieve a ranked list of documents whose content is highly related to a set of user-supplied keywords. This model has proven remarkably powerful for information retrieval tasks, such as locating the address of a restaurant. However, ranked lists of news contents are insufficient for more complex data exploration and analytical tasks where users try to understand the relations between complex concepts that span across multiple documents. One main research objective of this thesis aims at discovering relations among companies according to their impact that can be detected within their news. A company has an impact on another company if news about the impacted company in some systematic way reflect what is going on with the other company. Our first research question is:

RQ1: How can we model and quantify the impact of a company's news?

1.4.2 Customer Reviews Analysis

On the WWW there is a mass of information with multifarious opinions on a given topic may be generated from all over the world in a very short time. As the number of product reviews grows, it becomes difficult for a user to manually learn the panorama of an interesting topic from existing online information. Faced with this problem, research on opinion mining and sentiment analysis, e.g., [4, 5, 15], were proposed and have become a popular research topic at the crossroads of information retrieval and computational linguistics. Research on sentiment analysis can be classified into two different categories according to granularity of sentiments being analyzed against texts. One category is document overall sentiment analysis. The other category is aspect-based sentiment analysis. In this thesis, we study problems of sentiment analysis on customer reviews in both categories.

Carrying out sentiment analysis on customer reviews is not a trivial task. When we look into the details of each example of product reviews, we find that there are two intrinsic properties that might help us to solve the problem:

IP1: Customer reviews constitute domain-specific knowledge.

The product's attributes mentioned in reviews might have some relationships between each other. For example, for a digital camera, comments on image quality are usually mentioned. However, a sentence like "40D handles noise very well up to ISO 800", also refers to image quality of the camera 40D. Here we say "noise" is a sub-attribute factor of "image quality".

IP2: Vocabularies used in product reviews tend to be highly overlapping.

[5] http://www.nytimes.com

In online customer reviews, for one product there only exist a finite number of aspects (product's attributes) that can be commented on. For example, for a digital camera, attributes that are usually mentioned in reviews are "price", "LCD", "picture quality" and "battery life", etc. For each reviewed attribute, there are a finite number of vocabularies that are usually involved in sentiment expressing.

Document Overall Sentiment Analysis

Document overall sentiment classification is concerned with analyzing a document's overall sentiment. As indicated in the statement of Intrinsic Property 2 (IP2), vocabularies used in product reviews tend to be highly overlapping. Words referred to attributes of a product as well as words for describing sentiment on the attributes will co-occur in the customer reviews with high frequency. Furthermore, high frequent co-occur terms together indicate more clear sentiments than each only single term. For example, single term like "high" does not necessarily means positive sentiment. However, in a corpus of customer reviews on digital cameras terms "high" and "price" might co-occur together frequently and means definitely negative. Therefore, we have a research question:

RQ2: How can we capture high frequent co-occur terms as complex features to improve the accuracy of sentiment classification on product reviews?

As high frequent co-occur terms are generated as complex features for sentiment classifier, classification performance is expected to be improved. However, all the captured high frequent co-occur term patterns are not necessarily effective features for a sentiment classifier. For example, in customer reviews on hotels, the terms "staff", "nice", and "service" usually co-occur together. However, the generated complex feature like "staff service" is a noise or useless feature to classifiers. Therefore, we know the generated complex features by high frequent co-occur term pattern may generate noise as well and we have the following research question to deal with this problem:

RQ3: How can we identify and remove unwanted complex features from the generated co-occur terms without losing effective features for sentiment classification?

Attributed-based Sentiment Analysis

Attributes-based sentiment analysis is to analyze sentiment based on each attribute of a product. As indicated in the statement of Intrinsic Property 1 (IP1), customer reviews constitute domain-specific knowledge. There are relationship between attributes of a product in that domain. In computer science, one natural way to represent the structure and relationship of concepts of a domain knowledge is to use ontology. It is intuitive to investigate whether we can use the knowledge of an ontology structure to help us perform attributed-based sentiment analysis. Therefore we have the following research questions:

RQ4: How can we design an ontology-supported framework so that knowledge of the product ontology enhances the sentiment analysis process?

The above research question aims at developing an ontology-supported framework that can use the knowledge of ontology structure of a product domain to facilitate sentiment analysis process. If an ontology-supported sentiment analysis framework can be developed, it entails opportunities of improvement from several angles. Therefore, a further research question following RQ5 is:

RQ5: What are the important factors affecting the performance of the above ontology-supported sentiment analysis framework?

1.5 Research Approach

In this section, we discuss the research approaches used in this work.

The goal of the research process is to produce new knowledge or deepen understanding of a topic or issue[6]. Generally, there are three main forms of taking a research process, i.e., exploratory research, constructive research, and empirical research. Although it is difficult to define clear boundaries between them, each of them highlights different aspects of activities in the research process. Exploratory research is a type of research conducted for a problem that has not been clearly defined[7]. Constructive research is perhaps the most common computer science research method. This type of approach demands a form of validation that doesn't need to be quite as empirically based as in other types of research like exploratory research[8]. Empirical research is a way of gaining knowledge by means of direct and indirect observation or experience. Empirical evidence (the record of one's direct observations or experiences) can be analyzed quantitatively or qualitatively[9].

The research taken in this thesis involves activities of the above research approaches. Specifically, the research process includes following typical activities:

- **Research Problem Survey.** The important approach to starting a research process is to survey the research problems. The survey is mainly conducted by broad literature reviewing. The literatures are mainly from prestigious international conference proceedings, e.g., ACL, WWW, SIGIR, etc, and good international journals, e.g., TKDE, TOIS, etc. Through this broad reading process, we can find out the challenges of our research problems and understand the state-of-the-art techniques proposed in existing publications. In the research problem survey process, we learn all the preliminary knowledge of our research and make good preparation for further exploitation.

[6] http://en.wikipedia.org/wiki/Research#Research_methods
[7] http://en.wikipedia.org/wiki/Exploratory_research
[8] http://en.wikipedia.org/wiki/Constructive_research
[9] http://en.wikipedia.org/wiki/Empirical_research

- **Knowledge Learning and Approach Development.** The model and approach development process is to propose our own methods to tackle the research problems. In this process, we first analyze the challenges within the problems. Then we learn and study the knowledge that can be applied to the problems. The knowledge we need to learn in this stage involves several areas including natural language processing, linear algebra, probability theory, neural information processing, etc. After deep learning on the required knowledge, we can propose our own approaches to the research problems.
- **Data Preparation.** One important activity for research is to prepare data on which the proposed approaches can be evaluated. In our research, we use both public standard data sets and manually labeled self-created data sets. For task of sentiment classification, we use public standard data sets. e.g., the movie review data set[10] so that our approach can be easily compared with related work. For some tasks, such as product attribute detection and news impact analysis, we have to crawl data from online websites and manually label the data set for the specific experimental purposes.
- **Metrics for Result Analysis.** In empirical research, experiments can be analyzed quantitatively and qualitatively. The metrics used in the research of this thesis mainly use quantitatively empirical analysis. For each research problem, metrics are designed with each research questions raised in the research process so that advantages and disadvantages of proposed approaches can be revealed in an objective manner.

1.6 Research Contributions

This thesis has 6 research contributions listed as follows:

C1: ImpactWheel, an explorative visual analysis system that can reveal the impact of news articles.

In paper P1, we propose ImpactWheel, a new visual analysis technique that enables users to navigate and analyze large news corpora with rich topic contexts. Topic driven impact analysis provides a ranking mechanism that finds a set of companies that are deemed as most impacted by topics of news of a user-interested company. The idea of a probabilistic topic model is that documents are mixtures of topics, where a topic is semantically coherent and is formally treated as a probability distribution of words. In the paper P1 we discuss how we use a probabilistic topic model to naturally quantify impact of a company's news to the topic proportion in other companies' news collections, and propose a semi-supervised model estimation process to estimate the model's parameters which serves as quantity of impact of a company's news. In the paper P1, we also provide a new visualization design that helps us portray the relation ranking results and facilities data

[10] http://www.cs.cornell.edu/people/pabo/movie-review-data

understanding. Rich interactions are also provided that enable us to explore the analysis results in a dynamic and efficient way and also help to detect data patterns from rich context.

C2: An automatic approach for generating complex features for sentiment classifiers.

In paper P2, we propose an approach to generating complex features, called multi-unigram features, to enhance a negation-aware Naive Bayes classifier. The term “multi-unigram feature" is coined to represent the process that the generated features are produced by our algorithm that takes an initial set of unigram feature candidates as input. We further make the Naive Bayes classifier aware of negation expressions in the training and classification process to eliminate the confusions of the classifier that is caused by negation expressions within sentences. Experiments in the paper P2 not only qualitatively show the good quality of the generated features but also quantitatively demonstrate a significant effectiveness of ideas of both the multi-unigram features generation and the negation-aware classifier on improving the performance of the original Naive Bayes classifier.

C3: An Effective Feature Search (EFS) framework for enhancing the performance of sentiment classifiers.

As discussed in RQ3, the complex feature generation algorithm proposed in the paper P2 not only produce effective features but also bring useless noise for sentiment classifiers. In paper P3, we propose an Effective Feature Search (EFS) framework that makes a novel connection between feature candidate generation and a Stochastic Local Search (SLS) process to enhance performance of machine learning classifiers for sentiment classification. The EFS framework contains two steps. In the feature generation step, we utilize filter-based methods [16] to generate feature candidates including both complex feature and unigram feature taking advantage of high frequency Co-occurring Term (CoT) patterns. In the feature pruning step, we map the feature set optimization process to a Stochastic Local Search (SLS) process. In the proposed SLS model, a wrapper-based selection is adopted to score each selected feature subset with an objective function tailored to the classifier. A hill-climbing SLS algorithm is developed in the model to ensure quickly finding a local optima.

C4: A Hierarchical Learning via Sentiment Ontology Tree (HL-SOT) approach for sentiment analysis.

In paper p4 and P5, we study the problem of sentiment analysis on product reviews through a novel method, called the HL-SOT approach, namely Hierarchical Learning (HL) with Sentiment Ontology Tree (SOT).By sentiment analysis on product reviews we aim to fulfill two tasks, i.e., labeling a target text with: 1) the product’s attributes (attributes detection task), and 2) their corresponding sentiments mentioned therein (sentiment orientation task). The proposed HL-SOT approach is the first work to formulate the tasks of sentiment analysis to be a hierarchical classification problem. In the paper P4, we first propose a formal definition on SOT. A specific hierarchical learning algorithm is further proposed to achieve tasks of sentiment analysis in one hierarchical classification

process.

C5: A Localized Feature Selection (LFS) framework tailored to the HL-SOT approach.

In paper p6, we propose a Localized Feature Selection (LFS) framework tailored to the HL-SOT approach to sentiment analysis. In the proposed LFS framework, significant feature terms of each node can be selected to construct the locally customized index term space for the node so that the classification performance and computational efficiency of the existing HL-SOT approach are improved.

C6: A Hybrid Hierarchical Classification Process for Sentiment Analysis.

In paper P7, a Hybrid Hierarchical Classification Process (HHCP) is proposed to solve the two tasks, i.e., attributes detection task and sentiment orientation task, of sentiment analysis. The HHCP approach is proposed based on the paper P4 of the HL-SOT approach. Compared with the HL-SOT approach, the HHCP approach has the following contributions. First, the HHCP approach employs a linear Fisher classifier for attribute detection task, since Fisher classifier is developed by requiring maximum class separation in the output space, which is deemed as more competent than the Regularized Least Squares (RLS) employed by the HL-SOT approach. Second, the HHCP approach only performs the sentiment orientation task on the identified attributes that are leaf nodes of the hierarchical structure. Third, since the statistical linear classifiers that are designed for semantic classifications are evidently prone to errors when applied to classifying sentiment information, unlike the HL-SOT approach, the HHCP approach turns to a rule-based heuristic solution for the sentiment orientation task.

1.7 Papers

There are seven papers that are included in this thesis. These papers have all been published in international peer reviewed workshops, conferences, and journals. In this section, we present an overview of the papers. Details of the papers are included in the thesis in Part II.

P1: **Wei Wei**, Nan Cao, Jon Atle Gulla and Huamin Qu: “ImpactWheel : Visual Analysis of the Impact of Online News", in Proceedings of the 2011 IEEE/WIC/ACM International Conference on Web Intelligence.

P2: **Wei Wei**, Jon Atle Gulla and and Zhang Fu: “Enhancing Negation-Aware Sentiment Classification on Product Reviews via Multi-Unigram Feature Generation", in Proceedings of the Sixth International Conference on Intelligent Computing.

P3: **Wei Wei**, Ole J. Mengshoel and Jon Atle Gulla: “Stochastic Search for Effective Features for Sentiment Classification" Under submission to Data and Knowledge Engineering.

P4: **Wei Wei** and Jon Atle Gulla: "Sentiment Learning on Product Reviews via Sentiment Ontology Tree", in Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics.

P5: **Wei Wei**: "Analyzing Text Data for Opinion Mining", in Proceedings of 16th International Conference on Applications of Natural Language to Information Systems.

P6: **Wei Wei** and Jon Atle Gulla: "Enhancing the HL-SOT Approach to Sentiment Analysis via a Localized Feature Selection Framework", in Proceedings of the 2011 International Joint Conference on Natural Language Processing.

P7: **Wei Wei** and Jon Atle Gulla: "Sentiment Analysis in a Hybrid Hierarchical Classification Process", in Proceedings of the 7th International Conference on Digital Information Management.

1.8 Thesis Structure

This thesis contains two parts. Part I introduces background and motivation of research in this thesis and also presents an overview of technology context, related work, results and evaluations, etc. Part II contains a list of papers that are related to research in this thesis. The remainder of this thesis is organized as follows:

Part I

Chapter 2: Technology Context. In this chapter, an overview of background knowledge and technologies that are related to this thesis are reviewed.

Chapter 3: State of the Art. In this chapter, we discuss our research problems in terms of the state of the art approaches.

Chapter 4: Research Results. In this chapter, we summarize the contributions of this thesis. Each contribution is correspondent to a research question raised in the Section 1.4. For each contribution, a research question is revisited. Then we briefly describe each proposed approach, its evaluation, and roles of authors in the paper.

Chapter 5: Conclusion and Future Work. In this chapter, we conclude the work in this thesis and discuss potential research directions for future work.

Part II: Publication List. This part contains a list of the papers (P1-P7) that are used in this thesis and produced in the period of my PhD study.

Chapter 2

Technological Background

In this chapter, we briefly introduce fundamental techniques that are background knowledge for understanding the research papers that are included in this thesis. The chapter is organized as follows. Section 2.1 discusses five classic feature selection algorithms that are utilized in the papers P2, P3, and P6. Section 2.2 describes a list of machine learning classifiers that are used in the papers P2-P7. Section 2.3 presents a classic association rule learning algorithm that inspires complex feature generation algorithm proposed in the papers P2 and P3. Section 2.4 introduces statistical topic models that are background technique for developing the topic driven impact analysis model in the paper P1.

2.1 Feature Selection Approaches

Feature selection [1] also known as feature reduction, variable selection or data dimension reduction is a process of selecting a subset of most important and relevant features for building robust learning models. Research on feature selection techniques has become the focus of people working on statistical machine learning areas with the following objectives: improving the prediction performance of the models, providing faster and more cost-effective learning and classification process, and providing a better understanding of the underlying process that generated the data [17]. In this section, we review five classic feature selection algorithms, i.e., Document Frequency (DF) [18], Mutual Information (MI) [18, 19], χ^2-statistic (CHI) [18], Term Strength (TS) [20], and Information Gain (IG) [21], that are used in our research process.

2.1.1 Document Frequency Feature Selection

Document Frequency (DF) feature selection algorithm is a frequency-based feature selection process. The DF algorithm is the simplest and effective feature selection algorithm

[1] http://en.wikipedia.org/wiki/Feature_selection

that quantifies importance of a term as the number of documents in which the term occurs. Based on the intuition that rare terms are less important for category prediction, the DF algorithm counts the document frequency of each unique terms in the training data set and select a sub set of terms whose document frequency fulfills the threshold.

2.1.2 Mutual Information Feature Selection

Mutual Information (MI) is usually used to measure dependence of two random variables[2]. In text classification, MI can be used to indicate how much a term t is related to a class c. Therefore MI might serve as a criteria to select feature terms for some specific class and is calculated as [18]:

$$I(t,c) = \sum_{e_t \in \{1,0\}} \sum_{e_c \in \{1,0\}} P(e_t, e_c) \log_2 \frac{P(e_t, e_c)}{P(e_t)P(e_c)}, \tag{2.1}$$

where $e_t = 1$ means a training document contains t and $e_t = 0$ means a training document does not contain t, and $e_c = 1$ means a training document is in class c and $e_c = 0$ means a training document is not in class c.

2.1.3 χ^2-statistic Feature Selection

χ^2 feature selection is a popular algorithm for selecting features in text classification. The criteria used in the χ^2 feature selection algorithm is based on χ^2 test which is applied to test the independence of two events. Let t denote a term and c denote a class. The χ^2 score of t with c is calculated as [18]:

$$\chi^2(t,c) = \sum_{e_t \in \{1,0\}} \sum_{e_c \in \{1,0\}} \frac{(N_{e_t e_c} - E_{e_t e_c})^2}{E_{e_t e_c}}, \tag{2.2}$$

where e_t and e_c have the same definition as in the Formula 2.1, and N and E are respectively the observed frequency and the expected frequency in the training data.

2.1.4 Term Strength Feature Selection

Term Strength (TS) [20] estimates term importance based on how commonly a term is likely to appear in a cluster of highly related documents [22]. In the TS feature selection process, the algorithm first cluster documents in the training data according to their similarity. The number of clusters that can be generated from training data depends on the threshold on document similarity. Let documents x_i and x_j respectively represent any two different documents from the same cluster of training data. Score of term strength of

[2] http://en.wikipedia.org/wiki/Mutual_information

t is calculated based on the estimated conditional probability that t occur in the document x_i given that t also occur in the document x_j, i.e.,:

$$s(t) = P(t \in x_i | t \in x_j). \tag{2.3}$$

2.1.5 Information Gain Feature Selection

Information Gain (IG) is another frequently used measurement on feature selection in the field of machine learning. The IG utilizes entropy to calculate information change given more conditions. In information theory, entropy is used to measure uncertainty of a random variable. Let X denote a random variable with possible values $\{x_1, x_2, ..., x_n\}$. Then the entropy of X:

$$I(X) = -\sum_{i=1}^{n} P(x_i) log_2 P(x_i). \tag{2.4}$$

The IG feature selection algorithm quantifies the importance of a term t to a class c as how much difference is between entropy of c and conditional entropy of c given t:

$$IG(t, c) = P(c|t) log_2 P(c|t) - P(c) log_2 P(c). \tag{2.5}$$

2.2 Machine Learning Classifiers

Machine learning is a discipline within Artificial Intelligence (AI) that deals with algorithms that enable computers to learn from empirical data and solves problems on new observed data. In machine learning and statistics, classification is a key problem that predicts which categories new observed data belong to. In this section, we review a list of machine learning classifiers that are utilized in the research of this thesis.

2.2.1 Naive Bayes Classifier

The technique of a Naive Bayes (NB) classifier is based on the Bayesian theorem[3] with strong “naive" independence assumptions. The NB classifier is a simple but popular effective probabilistic classifier on solving classification problems. It can often outperform more sophisticated classification methods. In a typical text categorization problem, let d denote a document and let c denote a class. With the Bayes' rule, the probability of the document d being in the class c is calculated as:

$$P(c|d) = \frac{P(c)P(d|c)}{P(d)}.$$

[3] http://en.wikipedia.org/wiki/Bayes'_theorem

Since for each class c, the probability of a document d, i.e., $P(d)$, can be treated equally, with conditional independent assumption on words of the document d, the probability of d being in c can be derived as:

$$P(c|d) \propto P(c) \prod_{\forall w \in d} P(w|c),$$

where $P(w|c)$ is the conditional probability of a word w occurring in a document that belongs to class c.

2.2.2 Support Vector Machine

Support Vector Machine (SVM) is one of the best supervised machine learning techniques. It was proposed by Cortes and Vapnik [23]. The SVM classifier projects data to a high dimension which is typically much higher than the original feature space. The nonlinear function, say $\varphi(\cdot)$, can be polynomials, Gaussians, or other basis functions [24]. In a sufficiently high dimension transformed by $\varphi(\cdot)$, data from two categories can be separated by a hyperplane which has the largest distance to the nearest training data point of any class (see Fig. 2.1 [25]). The distance between the hyperplane and the nearest point is called margin. The nearest point is called support vector.

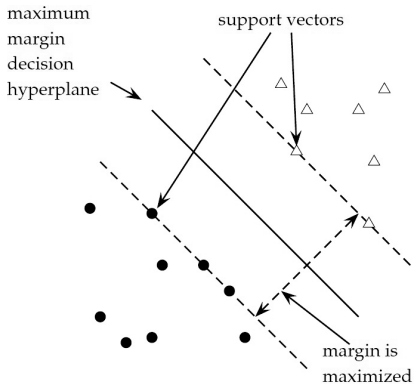


Figure 2.1: An Example of Support Vector Machine Hyperplane

Let feature vector v_i represent each data pattern in original space. Let x_i denote the vector that is transformed by an appropriate nonlinear function $\varphi(\cdot)$: $x_i = \varphi(v_i)$. Then the training data set in the transformed space is represented by $\{(x_i, y_i)\}$, where x_i is a vector in the transformed space and $y_i = \pm 1$ according to whether the data instance i is in or not in the category. Let w represent the gradient vector of the optimal hyperplane. The hyperplane can be represented by $w^T x + b$. Thus the margin between the hyperplane and x_i is:

$$\frac{y_i(w^T x_i + b)}{\|w\|}.$$

Assuming there is a positive margin δ exists [23], that is

$$\frac{y_i(w^T x_i + b)}{\|w\|} \geq \delta.$$

The goal of the SVM is to find appropriate parameters, i.e., w and b, of the hyperplane that maximize δ. To ensure a unique solution of w and b, we impose the constraint $\|w\|\delta = 1$ and the objective function can be represented as:

$$\begin{aligned} \max_{w,b} \quad & \frac{1}{\|w\|} \\ s.t. \quad & y_i(w^T x_i + b) \geq 1. \end{aligned}$$

Since maximizing $\frac{1}{\|w\|}$ is equal to minimizing $\frac{1}{2}\|w\|^2$, the objective function can be presented as:

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}\|w\|^2 \\ s.t. \quad & y_i(w^T x_i + b) \geq 1. \end{aligned}$$

The above optimization problem is a typical quadratic programming problem and have been studied in a number of alternative schemes [26, 23, 27].

2.2.3 K-Nearest Neighbor

K-Nearest Neighbor (KNN) classification is one of the classic algorithm in machine learning. It is a supervised learning algorithm and has been used in many applications in the field of data mining, statistical pattern recognition, and text processing. KNN is instance-based learning and assign objects to the class of its closest k neighbors, where k indicates the number of closest neighbors that are considered in the training and classification process. Here is an example of KNN in Fig. 2.2, where $k = 7$. From the example, we can see that within the seven closest neighbors around the object “X" there are five black circles and two white circles. Therefore, the “X" object will be assigned to the class of black circle.

The most popular metrics for calculating similarity between objects are Euclidean distance and Cosine similarity. Let n-dimensional vectors x_i and x_j respectively represent two objects. The Euclidean distance[4] is the ordinary distance between two points that one would measure with a ruler and is calculated as:

$$\text{distance}(x_i, x_j) = \sqrt{\sum_{t=1}^{n}(x_{i,t} - x_{j,t})^2}.$$

[4] http://en.wikipedia.org/wiki/Euclidean_distance

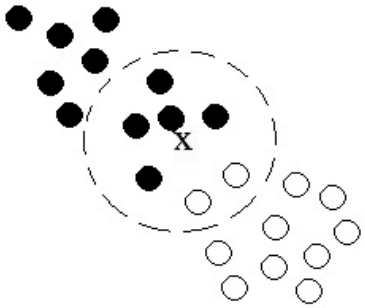


Figure 2.2: An Example of KNN Classifier ($k = 7$)

The Cosine similarity[5] is a measure of similarity between two vectors of an inner product space that measures the cosine of the angle between them and is calculated as:

$$\text{sim}(x_i, x_j) = \frac{x_i \cdot x_j}{\|x_i\|\|x_j\|} = \frac{\sum_{t=1}^{n} x_{i,t} \times x_{j,t}}{\sqrt{\sum_{t=1}^{n}(x_{i,t})^2}\sqrt{\sum_{t=1}^{n}(x_{j,t})^2}}.$$

2.2.4 Decision Tree Classification

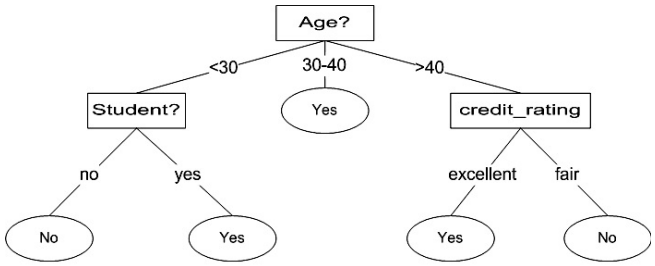


Figure 2.3: An Example of Decision Tree.

A decision tree[6] is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. In machine learning and data mining, a decision tree classifier is a predictive model using decision tree to map objects to class labels which are represented as leaves in the classification tree structure. An example of a decision tree classification is presented in Fig. 2.3. In the example, we see that each interior node, e.g., age and credit rating, represent a feature to be considered in the decision tree. Leaf nodes in Fig. 2.3 represent two classes, i.e., yes or no, of decisions on whether issuing a credit card. A decision tree

[5] http://en.wikipedia.org/wiki/Cosine_similarity
[6] http://en.wikipedia.org/wiki/Decision_tree

can be learned from training data. There are several decision tree learning algorithms including Iterative Dichotomiser 3 (ID3) [28], C4.5 algorithm [29], and Classification And Regression Tree (CART) [30], etc.

2.2.5 Linear Fisher Classifier

A linear Fisher classifier also known as Fisher's linear discriminant is a classification method that finds a linear combination of features and projects high-dimensional data onto a line and performs classification in this one-dimensional space. A linear Fisher classifier is developed by maximizing separation between classes while minimizing variance within each class. Let $x \in \mathcal{X}(\mathcal{X} = R^d)$ denote a vector representation for a text to be classified. Let c and $\bar{c}$ respectively denote the two classes: related to c and not related to c. The function of a linear Fisher classifier $f(.)$ is to project the d-dimensional input vector x down to one dimension $y \in R$ by:

$$y = f(x) = w^T \cdot x,$$

where $w = (w_1, w_2, ..., w_d)^T$ is a unit weight vector that defines the linear Fisher classifier. Imagine that if the two classes c and $\bar{c}$ are divisible in the d-dimensional space, after being projected down to the one dimension R, we still want to keep their divisibility. That is to say a projection needs to be selected so that the class separation can be maximized. Let the mean vectors x_c and $x_{\bar{c}}$ respectively represent the two classes of c and $\bar{c}$, i.e.,:

$$x_c = \frac{1}{N_c} \sum_{i \in c} x_i, \quad x_{\bar{c}} = \frac{1}{N_{\bar{c}}} \sum_{j \in \bar{c}} x_j.$$

We need to find a weight vector w that can maximize the separation distance between x_c and $x_{\bar{c}}$ when projected by w. However, the projection discovered in this way still suffers from a problem that the two classes that could be separated in the original space are still overlapping in the one dimensional output space, because the covariances of the two class distributions are non-diagonal. To alleviate this problem, Fisher [31] proposed a balanced function that maximizes separation between classes while minimizing variance within each class:

$$J(w) = \frac{w^T S_B w}{w^T S_I w}, \tag{2.6}$$

where S_B is the between-class covariance matrix given by:

$$S_B = (x_{\bar{c}} - x_c)(x_{\bar{c}} - x_c)^T, \tag{2.7}$$

and S_I is the inner-class covariance matrix given by:

$$S_I = \sum_{i \in c} (x_i - x_c)(x_i - x_c)^T + \sum_{j \in \bar{c}} (x_j - x_{\bar{c}})(x_j - x_{\bar{c}})^T. \tag{2.8}$$

The weight vector w that makes the optimized projection is the w that maximizes the $J(w)$ function in Formula 4.14, i.e.,:

$$w = \arg\max_{w} J(w) = \arg\max_{w} \frac{w^T S_B w}{w^T S_I w}. \tag{2.9}$$

To calculate the weight vector w and deal with the small sample size problem [32] in the training data set, following the similar idea in [33], we perform the singular value decomposition of S_I and have:

$$S_I = U \Sigma V^T, \tag{2.10}$$

where U and V are d-by-d orthogonal matrices and Σ is a d-by-d diagonal matrix. Let $V = [v_1, ..., v_r, v_{r+1}, ..., v_d]$, where r is the rank of S_I. Since S_I is a singular matrix, r is smaller than the dimensionality of the original space, i.e., $r < d$. Therefore, there must be a kernel $\mathcal{K}$ of S_I, where $\mathcal{K}$ is the null space of S_I and is a linear span of a set of vectors $\{x_k | S_I x_k = 0, 1 \leq k \leq (d - r)\}$. Let matrix Q be $[v_{r+1}, ..., v_d]$. Since the kernel $\mathcal{K}$ can be spanned by vectors $v_{r+1}, ..., v_d$ [34], the matrix QQ^T can be used when transforming samples from the original space to kernel. Let $\tilde{S_B}$ denote the scatter matrix of S_B and define:

$$\tilde{S_B} = QQ^T S_B (QQ^T)^T. \tag{2.11}$$

The weight vector w can be calculated as the eigenvector corresponding to the largest eigenvalues of scatter matrix of $\tilde{S_B}$.

2.3 Association Rule Mining

Association Rule Mining is a fundamental data mining method on identifying co-occurrence relationships, called associations, in large data sets. The problem of mining association rules can be presented as follows [35]. Let's take market basket analysis as an example. In a supermarket, customers purchase items. Shopping details of each purchase are recorded as transactions at cashiers. Let $I = \{i_1, i_2, ..., i_m\}$ denote a set of items. Let $T = (t_1, t_2, ..., t_n)$ be a set of transactions. An association rule is an implication of the rule $X \rightarrow Y$, where $X \subset I, Y \subset I$, and $X \cap Y = \varnothing$.

Apriori [36] is one of classic algorithms for learning association rules. The Apriori algorithm which relies on the downward closure [35] has two steps. First, it generate a frequent item set that has frequency in transactions above minimum threshold, say θ. Second, associate item associations are generated with co-occurrence frequency above the threshold θ based on the frequent item set generated in the first step. Details of the Apriori algorithm are presented in Algorithm 1.

Algorithm 1 Apriori Algorithm

```
1: F ← ∅;                                          ▷ initialized to be empty set
2: F_1 ← {f | f = i ∈ I and i.count ⩾ θ};          ▷ generate a frequent item set
3: for (k = 2; F_{k−1} ≠ ∅; k++) do
4:     F_cand ← ∅;                                 ▷ initialized to be empty set
5:     F_k ← ∅;                                    ▷ initialized to be empty set
6:     for all f, f' ∈ F_{k−1} do
7:         with f = {i_1, ..., i_{k−2}, i_{k−1}}
8:         and f' = {i_1, ..., i_{k−2}, i'_{k−1}}
9:         f_cand ← {i_1, ..., i_{k−2}, i_{k−1}, i'_{k−1}}
10:        if f_cand.count ⩾ θ then
11:            F_k ← F_k ∪ {f_cand};
12:        end if
13:    end for
14: end for
15: return F ← ∪_k F_k;        ▷ return the generated feature set F as a union of all the
    generated F_k
```

2.4 Statistical Topic Models

A topic model is a type of statistical model for discovering the abstract "topics" that occur in a collection of documents[7]. The idea of a probabilistic topic model is that documents are mixtures of topics, where a topic is semantically coherent and is formally treated as a probability distribution of words. In a probabilistic topic model, a document d is deemed to be generated by a mixture of topics. To generate each word w in d, a latent topic z is chosen with a probability and w is considered to be generated from a topic-specific multinomial distribution θ_z over words.

Probabilistic Latent Semantic Analysis (PLSA) is one of well known topic model developed by Thomas Hofmann in 1999 [37]. In PLSA documents are considered being made up of a mixture of latent topics from a topic set $Z = \{z_1, z_2, ..., z_K\}$ and model the whole document collection D as:

$$P(D) = \prod_{d \in D} p(d, w) = \prod_{d \in D} \prod_{w \in d} P(d)P(w|d) = \prod_{d \in D} \prod_{w \in d} P(d) \sum_{z \in Z} P(w|z)P(z|d). \quad (2.12)$$

Formula 2.12 models the process of generating each word w in D in a natural way. When “authors" are writing each document d, they first chose some specific topic with probability $P(z|d)$ and then choose a word from the topic z with probability $P(w|z)$. The number of parameters in the Formula 2.12 is $K|D| + K|V|$, where K is the number of latent topics, and $|D|$ is the number of documents in D, and $|V|$ is the number of words in the vocabulary set V. These parameters can be learned using the EM algorithm [37, 38].

[7] http://en.wikipedia.org/wiki/Topic_model

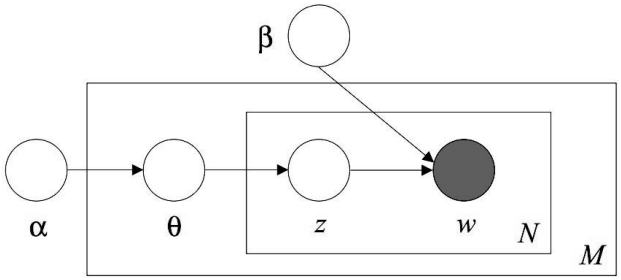


Figure 2.4: Graphical Model Representation of LDA.

Latent Dirichlet Allocation (LDA) is a generative probabilistic topic model and is proposed by Blei et al. in 2002 [39]. As described by Fig. 2.4 [39], the LDA is a three-level hierarchical Bayesian model, in which each document is modeled as a finite mixture over a set of latent topics. Each topic is modeled as infinite mixture over an underlying set of topic probabilities. The main contribution of LDA compared with the PLSA is that the authors introduce the Dirichlet distribution Dir(α) to model the parameters of topic distribution of documents. Different from the generation process of PLSA, the LDA assumes the following generative process for each document w in D. First, the parameter θ is chosen according to the Dirichlet distribution Dir(α). Then a topic z is chose from a multinomial distribution parameterized by θ. Finally, within the topic z a word w is chosen from a multinomial probability defined by a Dirichlet distribution Dir(β). The LDA model of the above generative process for describing a training document collection D is presented as:

$$P(D|\alpha,\beta) = \prod_{d\in D} \int P(\theta_d|\alpha)(\prod_{i=1}^{N_d} \sum_{z_{di}} P(z_{di}|\theta_d)P(w_{di}|z_{di},\beta))d\theta_d, \tag{2.13}$$

where α and β are super parameters. θ_d is the multinomial distribution that describes topic distributions for the document d. N_d is the total number of words in a document d. w_{di} represents the ith word of the document d. z_{di} represents the topic that generate the word w_{di}. The LDA model is a complete Bayesian model since it introduces Dirichlet distributions Dir(α) and Dir(β) to respectively model the parameters $P(z|d)$ and $P(w|z)$ of PLSA as random variables. In this way, the number of parameters to be estimated for a LDA model is $K + K|V|$. These parameters can be estimated by an efficient approximate inference technique based on variational methods and an EM algorithm [39].

Chapter 3

State of the Art

In this chapter, we give an overview of the state-of-the-art work that is related to research of this thesis. The related work is categorized into two sections that are respectively related to the two main problems studied in this thesis. Section 3.1 reviews the related work to modeling impact of online news. Section 3.2 discusses previous work on opinion mining and sentiment analysis.

3.1 Impact Modeling on News

With continuous growth of internet, huge amounts of news data becomes available on online medias. Research on online news analysis has been widely studied in the research communities of information retrieval and text mining. Unlike traditional search technologies that returns a ranked list of documents whose content is highly related to a set of user-supplied keywords, recent research relies on Topic Detection and Tracking (TDT) technologies to model and analyze news topic trend and impact [40].

3.1.1 News Impact Tracking and Modeling

Mori et al. [41] proposed a technique for topic-tracking from Web pages obtained by search engines. The technique relies on a KeyGraph to form concepts and topics with a collection of frequent words clustered together. Yang et al. [42] applied hierarchical and non-hierarchical document clustering algorithms to a corpus of 15,836 stories, focusing on the exploitation of both content and temporal information to automatically detect novel events from a temporally-ordered stream of news stories. Kumaran and Allan [43] use named entities as well as text classification techniques to improve performance of the new event detection task. Leskovec et al. [44] developed a framework for tracking short, distinctive phrases that travel relatively intact through online text and presented scalable algorithms for identifying and clustering textual variants of such phrases that scale to

a collection of 90 million articles, which offers quantitative analysis of the global news cycle and the dynamics of information propagation between mainstream and social media. Wang et al. [45] proposed an automatic online news topic ranking algorithm based on inconsistency analysis between media focus and user attention. Although existing work investigated on the topic detection and tracking problem in various way, none of them formulates a model that quantifies the impact of topics of news articles.

The first challenge in our research on news impact modeling is to develop a model that quantifies the impact of topics of news articles. Therefore different with existing TDT works [41, 42, 43, 44, 45] which focus on detecting and tracking topics of news articles, our work focus on the challenge of modeling impact of user-interested news topics and enables users to navigate among selected news topics to analyze and reveal news impact in an explorative manner.

3.1.2 News Relatedness Calculation

The second challenge in our research on news impact analysis is that in our proposed topic driven model we need to develop an approach to calculate impact relation between news articles based on proportion of one article's contents occupying the other article. A straightforward approach is to approximate the impact relation between news articles using document similarity measurement. Document similarity measurements calculate similarities between documents to indicate their relatedness and are usually employed in text classification [46] and text clustering [47] tasks. A simple but effective approach to measuring similarity between documents uses the vector space model, e.g., Cosine Similarity[1]. However, vector space model measures similarity between documents by treating each whole document as a vector. Without focusing on any semantic aspect, documents that are highly ranked as similar according to the vector space model might not necessarily relate to each other on the required semantic topic. Wang and Taylor [48] proposed to measure semantic similarities of documents based on concept forests that are generated with the assistance of a natural language ontology. In that work, document similarities are measured based on semantic concepts. However, the generation of concepts depends on the availability of an external ontology, while the semi-supervised topic model proposed in our work does not require knowledge from external resources. The proposed concept-tree-distance based document similarity [49] is a semantic concept based measurement without external knowledge. However, the approach proposed in [49] can neither be guided to focus on semantic topics nor work for the documents that might relate to more than one concerned topics.

[1] http://en.wikipedia.org/wiki/Cosine_similarity

3.1.3 Model Parameter Estimation

The proposed semi-supervised topic model in the paper P1 is based on probabilistic topic models, e.g., PLSA [37, 50], LDA [39], which have been proven effective for discovering latent semantic topics through modeling large collections of texts and have already been reported with promising performances on information retrieval [51], summarization [52, 15], clustering [53], classification [54], as well as various web intelligence tasks [6, 55, 56, 57, 58]. In our research, we adopt a topic model to model the news impact. The Maximum Likelihood (ML) estimator with the Expectation Maximization (EM) algorithm [59] is usually used for estimation of this kind of topic models. However, since this parameter estimation process is conducted in an unsupervised setting without any prior knowledge on the trained topics, the generated topic models will probably not be well agreed with required topics. Therefore, there is a challenge on how we can guide the news impact model training process to enforce the generated topic models to seemly represent the required topics.

3.2 Sentiment Analysis on Reviews

Sentiment analysis is a key problem studied under the research field of opinion mining which is at the crossroads of information retrieval and computational linguistics. There have already been a lot of research works that are dedicated to solving this problem. The task of sentiment analysis on reviews was originally performed to extract overall sentiment from the target texts, i.e., document overall sentiment classification. However, in [2], as the difficulty shown in the experiments, the whole sentiment of a document is not necessarily the sum of its parts. Then there came up with research works shifting focus from overall document sentiment to sentiment analysis based on product attributes [4, 60, 61, 5], i.e., attributed-based sentiment analysis. In our research on sentiment analysis on reviews, we investigate both on document overall sentiment classification and attribute-based sentiment analysis.

3.2.1 Document Overall Sentiment Analysis

Document overall sentiment analysis is to summarize the overall sentiment in the document. There have already been a lot of research works that are dedicated to solving this problem. With different grouping criterion existing research works can be grouped into different categories.

According to different granularity levels on which sentiment classification is to be analyzed, existing papers will mainly fall into two categories: *word-level sentiment classification* and *phrase-level sentiment classification*. The *word-level sentiment classification* is to utilize the polarity annotation of words in each sentence and summarize the overall sentiment of each sentiment-bearing word to infer the overall sentiment within the

text [1, 62, 3, 63, 64, 65, 8, 66]. The *phrase-level sentiment classification* focuses sentiment annotation on phrases not words with concerning that atomic units of expression is not individual words but rather appraisal groups [9]. In [67], the concepts of *prior polarity* and *contextual polarity* were proposed. This paper presented a system that is able to automatically identify the *contextual polarity* for a large subset of sentiment expressions. In [2], an unsupervised learning algorithm was proposed to classify reviews as recommended or not recommended by averaging sentiment annotation of phrases in reviews that contain adjectives or adverbs. However, the performances of these works are not good enough for sentiment analysis on product reviews, where sentiment on each attribute of a product could be so complicated that it is unable to be expressed by overall document sentiment.

According to techniques that sentiment classification mainly utilize, existing papers can be roughly grouped into rule-based sentiment classification and machine learning sentiment classification. Rule-based methods for sentiment classification is to develop a certain of rules based on which sentiment information can be extracted from texts. In [62], a rule-based algorithm was presented for extracting sentiment-bearing adjectives from WordNet[2]. In [61], the authors proposed to use some linguistic rules to deal with the sentiment classification problem together with a new opinion aggregation function. Machine learning sentiment classification is to utilize traditional machine learning techniques to classify texts by therein sentiment. In [10], it is found that the three employed machine learning methods did not perform as well on sentiment classification as on traditional topic-based categorization. In [12], the relationship between opinion detection and sentiment classification was examined. In that paper, text categorization technique was applied to extract subjective portions of text from documents. In [9], Whitelaw et al. proposed a Naive Bayes version of Turney's model and provided a framework that enabled human-provided information to be with unlabeled and labeled documents.

Early work, e.g., [64, 1], relies on adjectives to automatically decide sentiment orientation of documents. Pang et al. studied sentiment classification using machine learning techniques [10]. Although it has been claimed that standard machine learning techniques outperform human-produced baselines, there is also evidence that machine learning techniques do not perform as well on sentiment classification as on traditional topic-based text classification [10].

There exist at least two challenges for document overall sentiment classification using machine learning techniques. First, sentiment orientation of words is rather dependent on topics. Although there are some general applicable sentiment expression words, e.g., "good" and "bad", which always hold consistent sentiment orientation in different topics, it is also not difficult to find words that might have different and even opposite opinions in different contexts. Second, negation expression is another potential problem for machine learning classifiers in sentiment classification. Negation words including "not", "never", "no", etc., are considered meaningless stop words and usually filtered out from feature set in traditional topic-based text classification. However, these negation words are very im-

[2]http://wordnet.princeton.edu/

portant signals in the sentiment classification process since their existence might overturn the classification results. Therefore, our research on document overall sentiment analysis in the thesis focus on solve the above two challenges.

3.2.2 Attributed-based Sentiment Analysis

Attributes-based sentiment analysis is to analyze sentiment based on each attribute of a product. In [4], mining product features was proposed together with sentiment polarity annotation for each opinion sentence. In that work, sentiment analysis was performed on product attributes level. In [5], a system with framework for analyzing and comparing consumer opinions of competing products was proposed. The system made users be able to clearly see the strengths and weaknesses of each product in the minds of consumers in terms of various product features. In [60], Popescu and Etzioni not only analyzed polarity of opinions regarding product features but also ranked opinions based on their strength. In [68], Liu et al. proposed Sentiment-PLSA that analyzed blog entries and viewed them as a document generated by a number of hidden sentiment factors. These sentiment factors may also be factors based on product attributes. In [6], Lu et al. proposed a semi-supervised topic models to solve the problem of opinion integration based on the topic of a product's attributes. The work in [69] presented a multi-grain topic model for extracting the ratable attributes from product reviews. In [15], the problem of rated attributes summary was studied with a goal of generating ratings for major aspects so that a user could gain different perspectives towards a target entity. A special case of the attribute-based sentiment analysis is ontology-based sentiment analysis where ontology structure serves as external knowledge to organize a product's attributes [70, 71, 72]. The usage of ontology for opinion mining was firstly studied in [70]. It is reported in [70] that ontology-supported opinion mining outperforms methods without ontology support. Although the method proposed in [70] involved ontology to tackle the sentiment analysis problem, it ignored dependencies among attributes within an ontology's hierarchy.

Although all the above mentioned research work concentrated on attribute-based sentiment analysis, they did not sufficiently utilize the hierarchical relationships among a product attributes. When we look into the details of each example of product reviews, we find that there are some challenges and properties there we need to address. First of all, how can we utilize the domain-specific knowledge of product reviews. The product's attributes mentioned in reviews might have some relationships between each other. For example, for a digital camera, comments on image quality are usually mentioned. However, a sentence like "40D handles noise very well up to ISO 800", also refers to image quality of the camera 40D. Here we say "noise" is a sub-attribute factor of "image quality". Second, sentiments expressed in a review or even in a sentence might be opposite on different attributes and not every attributes mentioned are with sentiments. How can we deal with the complex sentiments expressed in one review? These challenges are focus of our research on attributes-based sentiment analysis.

3.3 Summary

In this section, we review the related work of our research and discuss some important issues and challenges in news impact and sentiment analysis. Specifically, on news impact analysis our research focus on the following issues: 1) developing a model that quantifies the impact of topics of news articles; 2) developing an approach to calculate topic impact relation between news articles; 3) developing a mechanism to guide the news impact model training process so that the generated topic models seemly represent required topics. On sentiment analysis, our research lies in both document overall sentiment classification and attributed-based sentiment analysis. On the task of document overall sentiment classification, there are following two challenges: 1) sentiment orientation of words is rather dependent on topics; 2) negation words are very important and might overturn the classification results. On the task of attribute-based sentiment analysis, we address the following questions: 1) how can we utilize the domain-specific knowledge of product reviews; 2) how can we deal with the complex sentiments expressed in one review.

Chapter 4

Research Results and Evaluation

This chapter presents a summary of the research results of this thesis. Each research result is a contribution to each research questions discussed in Section 1.4. To claim each research result, we first revisit its correspondent research question and describe the details and evaluation of the proposed approach. This chapter is structured in the order that corresponds to research contributions discussed in Section 1.6. Specifically, in Section 4.1 we present an overview of an explorative visual analysis system, called ImpactWheel, that enables users to navigate and analyze large news corpora with rich topic contexts. Section 4.2 summarizes an algorithm that generate complex features for sentiment classifiers. In Section 4.3, an effective feature search framework is briefly introduced that search for effective features for sentiment classifiers. Section 4.4 briefly presents a hierarchical learning framework, called HL-SOT approach, that deals with sentiment analysis tasks using knowledge from ontology structure. In Section 4.5, a localized feature selection framework is briefly discussed for performance enhancement of the HL-SOT approach. Finally, Section 4.6 gives an overview of a hybrid hierarchical classification process that is a further development based on the HL-SOT approach.

4.1 An Explorative Visual Analysis System

The first contribution of this thesis is:

C1: ImpactWheel, an explorative visual analysis system that can reveal the impact of news articles.

In the paper P1, we propose ImpactWheel, a new visual analysis technique that enables users to navigate and analyze large news corpora with rich topic contexts. This contribution is for the first research question:

RQ1: How can we detect the impact of a company through mining news collections and provide users an intuitive interaction to understand the affection of a spec-

ified event and its epidemic?

In the work of the paper P1, we take New York Times industry news as the example corpus to illustrate the power of the ImpactWheel system. Fig. 4.1 shows an overview of the ImpactWheel system. From the Fig. 4.1, we can see that the ImpactWheel as an explorative visual analysis system contains two fundamental components:

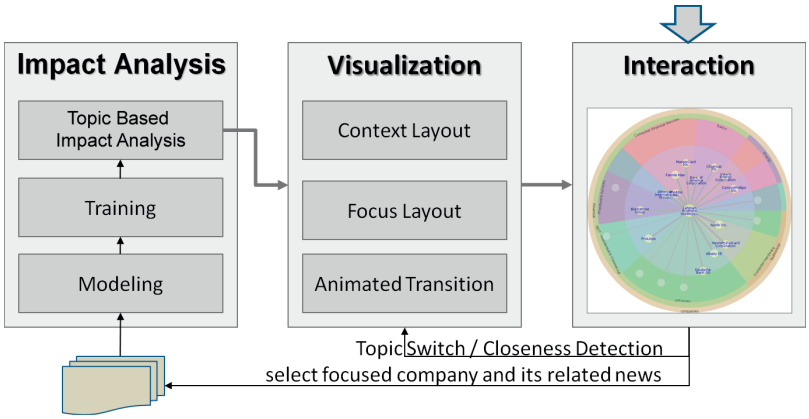


Figure 4.1: System Overview of ImpactWheel

- **Topic Driven Impact Analysis:** Given a news topic of a specified company u, ImpactWheel system provides a relation ranking mechanism that helps to find a set of companies that are deemed as most impacted by the topic of u's news.
- **Explorative Visualization:** ImpactWheel provides a new visualization design that helps to better portray the relation ranking results and facilitate data understanding. Rich interactions are also provided that enables explorative analysis. It helps to explore the analysis results in a dynamic and efficient way and also helps to detect data patterns from rich context.

Generally speaking, with a user-interested company and a set of interested news selected by the user, we want to find the company's impact on other companies based on its various news topics. To achieve this goal, the analysis in the ImpactWheel system contains three major steps as illustrated in Figure 4.1. First, a user selects an interested company and a set of interested news form the data corpus. After that, the topic driven impact analysis component calculates and ranks out a set of key companies that are most impacted by the user-interested company on all the topics of its news. Finally the analytic results are transformed into the visual form and represented by a well designed rich context visualization. Interactions are also provided to help users to explore the data on the visualization display and detect impact relations topic by topic. We describe the details of the two key components in the following sub sections.

4.1.1 Topic Driven Impact Analysis

Topic driven impact analysis provides a ranking mechanism that finds a set of companies that are deemed as most impacted by topics of news of a user-interested company. In this section, we first discuss topic driven news impact modeling that quantifies the impact of topic of a news article. Then we present how to estimate the parameters to calculate the modeled impact with a semi-supervised probabilistic topic model.

Topic Driven News Impact Modeling

Suppose we have a set of news articles in $D_u = \{d_{u,1}, d_{u,2}, ..., d_{u,k}\}$ of a user-interested company u. Let $D_c = \{d_{c,1}, d_{c,2}, ..., d_{c,n}\}$ denote a set of news articles of a company c. The topic driven impact analysis is to calculate relations between the company c and the company u according to how much c is impacted by u on each topic of u's news. We believe that if c is impacted by u on a topic there will exist some news of c mentioning the topic. It is reasonable to assume that the more c is impacted by a topic, the more proportion of the topic occupying the content of c's news. Hence, we model the impact of u's news $d_{u,j}$ $(1 \leq j \leq k)$ on the company c as how much proportion the topic of $d_{u,j}$ is mentioned in c's news:

$$\Gamma_{d_{u,j}}(c) = \sum_{d \in D_c} \rho_d(\tau(d_{u,j})), \tag{4.1}$$

where $\Gamma_{d_{c_u}}(c)$ denotes how much the news $d_{u,j}$ impacts the company c, and $\tau(d_{u,j})$ represents the topic of the news $d_{u,j}$, and $\rho_d(\tau(d_{u,j}))$ represents the proportion of the topic $\tau(d_{u,j})$ occupying the content of d. In order to calculate the value of $\Gamma_{d_{c_u}}(c)$, the news topic $\tau(d_{u,j})$ and the topic proportion $\rho_d(\tau(d_{u,j}))$ modeled in the Formula 4.1 need to be further defined. In the following of this section, we will describe a generation process of a probabilistic topic model in which $\tau(d_{u,j})$ and $\rho_d(\tau(d_{u,j}))$ can be naturally quantified.

The idea of a probabilistic topic model is that documents are mixtures of topics, where a topic is semantically coherent and is formally treated as a probability distribution of words. In a probabilistic topic model, each news topic $\tau(d_{u,j})$ can be modeled as a semantically coherent topic which is described by a multinomial distribution of words by a topic model θ_j: $\{p(w|\theta_j)\}_{w \in V}$. For all the words w in vocabulary V, θ_j is subject to the constraint: $\sum_{w \in V} p(w|\theta_j) = 1$. Let $\Theta_I = \{\theta_1, \theta_2, ..., \theta_k\}$ respectively denote topics[1] of k news articles in D_u. Each news d of a company c is deemed to be generated in a probabilistic sampling process in which each word w of d can be deemed to be generated from a mixture of topics in Θ_I. Intuitively, there might exist such a situation that no topic in Θ_I is covered by a news article. In order to smooth this case, we further define a general background model θ_B to model the common English words as well as contents that are not related to any topics in Θ_I. Let Θ denote the set of all the topics, i.e., $\Theta = \Theta_I \bigcup \{\theta_B\}$. In the probabilistic sampling process, when writing a word w of a news article d the "author" might make the following two stochastic decisions:

[1] If specified otherwise in the following of this paper "topic" has the same meaning with "topic model".

- the “author" might decide to use a word from a topic $\theta_j \in \Theta$ with probability $p(\theta_j|d)$;
- the “author" might choose a word w from the topic θ_j with probability $p(w|\theta_j)$.

Let $C = \{c_1, c_2, ..., c_m\}$ denote all the companies to be analyzed. Let D denote the set of all the news articles of companies in C: $D = \bigcup_{c \in C} D_c$. With the probabilistic generation process described above, the log likelihood of documents in D can be formally given by:

$$\begin{aligned} log(p(D)) = \sum_{D_c \in D} \sum_{d \in D_c} \sum_{w \in V} [f(w:d) \\ \times log(\sum_{\theta_j \in \Theta} p(\theta_j|d) \times p(w|\theta_j))], \end{aligned} \tag{4.2}$$

where $f(w:d)$ is the frequency of word w appearing in the document d.

In the probabilistic topic model described in Formula 4.2, the news topic $\tau(d_{u,j})$ is modeled as θ_j and the topic proportion $\rho_d(\tau(d_{u,j}))$ can be naturally quantified $p(\theta_j|d)$. In this way, the modeled impact of the news $d_{u,j}$ on the company c described in the Formula 4.1 can be calculated as:

$$\Gamma_{d_{u,j}}(c) = \sum_{d \in D_c} \rho_d(\tau(d_{u,j})) = \sum_{d \in D_c} p(\theta_j|d). \tag{4.3}$$

In next subsection, we will present a semi-supervised model estimation process to estimate the parameters $p(w|\theta_j)$ and $p(\theta_j|d)$ in the described probabilistic topic model.

Model Estimation in a Semi-supervised Setting

The parameters $p(w|\theta_j)$ ($\forall \theta_j \in \Theta$) and $p(\theta_j|d)$ ($\forall d \in D$) are estimated in the way that the probabilistic topic model can best explain the text data in D. The Maximum Likelihood (ML) estimator with the Expectation Maximization (EM) algorithm [59] is usually used for estimation of this kind of topic models. However, since this parameter estimation process is conducted in an unsupervised setting without any prior knowledge on the trained topics, the generated topic models will probably not be well agreed with topics in Θ_I. In order to guide the parameter estimation process to enforce the generated topic models to seemly represent topics in Θ_I, prior knowledge on those topics should be obtained. Then the prior knowledge can be incorporated with the ML estimator to make the estimation process with the Maximum A Posterior (MAP) estimator.

Different from existing works [56, 6] where the prior knowledge is collected from external resources, the prior knowledge on topics in Θ_I can be acquired from the company u's news set D_u. In D_u, the content of each news article $d_{u,j}$ tells us about what the topic θ_j is like. Analogically, all the news in D_u can also be deemed to be generated in a similar probabilistic sampling process as described above. In the same way, we model the log

likelihood of documents in D_u as:

$$\begin{aligned} log(p(D_u)) = \sum_{d_{u,i} \in D_{c_u}} \sum_{w \in V} [f(w : d_{u,i}) \\ \times log(\sum_{\theta_j \in \Theta} p(\theta_j | d_{u,i}) \times p(w|\theta_j))]. \end{aligned} \tag{4.4}$$

We use the ML estimator with the EM algorithm [59] with imposing constraints:

$$p(\theta_j | d_{u,i}) = \begin{cases} 1, & \text{if } j = i \\ 0, & \text{otherwise} \end{cases}$$

to estimate words distribution for each topic θ_j and denote each estimated distribution by $\hat{\Theta}_I = \{\hat{\theta}_1, \hat{\theta}_2, ..., \hat{\theta}_k\}$. In order to incorporate this prior knowledge, each $\hat{\theta}_j \in \hat{\Theta}_I$ is defined as a conjugate Dirichlet prior respectively for each topic model $\theta_j \in \Theta_I$ so that the topic model θ_j is a multinomial distribution parameterized by $Dir(1 + \mu_j p(w|\hat{\theta}_j))$, where μ_j represents a confidence parameter for the prior. Here μ_j can be deemed as the "equivalent sample size" which means that the effect of adding the prior would be equivalent to add $\mu_j p(w|\hat{\theta}_j)$ pseudo counts for word w for estimation of $p(w|\theta_j)$. Let Λ denote all the parameters $p(w|\theta_j)$ ($\forall \theta_j \in \Theta_I$) to be estimated. In the estimation process with the MAP estimator, the prior of Λ can be given by

$$p(\Lambda) \propto \prod_{j=1}^{k} \prod_{w \in V} p(w|\theta_j)^{\mu_j p(w|\hat{\theta}_j)}. \tag{4.5}$$

With the prior described in the Formula 4.5, Λ can be estimated with the MAP estimator, i.e., $\hat{\Lambda} = \arg\max_{\Lambda} p(D|\Lambda)p(\Lambda)$ by the EM algorithm. The updating formulas in EM steps are:

$$p(w|\theta_j, d) = \frac{p^{(n)}(\theta_j|d)p^{(n)}(w|\theta_j)}{\sum_{\theta_{j'} \in \Theta} p^{(n)}(\theta_{j'}|d)p^{(n)}(w|\theta_{j'})}; \tag{4.6}$$

$$p^{(n+1)}(\theta_j|d) = \frac{\sum_{w \in V} f(w : d)p(w|\theta_j, d)}{\sum_{\theta'_j \in \Theta} \sum_{w \in V} f(w : d)p(w|\theta_{j'}, d)}; \tag{4.7}$$

$$p^{(n+1)}(w|\theta_B) = \frac{\sum_{D_c \in D} \sum_{d \in D_c} f(w : d)}{\sum_{w' \in V} \sum_{D_c \in D} \sum_{d \in D_c} f(w' : d)}; \tag{4.8}$$

$$\begin{aligned} &\forall \theta_j \in \Theta_I : \quad p^{(n+1)}(w|\theta_j) = \\ &\frac{\sum_{D_c \in D} \sum_{d \in D_c} f(w : d)p(w|\theta_j, d) + \mu_j p(w|\hat{\theta}_j)}{\sum_{w' \in V} \sum_{D_c \in D} \sum_{d \in D_c} f(w : d)p(w'|\theta_j, d) + \mu_j}. \end{aligned} \tag{4.9}$$

With the above updating formulas, all the parameters are able to be estimated so that the estimated model can best fit to the text data set D. Then the modeled impact of the news $d_{u,j}$ on the company c can be calculated by the Formula 4.3.

4.1.2 Visual Analysis

In this subsection, we describe how to visualize the above analytic results to help users understand the data and find topic-based impact relation changes.

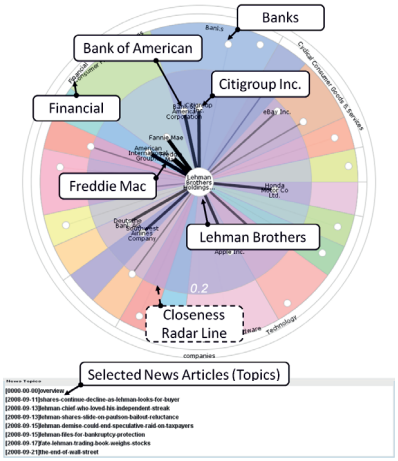


Figure 4.2: Interactive rich context visualization

Design Principles

Generally speaking, we represent the above analytic results using rich context visualization as illustrated in Figure 4.2. A screen capture demo is also available online.[2] More specifically, the design of the visualization follows several key design principles.

Focus + Context In our design, companies are encoded by circular nodes and treated as the information focus. The focused company is the user-interested company c_u that is laid out in the center surrounded by other related companies. Both lines and positions are used to encode the relation rankings. Context information is also considered in the design. A radial diagram in the background depicts the sector-industry hierarchy for each company. It uses colorful wedges and pie slices to identify different sectors and industries respectively. For example, in Figure 4.2, “Lehman Brothers" is the focused company which is depicted at the center of the view. “Freddie Mac" has a higher relation ranking to “Lehman Brothers" than any other companies. “Bank of American" and “Citigroup Inc." belong to “Banks" industry under the “Financial" sector since they are shown in the region of “Banks".

[2] http://www.youtube.com/watch?v=OQTEp5dUCr4

Overview + Detail With each news of the focused company, we are able to rank the relations between the companies based on news topics. Several star-like ranking graphs are generated by connecting them with the focused company according to their relation rankings. Thus, there are k ranking graphs if we have k news topics. These graphs reveal the differences of the relations ranked by different topics. An overview graph is computed by aggregating various relations together to provide an overview. All the relation details and overviews are encoded in the visualization in a uniform way. Only one type of relation is shown at a time. Users can switch between different topics to view different relations.

Rich Interaction To facilitate data exploration and visual analysis, ImpactWheel also provides a set of interactions. Besides some intuitive interactions like highlight, there are two important interactions: *Topic Switch.* By default, in the ImpactWheel visualization, the overview graph is shown. Users are able to switch between different relations by selecting the news topic listed at the bottom of the view. An animated transition is applied to depict the data changes in a smooth way. This interaction helps users to navigate and compare the relations of different topics.
Closeness Detection. Clicking on the focused company in the center will draw a circular radar line to show the rank. All the companies that are covered inside the line are highlighted (see Figure 4.2). Keep on clicking, the radii of the circle continues to increase and cover more nodes until it reaches the upper limitation. This interaction helps to quantitatively detect the closeness to the focused company in an efficient way.

Layout

The design outlined above introduces several constraints on the visualization layout. The visualization contains two kinds of layout: 1) the context layout in the background shows the industry hierarchies; 2) the focus layout in the foreground shows the relation ranking of companies.

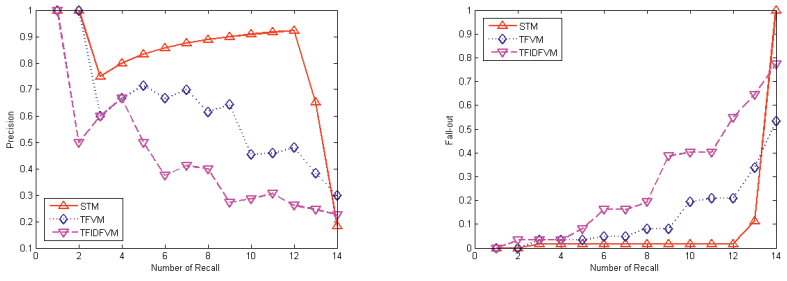
Context Layout The visualization of the ImpactWheel system encodes a hierarchical context in the background. In general, this hierarchical information is laid out based on a space filling radial layout which splits angles to assign space for each node in the hierarchy. More specifically, the root of the hierarchy takes the full angle range which is from 0 to 360 degree. This angle range is assigned to its children according to their weights. The angle splitting process is recursively preformed until it reaches the leave nodes of the hierarchy. The layout algorithm is widely used in many visualization designs such as SunBurst [73]. It provides some obvious advantages on compactness and aesthetics.

Focus Layout The focused star-graph of companies is also carefully laid out within the region of their related background context. To better organize the companies in view, we first order all the surrounding companies in a decreasing order by their ranking closeness

to the focused company. We use the radial layout technique again to put the companies into their related background region according to the above order. During the layout the radii of each company is also adjusted by their ranking closeness to the center. We put companies with higher rank close to the center, and the companies with a lower rank away from the center. In this way a spiral-liked view is automatically generated.

4.1.3 Evaluation

In this section, we conduct experiments for the performance evaluation on the ImpactWheel system on an example corpus collected from the original large data set of New York Times news[3]. On the collected corpus, we manually judge each news document $d \in D_c$ as "impacted" or "not-impacted" by the news d_u. Since this manual labeling process is time-consuming, we can not make this corpus too large. Using "Lehman Brothers" as a user-interested company, which was filed for Chapter 11 bankruptcy protection on Sep. 15, 2008[4], the collected corpus contains 76 pieces of news of 15 companies in the financial sector that occurred in the period from Sep. 11, 2008 to Sep. 18, 2008. Within the 76 news in the collected corpus, 14 news are manually judged as impacted by the topic of a news of Lehman titled "Lehman files for bankruptcy protection". The evaluation is divided on two parts. First, performance evaluation is conducted for topic driven impact analysis model. In addition, we demonstrate the effectiveness of the visualization on results with a case study.



(a) Interpolated Precision (b) Interpolated Fall-Out

Figure 4.3: Results of the Interpolated Precision and Interpolated Fall-out on Approaches

Performance Evaluation

The purpose of the proposed Semi-Supervised Topic Model (STM) approach is to calculate impact relations between each interested news d_u and news of a company c based

[3] http://www.nytimes.com/

[4] http://en.wikipedia.org/wiki/Bankruptcy_of_Lehman_Brothers

on the proportion of the topic of d_u occupying the content of news in D_c. Therefore the performance of impact relation analysis is reflected in the accuracy of the proportion value calculation, which can be approximated with a document similarity measurement. To investigate whether our proposed STM approach provides more effective mechanisms for calculating this proportion value, in the experiments we compare the proposed STM approach with two implemented baseline approaches: term-frequency based vector space model, called TFVM approach, and *tfidf*-based vector space model, called TFIDFVM approach, that approach the proportion value calculation by cosine similarity between d_u and news texts in D_c.

Evaluation Metrics Since it is not feasible to quantify the proportion value of each topic news occupying contents of news articles in test set, we evaluate the accuracy of proportion value calculations focusing on retrieval effectiveness. That is to say, more accuracy of the proportion value calculation will result in more effective ranking list on retrieving news documents that are labeled as "impacted". The retrieval effectiveness is measured by "Interpolated Precision" and "Interpolated Fall-out", which respectively record the precisions and fall-outs as the number of news labeled as "impacted" increases in the retrieved news. The precision here is defined as:

$$\text{precision} = \frac{|\{\text{“impacted" news}\} \bigcap \{\text{top } N \text{ ranked news}\}|}{N}, \tag{4.10}$$

which measures the fraction of news that are judged as "impacted" in the top N ranked news. The fall-out here is defined as:

$$\text{fall-out} = \frac{|\{\text{“not-impacted" news}\} \bigcap \{\text{top } N \text{ ranked news}\}|}{|\{\text{all “not-impacted" news in corpus}\}|}, \tag{4.11}$$

which measures the proportion of "not-impacted" news out of all "not-impacted" news in the corpus are ranked as top N retrieved news.

Experimental Results The experiments are conducted on performance comparison between the proposed STM approach with the TFVM approach and TFIDFVM approach. In the experiments, each approach will generate an impact relation value between each news in D_c and the news of bankruptcy of Lehmen. Ranked by impact relation values, the generated list of news are checked against the human-judged news list to calculate the precisions and fall-outs. The results of interpolated precision and interpolated fall-out of the three approaches are summarized in Fig. 4.3. From Fig. 4.3, we can observe that the proposed STM approach generally beats the TFVM approach and the TFIDFVM approach respectively on the two metrics. Although the STM approach reaches worst, when the number of recall is increased to 14, i.e. 100% recall, as shown in Fig. 4.3a and Fig. 4.3b, it is not difficult to observe that with incorporating prior knowledge on the topics, the STM approach result in better performance than the baseline approaches since more precision and less fall-out are obviously achieved as the number of recall increases.

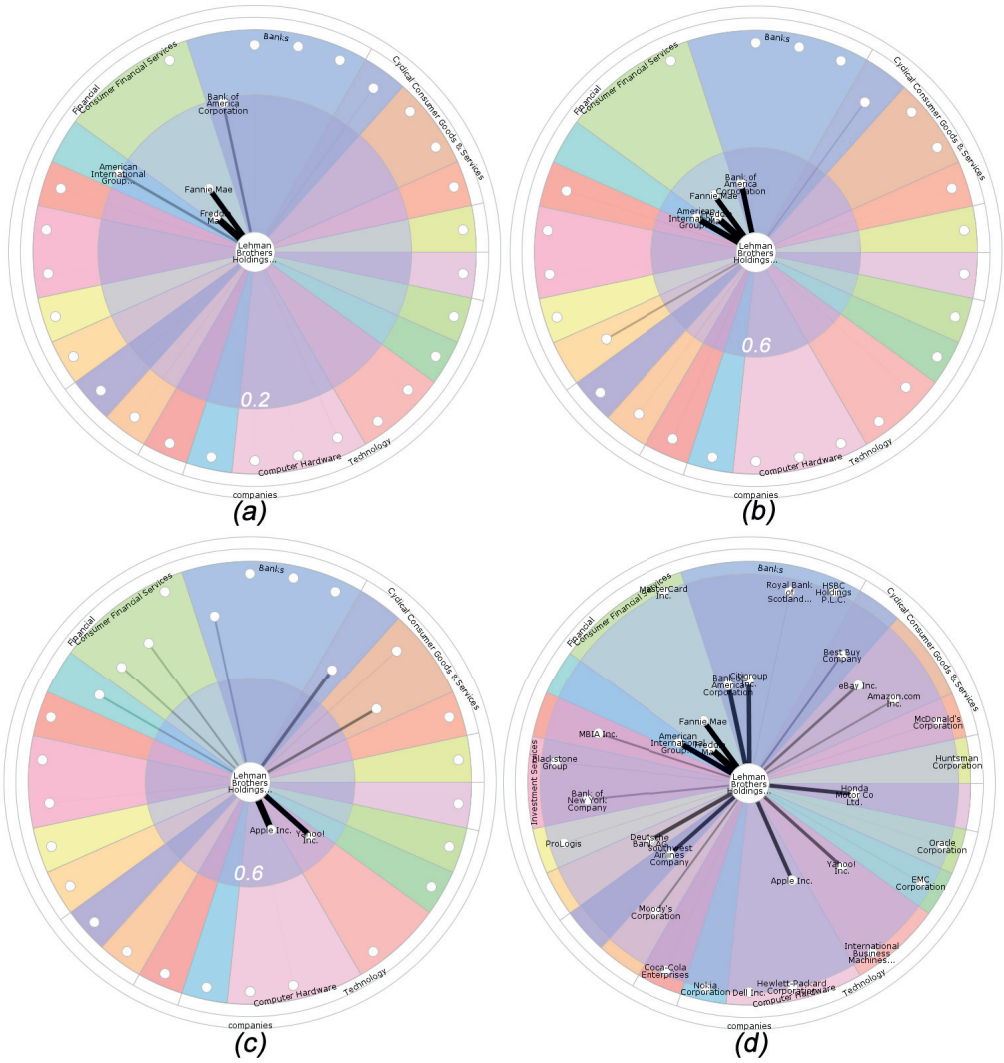


Figure 4.4: Case study on the bankruptcy of Lehmen Brothers.

On the contrary, without focusing on topic semantics the TFVM approach and TFIDFVM approach blindly retrieve more and more trivial news as the number of recall increases and therefore lose performances on the precision and the fall-out.

Case Study

We selected a series of news that occurred in the period from Sep. 11th to Sep. 21st 2008 which focused on “Lehmen Brothers" bankruptcy. We aim to help users better to understand the event’s impact on several major industries. An overview of the impact of all these news is depicted in Figure 4.2. As a whole, the Financial and Technology sectors are the two major sectors that are most affected by the bankruptcy of Lehmen. More specifically, “Freddie Mac" which is the company in the “Consumer Financial Services" industry is most affected by this bankruptcy event. When we explore the news topics one by one as illustrated in Figure 4.4, we find that the impact of the Lehmen bankruptcy is distributed to various industries in several key steps.

The whole story began as the news “Shares continue to decline as Lehmen looks for buyer" was published on Sep. 11th 2008. At this stage, as illustrated in Fig. 4.4(a), only the companies under the “Consumer Financial Services", “Bank" and “Insurance" industries were affected by the news. Until this moment, this early hint of financial disaster had not affected other industries. Two days latter, another news “Lehmen shares slide on paulson bailout reluctance" received more attention from financial companies like “Bank of American" and “Fannie Mae", as illustrated in Fig. 4.4(b). On Sep. 15th 2008, Lehmen announced the bankruptcy protection. This news immediately made great impact on several technique companies like “Apple" and “Yahoo". Some other retail companies such as “Best Buy" and “Amazon" were also affected as illustrated in Fig. 4.4(c). As the impact was distributed, until Sep. 21st, a piece of news with the title of “the end of wall street" generated great impact on most financial companies as illustrated in Figure 4.4(d).

From this study, using the ImpactWheel system, we easily detect the changes of the impact of the Lehmen bankruptcy event.

4.1.4 Summary

In the paper P1, we introduce ImpactWheel, an explorative visual analysis system that can detect the impact of the news articles. The system contains two major components, a topic driven impact analysis model and an interactive rich context visualization design. The experiments of performance evaluation on topic driven impact analysis show that our approach produce more precision and less fall-out on capturing semantics and context topics than other two baselines. A case study on the system demonstrates its powerfulness and effectiveness of visual analysis design. The result of this paper answers questions of RQ1 discussed in the Section 1.4.

My contribution: I was the first author of the paper P1 and did all model development, programming and paper writing work on the topic driven impact analysis. Nan Cao worked for the visual analysis part. Jon Atle Gulla and Huamin Qu gave feedback on the writing and empirical analysis.

4.2 An Automatic Complex Feature Generation Approach

The second contribution of this thesis is:

C2: An automatic approach to generate complex features for sentiment classifiers.

In the paper P2, an approach to generating complex features (called multi-unigram feature in the paper P2) to enhance a negation-aware Naive Bayes classifier for sentiment classification on sentences of product reviews. This contribution is for the second research question:

RQ2: Can we capture high frequent co-occur terms as complex features to improve the accuracy of sentiment classification on product reviews?

In a unigram language model, each unigram presence is usually treated as a feature of a sentiment classifier. In our method, we coin the term “multi-unigram feature" to represent a new kind of features that are generated with capturing high-frequently co-appeared unigrams in the training data. In this section, we will briefly discuss the process of automatically producing multi-unigram features.

4.2.1 Preprocessing for Unigram Feature Generation

For the purpose of generating a set of unigram feature candidates, we first employ an existing POS-tagger [74] to conduct a POS-tagging process on all the training data. According to the POS tags associated with each word, we select words only with interesting POS tags, e.g. verbs, nouns, adjectives, etc. to be unigram feature candidates. Then we perform a stop-word-removing process on the list of unigram feature candidates so that meaningless words, e.g. this, that, will be removed. Furthermore, we transform each word to its stem with the porter stemming algorithm [75] and get a set of stems. We use the set of stems as initial unigram feature candidate set and denote it by F_0.

4.2.2 Multi-Unigram Feature Generation Algorithm

Based on the initial unigram feature candidate set F_0, we propose a Multi-Unigram Feature Generation algorithm to generate a set of multi-unigram features in Algorithm 2. Let $D_{training}$ denote the training data set. F represents the set of features generated by the proposed algorithm. F is initialized to be an empty set. F_k ($k \geqslant 1$) represents the set of features generated in the k^{th} round. It is worth noting that each feature $f_k \in F_k$ generated in the k^{th} round must contain k stems. When k is equal to 1, any unigram feature candidate $f_0 \in F_0$ is selected to be a member of F_1, if $f_0.count$, i.e., the number of times f_0 occurring in $D_{training}$ is equal to or greater than the threshold θ_1. In the k^{th} ($k \geqslant 2$) round, if the generated feature set in the last round is not an empty set, i.e., $F_{k-1} \neq$, the algorithm will continue to generate F_k until the terminal condition is satisfied. In the module of F_k's generation (line 3 - 25 in the Algorithm 2), the candidate feature set F_{cand} and the

Algorithm 2 Multi-Unigram Feature Generation Algorithm

1: $F \leftarrow \emptyset$; ▷ initialized to be empty set
2: $F_1 \leftarrow \{f_0 | f_0 \in F_0, f_0.count \geqslant \theta_1\}$; ▷ select unigram features based on appearance frequency
3: **for** ($k = 2$; $F_{k-1} \neq \emptyset$; k++) **do**
4: $F_{cand} \leftarrow \emptyset$; ▷ initialized to be empty set
5: $F_k \leftarrow \emptyset$; ▷ initialized to be empty set
6: **for** all $f_0 \in F_0$ **do** ▷ feature f_0 contains one stem t_0
7: **for** all $f_{k-1} \in F_{k-1}$ **do** ▷ feature f_{k-1} contains $k-1$ stems $\{t_1, t_2, ..., t_{k-1}\}$
8: with $f_0 = \{t_0\}$ and $f_{k-1} = \{t_1, t_2, ..., t_{k-1}\}$
9: **if** $t_0 \notin f_{k-1}$ **then** ▷ if feature f_{k-1} does not contain stem t_0
10: $f_{cand} \leftarrow \{t_0, t_1, t_2, ..., t_{k-1}\}$; ▷ combine f_0 and f_{k-1} to generate a candidate
11: $F_{cand} \leftarrow F_{cand} \cup \{f_{cand}\}$; ▷ put candidate f_{cand} into set F_{cand}
12: **end if**
13: **end for**
14: **end for**
15: **for** all $f_{cand} \in F_{cand}$ **do**
16: **for** all sentences $d \in D_{training}$ **do**
17: **if** f_{cand} occurs in d **then**
18: f_{cand}.count++; ▷ record f_{cand} occurs in d
19: **end if**
20: **end for**
21: **if** $f_{cand}.count \geqslant \theta_2$ **then** ▷ if the number of occurrence satisfy the threshold
22: $F_k \leftarrow F_k \cup \{f_{cand}\}$; ▷ f_{cand} is promoted to be a feature in F_k
23: **end if**
24: **end for**
25: **end for**
26: **return** $F \leftarrow \cup_k F_k$; ▷ return the generated feature set F as a union of all the generated F_k

generated feature set F_k is initialized to be an empty set. The algorithm firstly generates a set of feature candidates in F_{cand} (line 6 - 14 in the Algorithm 2) and then filters out those candidates that occurs in $D_{training}$ less than a threshold of θ_2 times to generate the feature set F_k (line 15 - 24 in the Algorithm 2). In the module between line 6 and line 14 in the Algorithm 2, each feature f_{k-1} generated in the previous round will be combined with each unigram feature candidate f_0. If the stem t_0 of the unigram feature candidate f_0 is not in the stem set $\{t_1, t_2, ..., t_{k-1}\}$ of the feature f_{k-1}, f_0 and f_{k-1} can be combined into be a new feature candidate and be put into F_{cand}. In the module between line 15 and line 24 in the Algorithm 2, the number of occurrence f_{cand}.count of each candidate feature is recorded. The feature candidate f_{cand} in F_{cand} will become a member of F_k, if f_{cand}.count is equal to or greater than the threshold θ_2. Finally, the returned feature set F is a union of all the generated feature set F_k in each round.

Table 4.1: Performance Comparison

Approaches	Features	Negation-Aware	Parameters		Accuracies in Percent
NB-U	unigrams	no	θ_1= 1	N/A	0.6630
NB-MU	multi-unigrams	no	θ_1= 1	θ_2= 5	0.6850
NANB-U	unigrams	yes	θ_1= 1	N/A	0.7200
NANB-MU	multi-unigrams	yes	θ_1= 1	θ_2= 5	**0.7480**

4.2.3 Evaluation

The proposed approach of our second contribution is evaluated both quantitatively and qualitatively. The evaluation is conducted on a human-labeled data set that contains 700 sentences of customer reviews on digital cameras selected from a customer review website[5].

Quantitative Evaluation on Performance Comparison

In order to show the effectiveness of generated multi-unigram features and negation-aware concept respectively, we implement baseline approaches of NB classifier on Unigram features (denoted by NB-U), NB classifier on Multi-Unigram features (denoted by NB-MU), and Negation-Aware NB classifier on Unigram features (denoted by NANB-U). We compare our proposed Negation-Aware NB classifier on Multi-Unigram features (denoted by NANB-MU) with the three implemented baseline methods. In the experiments, the threshold θ_1 is set to 1 for all the approaches, which means that all the unigram feature candidates in the initial set F_0 are selected as features and are utilized to generate new multi-unigram features. The threshold θ_2 is set to 5 for approaches NB-MU and NANB-MU, which means that for a new generated multi-unigram feature candidate, it will be selected as a feature only if it appears at least 5 times in the training data.

Qualitative Evaluation on Generated Features

Tab. 4.2 presents top ranked generated features according to their frequencies in the training data of “positive" and “negative" classes with our proposed NANB-MU approach in one execution running. From Tab. 4.2, we can see that most top ranked features for respectively “positive" and “negative" classes are reasonable and consistent with human beings intuition. Especially, there are features containing multiple unigram stems, such as <good qualiti build> for “positive" class and <heavi weight> for “negative" class,

[5]http://www.consumerreview.com/

Table 4.2: Top Ranked Features according to Frequencies in Each Class

Class	Top Frequency Features
Positive	<good qualiti build>,<batteri long life>,<excel qualiti imag>, <great len>,<great pictur>,<valu monei>,<good life>,<good grip>,<good pictur>,<high qualiti>, <best camera>,<camera solid>,<feel hold>,<big lcd>,<long life>,<imag fantast>,<imag sharp>, <batteri long>,<great imag>,<built bodi>,<good camera>,<good feel>,<good build>,<feel solid>, <low nois>,<qualiti build>,<work>,<great>,<solid>,<good qualiti>,<good>,<great camera>, <good imag>,<camera like>,<excel>,<nice>,<ergonom>,<best>,<focu fast>
Negative	<heavi weight>,<limit>,<poor>,<problem>,<wait>,<heavi>, <disappoint>,<less>,<paid>,<problem camera>,<bit heavi>,<small>,<disapoint>,<regret>, <plastic>,<hate>,<unaccept>,<creep>,<gear>, <weak>,<flaw>,<shake>,<distort>,<clip>, <late>,<stretch>,<dislik>,<slow speed>,<cost>,<bad>,<dark>,<feel grip>,<nois iso>,<expens>

that won't exist in unigram feature set. These multi-unigram features generated by our proposed algorithm are believed to be pivotal features that benefit the NB classifier.

4.2.4 Summary

In the paper P2, we propose an approach to generating multi-unigram features to enhance a negation-aware Naive Bayes classifier. The term "multi-unigram feature" is coined to represent the process that the generated features are produced by our generation algorithm that takes an initial set of unigram feature candidates as input. We further make the Naive Bayes classifier aware of negation expressions in the training and classification process to eliminate the confusions of the classifier that is caused by negation expressions within sentences. Experiments not only qualitatively show the quality of the generated features but also quantitatively demonstrate that our proposed approach beats other three baseline methods. The result of this paper answers questions of RQ2 discussed in the Section 1.4.

My contribution: I was the first author of the paper P2 and did all the algorithm design, programming and paper writing work. Jon Atle Gulla gave feedback on the writing and empirical analysis. Zhang Fu was in the discussion rounds on algorithm design.

4.3 An Effective Feature Search Framework

The third contribution of this thesis is:

C3: An Effective Feature Search (EFS) framework for enhancing performance of sentiment classifiers.

In the paper P3, we propose an Effective Feature Search (EFS) framework that makes a novel connection between feature candidate generation and a Stochastic Local Search (SLS) process to enhance performance of machine learning classifiers for sentiment classification. This contribution is for the third research question:

RQ3: Can we remove the noise of the generated candidates while keep effective features for sentiment classification?

There are two important steps, i.e., **feature generation step** and **feature pruning step**, in our proposed framework. In the feature generation step, we utilize filter-based methods [16] to select feature candidates not only considering unigram features but also taking complex features into consideration. The feature pruning step is developed and devoted to searching for an optimized feature subset out of the feature candidate set from the feature generation step. The first feature generation step takes similar advantage of high frequency Co-occurring Term (CoT) patterns to generate complex feature candidates. Therefore, we do not discuss complex feature generation process based on the similar rationale as in the Algorithm 2 again. Instead, we will focus on presenting the SLS process in the feature pruning step.

Let F denote the generated feature candidate set from the generation step and F is a union of the unigram feature candidate set F_u and the CoT feature candidate set F_{cot}, i.e., $F = F_u \cup F_{cot}$. F might contain both effective and useless features. The purpose of the feature pruning step is to select a subset of features from F which are considered as useful for sentiment classifiers, while ignoring the rest. In the feature pruning step, we map the feature subset optimization problem to a Stochastic Local Search (SLS) model as follows.

4.3.1 Stochastic Local Search Model

Let $f_i \in F$ $(1 \leq i \leq n)$ respectively represent each feature candidate from F_u and $n = |F_u|$. Let $f_j \in F$ $(n+1 \leq j \leq n+m)$ respectively represent each feature candidate from F_{cot} and $m = |F_{cot}|$. An Stochastic Local Search Model (SLSM) for feature subset optimization is formulated as follows.

An SLSM is formally defined as a 4-tuple $M = (S, \mathcal{N}, \mathcal{G}, \mathcal{O})$ where S is a set of state vectors and forms the search space. Each state vector s $(s \in S)$ represents a feature subset of F_s[6], where s is an $(n+m)$-dimensional binary vector $(s_1, s_2, ..., s_n, s_{n+1}, ..., s_{n+m})$ and each value s_k $(1 \leq k \leq n+m)$ encodes whether the feature f_k $(f_k \in F)$ is selected: 1 means being selected while 0 means the opposite. $\mathcal{N}$ is a neighborhood relation , i.e., $\mathcal{N} \subseteq S \times S$; $\mathcal{G} : S \to \mathbb{R}$ is an evaluation function that maps each state $s (s \in S)$ to a real number score $g (g \in \mathbb{R})$; $\mathcal{O}$ defines optimal states $\mathcal{O} = \{s^* | s^* = \arg_{max}\, \mathcal{G}(s)\}$.

[6] In this paper, when we say a feature subset s it means F_s.

Algorithm 3 SLS Algorithm

Input: c: machine learning classifier
F: feature candidate set
$S_t \leftarrow \emptyset$: initialize an empty tabu list
θ_u: unigram candidates initialization parameter
θ_{cot}: CoT candidates initialization paramete
λ: noise parameter
κ: greedy parameter
R: number of flips per try
MAX-TRIES: number of tries
Output: s^*: optimized state of feature set

$t \leftarrow 1$;
$g^* \leftarrow 0$; ▷ performance score record
while $t \leq$ MAX-TRIES **do**
  $s \leftarrow$ INITIALIZE(θ_u, θ_{cot});
  $g \leftarrow \mathcal{G}_c(s)$; ▷ calculate performance
  if $g \geq g^*$ **then**
    $g^* \leftarrow g$; ▷ record performance
    $s^* \leftarrow s$; ▷ record feature set
  end if
  $S_t \leftarrow S_t \cup \{s\}$; ▷ add to taboo list
  $r \leftarrow 1$;
  while $r \leq R$ **do**
    next$\leftarrow$NEXTSTEP(λ);
    if next is a noise step **then**
      $s \leftarrow$NEIGHBOR(s, S_t);
      $g \leftarrow \mathcal{G}_c(s)$;
      $S_t \leftarrow S_t \cup \{s\}$;
      if $g \geq g^*$ **then**
        $g^* \leftarrow g$;
        $s^* \leftarrow s$;
      end if
    end if
    if next is a greedy step **then**
      $j \leftarrow 1$;
      $g_j^* \leftarrow 0$;
      while $j \leq \kappa$ **do** ▷ try κ neighbors
        $s_j \leftarrow$NEIGHBOR(s, S_t);
        $g_j \leftarrow \mathcal{G}_c(s_j)$;
        $S_t \leftarrow S_t \cup \{s_j\}$;
        if $g_j \geq g_j^*$ **then**
          $g_j^* \leftarrow g_j$;
          $s_j^* \leftarrow s_j$;
        end if
        $j \leftarrow j + 1$;
      end while
      $s \leftarrow s_j^*$; ▷ choose best neighbor
      if $g_j^* \geq g^*$ **then**
        $g_j^* \leftarrow g^*$;
        $s^* \leftarrow s_j^*$;
      end if
    end if
    $r \leftarrow r + 1$;
  end while
  $t \leftarrow t + 1$;
end while
return s^*;

In the above definition, the neighborhood relation $\mathcal{N}$ defines neighboring states of each state $s \in S$. In an SLS algorithm, the search iteratively moves from one state to its neighboring states and scores are calculated by the evaluation function $\mathcal{G}$. Unlike in the feature

generation step, where feature candidate selection is performed in a filter-based manner which ranks each feature candidate separately, in this feature pruning step each feature subset is evaluated as a whole by $\mathcal{G}$. In order that the optimized feature subset is tailored to a particular machine learning classifier c, we adopt a wrapper-based approach and the evaluation function $\mathcal{G}_c$ is a mapping function from a feature subset s to the classifier c's empirical accuracy on s. In the rest of this section, we propose an SLS algorithm that serves as a heuristic approach to feature subset optimization.

4.3.2 Stochastic Local Search Algorithm

Searching for an optimal feature subset is an NP-hard problem. The stochastic local search (SLS) approach has proven to be highly competitive for solving a range of hard computational problems including satisfiability of propositional logic formulas [76] as well as computing the most probable explanation [77] and the maximum a posteriori hypothesis [78] in Bayesian networks.

Our proposed SLS algorithm is described in Algorithm 3, where c indicates an employed machine learning classifier. F denotes the feature candidate set. S_t serves as a tabu list that records all the visited states so that the algorithm does not consider a state repeatedly. The parameters θ_u and θ_{cot} are inputs to the function INITIALIZATION(θ_u, θ_{cot}) and respectively control how many percentage of unigram candidates in F_u and CoT candidates in F_{cot} that are included in an initial state by random selection. The parameter λ is input to the function NEXTSTEP(λ), which decides the next step to be a noise step with probability of λ or a greedy step with probability $1 - \lambda$. The NEIGHBOR(s, S_t) function returns a random neighbor state of s that is not recorded in the tabu list S_t.[7] The parameter κ limits how many neighbor states are evaluated in a greedy step. The parameter R defines how many steps are performed in each try, and MAX-TRIES defines the maximum number of tries before termination performed by the algorithm.

Algorithm 3 works in the following way. In each try, an initial state is randomly created. The search begins with this initial state. Then the algorithm goes through an iterative hill-climbing process in R steps. Each step of the process is either a noise step or a greedy step. If it is a noise step, the search moves to a neighbor state. If it is greedy step, the search test κ neighbors of the current state and goes to a neighbor state with maximum score. In each step, if the score p in that step is not worse than the recorded score p^*, then p^* is updated with p and the recorded state s^* is updated with s. After algorithm finishes with R steps in MAX-TRIES tries, the recorded state s^* is returned. In this way, the correspond feature subset F_{s^*} is optimized for the classifier c.

[7] A neighbor state of s is a state s' so that hamming distance between s and s' is 1.

4.3.3 Evaluation

In this subsection, we present experiments to evaluate the performance of the proposed EFS framework. In the evaluation we compare our proposed approach with existing methods on two standard datasets, i.e., the datasets DM1400 and DM2000, which are from the movie review domain and were originally introduced by Pang et al. in 2002 [10] and 2004 [12] respectively.

Metrics

To evaluate the performance of sentiment classifiers, we adopt the evaluation metric *accuracy* which is the percentage of correctly labeled reviews out of total reviews and is generally used in most of the previous work:

$$\text{accuracy} = \frac{\#\text{correctly labeled reviews}}{\#\text{total reviews}}.$$

Performance Comparison

Table 4.3 chronologically summarizes results on the two standard datasets reported in recent ten years. The column “Ex-Efforts" indicates whether the related work uses extra human efforts as part of their proposed methods. Performance above 90% on each dataset is bolded. Best performance on each dataset is underlined. From the Table 4.3, we can see that our approach achieves the best performance (90.13% accuracy) on DM1400. Among the six methods that achieve classification performance above 90% on DM2000, most of them require extra human efforts, e.g., manually built lexicons [9], predefined extraction patterns [80, 92], and preselected feature categories [84], as inputs to their methods. In comparison, our proposed EFS framework is a fully automatic process. Although Bai's method [82] currently is best on DM2000, our approach shows comparably good (92.70% v.s. 92.37%) on the same dataset. In addition, our approach beats Bai's method [82] on the DM1400 with more than 11% accuracy, which suggests our approach is more robust than Bai's approach [82]. Therefore, through performance comparison on two standard datasets, we conclude that our proposed EFS framework is generally superior to existing state-of-the-art approaches in that our approach is high accuracy, more robust, and needs small human efforts.

4.3.4 Summary

The paper P3 is a further development on the contribution of the paper P2. In the paper P3, we propose an Effective Feature Search (EFS) framework to enhance the performance of sentiment classifiers. The proposed EFS framework takes advantages of CoT patterns and

Table 4.3: Performance Comparison

Dataset	Work	Year	Ex-Efforts	Accuracy
DM1400	Pang et al. [10]	2002	No	82.90%
DM1400	Mullen&Collier [79]	2004	Yes	86.00%
DM1400	Riloff et al. [80]	2006	Yes	82.70%
DM1400	Zhai et al. [81]	2010	No	84.30%
DM1400	Bai [82]	2011	No	78.08%
DM1400	our EFS framework	2012	No	**90.13%**
DM2000	Pang&Lee [12]	2004	No	87.20%
DM2000	Whitelaw et al. [9]	2005	Yes	**90.20%**
DM2000	Kennedy&Inkpen [83]	2006	Yes	86.20%
DM2000	König&Brill [14]	2006	Yes	**91.00%**
DM2000	Zaidan et al. [13]	2007	Yes	**92.20%**
DM2000	Abbasi et al. [84]	2008	Yes	**91.70%**
DM2000	Martineau&Finin [85]	2009	No	88.10%
DM2000	O'Keefe&Koprinska [86]	2009	No	87.15%
DM2000	Taboada et al. [7]	2011	Yes	76.63%
DM2000	Pak&Paroubek [87]	2011	Yes	85.10%
DM2000	Saleh et al. [88]	2011	No	86.19%
DM2000	Heerschop et al. [89]	2011	Yes	81.00%
DM2000	Mejova et al. [90]	2011	Yes	87.50%
DM2000	Maas et al. [91]	2011	No	88.90%
DM2000	Abbasi et al. [92]	2011	Yes	89.65%
DM2000	Bai [82]	2011	No	**92.70%**
DM2000	our EFS framework	2012	No	**92.37%**

search for effective features in an SLS process. Performance comparison on two standard datasets shows that our proposed EFS framework is comparatively superior to existing state-of-the-art approaches in that our EFS framework is highly accurate, robust, and needs small human efforts. The result of this paper answers questions of RQ3 discussed in the Section 1.4.

My contribution: I was the first author of the paper P3 and did the implementation of the work, analysis of the results, and writing the paper. Ole J. Mengshoel gave suggestions and feedback on the development of the SLS model and the algorithm. Jon Atle Gulla on result analysis and writing on improving the paper.

4.4 A Hierarchical Learning Approach on Sentiment Ontology Tree

The forth contribution of this thesis is:

C4: A Hierarchical Learning via Sentiment Ontology Tree (HL-SOT) approach for sentiment analysis.

In the paper P4, we study the problem of sentiment analysis on product reviews through a novel method, called the HL-SOT approach, namely Hierarchical Learning (HL) with Sentiment Ontology Tree (SOT). This contribution is for the forth research question:

RQ4: How can we design an ontology-supported framework so that knowledge of the ontology of a product can naturally help sentiment analysis process?

By sentiment analysis on product reviews we aim to fulfill two tasks, i.e., labeling a target text[8] with: 1) the product's attributes (attributes detection task), and 2) their corresponding sentiments mentioned therein (sentiment orientation task). In this section, before we formulate the overview of the HL-SOT approach, we first present a formal definition on what the SOT is.

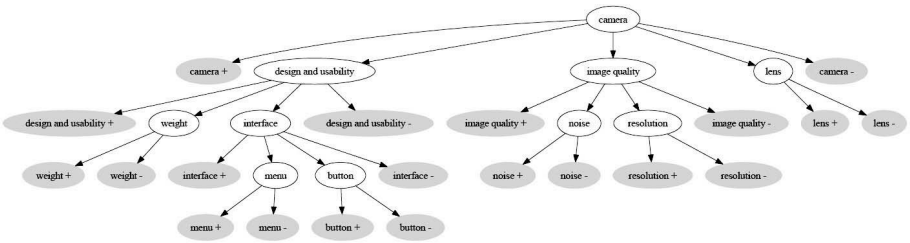


Figure 4.5: An example of part of a SOT for digital camera

4.4.1 Sentiment Ontology Tree

As we discussed in Section 1.4, ontology is a knowledge structure that organizes the hierarchical relationships among a product's attributes. Our goal is to develop an ontology-supported framework so that knowledge of the ontology of a product can naturally help sentiment analysis process. In the paper P6, we propose to use a tree-like ontology structure SOT, i.e., Sentiment Ontology Tree, to formulate relationships among a product's attributes. Here,we give a formal definition on what a SOT is.

Definition 1 *[SOT] SOT is an abbreviation for Sentiment Ontology Tree that is a tree-like ontology structure* $T(v, v^+, v^-, \mathbb{T})$. v *is the root node of* T *which represents an attribute*

[8]Each product review to be analyzed is called target text in the following of this section.

of a given product. v^+ is a positive sentiment leaf node associated with the attribute v. v^- is a negative sentiment leaf node associated with the attribute v. $\mathbb{T}$ is a set of subtrees. Each element of $\mathbb{T}$ is also a SOT $T'(v', v'^+, v'^-, \mathbb{T}')$ which represents a sub-attribute of its parent attribute node.

By the Definition 1, we define a root of a SOT to represent an attribute of a product. The SOT's two leaf child nodes are sentiment (positive/negative) nodes associated with the root attribute. The SOT recursively contains a set of sub-SOTs where each root of a sub-SOT is a non-leaf child node of the root of the SOT and represent a sub-attribute belonging to its parent attribute. This definition successfully describes the hierarchical relationships among all the attributes of a product. For example, in Fig. 4.5 the root node of the SOT for a digital camera is its general overview attribute. Comments on a digital camera's general overview attribute appearing in a review might be like "this camera is great". The "camera" SOT has two sentiment leaf child nodes as well as three non-leaf child nodes which are respectively root nodes of sub-SOTs for sub-attributes "design and usability", "image quality", and "lens". These sub-attributes SOTs recursively repeat until each node in the SOT does not have any more non-leaf child node, which means the corresponding attributes do not have any sub-attributes, e.g., the attribute node "button" in Fig. 4.5.

4.4.2 Sentiment Analysis with SOT

In this subsection, we present the HL-SOT approach. With the defined SOT, the problem of sentiment analysis is able to be formulated to be a hierarchical classification problem. Then a specific hierarchical learning algorithm is further proposed to solve the formulated problem.

Problem Formulation

In the proposed HL-SOT approach, each target text is to be indexed by a unit-norm vector $x \in \mathcal{X}, \mathcal{X} = \mathbb{R}^d$. Let $\mathscr{Y} = \{1, ..., N\}$ denote the finite set of nodes in SOT. Let $y = \{y_1, ..., y_N\} \in \{0, 1\}^N$ be a label vector to a target text x, where $\forall i \in \mathscr{Y}$:

$$y_i = \begin{cases} 1, & \text{if } x \text{ is labeled by the classifier of node } i, \\ 0, & \text{if } x \text{ is not labeled by the classifier of node } i. \end{cases}$$

A label vector $y \in \{0, 1\}^N$ is said to respect SOT if and only if y satisfies $\forall i \in \mathscr{Y}, \forall j \in \mathcal{A}(i)$: if $y_i = 1$ then $y_j = 1$, where $\mathcal{A}(i)$ represents a set ancestor nodes of i, i.e.,$\mathcal{A}(i) = \{x | ancestor(i, x)\}$. Let $\mathcal{Y}$ denote a set of label vectors that respect SOT. Then the tasks of sentiment analysis can be formulated to be the goal of a hierarchical classification that is to learn a function $f : \mathcal{X} \to \mathcal{Y}$, that is able to label each target text $x \in \mathcal{X}$ with classifier of each node and generating with x a label vector $y \in \mathcal{Y}$ that respects SOT. The requirement of a generated label vector $y \in \mathcal{Y}$ ensures that a target text is to be labeled with a node only if its parent attribute node is labeled with the target text. For example, in

Fig. 4.5 a review is to be labeled with “image quality +" requires that the review should be successively labeled as related to “camera" and “image quality". This is reasonable and consistent with intuition, because if a review cannot be identified to be related to a camera, it is not safe to infer that the review is commenting a camera’s image quality with positive sentiment.

HL-SOT Algorithm

The algorithm H-RLS studied in [93] solved a similar hierarchical classification problem as we formulated above. However, the H-RLS algorithm was designed as an online-learning algorithm which is not suitable to be applied directly in our problem setting. Moreover, the algorithm H-RLS defined the same value as the threshold of each node classifier. We argue that if the threshold values could be learned separately for each classifiers, the performance of classification process would be improved. Therefore we propose a specific hierarchical learning algorithm, named HL-SOT algorithm, that is able to train each node classifier in a batch-learning setting and allows separately learning for the threshold of each node classifier.

Defining the f function Let $w_1, ..., w_N$ be weight vectors that define linear-threshold classifiers of each node in SOT. Let $W = (w_1, ..., w_N)^\top$ be an $N \times d$ matrix called weight matrix. Here we generalize the work in [93] and define the hierarchical classification function f as:

$$\hat{y} = f(x) = g(W \cdot x),$$

where $x \in \mathcal{X}, \hat{y} \in \mathcal{Y}$. Let $z = W \cdot x$. Then the function $\hat{y} = g(z)$ on an N-dimensional vector z defines:
$\forall i = 1, ..., N$:

$$\hat{y}_i = \begin{cases} \mathfrak{B}(z_i \geq \theta_i), & \text{if } i \text{ is a root node in SOT} \\ & \text{or } y_j = 1 \text{ for } j = \mathcal{P}(i), \\ 0, & \text{else} \end{cases}$$

where $\mathcal{P}(i)$ is the parent node of i in SOT and $\mathfrak{B}(S)$ is a boolean function which is 1 if and only if the statement S is true. Then the hierarchical classification function f is parameterized by the weight matrix $W = (w_1, ..., w_N)^\top$ and threshold vector $\theta = (\theta_1, ..., \theta_N)^\top$. The hierarchical learning algorithm HL-SOT is proposed for learning the parameters of W and θ.

Parameters Learning for f function Let D denote the training data set: $D = \{(r, l)|r \in \mathcal{X}, l \in \mathcal{Y}\}$. In the HL-SOT learning process, the weight matrix W is firstly initialized to be a 0 matrix, where each row vector w_i is a 0 vector. The threshold vector is initialized to be a 0 vector. Each instance in the training set D goes into the training process. When

a new instance r_t is observed, each row vector $w_{i,t}$ of the weight matrix W_t is updated by a regularized least squares estimator given by:

$$\begin{aligned} w_{i,t} = (I + S_{i,Q(i,t-1)} S_{i,Q(i,t-1)}^{\top} + r_t r_t^{\top})^{-1} \\ \times S_{i,Q(i,t-1)} (l_{i,i_1}, l_{i,i_2}, ..., l_{i,i_{Q(i,t-1)}})^{\top} \end{aligned} \tag{4.12}$$

where I is a $d \times d$ identity matrix, $Q(i, t-1)$ denotes the number of times the parent of node i observes a positive label before observing the instance r_t, $S_{i,Q(i,t-1)} = [r_{i_1}, ..., r_{i_{Q(i,t-1)}}]$ is a $d \times Q(i, t-1)$ matrix whose columns are the instances $r_{i_1}, ..., r_{i_{Q(i,t-1)}}$, and $(l_{i,i_1}, l_{i,i_2}, ..., l_{i,i_{Q(i,t-1)}})^{\top}$ is a $Q(i, t-1)$-dimensional vector of the corresponding labels observed by node i. The Formula 4.12 restricts that the weight vector $w_{i,t}$ of the classifier i is only updated on the examples that are positive for its parent node. Then the label vector $\hat{y}_{r_t}$ is computed for the instance r_t, before the real label vector l_{r_t} is observed. Then the current threshold vector θ_t is updated by:

$$\theta_{t+1} = \theta_t + \epsilon(\hat{y}_{r_t} - l_{r_t}), \tag{4.13}$$

where ϵ is a small positive real number that denotes a corrective step for correcting the current threshold vector θ_t. To illustrate the idea behind the Formula 4.13, let $y'_t = \hat{y}_{r_t} - l_{r_t}$. Let $y'_{i,t}$ denote an element of the vector y'_t. The Formula 4.13 correct the current threshold $\theta_{i,t}$ for the classifier i in the following way:

- If $y'_{i,t} = 0$, it means the classifier i made a proper classification for the current instance r_t. Then the current threshold θ_i does not need to be adjusted.

- If $y'_{i,t} = 1$, it means the classifier i made an improper classification by mistakenly identifying the attribute i of the training instance r_t that should have not been identified. This indicates the value of θ_i is not big enough to serve as a threshold so that the attribute i in this case can be filtered out by the classifier i. Therefore, the current threshold θ_i will be adjusted to be larger by ϵ.

- If $y'_{i,t} = -1$, it means the classifier i made an improper classification by failing to identify the attribute i of the training instance r_t that should have been identified. This indicates the value of θ_i is not small enough to serve as a threshold so that the attribute i in this case can be recognized by the classifier i. Therefore, the current threshold θ_i will be adjusted to be smaller by ϵ.

The hierarchical learning algorithm HL-SOT is presented as in Algorithm 4. The HL-SOT algorithm enables each classifier to have its own specific threshold value and allows this threshold value can be separately learned and corrected through the training process. It is not only a batch-learning setting of the H-RLS algorithm but also a generalization to the latter. If we set the algorithm HL-SOT's parameter ϵ to be 0, the HL-SOT becomes the H-RLS algorithm in a batch-learning setting.

Algorithm 4 Hierarchical Learning Algorithm HL-SOT

INITIALIZATION:
Each vector $w_{i,1}, i = 1, ..., N$ of weight matrix W_1 is set to be 0 vector
Threshold vector θ_1 is set to be 0 vector
BEGIN
for $t = 1, ..., |D|$ **do**
    Observe instance $r_t \in \mathcal{X}$
    for $i = 1, ...N$ **do**
        Update each row $w_{i,t}$ of weight matrix W_t by Formula 4.12
    end for
    Compute $\hat{y}_{r_t} = f(r_t) = g(W_t \cdot r_t)$
    Observe label vector $l_{r_t} \in \mathcal{Y}$ of the instance r_t
    Update threshold vector θ_t by Formula 4.13
end for
END

4.4.3 Evaluation

The evaluation on the HL-SOT approach is conducted on a human-labeled data set from a customer review website[9]. Since the proposed HL-SOT approach is a hierarchical classification process, we use three classic loss functions for measuring classification performance.

Metrics

The three loss functions are respectively the One-error Loss (O-Loss) function, the Symmetric Loss (S-Loss) function, and the Hierarchical Loss (H-Loss) function:

- One-error loss (O-Loss) function is defined as:

$$L_O(\hat{y}, l) = \mathfrak{B}(\exists i : \hat{y}_i \neq l_i),$$

  where $\hat{y}$ is the prediction label vector and l is the true label vector; $\mathfrak{B}$ is the boolean function as defined in Section 4.4.2.

- Symmetric loss (S-Loss) function is defined as:

$$L_S(\hat{y}, l) = \sum_{i=1}^{N} \mathfrak{B}(\hat{y}_i \neq l_i),$$

- Hierarchical loss (H-Loss) function is defined as:

$$L_H(\hat{y}, l) = \sum_{i=1}^{N} \mathfrak{B}(\hat{y}_i \neq l_i \wedge \forall j \in \mathcal{A}(i), \hat{y}_j = l_j),$$

[9] http://www.consumerreview.com/

Table 4.4: Performance Comparison (A Smaller Loss Value Means a Better Performance)

Metrics	Dimensionality=110			Dimensionality=220		
	H-RLS	HL-flat	HL-SOT	H-RLS	HL-flat	HL-SOT
O-Loss	0.9812	0.8772	**0.8443**	0.9783	0.8591	**0.8428**
S-Loss	8.5516	2.8921	**2.3190**	7.8623	2.8449	**2.2812**
H-Loss	3.2479	1.1383	**1.0366**	3.1029	1.1298	**1.0247**

where $\mathcal{A}$ denotes a set of nodes that are ancestors of node i in SOT.

Unlike the O-Loss function and the S-Loss function, the H-Loss function captures the intuition that loss should only be charged on a node whenever a classification mistake is made on a node of SOT but no more should be charged for any additional mistake occurring in the subtree of that node. It measures the discrepancy between the prediction labels and the true labels with consideration on the SOT structure defined over the labels. In our experiments, the recorded loss function values for each experiment running are computed by averaging the loss function values of each testing snippets in the testing set.

Performance Comparison

In order to find out whether utilizing the hierarchical relationships among labels and the introduction of separately learning threshold for each classifier help to improve the accuracy of the classification, we compare our HL-SOT approach with the following two baseline approaches:

- HL-flat: The HL-flat approach involves an algorithm that is a “flat" version of HL-SOT algorithm by ignoring the hierarchical relationships among labels when each classifier is trained. In the training process of HL-flat, the algorithm reflexes the restriction in the HL-SOT algorithm that requires the weight vector $w_{i,t}$ of the classifier i is only updated on the examples that are positive for its parent node.
- H-RLS: The H-RLS approach is implemented by applying the H-RLS algorithm studied in [93]. Unlike our proposed HL-SOT algorithm that enables the threshold values to be learned separately for each classifiers in the training process, the H-RLS algorithm only uses an identical threshold values for each classifiers in the classification process.

Experiments are conducted on the performance comparison between the proposed HL-SOT approach with HL-flat approach and the H-RLS approach. The dimensionality d of the index term space is set to be 110 and 220. The corrective step ϵ is set to be 0.005. The experimental results are summarized in Table 4.5. From Table 4.5, we can observe that the HL-SOT approach generally beats the H-RLS approach and HL-flat approach on O-Loss, S-Loss, and H-Loss respectively. The H-RLS performs worse than the HL-flat and the HL-SOT, which indicates that the introduction of separately learning threshold for each classifier did improve the accuracy of the classification. The HL-SOT approach performs

better than the HL-flat, which demonstrates the effectiveness of utilizing the hierarchical relationships among labels.

4.4.4 Summary

In the paper P4, we propose a novel and effective approach to sentiment analysis on product reviews. In our proposed HL-SOT approach, we define SOT to formulate the knowledge of hierarchical relationships among a product's attributes and tackle the problem of sentiment analysis in a hierarchical classification process with the proposed algorithm. The performance comparison shows that the proposed HL-SOT approach outperforms two baselines: the HL-flat and the H-RLS approach. This confirms two intuitive motivations based on which our approach is proposed: 1) separately learning threshold values for each classifier improve the classification accuracy; 2) knowledge of hierarchical relationships of labels improve the approach's performance. The result of this paper answers questions of RQ4 discussed in the Section 1.4.

My contribution: I was the first author of the paper P4 and was responsible for the design and implementation of the work, analysis of the results, and writing the paper. Jon Atle Gulla gave feedback and discussed on the implementation, the results, and the writing of the paper.

4.5 A Localized Feature Selection Framework

The fifth contribution of this thesis is:

C5: A Localized Feature Selection (LFS) framework tailored to the HL-SOT approach.

In the paper P6, we propose a Localized Feature Selection (LFS) framework tailored to the HL-SOT approach discussed in Section 4.4. This contribution is for the fifth research question:

RQ5: If an ontology-supported sentiment analysis framework can be developed, how can we enhance its performance from different angles?

In this section, we present the LFS framework to generate a locally customized index term space for each node of SOT respectively. We first discuss why localized feature selection is needed for the HL-SOT approach. Then we define the concept of local hierarchy of SOT to introduce the local feature selection scope of a node, followed by a presentation on the local hierarchy based feature selection process.

4.5.1 Why Localized Feature Selection for the HL-SOT

One deficiency of the HL-SOT approach is that it uses a globally unified index term space to index target texts, which cannot efficiently encode feature information required by each local individual node of SOT. When we look into the detailed classification process of each node of SOT, we observe the following two types of phenomena. Firstly, SOT organizes domain knowledge in a tree like structure. Within a particular domain knowledge represented by SOT, nodes that stay in different branches of SOT represent independent different attributes in that domain. In this way, feature terms (e.g., the term "ergonomics") that are relevant to a node (e.g., the node "design and usability") might be irrelevant to other nodes (e.g., the node "image quality") that stay at another branches of SOT; Secondly, the HL-SOT approach labels each target text in a hierarchical order which ensures that each target text that comes to be handled by a node has already been labeled as true by its parent node. Due to this characteristic, feature terms (e.g., the term "noise") that are significant to a node i (e.g., the node "noise") might become a trivial term for i's child nodes (e.g., the nodes "noise +" and "noise -"). Therefore, the purpose of the localized feature selection is to filter out irrelevant terms that are insignificant to each individual node and build a locally customized index term space for the node so that the performance of the node can be improved.

4.5.2 Local Feature Selection Scope for a Node

In order to select locally customized feature terms for each individual node, we need to define a suitable scope, called local feature selection scope[10], within which the feature selection process can be effectively conducted for the node. Since the HL-SOT approach is a hierarchical classification process, before we introduce the local scope for a node we first give a formal definition on local hierarchy of SOT.

Definition 2 *[Local Hierarchy] A local hierarchy Δ_u of SOT is defined to be formed by all the child nodes of u in SOT, where the node u must be a non-leaf node of the SOT.*

By the Definition 2, we say all the child nodes of u are on the same local hierarchy under u which is denoted by Δ_u. For examples, in Fig. 4.6 nodes "camera +", "design and usability", "image quality", "lens", "camera -" are deemed on the same local hierarchy under the node "camera" and nodes "weight +", "weight -" are deemed on the same local hierarchy under the node "weight", etc. In the hierarchical labeling process of the HL-SOT approach, after a target text is labeled as true by a node i it will go further to the local hierarchy under i and is to be labeled by all nodes on the local hierarchy Δ_i. For a target text the labeling processes of nodes on Δ_i locally can be considered as a multi-label classification process where each node is a local label. Therefore, the measurement for selecting terms as features should be calculated among nodes on the same hierarchy.

[10] In this paper, we also call it "local scope" for short.

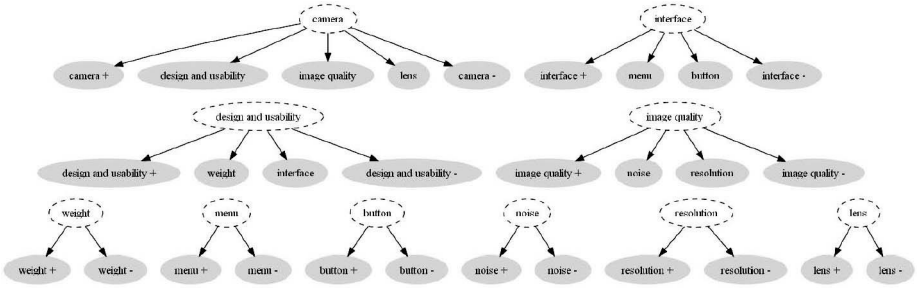


Figure 4.6: All local hierarchies of the example SOT: the grey nodes sharing the same parent node in dashed line are called on the same local hierarchy under the parent node

Hence, the local scope for a node is defined within the local hierarchy which the node is on.

4.5.3 Local Hierarchy Based Feature Selection

In the proposed LFS framework, local feature selection for a node i of SOT is performed within the *local scope* of the node i. Since nodes on the same local hierarchy share the same local scope, local feature selection process for all nodes of SOT is achieved in local hierarchy based manner. Specifically, for the feature selection process on a local hierarchy Δ, let $c_1, c_2, ..., c_K$ denote the K nodes on Δ. Let D denote the training data set for the HL-SOT approach. Let D_{c_k} denote the set of instances in D that contains the label of the node $c_k (1 \leqslant k \leqslant K)$. Let D_Δ denote the training corpus for the local hierarchy Δ which is the set of all instances in the training data set D that contain any label of nodes on the local hierarchy Δ: $D_\Delta = \bigcup_{k=1}^{K} D_{c_k}$. Let V_{c_k} denote the set of all the vocabularies that appears in D_{c_k}. Let $s_{c_k}(w)$ denote the term score that measures the suitability of w as a feature for node c_k. Let F_{c_k} denote the set of feature terms selected for c_k. Let d_{c_k} denote the number of features to be selected in F_{c_k}. A local feature selection process for nodes on the local hierarchy Δ is described in Algorithm 5.

In the data initialization phase of the Algorithm 5, the data instance set D_{c_k} and vocabulary set V_{c_k} for each node on the local hierarchy Δ as well as the training corpus D_Δ are established. In a local feature selection process, a term score $s_{c_k}(w)$ for each term $w \in V_{c_k}$ can be calculated by a specified feature selection algorithm, taking D_Δ as the training corpus and D_{c_k} as the data instance set in the class c_k. The local feature selection process can employ any specific feature selection algorithm to calculate the term scores. After all terms in V_{c_k} are calculated, those terms with top d_{c_k} scores are selected to establish the feature space F_{c_k} for the node c_k. Since the number of terms in V_{c_k} varies from node to node, in order to produce a rational dimensionality d_{c_k} for the established feature space F_{c_k}, we introduce a feature selection rate, denoted by γ, to control d_{c_k} for each node c_k, i.e., $d_{c_k} = \ulcorner |V_{c_k}| \times \gamma \urcorner$.

Algorithm 5 Localized Feature Selection Algorithm

DATA INITIALIZATION:
for each node c_k on Δ **do**
  Establish D_{c_k} containing instances being labeled true by c_k;
  Establish the vocabulary set V_{c_k};
  Remove stop words from V_{c_k};
end for
Establish the training corpus: $D_\Delta = \bigcup_{k=1}^{K} D_{c_k}$;
BEGIN
for each node c_k on Δ **do**
  for each term $w \in V_{c_k}$ **do**
    with training corpus D_Δ and data instance set D_{c_k}:
    Calculate $s_{c_k}(w)$ with a specified feature selection algorithm;
  end for
  Establish feature space F_{c_k} with top d_{c_k} terms from V_{c_k};
end for
END

After local feature selection processes for all the nods of SOT are accomplished, a locally customized index term space F_{c_k} for each node c_k is established. Each target text will be respectively indexed by a customized vector $x_{c_k} \in \mathcal{X}_{c_k} (\mathcal{X}_{c_k} = \mathbb{R}^{d_{c_k}})$ when it goes through the hierarchical classification process of the HL-SOT approach.

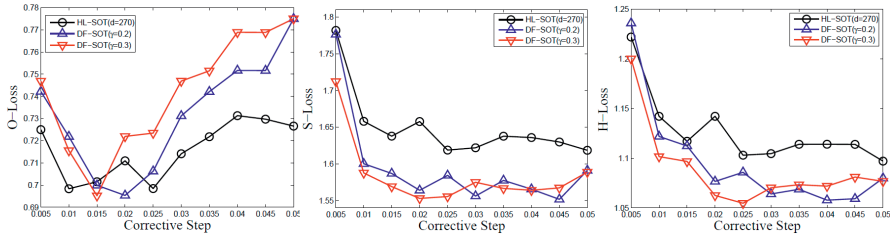
4.5.4 Evaluation

In this section, we present the evaluation on the proposed LFS framework on a dataset from a customer review website[11]. We use the same three loss functions, i.e., the One-error Loss (O-Loss) function, the Symmetric Loss (S-Loss) function, and the Hierarchical Loss (H-Loss) function, for measuring classification performance as described in the Section 4.4.3. We use the existing HL-SOT approach as a baseline. Since the HL-SOT approach used terms' document frequencies (DF) [18] algorithm to select features to build the globally unified index term space, employing the same DF feature selection algorithm we apply the proposed LFS framework on the HL-SOT approach and call the implemented method "DF-SOT". The only difference between HL-SOT and DF-SOT is the index term space for each node of SOT, i.e., in the HL-SOT all the nodes using the globally unified index term space while in the DF-SOT each node respectively using a locally customized index term space. In this way, the performance difference between the two methods will indicate the effect of the proposed LFS framework.

Comparison on Classification Performance

We conduct experiments to investigate whether the classification performance of the HL-SOT can be improved when it is implemented with the LFS framework. Fig. 4.7 presents the experimental results of classification accuracies between HL-SOT and DF-SOT. In the experiments, the dimensionality d of the globally unified index term space of the HL-SOT approach is set to 270, which is large enough for the HL-SOT approach to reach

[11] http://www.consumerreview.com/



(a) O-Loss (b) S-Loss (c) H-Loss

Figure 4.7: Classification Performance (A Smaller Loss Value Means Better Classification Performance)

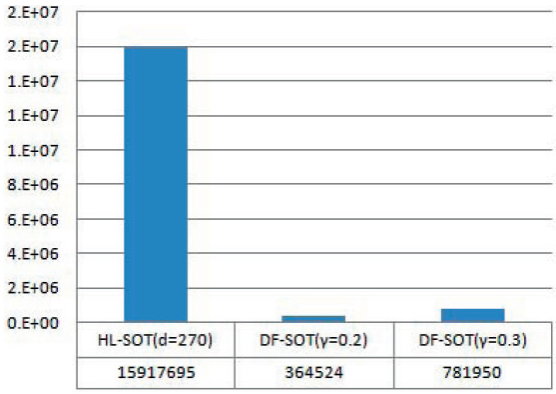


Figure 4.8: Time Consuming (ms)

its best performance level. The feature selection rate γ for the locally customized index term space of the DF-SOT approach is set to 0.2 and 0.3, which brings respectively 80% and 70% vocabulary reduction. The value of the corrective step ϵ is set to varying from 0.005 to 0.05 with each step of 0.005 so that each running approach can achieve its best performance with a certain value of ϵ. From Fig. 4.7, we can observe that when $\gamma = 0.2$ the DF-SOT approach reaches its best performance with 0.6953 ($\epsilon = 0.02$) on O-Loss, 1.5516 ($\epsilon = 0.045$) on S-Loss, and 1.0578 ($\epsilon = 0.04$) on H-Loss, and that when $\gamma = 0.3$ the DF-SOT approach reaches its best performance with 0.6953 ($\epsilon = 0.015$) on O-Loss, 1.5531 ($\epsilon = 0.02$) on S-Loss, and 1.0547 ($\epsilon = 0.025$) on H-Loss, which outperforms the best performance of the HL-SOT approach on O-Loss 0.6984 ($\epsilon = 0.025$), on S-Loss 1.6188 ($\epsilon = 0.025$), and on H-Loss 1.0969 ($\epsilon = 0.05$). This indicates that with the proposed LFS framework, compared with the HL-SOT approach, the DF-SOT approach generally improves the classification performance.

Comparison on Computational Efficiency

We conduct further experiments to analyze computational efficiency gained through the proposed LFS framework. All the experiments are conducted on a normal personal computer containing an Intel Pentium D CPU (2.4 GHz, Dual Core) and 4G memory. Fig. 4.8 summarizes the computational time consumed by experiment runs respectively for HL-SOT ($d = 270$) and DF-SOT ($\gamma = 0.2$ and $\gamma = 0.3$). From Fig. 4.8, we can observe that the HL-SOT approach consumes 15917695 ms to finish an experimental run, although the DF-SOT approach only takes respectively 2.29% (with $\gamma = 0.2$) and 4.91% (with $\gamma = 0.2$) of computational time as the existing HL-SOT approach consumes and achieves even better classification performance than the HL-SOT approach (see Fig.4.7). This confirms that much computational efficiency can be gained for the HL-SOT approach to be implemented in the LFS framework while better classification performance is ensured. Since the computational complexity of each node classifier of DF-SOT is the same as HL-SOT, the computational efficiency gained from the proposed LFS framework should be attributed to the dimension reduction of the index term space.

4.5.5 Summary

In the paper P6, we propose a Localized Feature Selection (LFS) framework tailored to the HL-SOT approach to sentiment analysis. Within the proposed LFS framework, each node classifier of the HL-SOT approach is able to perform classification on target texts in a locally customized index term space. Experiments against a human-labeled data set demonstrates that with the proposed LFS framework the classification performance of the HL-SOT approach is enhanced with computational efficiency being greatly gained. The result of this paper answers questions of RQ5 discussed in the Section 1.4.

My contribution: I was the first author of the paper P6 and was responsible for the design and implementation of the work, analysis of the results, and writing the paper. Jon Atle Gulla gave feedback and discussed on the implementation, the results, and the writing of the paper.

4.6 A Hybrid Hierarchical Classification Process

The sixth contribution of this thesis is:

C6: A Hybrid Hierarchical Classification Process for Sentiment Analysis.

In the paper P7, we propose a novel hybrid approach to solve the Attribute Detection (AD) task and Sentiment Orientation (SO) tasks in a Hybrid Hierarchical Classification Process (HHCP). This contribution is for the fifth research question:

RQ5: If an ontology-supported sentiment analysis framework can be developed, how can we enhance its performance from different angles?

Specifically, compared with the HL-SOT approach, the HHCP approach makes the following improvements. First, the HL-SOT approach employs a linear classifier that is estimated by a Regularized Least Squares (RLS) estimator. As we know, the goal of a linear RLS classifier is to make the model prediction as close as possible to a set of target values [94]. Thus, we have reasons to argue that the linear RLS classifier is not competent enough to be employed by the HL-SOT approach, where each node demands a binary classifier that should have had the capability to find maximum class separation in the output space. Therefore, in the proposed HHCP approach, for the AD task, a linear Fisher classifier is employed for identifying each attribute, since Fisher classifier is developed by requiring maximum class separation in the output space. Second, in the HL-SOT approach the SO task is performed on every attribute identified in its AD task. However, we found that with the knowledge of hierarchical relationships between labels not all the identified attributes need go through the SO process. Therefore, our proposed HHCP approach only performs the SO task on the identified attributes that are leaf nodes of the hierarchical structure. Third, like the HL-SOT approach, we could continue to apply the linear Fisher classifier on the SO task. However, when we looked into the failure cases made by the HL-SOT approach, we found that there were frequent cases where product attributes were successfully identified while polarity of sentiment information was misclassified, which indicates that common classifiers that work well for semantic classifications in the AD task might not be sensitive enough for classifying sentiment information in the SO task. Since the statistical linear classifiers that are designed for semantic classifications are evidentally prone to errors when applied to classifying sentiment information, our proposed HHCP approach turns to a rule-based heuristic solution for the SO task. In this section, we formulate a Hybrid Hierarchical Classification Process (HHCP) for both the attribute detection task and the sentiment orientation task.

4.6.1 Attribute Detection Task

In the AD task, a review text is to be labeled with product attributes that are mentioned. The AD task is the pivotal part of the whole sentiment analysis process, since it will further affect the performance of the SO task. To utilize knowledge of hierarchical relationships between attributes, each review text is to be analyzed by attribute nodes of SOT in a hierarchical manner: a text is to be classified with a node only if it is labeled as “true" by the node’s parent node. In this sub-section, we first introduce a linear Fisher classifier that is employed by each attribute node and then formulate the weight vector calculation algorithm as well as the decision boundary optimization process.

A linear Fisher classifier

A linear Fisher classifier serves as a binary classifier and is utilized for each attribute of a product. Let $x \in \mathcal{X}(\mathcal{X} = \mathbb{R}^d)$ denote an vector representation for a review text. Let c and $\bar{c}$ respectively denote the two classes: related to c and not related to c. The function of a linear Fisher classifier $f(.)$ is to project the d-dimensional input vector x down to one dimension $y \in \mathbb{R}$ by:

$$y = f(x) = w^T \cdot x,$$

where $w = (w_1, w_2, ..., w_d)^T$ is a unit weight vector that defines the linear Fisher classifier. Imagine that if the two classes c and $\bar{c}$ are divisible in the d-dimensional space, after being projected down to the one dimension $\mathbb{R}$, we still want to keep their divisibility. That is to say a projection needs to be selected so that the class separation can be maximized. Let the mean vectors x_c and $x_{\bar{c}}$ respectively represent the two classes of c and $\bar{c}$, i.e.,:

$$x_c = \frac{1}{N_c} \sum_{i \in c} x_i, \quad x_{\bar{c}} = \frac{1}{N_{\bar{c}}} \sum_{j \in \bar{c}} x_j.$$

We need to find a weight vector w that can maximize the separation distance between x_c and $x_{\bar{c}}$ when projected by w. However, the projection discovered in this way still suffers a problem that the two classes that could be separated in the original space are still overlapping in the one dimensional output space, because the covariances of the two class distributions are non-diagonal. To alleviate this problem, Fisher [31] proposed a balanced function that maximizes separation between classes while minimizing variance within each class:

$$J(w) = \frac{w^T S_B w}{w^T S_I w}, \tag{4.14}$$

where S_B is the between-class covariance matrix given by:

$$S_B = (x_{\bar{c}} - x_c)(x_{\bar{c}} - x_c)^T, \tag{4.15}$$

and S_I is the inner-class covariance matrix given by:

$$S_I = \sum_{i \in c} (x_i - x_c)(x_i - x_c)^T + \sum_{j \in \bar{c}} (x_j - x_{\bar{c}})(x_j - x_{\bar{c}})^T. \tag{4.16}$$

The weight vector w that makes the optimized projection is the w that maximizes the $J(w)$ function in Formula 4.14, i.e.,:

$$w = \arg\max_w J(w) = \arg\max_w \frac{w^T S_B w}{w^T S_I w}. \tag{4.17}$$

Calculating the weight vector w

One solution [94] for calculating the weight vector w is to differentiate $J(w)$ with respect to w and maximize $J(w)$ when:

$$(w^T S_B w) S_I w = (w^T S_I w) S_B w. \tag{4.18}$$

Since only the direction not the magnitude of w is concerned in projection, the scalar factors $w^T S_B w$ and $w^T S_I w$ can be ignored in the Equation 4.18. Considering that $S_B w$ is in the same direction of $x_{\bar{c}} - x_c$, when it is multiplied by S_I^{-1} on both sides of the Equation 4.18, we have:

$$w \propto S_I^{-1}(x_{\bar{c}} - x_c). \tag{4.19}$$

We could use the Formula 4.19 to calculate the weight vector w in the training process. However, due to the small sample size problem [32] in the training data set, this method is not guaranteed to always work since the S_I matrix is not always invertible or nonsingular. Therefore, we have to find another way to calculate w when S_I is a singular matrix. Following the similar idea in [33], we perform the singular value decomposition of S_I and have:

$$S_I = U \Sigma V^T, \tag{4.20}$$

where U and V are d-by-d orthogonal matrices and Σ is a d-by-d diagonal matrix. Let $V = [v_1, ..., v_r, v_{r+1}, ..., v_d]$, where r is the rank of S_I. Since S_I is a singular matrix, r is smaller than the dimensionality of the original space, i.e., $r < d$. Therefore, there must be a kernel $\mathcal{K}$ of S_I, where $\mathcal{K}$ is the null space of S_I and is a linear span of a set of vectors $\{x_k | S_I x_k = 0, 1 \leq k \leq (d - r)\}$. Let matrix Q be $[v_{r+1}, ..., v_d]$. Since the kernel $\mathcal{K}$ can be spanned by vectors $v_{r+1}, ..., v_d$ [34], the matrix QQ^T can be used when transforming samples from the original space to kernel. Let $\tilde{S_B}$ denote the scatter matrix of S_B and define:

$$\tilde{S_B} = QQ^T S_B (QQ^T)^T. \tag{4.21}$$

The weight vector w can be calculated as the eigenvector corresponding to the largest eigenvalues of scatter matrix of $\tilde{S_B}$.

The calculation process for the weight vector w is summarized in Algorithm 6. In this calculation algorithm, the matrices S_B and S_I are first initialized. If the rank of S_I is equal to d, it means S_I is invertible. Then the weight vector w is calculated using the Formula 4.19. If the rank of S_I is smaller than d, we cannot calculate the inverse matrix of S_I directly. Instead, we calculate the matrix $\tilde{S_B}$ using the Formula 4.21 and let w be the eigenvector corresponding to $\tilde{S_B}$'s the largest eigenvalue.

Optimization for the decision boundary

After the weight vector w is calculated, the function of the Fisher classifier $f(.)$ is decided and each input vector x can be projected from the original d-dimensional space down to the one-dimensional real number space $\mathbb{R}$. In order to classify data that are projected onto $\mathbb{R}$, we need to find a decision boundary that partitions $\mathbb{R}$ into two sets, one for each class. Although classes are divisible in the original d-dimensional space, the decision boundary in $\mathbb{R}$ might not be always clear cut, since the projection from the d-dimensional space onto the one dimensional space might lead to information loss. However, as we can imagine, there always exists an optimized decision boundary that classifies data with minimum classification error.

Algorithm 6 Calculation algorithm for w

BEGIN
Initialize S_B using the Formula 4.15
Initialize S_I using the Formula 4.16
if $r == d$ **then** ▷ r is the rank of S_I
    Calculate w using the Formula 4.19
else
    Calculate the SVD: $S_I = U\Sigma V^T$
    Let Q be $[v_{r+1}, ..., v_d]$
    Calculate $\tilde{S_B}$ using the Formula 4.21
    Get the largest eigenvalue e of $\tilde{S_B}$
    Let w be the eigenvector corresponding to e
end if
END

In order to discover the decision boundary with minimum classification error, we define an error recording function $E(y)$ that records every error made by the classifier on the training data with the decision boundary y:

$$\begin{aligned} E(y) = &\sum_{x \in c} B(f(x) - y \geq 0) \oplus B(\bar{y}_c - y \geq 0) \\ &+ \sum_{x \in \bar{c}} B(f(x) - y \geq 0) \oplus B(\bar{y}_{\bar{c}} - y \geq 0), \end{aligned} \tag{4.22}$$

where $\bar{y}_c$ and $\bar{y}_{\bar{c}}$ respectively denote mean values of projected values of samples from c and $\bar{c}$, i.e.,:

$$\bar{y}_c = \frac{1}{N_c} \sum_{x \in c} f(x), \quad \bar{y}_{\bar{c}} = \frac{1}{N_{\bar{c}}} \sum_{x \in \bar{c}} f(x),$$

and $B(.)$ is a boolean function which is 1 if the statement in $B(.)$ is true otherwise 0, and $\oplus$ is the XOR logistical operation. Assuming that the optimized decision boundary y_{opt} lies at somewhere between $\bar{y}_c$ and $\bar{y}_{\bar{c}}$, to locate y_{opt}, a traversal search method is employed starting from $\bar{y}_c$ to $\bar{y}_{\bar{c}}$. The optimized decision boundary y_{opt} is the y that achieve minimum value of the error recording function, i.e.,:

$$y_{opt} = \arg\min_y E(y). \tag{4.23}$$

4.6.2 Sentiment Orientation Task

The SO task is to find out sentiment polarity based on attributes identified in the AD task. In our proposed approach, not all the identified attributes need to go through the SO process. The SO task is only performed on the identified attributes that are leaf nodes of the hierarchical structure. We could continue to employ the linear Fisher classifier to

analyze sentiment of each attribute. However, when we looked into the failure cases, it is found in many cases that results of the AD tasks are correct while results of the SO tasks are wrong. Therefore, it is reasonable to doubt that the linear Fisher classifier that works well for semantic classification (e.g., for the AD task) might not be effective on sentiment classification (e.g., for the SO task). Hence, in our proposed HHCP approach, the linear Fisher classifier is given up for the SO task. Instead, we turn to a heuristic classification method inspired by rules.

Sentiment indication terms

One important rule that motivates our approach is that in review texts vocabularies that express sentiments tend to be highly overlapping for the same attribute of a product. For example, when we search our brain to lookup a term to praise the “LCD screen" attribute of a digital camera, terms such as “big", “clear", etc., usually jump out of our minds. In our approach, we call the terms that are utilized to indicate sentiments as Sentiment Indication Terms (SITs). Furthermore, SITs are usually dependent on attributes. For example, the term “big" is positive for the “LCD screen" but is usually used by people with small hands to complain about the size of a digital camera. Fortunately, when the SO task is performed, it is assumed that it targets on each known attribute that has been identified in the AD task. Therefore, it is suggested that we might utilize a set of SITs of an attribute to judge the sentiment expressed on the attribute.

A set of SITs for an attribute, say α, can be obtained from the training texts that are labeled with α in the set D_α . Let V_{D_α} denote a set of words that appear in D_α. Since each review text analyzed in the SO task is already labeled with α, terms that describe the attribute α are not useful for the current SO task. These attribute description words[12] together with stop words are all removed from V_{D_α} and the newly obtained word set V_α is treated as the SIT set of α.

In order to estimate sentiment indication of each word v in V_α, based on the rule that SITs of α are respectively used frequently for each sentiment expression of α, two heuristic measurements for v are respectively defined:

$$s_\alpha^+(v) = \frac{N_\alpha^+(v)}{N_\alpha}, \quad s_\alpha^-(v) = \frac{N_\alpha^-(v)}{N_\alpha}, \tag{4.24}$$

where $s_\alpha^+(v)$ and $s_\alpha^-(v)$ are respectively sentiment indication scores for $\alpha+$ and $\alpha-$[13], $N_\alpha^+(v)$ and $N_\alpha^-(v)$ are respectively numbers of texts containing v in $D_{\alpha+}$ and $D_{\alpha-}$, and N_α is the total number of texts in D_α.

Due to the limited number of samples in the training data set, for an attribute α there always are some SITs that are missed by V_α. Especially for some attributes that are

[12]In the experiments, we treat each term of an attribute label as description words of the label. For example, description words of the attribute “image quality" is “image" and “quality".

[13]$\alpha+$ and $\alpha-$ respectively represent positive opinions and negative opinions on α.

Algorithm 7 SO algorithm for a review text t

BEGIN
Get attribute set $\mathcal{A}_t$ that need SO task
Initialize $\mathcal{A}_t^*$ to be $\emptyset$ ▷ $\mathcal{A}_t^*$: the sentiment label set
for $\forall \alpha \in \mathcal{A}_t$ **do**
    Initialize $s_\alpha^+ = 0$
    Initialize $s_\alpha^- = 0$
    Collect D_α from D
    Establish V_α
    for $\forall v \in V_\alpha$ **do**
        Calculate $s_\alpha^+(v)$ and $s_\alpha^-(v)$ with the Formula 4.24
    end for
    for $\forall v \in V_t$ **do** ▷ V_t: the vocabulary set of t
        if $v \in V_\alpha$ **then**
            if Negation is caught within r step **then**
                s_α^- += $s_\alpha^+(v)$
                s_α^+ += $s_\alpha^-(v)$
            else
                s_α^+ += $s_\alpha^+(v)$
                s_α^- += $s_\alpha^-(v)$
            end if
        else
            if $v \in U^+$ **then**
                if Negation is caught within r step **then**
                    s_α^- += σ
                else
                    s_α^+ += σ
                end if
            end if
            if $v \in U^-$ **then**
                if Negation is caught within r step **then**
                    s_α^+ += σ
                else
                    s_α^- += σ
                end if
            end if
        end if
    end for
    $\mathcal{A}_t^* \leftarrow \alpha^* = \arg\max_* s_\alpha^*$
end for
return $\mathcal{A}_t^*$
END

rarely mentioned or some attributes for which training sample size is not big enough, even common universally sentiment words, e.g., “great" for positive and “bad" for negative, are absent from the SIT set. To alleviate this problem, we introduce two extended SIT sets U_α^+ and U_α^- respectively for positive and negative SITs of the attribute α. U_α^+ and U_α^- are reserved in case a universally positive/negative word is not collected in V_α.

Dealing with negation expressions

Until now, only SITs are concerned, although there is another pivotal factor that determines sentiment. That is the negation expression which is so important that it usually converts the whole sentiment of an expression. Unfortunately, most statistical classifiers treat negation words, e.g., “no", “not" and “never", as stop words and do not consider them in the classification processes. However, our proposed rule-based heuristic method

can be enabled to be aware of negation. To achieve this function, a set of negation words are collected in V_N. For each SIT v, negation expression is checked for v in the range $r \in Z^{+}$[14] around v, i.e., the r words before v and the r words after v in the review text are checked to find out whether they are negation words. If a negation is caught in the r range around v, the sentiment contribution from v is counted contrarily.

Sentiment orientation algorithm

The process of performing the SO task on a review text t is summarized in Algorithm 7. In the Algorithm 7, attributes that need to be analyzed in the SO task are firstly retrieved in set $\mathcal{A}_t$ from the AD task. $\mathcal{A}_t^*$ is initialized to be an empty set to store sentiment labels. For each attribute α that needs go through the SO process, the algorithm first establishes V_α and then calculates sentiment indication scores $s_\alpha^+(v)$ and $s_\alpha^-(v)$ for each $v \in V_\alpha$. After that, each word v from the vocabulary set V_t of the testing text t is analyzed. In this process, if v is a SIT of the attribute α, the positive and the negative sentiment indication scores of v are respectively added to the total positive score s_α^+ and the total negative score s_α^-. If v is not a SIT of α, v is checked whether it is a universally positive/negative word. If v is a universally positive/negative word, a defined universal sentiment score, say $\sigma \in \mathbb{R}^{+}$[15], is added to the total positive/negative score. In the analytic process of v, negation expression is always checked against words within r steps from v. If a negation expression is caught, the contribution of the scores from v will be added to the total positive and negative score oppositely. After finishing analyzing all the words in V_t, the sentiment with maximum total sentiment score is assigned to α. When all the attribute in $\mathcal{A}_t$ is processed, the algorithm outputs all the sentiment labels in $\mathcal{A}_t^*$ for the text t.

4.6.3 Evaluation

In this section, we present the evaluation on the proposed HHCP approach on a dataset from a customer review website[16]. We employ three similar loss functions as in the Section 4.4.3 that are designed for evaluating hierarchical labeling process. They are respectively one-error loss function, hierarchical loss function, and symmetric loss function. Using the existing HL-SOT approach as one baseline method, we also set up another two baselines, namely F-SOT method and HFCP method. The F-SOT method is developed by replacing the linear RLS classifiers employed in the HL-SOT approach with linear Fisher classifiers. The purpose of development of the F-SOT approach is to directly show performance improvement from the linear Fisher classifier. The HFCP approach is developed with applying the linear Fisher classifier on both the AD task and the SO task. The development of the HFCP approach aims at revealing benefits of turning to a rule-based heuristic classification method employed in the proposed HHCP approach.

[14] Z^+ denotes the positive integer set.
[15] $\mathbb{R}^+$ denotes the positive real number set.
[16] http://www.consumerreview.com/

Performance Comparison

Experiments on performance comparison are conducted among the three baseline methods and the proposed HHCP approach. Performances are compared when the dimensionality d of the input vector space is set to 150 and 300 respectively. The parameter ϵ which serves as the corrective step for training the HL-SOT approach is set to 0.01. The universal sentiment score σ that is involved in the rule-based SO task is set to 1. The experimental results are summarized in Table 4.5.

Table 4.5: Performance Comparison (A Smaller Loss Value Means a Better Performance)

Metrics	d=150				d=300			
	HL-SOT	F-SOT	HFCP	HHCP	HL-SOT	F-SOT	HFCP	HHCP
O-Loss	0.6807	0.6500	0.6300	**0.5473**	0.6720	0.6347	0.6220	**0.5493**
H-Loss	1.1067	1.0060	0.9513	**0.9471**	1.0853	1.0146	0.9773	**0.9630**
S-Loss	1.4227	1.3073	1.2494	**1.1907**	1.3980	1.2780	1.2335	**1.1704**

From the Table 4.5, we can observe that our proposed HHCP approach generally outperforms the other three baseline methods on the three evaluation metrics. The F-SOT is generally better than the HL-SOT, which confirms that compared with the linear RLS classifier the linear Fisher classifier enhances the performances on achieving sentiment analysis tasks. The HFCP is generally better than F-SOT, which indicates that the new treatment on identified attributes in the SO task help improve performance. The proposed HHCP is the best and specifically better than HFCP, which shows the success of the strategy of turning to a rule-based heuristic classification method in the SO task.

4.6.4 Summary

In the paper P7, we propose a novel approach to tackle two complementary sub-tasks of sentiment analysis on review texts, i.e., the Attribute Detection (AD) task and the Sentiment Orientation (SO) task, in a Hybrid Hierarchical Classification Process (HHCP). Specifically, the HHCP approach employs a linear Fisher classifier to achieve the AD task in an ontology-based hierarchical classification process. As evidences show that common statistical classifiers that have superior performances on semantic classifications do not necessarily work well on classifying sentiment information, we did not continue to use the linear Fisher classifier in the SO task. Instead, we turn to a rule-based heuristic classification method on performing sentiment orientation for attributes identified from the AD task. Experiments conducted for performance comparison not only show that our proposed HHCP approach outperforms the HL-SOT approach and the other two baseline methods. The HHCP approach is based on the HL-SOT approach and enhance its performance. The result of this paper answers questions of RQ5 discussed in the Section 1.4.

My contribution: I was the first author of the paper P7 and was responsible for the design and implementation of the work, analysis of the results, and writing the paper. Jon Atle

Gulla gave feedback and discussed on the implementation, the results, and the writing of the paper.

4.7 Contributions in Relation to Related Work

This section summarizes the contributions of each research result presented in previous sections in the light of related work with referring the reader to the research challenges and requirements discussed in the Section 3.3.

4.7.1 News impact analysis

As discussed in the Section 3.1, the three requirements for news impact analysis with respect to the related work are: 1) developing a model that quantifies the impact of topics of news articles; 2) developing an approach to calculate topic impact relation between news articles; 3) developing a mechanism to guide the news impact model training process so that the generated topic models seemly represent required topics. The work presented in the Section 4.1 focus on addressing the above three issues. Unlike the previous work [41, 42, 43, 44, 45] which focus on detecting and tracking topics of news articles, the proposed topic driven new impact model in our work is able to quantify the impact of topic of a news articles. In our work, a topic model technique is utilized to the topic impact relation between news articles, which analyze news article relations not only on keywords aspect but on the topic level. In the model training process, our work incorporate prior knowledge on required topics and guide the parameter estimation process so that the generated topic models seemly represent required topics.

4.7.2 Document overall sentiment classification

As discussed in the Section 3.2.1, there are following two challenges in document overall sentiment classification: 1) sentiment orientation of words is rather dependent on topics; 2) negation words are very important and might overturn the classification results. The motivations of the work described in the Section 4.2 are based on the above two challenges. In the work, we proposed an approach to generating complex features taking the advantage of high frequency of Co-occurring Term (CoT) patterns within customer review data sets. Since words that are used to express opinions are topic dependent, we believe that within a set of customer reviews on the same product the frequently co-occurring terms form a good resource of complex features that are highly capable of deciding the expressed sentiment orientation. For example, single terms like ąřhighąś does not necessarily means positive sentiment. However, in a corpus of customer reviews on digital cameras terms ąřhighąś and ąřpriceąś might co-occur together frequently and means def-

initely negative. In addition, negation words like ąřnotąś cannot be treated as an obvious feature for either positive class or negative class.

However, terms like ąřnotąś and ąřgoodąś together can be deemed as negative and usually co-occur in many negative expressions. However, high using only frequency CoT patterns not only produce effective complex features like but also bring useless candidates. Therefore, our work presented in the Section 4.3 proposed an Effective Feature Search framework which is the first approach that takes advantage of CoT patterns and search for effective features in an SLS process. Our EFS framework is a fully automatic process in that unlike previous work it requires no extra resources [92], no human-developed lexicons [95], and no human efforts and interactions [14] in the training and classification process. Hence, our proposed EFS framework is quite general and can be applied on data from different topic domains.

4.7.3 Attributed-based Sentiment Analysis

As discussed in the Section 3.2.2, we need to address the following two questions in the task of attributed-based sentiment analysis: 1) how can we utilize the domain-specific knowledge of product reviews; 2) how can we deal with the complex sentiments expressed in one review. Our work presented in the Section 4.4 deals with the above two problems. In that work, we proposed the concept of Sentiment Ontology Tree (SOT) that organizes a product's domain-specific knowledge such as relations between a product's attributes and the sentiment in a tree-like ontology structure. A specific hierarchical learning algorithm was developed in the HL-SOT approach which allows multiple-path labeling (input target text can be labeled with nodes belonging to more than one path in the SOT) and partial-path labeling (the input target text can be labeled with nodes belonging to a path that does not end on a leaf). This property makes the approach well suited for the situation where complicated sentiments on different attributes are expressed in one target text. To the best of our knowledge, the proposed HL-SOT approach is the first work to formulate the sentiment analysis task to be a hierarchical classification problem. The proposed HL-SOT approach can be generalized to make it possible to perform sentiment analysis on target texts that are a mix of reviews of different products, whereas existing works mainly focus on analyzing reviews of only one type of product.

The HL-SOT approach uses a globally unified index term space to encode target texts for different nodes which is deemed to limit its performance. The work described in the Section 4.5 aims at overcoming this weakness of the unique index term space and proposed a Localized Feature Selection (LFS) framework tailored to the HL-SOT approach. The proposed LFS framework enhances both the classification performance and the computational efficiency of the HL-SOT approach. In addition, the linear classifier utilized in the HL-SOT approach was estimated by a RLS estimator which is not competent enough to find max class separation. The HHCP approach described in the Section 4.6 employs a Fisher classifier, which not only maximizes the class separation but also minimizes the within-class variance, for the attribute detection task. Furthermore, our proposed HHCP

approach turns to a rule-based solution for classifying sentiment information since the statistical linear classifier has been evidentally proven its fallibility, while the HL-SOT approach uses the same linear RLS classifier for the sentiment orientation task.

Chapter 5

Conclusions

In this chapter, we present conclusions of the work in this thesis by summarizing the results and suggesting directions of future work.

5.1 Summary of Contributions

The overall research goal of this thesis is “extracting implicit knowledge from mining online text data". Therefore, the work in this thesis aims at discovering knowledge from online text corpora that cannot be directly retrieved through traditional keyword-based searching strategies. Two kinds of online information of WWW, i.e., objective online news and subjective customer reviews, have been analyzed in this thesis. For online business news, we intended to discover company’s impact through mining news collections and understand the affection of a specified event of a company. For online customer reviews, we aimed at extracting sentiments expressed in online review texts. To guide the research process, we raise five research questions in the Section 1.4. Each of the research question is studied in research publications. Approaches and solutions that are proposed in the publications constitute the six contributions (see the Section 1.6) of this thesis.

The first research question is how to detect the impact of companies through mining news collections. For this problem, we propose a topic driven impact analysis model that captures topic context in each news article through a semi-supervised topic model. The proposed topic driven impact analysis model provides a ranking mechanism that quantify the impact of the topic of each company’s news. For estimating parameters in the model, we have developed a semi-supervised parameter estimation process with Maximum A Posterior (MAP) estimator. The proposed topic driven impact analysis model is more focused on focusing on topic semantics than traditional vector space model based on cosine similarity.

The second research question is whether we can capture high frequent co-occur terms as complex features to improve sentiment classifiers. This question is asked based on the

observation that vocabularies used in product reviews tend to be highly overlapping. We believe that within highly-repeated co-occur term pairs, there contains useful knowledge that can benefit sentiment classifiers in the training and classification process. Therefore, we propose an automatic complex feature generation algorithm that takes advantages of high-frequently co-occur terms. Empirical analysis on the proposed algorithm not only qualitatively demonstrate good quality of the generated complex features but also quantitatively show the effectiveness on improving performance of sentiment classifiers.

The third research question is a further development on the approached proposed for the second question. It asks how we can remove the noise of the generated candidates while keep effective features for sentiment classification. Accordingly, we propose a fully-automatic Effective Feature Search (EFS) framework that makes a novel connection between complex feature generation and a Stochastic Local Search (SLS) process. In the SLS process, an optimized subset is searched for from the originally generated feature set. Performance of the proposed EFS framework is compared with existing techniques on two standard datasets, which indicate that our EFS framework is highly accurate, robust, and needs small human efforts.

When we looked into a lot of cases of online customer reviews. We found out that there exist relationships among attributes mentioned in reviews. We also notice that ontology is a good representation that organize various attributes of an object. Therefore, we have our the forth research question and state it as “how can we design an ontology-supported framework so that knowledge of the ontology of a product can naturally help sentiment analysis process" To answer this question, we define a concept of Sentiment Ontology Tree (SOT). The SOT organize each attribute of a product with two opposite sentiments. We further developed a hierarchical learning approach (HL-SOT) that naturally involves the SOT in sentiment analysis process. Evaluation on the HL-SOT approach show that utilizing the hierarchical relationships represented in the SOT help to improve the accuracy of sentiment analysis.

The fifth research question is raised after the HL-SOT approach was proposed. It aims at enhancing performance of the HL-SOT approach. First, since we found out that the HL-SOT approach uses a globally unified index term space, we believe that this cannot efficiently encode feature information required by each local individual node of SOT. Therefore, we proposed a Localized Feature Selection (LFS) framework to deal with this problem, which is empirically proven to be effective on improving both accuracy and efficiency of the HL-SOT approach. Second, we found out that the linear RLS classifier is not competent as needed to be employed in the HL-SOT approach, we propose to use a linear Fisher classifier instead in our proposed Hybrid Hierarchical Classification Process (HHCP) approach. Furthermore, since the statistical linear classifiers that are designed for semantic classification are evidentally prone to errors when being applied to classifying sentiment information, in the HHCP approach, we turns to a rule-based heuristic solution for sentiment orientation task. Experimental results on analyzing the proposed HHCP approach show that the HHCP approach is superior to the HL-SOT approach and the other two baseline methods.

5.2 Future Work

We would like to make a few suggestions on the future work based on the research in this thesis:

- Sentiment classification is a very challenging task. Since natural language expression is complicated and varies from time to time, no method can deal with every situation exhaustively. When we look at some cases where our proposed methods fail, we can still find some situations, e.g., negation expressions and ironic expressions, that are not dealt with very well. Research on detecting negation expressions and ironic expressions is worth of being investigated.
- In the proposed EFS framework, a hill-climbing SLS algorithm is developed to solve the feature subset selection problem. There is a need to better appreciate how the SLS algorithm should be designed with intelligent parameter techniques such as tabu list update strategy, guided noise search, and clever initialization so that the SLS algorithm can search for optimized solutions in a fast process.
- In the HL-SOT approach, the SOT is defined to formulate this knowledge in the proposed approach. However, what attributes to be included in a product's SOT and how to structure these attributes in the SOT is an effort of human beings. The sizes and structures of SOTs constructed by different individuals may vary. How the classification performance will be affected by variances of the generated SOTs is worthy of study. In addition, the SOT is constructed as a result of human efforts. An automatic method to learn a product's attributes and the structure of SOT from existing product review texts will greatly benefit the efficiency of the proposed approach.
- Although the proposed LFS framework shows its effectiveness of improving on the HL-SOT approach, its improvement on the classification performance is not so obvious compared with its much improvement on computational efficiency. Due to the limited number of instances in the training data set, the classification performance still suffers from the problem that unobserved terms appear in testing cases. This problem is inherently raised by the bag-of-word model. A concept-based indexing scheme that can infer concepts of unobserved terms might alleviate the problem.
- The proposed SOT-based hierarchical learning framework such as the HL-SOT approach and the HHCP approach focus on sentiment analysis on reviews of one product. We expect that this kind of analysis can be naturally generalized to analyzing a mix of reviews of more than one products. An interesting suggestion for future work would be to study performance of the HL-SOT approach and the HHCP approach on a set of customer reviews of a mix of different products.

Bibliography

[1] Vasileios Hatzivassiloglou and Janyce M. Wiebe. Effects of adjective orientation and gradability on sentence subjectivity. In *Proceedings of 18th International Conference on Computational Linguistics (COLING'00)*, Saarbrüken, Germany, 2000.

[2] Peter D. Turney. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, Philadelphia, USA, 2002.

[3] Andrea Esuli and Fabrizio Sebastiani. Determining the semantic orientation of terms through gloss classification. In *Proceedings of 14th ACM Conference on Information and Knowledge Management (CIKM'05)*, 2005.

[4] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of 10th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'04)*.

[5] Bing Liu, Minqing Hu, and Junsheng Cheng. Opinion observer: analyzing and comparing opinions on the web. In *Proceedings of 14th International World Wide Web Conference (WWW'05)*.

[6] Yue Lu and Chengxiang Zhai. Opinion integration through semi-supervised topic modeling. In *Proceedings of 17th International World Wide Web Conference (WWW'08)*.

[7] Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly D. Voll, and Manfred Stede. Lexicon-based methods for sentiment analysis. *Computational Linguistics*, 37(2):267–307, 2011.

[8] Ann Devitt and Khurshid Ahmad. Sentiment polarity identification in financial news: A cohesion-based approach. In *Proceedings of 45th Annual Meeting of the Association for Computational Linguistics (ACL'07)*, Prague, Czech Republic, 2007.

[9] Casey Whitelaw, Navendu Garg, and Shlomo Argamon. Using appraisal groups for sentiment analysis. In *Proceedings of the 14th ACM international Conference on Information and Knowledge Management*, pages 625–631, Bremen, Germany, November 2005. ACM Press.

[10] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of EMNLP-02, 7th Conference on Empirical Methods in Natural Language Processing*, pages 79–86, Philadelphia, US, 2002.

[11] Chenghua Lin and Yulan He. Joint sentiment/topic model for sentiment analysis. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM'11)*, 2009.

[12] Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *ACL*, pages 271–278, 2004.

[13] Omar F. Zaidan and Jason Eisner. Using "annotator rationales" to improve machine learning for text categorization. In *Proceedings of NAACL HLT 2007*, 2007.

[14] Arnd Christian König and Eric Brill. Reducing the human overhead in text categorization. In *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia (KDD'06)*, 2006.

[15] Yue Lu, ChengXiang Zhai, and Neel Sundaresan. Rated aspect summarization of short comments. In *Proceedings of 18th International World Wide Web Conference (WWW'09)*.

[16] Ron Kohavi and George John. Wrappers for feature subset selection. *Artificial Intelligence*, 97:273–324, 1997.

[17] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[18] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schutze. *Introduction to Information Retrieval*, chapter 13, pages 271–278. Cambridge University Press, 2008.

[19] Kenneth Ward Church and Patrick Hanks. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29, 1990.

[20] J. W. Wilbur and K. Sirotkin. The automatic identification of stop words. *Journal of the American Society for Information Science*, 18:45–55, 1992.

[21] Tom Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[22] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML-97)*, pages 412–420, San Francisco, CA, 1997. Morgan Kaufmann.

[23] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[24] Richard Duda, Peter Hart, and David Stork. *Pattern Classification*. John Wiley and Sons, 2001.

[25] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schutze. *Introduction to Information Retrieval*, chapter 15, pages 294–294. Cambridge University Press, 2008.

[26] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3:95–110, 1956.

[27] Mary E. Hribar Nicholas I. M. Gould and Jorge Nocedal. On the solution of equality constrained quadratic programming problems arising in optimization. *SIAM Journal of Scientific Computing*, 23:1376ÍC1395, 2006.

[28] R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[29] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.

[30] T. G. Dietterich. Random forests. *Machine Learning*, 40(2):139–157, 2000.

[31] Ronald Aylmer Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.

[32] J. S. Raudys and A. K. Jain. Small sample size effects in statistical pattern recognition: Recommendations for practitioners. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(3):252–263, 1990.

[33] L. F. Chen, H. Y. M. Liao, M. T. Ko, J. C. Lin, and G. J. Yu. A new LDA-based face recognition system which can solve the small sample size problem. *Pattern Recognition*, 33(10):1713–1726, 2000.

[34] B. Noble and J. W. Daniel. *Applied Linear Algebra*. Prentice-Hall, Englewood Cliffs, NJ, USA, 1977.

[35] Bing Liu. *Pattern Classification*. Springer, 2011.

[36] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 487–499, 1994.

[37] Thomas Hofmann. Probabilistic latent semantic analysis. In *UAI*, 1999.

[38] G. J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. Wiley, 1996.

[39] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[40] James Allan. *Introduction to topic detection and tracking*, pages 1–16. Kluwer Academic Publishers, 2002.

[41] Masaki Mori, Takao Miura, and Isamu Shioya. Topic detection and tracking for news web pages. In *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 338–342, 2006.

[42] Yiming Yang, Thomas Pierce, and Jaime G. Carbonell. A study of retrospective and on-line event detection. In *SIGIR'98*, pages 28–36, 1998.

[43] Kumaran, Giridhar and Allan, James. Text classification and named entities for new event detection. In *SIGIR'04*, pages 297–304, 2004.

[44] Jure Leskovec, Lars Backstrom, and Jon M. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *KDD'09*, pages 497–506, 2009.

[45] Canhui Wang, Min Zhang, Liyun Ru, and Shaoping Ma. Automatic online news topic ranking using media focus and user attention based on aging theory. In *CIKM'08*, pages 1033–1042, 2008.

[46] Yihua Chen, Eric K. Garcia, Maya R. Gupta, Ali Rahimi, and Luca Cazzanti. Similarity-based classification: Concepts and algorithms, 2009.

[47] Hung Chim and Xiaotie Deng. A new suffix tree similarity measure for document clustering. In *Proceedings of the 16th international conference on World Wide Web*, pages 121–130, 2007.

[48] James Z. Wang and William Taylor. Concept forest: A new ontology-assisted text document similarity measurement method. In *IEEE/WIC/ACM International Conference on Web Intelligence*, pages 395–401, 2007.

[49] Praveen Lakkaraju, Susan Gauch, and Mirco Speretta. Document similarity based on concept tree distance. In *HYPERTEXT 2008, Proceedings of the 19th ACM Conference on Hypertext and Hypermedia, Pittsburgh, PA, USA, June 19-21, 2008*, pages 127–132. ACM, 2008.

[50] Thomas Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42(1/2):177–196, January 2001.

[51] Xing Wei and W. Bruce Croft. LDA-based document models for ad-hoc retrieval. In *SIGIR 2006: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, Washington, USA, August 6-11, 2006*, pages 178–185, 2006.

[52] Leonhard Hennig, Ernesto William De Luca, and Sahin Albayrak. Learning summary content units with topic modeling. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, pages 391–399, 2010.

[53] Deng Cai, Qiaozhu Mei, Jiawei Han, and Chengxiang Zhai. Modeling hidden topics on document manifold. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM 2008, Napa Valley, California, USA, October 26-30, 2008*, pages 911–920, 2008.

[54] W. Chong, D. Blei, and L. Fei Fei. Simultaneous image classification and annotation. In *Proceedings of 22nd IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2009)*, pages 1903–1910, 2009.

[55] Cindy Xide Lin, Bo Zhao, Qiaozhu Mei, and Jiawei Han. PET: a statistical model for popular events tracking in social communities. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010*, pages 929–938, 2010.

[56] Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and ChengXiang Zhai. Topic sentiment mixture: modeling facets and opinions in weblogs. In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*, pages 171–180, 2007.

[57] Jing Zhang, Jie Tang, Liu Liu, and Juan-Zi Li. A mixture model for expert finding. In *PAKDD'08*, pages 466–478, 2008.

[58] Ali Daud, Juanzi Li, Lizhu Zhou, and Faqir Muhammad. Temporal expert finding through generalized time topic modeling. *Knowledge-Based System*, 23(6):615–625, 2010.

[59] A. P. Dempster, N. M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39:1–38, 1977.

[60] Ana-Maria Popescu and Oren Etzioni. Extracting product features and opinions from reviews. In *Proceedings of Human Language Technology Conference and Empirical Methods in Natural Language Processing Conference (HLT/EMNLP'05)*, 2005.

[61] Xiaowen Ding and Bing Liu. The utility of linguistic rules in opinion mining. In *Proceedings of the 30th Annual International ACM Special Interest Group on Information Retrieval Conference (SIGIR'07)*.

[62] Alina Andreevskaia and Sabine Bergler. Mining wordnet for a fuzzy sentiment: Sentiment tag extraction from wordnet glosses. In *Proceedings of 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL'06)*, 2006.

[63] Andrea Esuli and Fabrizio Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of 5th International Conference on Language Resources and Evaluation (LREC'06)*, 2006.

[64] Vasileios Hatzivassiloglou and Kathleen R. McKeown. Predicting the semantic orientation of adjectives. In *Proceedings of 35th Annual Meeting of the Association for Computational Linguistics (ACL'97)*, 1997.

[65] Jaap Kamps, Maarten Marx, Robert. Mokken, and Maarten de Rijke. Using WordNet to measure semantic orientation of adjectives. In *Proceedings of 4th International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal, 2004.

[66] Hong Yu and Vasileios Hatzivassiloglou. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of 8th Conference on Empirical Methods in Natural Language Processing (EMNLP'03)*, 2003.

[67] Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of Human Language Technology Conference and Empirical Methods in Natural Language Processing Conference (HLT/EMNLP'05)*.

[68] Yang Liu, Xiangji Huang, Aijun An, and Xiaohui Yu. ARSA: a sentiment-aware model for predicting sales performance using blogs. In *Proceedings of the 30th Annual International ACM Special Interest Group on Information Retrieval Conference (SIGIR'07)*, 2007.

[69] Ivan Titov and Ryan T. McDonald. Modeling online reviews with multi-grain topic models. In *Proceedings of 17th International World Wide Web Conference (WWW'08)*.

[70] Lina Zhou and Pimwadee Chaovalit. Ontology-supported polarity mining. *Journal of the American Society for Information Science and Technology (JASIST)*, 59(1):98–110, 2008.

[71] Wei Wei and Jon Atle Gulla. Sentiment learning on product reviews via sentiment ontology tree. In *Proceedings of 48th Annual Meeting of the Association for Computational Linguistics (ACL'10)*, 2010.

[72] Yue Lu, Huizhong Duan, Hongning Wang, and ChengXiang Zhai. Exploiting structured ontology to organize scattered online opinions. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING'10)*, pages 734–742, 2010.

[73] J. Stasko and E. Zhang. Focus + context display and navigation techniques for enhancing radial, space-filling hierarchy visualizations. In *IEEE Symposium on Information Visualization*, pages 57–65, 2000.

[74] Kristina Toutanova and Christopher D. Manning. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the Empirical Methods in Natural Language Processing Conference (EMNLP'00)*, Hong Kong, China, 2000.

[75] M. F. Porter. An algorithm for suffix stripping. *Readings in Information Retrieval*, pages 313–316, 1997.

[76] H. H. Hoos and T. Stützle. *Stochastic Local Search: Foundations and Applications*. Morgan Kaufmann, San Francisco, 2005.

[77] O. J. Mengshoel. *Efficient Bayesian Network Inference: Genetic Algorithms, Stochastic Local Search, and Abstraction*. PhD thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, April 1999.

[78] J. D. Park and A. Darwiche. Complexity results and approximation strategies for MAP explanations. *Journal of Artificial Intelligence Research (JAIR)*, 21:101–133, 2004.

[79] Tony Mullen and Nigel Collier. Sentiment analysis using support vector machines with diverse information sources. In *Proceedings of EMNLP'04*, pages 412–418.

[80] Ellen Riloff, Siddharth Patwardhan, and Janyce Wiebe. Feature subsumption for opinion analysis. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP'06)*, pages 440–448, 2006.

[81] Zhongwu Zhai, Hua Xu, Jun Li, and Peifa Jia. Feature subsumption for sentiment classification in multiple languages. In *Proceedings of 14th Pacific-Asia Conference (PAKDD'10)*, volume 6119 of *Lecture Notes in Computer Science*, pages 261–271, 2010.

[82] Xue Bai. Predicting consumer sentiments from online text. *Decision Support Systems*, 50(4):732–742, 2011.

[83] Alistair Kennedy and Diana Inkpen. Sentiment classification of movie reviews using contextual valence shifters. *Computational Intelligence*, 22(2):110–125, 2006.

[84] Ahmed Abbasi, Hsinchun Chen, and Arab Salem. Sentiment analysis in multiple languages: Feature selection for opinion classification in web forums. *ACM Trans. Inf. Syst*, 26(3), 2008.

[85] Justin Martineau and Tim Finin. Delta TFIDF: An improved feature space for sentiment analysis. In *Proceedings of the Third International Conference on Weblogs and Social Media (ICWSM'09)*, 2009.

[86] Irena Koprinska Tim O Keefe. Feature selection and weighting methods in sentiment analysis. In *Proceedings of the 14th Australian Document Computing Symposium*, 2009.

[87] Alexander Pak and Patrick Paroubek. Text representation using dependency tree subgraphs for sentiment analysis. In *Proceedings of DASFAA 2011 Workshops*, volume 6637 of *Lecture Notes in Computer Science*, pages 323–332. Springer, 2011.

[88] Mohammed Rushdi-Saleh, Maria Teresa Martín-Valdivia, Arturo Montejo Ráez, and Luis Alfonso Ureña López. Experiments with SVM to classify opinions in different domains. *Expert Syst. Appl*, 38(12):14799–14804, 2011.

[89] Bas Heerschop, Frank Goossen, Alexander Hogenboom, Flavius Frasincar, Uzay Kaymak, and Franciska de Jong. Polarity analysis of texts using discourse structure. In *Proceedings of the 20th ACM Conference on Information and Knowledge Management (CIKM'11)*, 2011.

[90] Yelena Mejova and Padmini Srinivasan. Exploring feature definition and selection for sentiment classifiers. In *Proceedings of the Fifth International Conference on Weblogs and Social Media*, 2011.

[91] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of The 49th Annual Meeting of the Association for Computational Linguistics (ACL'11)*, pages 142–150, 2011.

[92] Ahmed Abbasi, Stephen L. France, Zhu Zhang, and Hsinchun Chen. Selecting attributes for sentiment classification using feature relation networks. *IEEE Trans. Knowl. Data Eng*, 23(3):447–462, 2011.

[93] Nicolò Cesa-Bianchi, Claudio Gentile, and Luca Zaniboni. Incremental algorithms for hierarchical classification. *Journal of Machine Learning Research (JMLR)*, 7:31–54, 2006.

[94] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., 2006.

[95] Vincent Ng, Sajib Dasgupta, and S. M. Niaz Arifin. Examining the role of linguistic knowledge sources in the automatic identification and classification of reviews. In *Proceedings of 44th Annual Meeting of the Association for Computational Linguistics (ACL'06)*, 2006.

Part II

Selected Papers

Chapter 6

Paper One

Wei Wei, Nan Cao, Jon Atle Gulla and Huamin Qu, “ImpactWheel : Visual Analysis of the Impact of Online News", Proceedings of the 10th IEEE/WIC/ACM International Conference on Web Intelligence, pp 465-474, 2011, IEEE Computer Society Washington, DC, USA, ISBN 978-0-7695-4513-4.

Is not included due to copyright

Chapter 7

Paper Two

Wei Wei, Jon Atle Gulla, and Zhang Fu, “Enhancing Negation-Aware Sentiment Classification on Product Reviews via Multi-Unigram Feature Generation", Proceedings of the Sixth International Conference on Intelligent Computing, Advanced Intelligent Computing Theories and Applications, Lecture Notes in Computer Science Volume 6215, 2010, pp 380-391, Springer-Verlag Berlin, Heidelberg, ISBN 3-642-14921-9 978-3-642-14921-4.

Is not included due to copyright

Chapter 8

Paper Three

Wei Wei, Ole J. Mengshoel and Jon Atle Gulla,“Stochastic Search for Effective Features for Sentiment Classification", in review with Data & Knowledge Engineering.

Stochastic Search for Effective Features for Sentiment Classification

Wei Wei[1] Ole J. Mengshoel[2] Jon Atle Gulla[1]
[1]Department of Computer and Information Science, Norwegian University of Science and Technology
{wwei,jag}@idi.ntnu.no
[2]Department of Electrical and Computer Engineering, Carnegie Mellon University
ole.mengshoel@sv.cmu.edu

Abstract

In this paper, we propose a fully-automatic Effective Feature Search (EFS) framework to enhance the performance of sentiment classifiers from the perspective of feature selection. Taking advantage of high frequency Co-occurring-Term (CoT) patterns, our EFS framework first generates unigram feature candidates and complex CoT feature candidates in the feature generation step. In the feature pruning step, a Stochastic Local Search (SLS) process addresses the feature subset selection problem. A hill-climbing SLS algorithm is developed to search for an optimized feature subset in the SLS process. The proposed EFS framework is empirically analyzed in extensive experiments. Performance comparison on two standard datasets shows that our proposed EFS framework is comparatively superior to existing state-of-the-art approaches in that our EFS framework is highly accurate, robust, and needs small human efforts. Experiments using three unigram generation algorithms, i.e., Term Frequency (TF), χ^2 (CHI), and Information Gain (IG), and four machine learning classifiers, i.e., Support Vector Machine (SVM), Naive Bayes (NB), Decision Tree (DT), and k Nearest Neighbor (KNN), demonstrate the general effectiveness of our EFS framework. Further impact analysis on the parameters λ, κ, and R of the proposed hill-climbing SLS algorithm empirically studies the trade-offs for setting these parameters in the SLS process.

Keywords: Sentiment Analysis, Sentiment Classification, Feature Selection, Stochastic Local Search

1. Introduction

In this fast-paced information era, it becomes more and more easy for people to access and share their opinions on the World Wide Web (WWW). People generate a large amount of text on various Web sites such as TripAdvisor[1] and Twitter[2],etc. The online user-generated reviews are usually rich in opinions and can be very useful information for potential customers, online advertisers as well as product manufacturers. However, as the amount of opinion information grows rapidly, it becomes impossible for humans to manually collect and digest these opinion-rich texts exhaustively. To alleviate this problem, research on sentiment analysis has emerged as a popular topic at the crossroads of information retrieval and computational linguistics.

One key problem of sentiment analysis is sentiment classification, which aims at classifying a review text as positive or negative. Early work, e.g., [1, 2], relies on adjectives to automatically decide sentiment orientation of documents. Pang et al. studied sentiment classification using machine learning techniques [3]. Although it has been claimed that standard machine learning techniques outperform human-produced baselines, there is also evidence that machine learning techniques do not perform as well on sentiment classification as on traditional topic-based text classification [3].

There exist at least two challenges for sentiment classification using machine learning techniques. First, sentiment orientation of words is rather dependent on topics. Although there are some general applicable sentiment expression words, e.g., "good" and "bad", which always hold consistent sentiment orientation in different topics, it is also not difficult to find words that might have different and even opposite opinions in different contexts. For example, the term "high" is a positive adjective to describe screen resolution of a camera while it becomes a negative adjective when it is used with a camera's price. In this way, when the term "high" is used as a feature in a machine learning classifier it might mislead the classifier in its learning process even if all the training data are collected from the same domain, e.g., product reviews on cameras. Second, negation expression is another potential problem for machine learning classifiers in sentiment classification. Negation words including "not", "never", "no", etc., are considered meaningless stop words and usually filtered out from feature set in traditional topic-based text classification. However, these negation words are very important

[1] http://www.tripadvisor.com/

[2] http://twitter.com/

signals in the sentiment classification process since their existence might overturn the classification results. Therefore, without elegantly dealing with significant negation words, machine learning classifiers are prone to err in sentiment classification.

In this paper, we propose an Effective Feature Search (EFS) framework that makes a novel connection between feature candidate generation and a Stochastic Local Search (SLS) process to enhance performance of machine learning classifiers for sentiment classification. The purpose of the proposed EFS framework is to search for effective features that can alleviate challenges of sentiment classification by machine learning classifiers. There are two important steps, i.e., **feature generation step** and **feature pruning step**, in our proposed framework.

First, in the feature generation step, we utilize filter-based methods [4] to select feature candidates not only considering unigram features but also taking complex features[3] into consideration. Unlike previous work, where complex features are only extracted from a manually built lexicons [5], human-defined patterns [6, 7], or preselected categories [8], our work takes advantage of high frequency Co-occurring Term (CoT) patterns and calls the generated complex features CoT features[4]. The rationale of making use of CoT patterns is based on an intrinsic property of online opinion-rich review texts: vocabularies used in opinion expression on the same topic are limited and tend to be highly repeated [10]. For example, single terms like "high" does not necessarily means positive sentiment. However, in a corpus of customer reviews on digital cameras terms "high" and "price" might co-occur together frequently and means definitely negative. In addition, negation words like "not" cannot be treated as an obvious feature for either positive class or negative class. However, terms like "not" and "good" together can be deemed as negative and usually co-occur in many negative expressions. Hence, we believe that within frequently co-occurring terms there might exist effective complex features that are highly capable of deciding the expressed sentiment orientation.

Second, high frequency CoT patterns not only produce effective complex features like "staff nice" but also bring useless candidates like "staff service". Therefore, a feature pruning step is developed and devoted to searching for an optimized feature subset out of the feature candidate set from the feature generation

[3]Compared with simple unigram features, a complex feature is a combination of more than one terms.

[4]To avoid unnecessary confusion with n-gram which is a contiguous sequence of terms in a text by definition [9], we coin the term "CoT" to capture the significance of frequently co-occurring term patterns.

step. In the feature pruning step, we map the feature set optimization process to a Stochastic Local Search (SLS) process. In the proposed SLS model, a wrapper-based selection is adopted to score each selected feature subset with an objective function tailored to the classifier. A hill-climbing SLS algorithm is developed in the model to ensure quickly finding a local optima.

Our proposed EFS framework is empirically analyzed in extensive experiments. Performance comparison on two standard datasets shows that our proposed EFS framework is comparatively superior to existing state-of-the-art approaches in that our EFS framework is highly accurate, robust, and needs small human efforts. Experiments using three unigram generation algorithms, i.e., Term Frequency (TF), χ^2 (CHI), and Information Gain (IG), and four machine learning classifiers, i.e., Support Vector Machine (SVM), Naive Bayes (NB), Decision Tree (DT), and k Nearest Neighbor (KNN), demonstrate the general effectiveness of the proposed EFS framework. Further impact analysis on the parameters λ, κ, and R of the proposed hill-climbing SLS algorithm empirically studies the trade-offs for setting these parameters in the SLS process.

As far as we know, our proposed EFS framework is the first approach that takes advantage of CoT patterns and search for effective features in an SLS process. The EFS framework is a fully automatic process in that unlike previous work it requires no extra resources [11], no human-developed lexicons [12], and no human efforts and interactions [13] in the training and classification process. Hence, our proposed EFS framework is quite general and can be applied on data from different topic domains.

The remainder of the paper is organized as follows. In Section 2, we discuss an overview of related work on sentiment analysis. In Section 3, we present our proposed EFS feature selection framework. Empirical analysis on the proposed framework is presented in Section 4. Finally, we conclude the paper and discuss our future work in Section 5.

2. Related Work

Sentiment analysis is usually concerned with opinion detection[5] and sentiment classification[6]. Opinion detection attempts to determine whether a text is objective

[5]"Opinion detection" is also called "subjectivity/objectivity identification" in some literatures.

[6]"Sentiment classification" is also called "sentiment orientation" or "polarity classification", etc. in some literatures.

or subjective (e.g., [14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27]). Sentiment classification aims at classifying whether a subjective text contains positive or negative sentiments. Sentiment classification on subjective texts (e.g. online reviews) can be conducted either on an overall document level (e.g., [3, 28, 5], etc.) or on more fine-grained aspect-based level (e.g. [29, 30, 31, 32, 33, 10, 34, 35, 36, 37, 38]). This work focuses on document overall sentiment classification.

Document overall sentiment classification is concerned with analyzing a document's overall sentiment, which can be solved by two main approaches. Lexicon-based methods [39] conduct sentiment analysis by inferring a document's overall sentiment from sentiments of words [2, 40, 41, 42, 1, 43, 44, 25] or phrases [45, 46, 47]. Machine learning approaches build classifiers to classify a document's overall sentiment through a supervised [3] or unsupervised [48] learning process. This work focuses on supervised sentiment classification.

Since Pang et al. [3] studied sentiment classification using machine learning techniques, a lot of work has addressed the document overall sentiment classification problem in a supervised text classification process. Within exiting publications there exist various techniques to improve performance of traditional topic-based classifiers on sentiment classifications. These techniques include feature selection approaches, identifying more important subjective portions of texts [28, 49], using POS [50] or syntax [51] information, learning from human-annotator rationale [52] or human interaction [13]. This paper aims at improving performance of sentiment classifiers from the perspective of feature selection.

There are various feature selection techniques for sentiment classification. Like feature selection for traditional topic-based text classification, weighting methods are usually used to score features, ranging from purely using TFIDF and its variants [53, 54, 55] to involving extra lexicon [42] to assist feature scoring process [56, 57]. Besides using knowledge from lexicons, feature selection schemes for sentiment classification also make use of Natural Language Processing (NLP) techniques such as stemming and Part-Of-Speech (POS) tagging [58] as well as syntax models [51]. Instead of using only unigram features, existing works also try to capture dependencies among words [59] aiming at extracting complex features, e.g., N-gram features. Whitelaw et al. present a method to extract appraisal group features for sentiment classification [5]. However, this work [5] has the limitation of relying on a manually built lexicons, which generates effective features limited by the involved lexicons. Riloff et al. [6] propose a feature subsumption hierarchy to first generate complex features and then reduce the unnecessary ones. Ahmed et al. follow the similar generation and reducing framework in their later works [7, 8]. However, these existing works requires either human-defined ex-

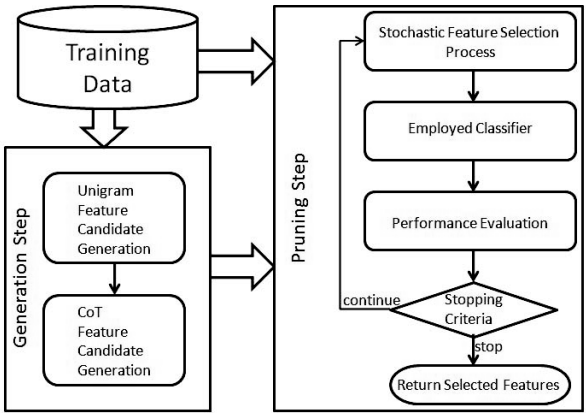


Figure 1: An Overview of EFS Framework

traction patterns [6, 7] or human-selected rules and feature categories [8] as input to feature generation step. In addition to human efforts, this also limits the generated complex features to predefined patterns and categories. In contrast, with less human efforts our proposed EFS framework is a fully automatic process which takes the advantage of the property of review texts and generate complex feature candidates through mining of CoT patterns.

3. Effective Feature Search Framework

In this section, we present the proposed Effective Feature Search (EFS) framework. An overview of the EFS framework is presented in Fig. 1. As depicted in the Fig. 1, the proposed EFS framework consists of two steps: the feature generation step and the feature pruning step. The feature generation step produces an initial set of feature candidates. The feature pruning step removes redundant, useless, or noisy feature candidates from the initial candidate set so that performance of the employed sentiment classifiers will be maximized in an SLS process. We now discuss these two steps in more detail.

3.1. Feature Generation Step

As discussed in Section 1, effective features for sentiment classifiers involve both simple features and complex features. Frequent CoT patterns are believed to contain effective complex features. In the feature generation step, unigram feature candidates are first generated. Based on these unigram feature candidates, complex feature candidates are generated from frequent CoT patterns.

3.1.1. Unigram feature candidate generation

Like feature selection for traditional topic-based text classifications, unigram feature candidates can be generated using different feature selection algorithms[7], e.g., Term Frequency (TF) based algorithm [60], χ^2-statistic algorithm [60], and information gain based algorithm [61], etc. The generated unigram feature candidate set, denoted by F_u, in this step will not only be included in the initial feature candidate set but also form the basis input to CoT feature candidate generation.

3.1.2. CoT feature candidate generation

Before we present the CoT feature candidate generation, we first give a formal definition on what CoT is.

Definition 1. *(CoT): CoT is an abbreviation for Co-occurring Terms. Terms are considered as CoT if they are presented together in one document and occur together in a sequence of text without being separated by any punctuation*[8].

By Definition 1 we know that two terms occurring in one document are not necessarily CoT. For example, in a corpus of reviews on hotels it is very common to find a snippet like "friendly staff, good location". Here, "friendly-staff" is considered as CoT though "staff-location" cannot be considered as CoT by the above definition, since they are separated by a comma.

Algorithm 1 describes our CoT feature candidate generation process, where F_u is a set of unigram feature candidates generated by a classic feature generation algorithm. The parameter l limits the max number of terms being considered as CoT. The parameter t_{cot} serves as a threshold to filter out low-frequency CoT patterns. The operation $u \ncong f$ means that the unigram term u is not equal to any term of CoT f. The operation $(u \oplus f)$ means that a new CoT which is made up of the unigram term u and all the terms of CoT f. After finishing the CoT feature candidate generation process described by the Algorithm 1, all the CoT with frequency above t_{cot} and length within l will be selected to the set F_{cot}. The generated candidates from CoT patterns are named CoT features candidates. We believe the generated CoT candidate set F_{cot} might contain effective features of significant opinion information, since vocabularies used in opinion expression on the same topic are limited and tend to be highly repeated [10]. For example, single

[7]In this work, "feature selection algorithms" and "unigram generation algorithms" are exchangeably used.

[8]A list of punctuation in our experiments include comma, periods, question marks, exclamation marks, colons, and semicolons.

Algorithm 1: CoT feature candidate generation

Input: D: a document set
F_u: unigram feature candidate set
$F_t \leftarrow F_u$: initialize temporary set
$F_{cot} \leftarrow \emptyset$: empty CoT feature candidate set
$l(l \in Z^+)$: max number of terms as CoT
t_{cot}: CoT frequency threshold
Output: F_{cot}

$r \leftarrow 2$;
while $r \leq l$ & $F_t \neq \emptyset$ **do**
  $F_r \leftarrow \emptyset$;
  forall $u \in F_u$ **do**
    forall $f \in F_t$ **do**
      if $u \not\cong f$ & $(u \oplus f) \notin F_{cot}$ **then**
        $t \leftarrow 0$;
        forall $d \in D$ **do**
          $c \leftarrow$ count $(u \oplus f)$ in d;
          $t \leftarrow t + c$
        end
        if $t \geq t_{cot}$ **then**
          $F_{cot} \leftarrow F_{cot} \bigcup \{(u \oplus f)\}$;
          $F_r \leftarrow F_r \bigcup \{(u \oplus f)\}$;
        end
      end
    end
    $F_t \leftarrow F_r$;
  end
  $r \leftarrow r + 1$;
end
return F_{cot};

terms like "high" does not necessarily means positive sentiment. However, in a corpora of customer reviews on digital cameras terms "high" and "price" might co-occur together frequently and means definitely negative. In next section, we present our feature pruning step on selecting effective features from the generated feature candidate sets.

3.2. Feature Pruning Step

The feature candidate set F to be pruned is a union of the unigram feature candidate set F_u and the CoT feature candidate set F_{cot}, i.e., $F = F_u \cup F_{cot}$. F might contain both effective and useless features. The purpose of our feature pruning step is to select a subset of features from F which are considered as useful for sentiment classifiers, while ignoring the rest. In the feature pruning step, we map the feature subset optimization problem to a Stochastic Local Search (SLS) model as follows.

3.2.1. Stochastic Local Search Model

Let $f_i \in F$ $(1 \leq i \leq n)$ respectively represent each feature candidate from F_u and $n = |F_u|$. Let $f_j \in F$ $(n+1 \leq j \leq n+m)$ respectively represent each feature candidate from F_{cot} and $m = |F_{cot}|$. An Stochastic Local Search Model (SLSM) for feature subset optimization is formulated as follows.

An SLSM is formally defined as a 4-tuple $M = (S, \mathcal{N}, \mathcal{G}, \mathcal{O})$ where S is a set of state vectors and forms the search space. Each state vector s $(s \in S)$ represents a feature subset of F_s[9], where s is an $(n+m)$-dimensional binary vector $(s_1, s_2, ..., s_n, s_{n+1}, ..., s_{n+m})$ and each value s_k $(1 \leq k \leq n+m)$ encodes whether the feature f_k $(f_k \in F)$ is selected: 1 means being selected while 0 means the opposite. $\mathcal{N}$ is a neighborhood relation , i.e., $\mathcal{N} \subseteq S \times S$; $\mathcal{G} : S \to \mathbb{R}$ is an evaluation function that maps each state $s(s \in S)$ to a real number score $g(g \in \mathbb{R})$; $\mathcal{O}$ defines optimal states $\mathcal{O} = \{s^* | s^* = \arg_{max}\, \mathcal{G}(s)\}$.

In the above definition, the neighborhood relation $\mathcal{N}$ defines neighboring states of each state $s \in S$. In an SLS algorithm, the search iteratively moves from one state to its neighboring states and scores are calculated by the evaluation function $\mathcal{G}$. Unlike in the feature generation step, where feature candidate selection is performed in a filter-based manner which ranks each feature candidate separately, in this feature pruning step each feature subset is evaluated as a whole by $\mathcal{G}$. In order that the optimized feature subset is tailored to a particular machine learning

[9]In this paper, when we say a feature subset s it means F_s.

classifier c, we adopt a wrapper-based approach and the evaluation function $\mathcal{G}_c$ is a mapping function from a feature subset s to the classifier c's empirical accuracy on s. In the rest of this section, we propose an SLS algorithm that serves as a heuristic approach to feature subset optimization.

3.2.2. Stochastic Local Search Algorithm

To search for an optimal feature subset is an NP-hard problem. The stochastic local search (SLS) approach has proven to be highly competitive for solving a range of hard computational problems including satisfiability of propositional logic formulas [62, 63, 64, 65] as well as computing the most probable explanation [66, 67, 68] and the maximum a posteriori hypothesis [69, 70] in Bayesian networks.

Our proposed SLS algorithm is described in Algorithm 2, where c indicates an employed machine learning classifier. F denotes the feature candidate set. S_t serves as a tabu list that records all the visited states so that the algorithm does not consider a state repeatedly. The parameters θ_u and θ_{cot} are inputs to the function INITIALIZATION(θ_u, θ_{cot}) and respectively control how many percentage of unigram candidates in F_u and CoT candidates in F_{cot} that are included in an initial state by random selection. The parameter λ is input to the function NEXTSTEP(λ), which decides the next step to be a noise step with probability of λ or a greedy step with probability $1 - \lambda$. The NEIGHBOR(s, S_t) function returns a random neighbor state of s that is not recorded in the tabu list S_t.[10] The parameter κ limits how many neighbor states are evaluated in a greedy step. The parameter R defines how many steps are performed in each try, and MAX-TRIES defines the maximum number of tries before termination performed by the algorithm.

Algorithm 2 works in the following way. In each try, an initial state is randomly created. The search begins with this initial state. Then the algorithm goes through an iterative hill-climbing process in R steps. Each step of the process is either a noise step or a greedy step. If it is a noise step, the search moves to a neighbor state. If it is greedy step, the search test κ neighbors of the current state and goes to a neighbor state with maximum score. In each step, if the score p in that step is not worse than the recorded score p^*, then p^* is updated with p and the recorded state s^* is updated with s. After algorithm finishes with R steps in MAX-TRIES tries, the recorded state s^* is returned. In this way, the correspond feature subset F_{s^*} is optimized for the classifier c.

[10] A neighbor state of s is a state s' so that hamming distance between s and s' is 1.

Algorithm 2: SLS Algorithm

Input: c: machine learning classifier F: feature candidate set
$S_t \leftarrow \emptyset$: initialize an empty tabu list θ_u: unigram candidates initialization parameter
θ_{cot}: CoT candidates initialization parameter λ: noise parameter
κ: greedy parameter R: number of flips per try
MAX-TRIES: number of tries
Output: s^*: optimized state of feature set

$t \leftarrow 1$;
$g^* \leftarrow 0$; // performance score record
while $t \leq$ *MAX-TRIES* **do**
  $s \leftarrow$ INITIALIZE(θ_u, θ_{cot});
  $g \leftarrow \mathcal{G}_c(s)$; // calculate performance
  if $g \geq g^*$ **then**
    $g^* \leftarrow g$; // record performance
    $s^* \leftarrow s$; // record feature set
  end
  $S_t \leftarrow S_t \cup \{s\}$; // add to taboo list
  $r \leftarrow 1$;
  while $r \leq R$ **do**
    next$\leftarrow$NEXTSTEP(λ);
    if *next is a noise step* **then**
      $s \leftarrow$NEIGHBOR(s, S_t);
      $g \leftarrow \mathcal{G}_c(s)$;
      $S_t \leftarrow S_t \cup \{s\}$;
      if $g \geq g^*$ **then**
        $g^* \leftarrow g$;
        $s^* \leftarrow s$;
      end
    end
    if *next is a greedy step* **then**
      $j \leftarrow 1$;
      $g_j^* \leftarrow 0$;
      while $j \leq \kappa$ **do** // try κ neighbors
        $s_j \leftarrow$NEIGHBOR(s, S_t);
        $g_j \leftarrow \mathcal{G}_c(s_j)$;
        $S_t \leftarrow S_t \cup \{s_j\}$;
        if $g_j \geq g_j^*$ **then**
          $g_j^* \leftarrow g_j$;
          $s_j^* \leftarrow s_j$;
        end
        $j \leftarrow j + 1$;
      end
      $s \leftarrow s_j^*$; // choose best neighbor
      if $g_j^* \geq g^*$ **then**
        $g_j^* \leftarrow g^*$;
        $s^* \leftarrow s_j^*$;
      end
    end
    $r \leftarrow r + 1$;
  end
  $t \leftarrow t + 1$;
end
return s^*;

4. Experiments

In this section, we conduct extensive experiments to empirically analyze the proposed EFS framework. Our experiments are intended to address the following questions:

1. How does our proposed framework compare with published state-of-the-art approaches?
2. How does our proposed EFS framework behave when different unigram generation algorithms and different machine learning classifiers are employed in the framework.
3. How do the SLS algorithm's parameters λ, κ, and R impact on the SLS process?

4.1. Metrics

To evaluate performance of sentiment classifiers, we adopt evaluation metric *accuracy* which is the percentage of correctly labeled reviews out of total reviews and is generally used in most of the previous work:

$$\text{accuracy} = \frac{\#\text{correctly labeled reviews}}{\#\text{total reviews}}.$$

4.2. Datasets and Tools

Our proposed EFS framework is empirically analyzed using four datasets. Table 1 summarizes statistics for the four datasets.[11] The datasets DM1400 and DM2000 are from the movie review domain. They were originally introduced by Pang et al. in 2002 [3] and 2004 [28] respectively. Both the DM1400 and DM2000 have been extensively used in the past. In our experiments, we compare our proposed EFS framework with recent published state-of-the-art methods on these two standard datasets. The dataset DH5000 are collected from a hotel booking website.[12] On the hotel booking website, positive reviews and negative reviews are separated presented. We crawled 1505605 positive reviews and 1015700 negative reviews and randomly selected 2500 reviews from both categories to form the DH5000 dataset. Unlike reviews in DM1400 and DM2000, where each review

[11] Headers of the table means respectively names of dataset, domain of dataset, number of positive reviews, number of negative reviews, total number of words, total number of unique terms, and average number of words per review in each dataset.

[12] `http://www.booking.com/`

Table 1: Dataset Statistics

Name	Domain	Pos	Neg	Words	Unique	Avg.
DM1400	Movie Review	700	700	909546	41389	649.68
DM2000	Movie Review	1000	1000	1289584	47986	644.79
DH5000	Hotel Review	2500	2500	98093	6556	19.62
DH1000	Hotel Review	500	500	19070	2669	19.07

is on average made up of around 650 words, each review in DH5000 is rather shorter (19.62 words per review on average). Introduction of DH5000 serves as a supplement to validate our approach on classifying short statements.[13] The dataset DH1000 is a small sample of DH5000. Since running for each parameter setting of the SLS algorithm is a time consuming task, due to limitation of time and computational devices, DH1000 is used to test the SLS algorithm's parameters' impact on search process.

In our experiments, each dataset is preprocessed to remove stop words. Unlike traditional list of stop words, negation words, e.g., "no," "not," "never," etc. are not treated as stop words in our preprocess. The Porter stemmer algorithm [71] is used to stem all terms remaining in the dataset. The machine learning classifiers employed in this work are implemented using an open source machine learning software Weka[14] [72]. To catch the statistical significance, all results reported in this work are based on 10-fold cross-validation.

4.3. Performance Comparison

In this section, we report on experiments that evaluate the performance of our proposed EFS framework. We compare our approach with the state-of-the-art methods on the two standard datasets DM1400 and DM2000. In the experiments, Information Gain (IG) feature selection algorithm is employed in the proposed EFS framework for unigram feature candidate generation. The parameter t_{cot} for CoT feature candidate generation is set to 10. The parameters of the SLS algorithm are respectively set: $\theta_u = 1$, $\theta_{cot} = 0.9$, $\lambda = 0.1$, $\kappa = 300$, $R = 500$. The SLS algorithm is run in 5 tries.

Table 2 chronologically summarizes results on the two standard datasets reported in recent ten years. The column "Ex-Efforts" indicates whether the related

[13] As a contribution of this work, we make this dataset accessible at `http://anonymous-for-blind-review/hotelreivew.zip`

[14] `http://www.cs.waikato.ac.nz/ml/weka/`

Table 2: Performance Comparison

Dataset	Work	Year	Ex-Efforts	Accuracy
DM1400	Pang et al. [3]	2002	No	82.90%
DM1400	Mullen&Collier [56]	2004	Yes	86.00%
DM1400	Riloff et al. [6]	2006	Yes	82.70%
DM1400	Zhai et al. [73]	2010	No	84.30%
DM1400	Bai [59]	2011	No	78.08%
DM1400	our EFS framework	2012	No	**90.13%**
DM2000	Pang&Lee [28]	2004	No	87.20%
DM2000	Whitelaw et al. [5]	2005	Yes	**90.20%**
DM2000	Kennedy&Inkpen [74]	2006	Yes	86.20%
DM2000	König&Brill [13]	2006	Yes	**91.00%**
DM2000	Zaidan et al. [52]	2007	Yes	**92.20%**
DM2000	Abbasi et al. [8]	2008	Yes	**91.70%**
DM2000	Martineau&Finin [53]	2009	No	88.10%
DM2000	O'Keefe&Koprinska [57]	2009	No	87.15%
DM2000	Taboada et al. [39]	2011	Yes	76.63%
DM2000	Pak&Paroubek [75]	2011	Yes	85.10%
DM2000	Saleh et al. [54]	2011	No	86.19%
DM2000	Heerschop et al. [49]	2011	Yes	81.00%
DM2000	Mejova et al. [75]	2011	Yes	87.50%
DM2000	Duric&Song [51]	2011	No	87.50%
DM2000	Maas et al. [76]	2011	No	88.90%
DM2000	Abbasi et al. [7]	2011	Yes	89.65%
DM2000	Bai [59]	2011	No	**92.70%**
DM2000	our EFS framework	2012	No	**92.37%**

work uses extra human efforts as part of their proposed methods. Performance above 90% on each dataset is bolded. Best performance on each dataset is underlined. From the Table 2, we can see that our approach achieves the best performance (90.13% accuracy) on DM1400. Among the six methods that achieve classification performance above 90% on DM2000, most of them require extra human efforts, e.g., manually built lexicons [5], predefined extraction patterns [6, 7], and preselected feature categories [8], as inputs to their methods. In comparison, our proposed EFS framework is a fully automatic process. Although Bai's method [59] currently is best on DM2000, our approach shows comparably good (92.70% v.s. 92.37%) on the same dataset. In addition, our approach beats Bai's method [59] on the DM1400 with more than 11% accuracy, which suggests our approach is more robust than Bai's approach [59]. Therefore, through performance comparison on two standard datasets, we conclude that our proposed EFS framework is generally superior to existing state-of-the-art approaches in that our approach is high accuracy, more robust, and needs small human efforts.

4.4. Initialization Test and Performance Analysis

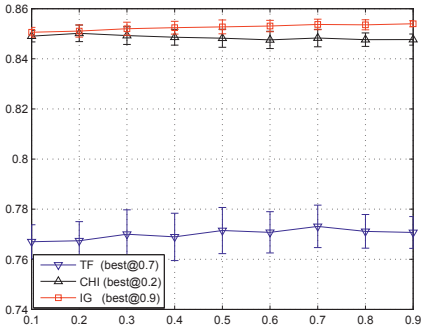
In this section, we conduct experiments to analyze general effectiveness of the proposed EFS framework. We aim at discovering how is our framework affected when different unigram generation algorithms and different machine learning classifiers are employed. To cover characteristics of long reviews and short reviews in domains movie review and hotel review, the analysis is conducted against two datasets, i.e., DM1400 and DM2000. Three different unigram generation algorithms[15], i.g., Term Frequency (TF) based algorithm [60], χ^2-statistic algorithm [60], and information gain based algorithm [61], and four machine learning classifiers, i.e., Support Vector Machine (SVM), Naive Bayes (NB), Decision Tree (DT), and k Nearest Neighbor (KNN), are studied in the EFS framework.[16]

4.4.1. Initialization Test

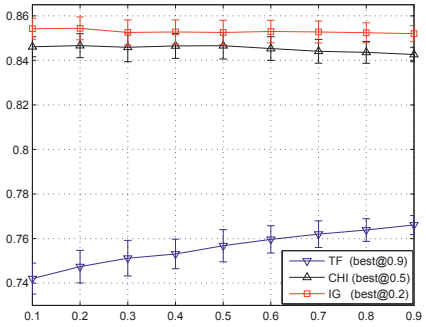
In Algorithm 2, the parameters θ_u and θ_{cot} in the INITIALIZE(θ_u, θ_{cot}) function control respectively initial percentage of unigram candidates and CoT candidates

[15]We does not report on Mutual Information (MI) based algorithm since we found out that too few CoT candidates can be generated based on the unigram candidate set generated from MI algorithm.

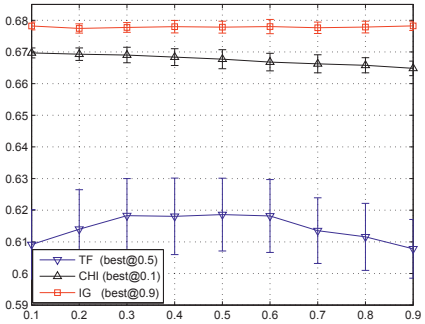
[16]In our experiment, we also tried to evaluate Logistic Regression in the proposed EFS framework. Since the SLS process for LR consumes too much time, due to limitation of time and our current computing devices, we do not report on LR in this submission.



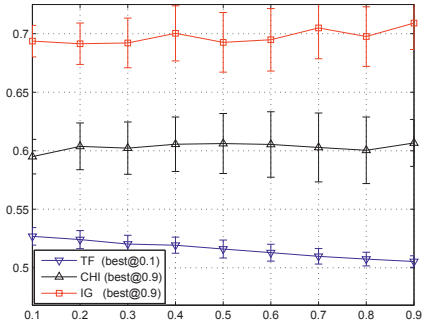
(a) NB Initialization on DM1400



(b) SVM Initialization on DM1400

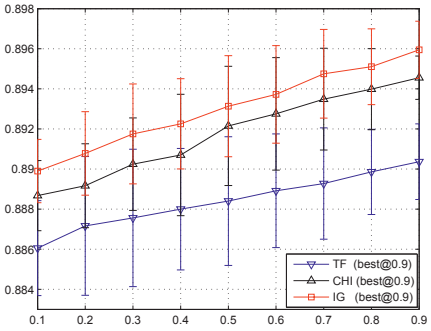


(c) DT Initialization on DM1400

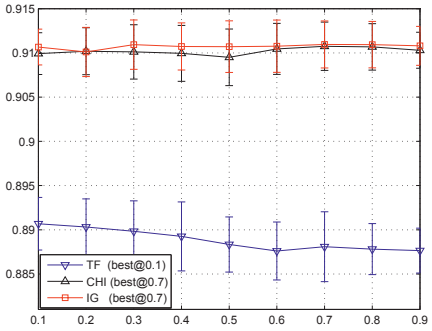


(d) KNN Initialization on DM1400

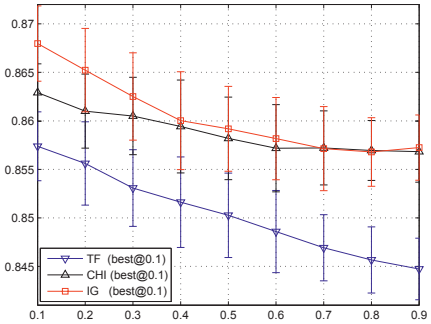
Figure 2: Experimental results on testing initialization parameter θ_{cot} (x-axis) of the Algorithm 2, varying unigram generation algorithms (TF, CHI, IG) and machine learning classifiers (SVM, NB, DT, KNN) on the datasets DM1400. Best values of the initialization parameter θ_{cot} for each variants are presented in the figure.



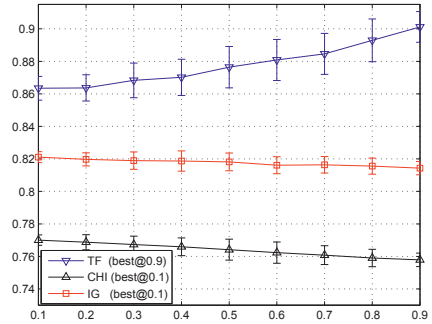
(a) NB Initialization on DH5000



(b) SVM Initialization on DH5000



(c) DT Initialization on DH5000



(d) KNN Initialization on DH5000

Figure 3: Experimental results on testing initialization parameter θ_{cot} (x-axis) of the Algorithm 2, varying unigram generation algorithms (TF, CHI, IG) and machine learning classifiers (SVM, NB, DT, KNN) on the datasets DH5000. Best values of the initialization parameter θ_{cot} for each variants are presented in the figure.

Table 3: Performance of EFS on DM1400

Experiments on DM1400	Unigram Generation			CoT Generation		Classification Performance			
	FS	Parameter	# unigrams	t_{cot}	# CoTs	SVM	NB	DT	KNN
Exp.1.1	TF	tf=100	887	-	-	73.70%	76.86%	60.14%	54.67%
Exp.1.2	TF	tf=100	887	10	6814	79.54%	81.57%	66.49%	63.71%
Exp.1.3	CHI	TopN=887	887	-	-	84.83%	84.91%	67.03%	58.95%
Exp.1.4	CHI	TopN=887	887	10	618	88.51%	86.72%	68.23%	74.91%
Exp.1.5	IG	TopN=887	1009	-	-	85.69%	84.95%	67.85%	69.19%
Exp.1.6	IG	TopN=887	1009	10	414	**90.09%**	86.21%	68.91%	85.29%

Table 4: Performance of EFS on DH5000

Experiments on DH5000	Unigram Generation			CoT Generation		Classification Performance			
	FS	Parameter	# unigrams	t_{cot}	# CoTs	SVM	NB	DT	KNN
Exp.2.1	TF	tf=10	743	-	-	89.16%	88.55%	85.99%	86.20%
Exp.2.2	TF	tf=10	743	5	2805	91.39%	90.49%	87.57%	92.69%
Exp.2.3	CHI	TopN=743	845	-	-	90.96%	88.83%	86.49%	77.23%
Exp.2.4	CHI	TopN=743	845	5	1838	93.69%	90.98%	88.36%	82.68%
Exp.2.5	IG	TopN=743	784	-	-	91.15%	88.87%	87.13%	82.25%
Exp.2.6	IG	TopN=743	784	5	1718	**93.79%**	90.97%	88.73%	87.42%

to be included in the SLS process. If these parameters are set too large, a lot of noisy candidates might be included in the initial set. If they are set too small, some useful candidates might be filtered out of the initial set. Either way might lower the probability of finding an optimal solution. In this subsection, we test the initialization parameter θ_{cot} from 0.1 to 0.9 and set θ_u to 1 under assumption that the generated unigram candidates created by traditional feature selection algorithms are reasonable. For each value of θ_{cot}, we test 100 times initialization performance of our proposed framework. Average performance and deviations as well as best values of θ_{cot} for each combination of unigram generation algorithm and machine learning classifier on the two datasets are respectively presented in Fig. 2 and Fig. 3.

4.4.2. Performance Analysis

We study performance of the proposed EFS framework on DM1400 and DH5000 using different unigram generation algorithms and machine learning classifiers. In order to investigate direct effectiveness of our proposed EFS framework, in Exp.$1.x$ and $2.x$ ($x = 1, 3, 5$), we implement baseline methods as traditional text classification on selected unigram features: applying different machine learning classifiers (i.e., SVM, NB, DT, KNN) on features generated by different unigram generation algorithms (i.e., TF, CHI, IG) on both datasets DM1400 and DH5000. In addition, for each unigram generation algorithm and machine learning classi-

fier combination used in baseline methods, we implement our approach in Exp. $1.z$ and $2.z$ ($z = 2, 4, 6$). To avoid producing bias by different number of generated features in our experiments, we use comparable parameter setting for the three feature selection algorithms[17] so that a similar number of unigram candidates are generated. For the Algorithm 2, the initialization parameter θ_u is set to 1 and θ_{cot} is set to the value that produces the best performance recorded in the Fig. 2 and Fig. 3. Other parameters are set as $\lambda = 0.1$, $\kappa = 300$, $R = 200$ and the Algorithm 2 is run for just one try for each parameter setting.

The experimental results are summarized in Table 3 and Table 4. From both the tables, it is observed that experiments that using our approach (i.e., Exp. $1.z$ and $2.z$) beat their corresponding baselines using only unigram features (i.e., Exp. $1.x$ and $2.x$) on both datasets. This indicates that the our proposed EFS framework is stably effective independent of unigram generation algorithms and machine learning classifiers. Especially, in many experiments, e.g., Exp.1.6 on SVM and Exp. 2.2 on KNN, performance is observed with more than 4.5% boosts. Although KNN generally performs the worst, it is also the classifier that benefits from the EFS framework most. For example, in Exp.1.6 on KNN, a performance boost (i.e., 16.10% improvement) is observed. In all the experiments on both datasets DM1400 and DH5000, IG and SVM are confirmed to be the best combination that are employed in our proposed EFS framework.

4.5. Study Impact of Parameters

In this section, we design experiments to study the SLS parameters λ, κ, and R of the Algorithm 2 on a small sample dataset DH1000. In the experiments, Information Gain (IG) algorithm is used for unigram candidate generation. A Naive Bayes (NB) classifier is employed in the SLS process. The initialization parameters θ_u and θ_{cot} are respectively set to 1 and 0.9.

4.5.1. Impact of Parameter λ

The parameter λ controls the probability of taking a noise step. A low noise value enables an SLS algorithm to greedily climb hills without taking unnecessary downhill noise steps. A high noise value might provide the SLS algorithm with a

[17]For example on DH5000, we use $tf = 10$ as frequency threshold and get 743 unigram candidates by TF. Then we use the score assigned by CHI and IG at the 743th position to select unigram candidates. This usually results in more than 743 candidates being selected, e.g. 845 for CHI and 784 for IG.

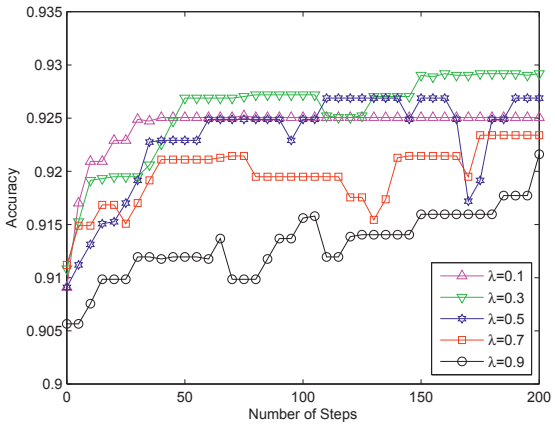


Figure 4: Impact Analysis on the Parameter λ

powerful mechanism to escape local (but non-global) optima. Therefore there is a fundamental trade-off between using low and high levels of noise in SLS.

In our experiment, the parameter κ is set to 100; the parameter R is set to 200; the parameter λ is varying from 0.1 to 0.9 with interval of 0.2 between steps. The experimental results are shown in Fig. 4. From Fig. 4, it is observed that the SLS algorithm reaches its best performance when the noise $\lambda = 0.3$. With a lower noise, e.g., $\lambda = 0.1$, the search process tends to be stuck in local optima and is not jumping out towards a better solution. With a higher noise, e.g., $\lambda = 0.9$, the search process takes too many downhill noise steps making it hard to reach an optimal solution.

4.5.2. Impact of Parameter κ

The parameter κ controls the number of neighbor states being tested in each greedy step. If κ is set too small, it increases the risk of not finding a better state in the step. If κ is set too large, it raises the probability of finding a better state in each greedy step although it increases computational time being consumed in search process. Hence, there is also a trade-off between testing a small number of neighbors and testing a large number of neighbors.

In this subsection, we present an experiment that studies the impact of the parameter κ on the Algorithm 2. In our experiment, the parameter λ is set to 0.1; the parameter R is set to 200. Algorithm 2 runs for 200 steps for each value of the parameter κ being set to 10, 20, 50, 100, 200. The experimental results are

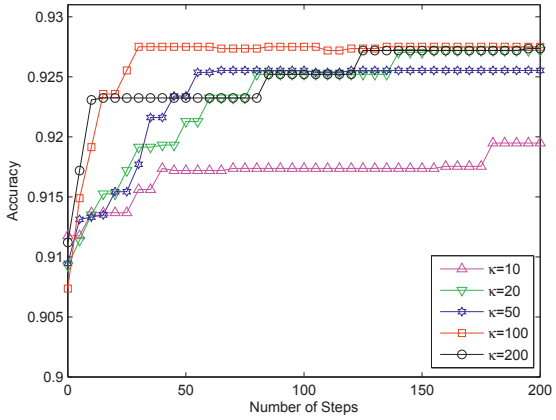


Figure 5: Impact Analysis on the Parameter κ

presented in Fig. 5. From Fig. 5, we see that a larger value of κ generally gives a better solution in the SLS process. It is also observed that the SLS process reaches comparable good results when $\kappa = 100$ and $\kappa = 200$. This indicate that when κ is "large enough", only increasing κ does not necessarily give much performance increase, while losing much on computational efficiency since more neighbor states need to be evaluated in each greedy step.

4.5.3. Impact of Parameter R

The parameter R indicates the number of iterative steps in a try of the SLS algorithm. It controls when the SLS algorithm should restart through re-initialization. If R is set too large, the SLS process lacks of more opportunity to search for other local optima from restart. If R is set too small, the SLS process will restart frequently so that local optima can hardly be reached in one try of the hill-climbing process.

In this subsection, we study the impact of varying the parameter R on the Algorithm 2. In our experiment, the parameter κ is set to 100; the parameter λ is set to 0.1. Algorithm 2 runs for 200 steps respectively for each value of the parameter R being set to 10, 20, 50, 100, 200. The experimental results are shown in Fig. 6. Fig. 6 illustrates that the SLS process reaches its best performance when $R = 50$. A larger value of R, e.g., $R = 200$, makes the SLS process trapped into a poor area of search space. A smaller value of R, e.g., $R = 10$, make search interrupted frequently by restart so that there is not enough time for the

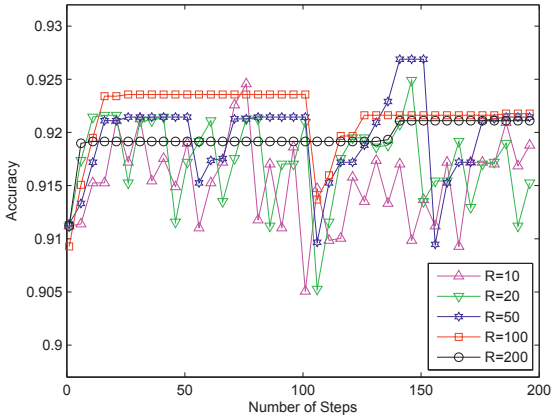


Figure 6: Impact Analysis on the Parameter R

SLS process to reach the local optima.

5. Conclusion

In this paper, we propose an Effective Feature Search (EFS) framework to enhance the performance of sentiment classifiers. The proposed EFS framework takes advantages of CoT patterns and search for effective features in an SLS process. The proposed EFS framework is a fully automatic approach with two steps. In the feature generation step, our approach employs an existing feature selection algorithm and relies on high frequency CoT patterns to generate unigram and complex (CoT) feature candidates. In the feature pruning step, we map the feature subset optimization problem to a Stochastic Local Search (SLS) model. Our hill-climbing SLS algorithm searches for an optimal feature subset using a wrapper approach.

The proposed EFS framework is empirically analyzed in systematic experiments. Performance comparison on two standard datasets shows that our proposed EFS framework is comparatively superior to existing state-of-the-art approaches in that our EFS framework is highly accurate, robust, and needs small human efforts. The proposed EFS framework is evaluated for different combinations of three unigram generation algorithms and four machine learning classifiers. The experimental results suggest that the EFS framework using the IG feature selection algorithm for unigram generation and employing the SVM classifier can achieve

the best performance for sentiment classification task. Further empirical analysis on the parameters λ, κ, and R of the Algorithm 2 studies the impact of these parameters to the SLS process and suggest empirical trade-offs for setting these parameters.

This work mainly focuses on elaborating a general overview of the proposed framework that makes a novel connection between feature candidate generation and an SLS process. A hill-climbing SLS algorithm is developed in this paper to solve the feature subset selection problem in the SLS model. There is a need to better appreciate how the SLS algorithm should be designed with intelligent parameter techniques such as tabu list update strategy, guided noise search, and clever initialization so that the SLS algorithm can search for optimized solutions in a fast process. In the Algorithm 1, the parameter l controls the number of terms that are considered as CoT candidates. Due to limitation of time and computational devices, in all experiments reported the parameter l is set to 2. Further experiments are demanded to test whether performance of our proposed EFS framework grows as the parameter l increases. We plan to investigate on these issues in our future work.

References

[1] V. Hatzivassiloglou, K. R. McKeown, Predicting the semantic orientation of adjectives, in: Proceedings of 35th Annual Meeting of the Association for Computational Linguistics, 1997.

[2] V. Hatzivassiloglou, J. M. Wiebe, Effects of adjective orientation and gradability on sentence subjectivity, in: Proceedings of 18th International Conference on Computational Linguistics, 2000.

[3] B. Pang, L. Lee, S. Vaithyanathan, Thumbs up? sentiment classification using machine learning techniques, in: Proceedings of the 7th Conference on Empirical Methods in Natural Language Processing (EMNLP'02).

[4] R. Kohavi, G. John, Wrappers for feature subset selection, Artificial Intelligence 97 (1997) 273–324.

[5] C. Whitelaw, N. Garg, S. Argamon, Using appraisal groups for sentiment analysis, in: Proceedings of the 14th ACM international Conference on Information and Knowledge Management, ACM Press, 2005, pp. 625–631.

[6] E. Riloff, S. Patwardhan, J. Wiebe, Feature subsumption for opinion analysis, in: Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP'06), 2006, pp. 440–448.

[7] A. Abbasi, S. L. France, Z. Zhang, H. Chen, Selecting attributes for sentiment classification using feature relation networks, IEEE Trans. Knowl. Data Eng 23 (3) (2011) 447–462.

[8] A. Abbasi, H. Chen, A. Salem, Sentiment analysis in multiple languages: Feature selection for opinion classification in web forums, ACM Trans. Inf. Syst 26 (3).

[9] Wikipedia, N-gram definition, http://en.wikipedia.org/wiki/N-gram.

[10] W. Wei, J. A. Gulla, Sentiment learning on product reviews via sentiment ontology tree, in: Proceedings of 48th Annual Meeting of the Association for Computational Linguistics, 2010.

[11] A. Abbasi, S. L. France, Z. Zhang, H. Chen, Selecting attributes for sentiment classification using feature relation networks, IEEE Trans. Knowl. Data Eng 23 (3).

[12] V. Ng, S. Dasgupta, S. M. N. Arifin, Examining the role of linguistic knowledge sources in the automatic identification and classification of reviews, in: Proceedings of 44th Annual Meeting of the Association for Computational Linguistics (ACL'06), 2006.

[13] A. C. König, E. Brill, Reducing the human overhead in text categorization, in: Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia (KDD'06), 2006.

[14] J. M. Wiebe, Learning subjective adjectives from corpora, in: Proceedings of the 2000 National Conference on Artificial Intelligence, AAAI, 2000.

[15] J. Wiebe, T. Wilson, R. Bruce, M. Bell, M. Martin, Learning subjective language, Comput. Linguist. 30 (3) (2004) 277–308.

[16] K. Dave, S. Lawrence, D. M. Pennock, Mining the peanut gallery: opinion extraction and semantic classification of product reviews, in: Proceedings of 12nd International World Wide Web Conference (WWW'03), 2003.

[17] E. Riloff, J. Wiebe, Learning extraction patterns for subjective expressions, in: Proceedings of EMNLP-03, 8th Conference on Empirical Methods in Natural Language Processing, 2003.

[18] J. Wiebe, E. Riloff, Creating subjective and objective sentence classifiers from unannotated texts, in: Proceeding the International Conference on Intelligent Text Processing and Computational Linguistics., 2005.

[19] T. Wilson, P. Hoffmann, S. Somasundaran, J. Kessler, J. Wiebe, Y. Choi, C. Cardie, E. Riloff, S. Patwardhan, Opinionfinder: A system for subjectivity analysis, in: Proceedings of Human Language Technology Conference and Empirical Methods in Natural Language Processing Conference, 2005.

[20] S. Bethard, H. Yu, A. Thornton, V. Hatzivassiloglou, D. Jurafsky, Automatic extraction of opinion propositions and their holders, in: Proceedings of the AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications, 2004.

[21] J. M. Wiebe, R. F. Bruce, T. P. O'Hara, Development and use of a gold-standard data set for subjectivity classifications, in: Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics, 1999, pp. 246–253.

[22] T. Wilson, J. Wiebe, R. Hwa, Just how mad are you? finding strong and weak opinion clauses, in: Proceedings of the Nineteenth National Conference on Artificial Intelligence, 2004, pp. 761–769.

[23] E. M. Riloff, J. Wiebe, T. Wilson, Learning subjective nouns using extraction pattern bootstrapping, in: Proceedings of the Seventh Conference on Natural Language Learning (CoNLL-2003), 2003.

[24] J. Wiebe, T. Wilson, Learning to disambiguate potentially subjective expressions, in: Proceedings of the Sixth Conference on Natural Language Learning (CoNLL-2002), 2002, pp. 112–118.

[25] H. Yu, V. Hatzivassiloglou, Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences, in: Proceedings of 8th Conference on Empirical Methods in Natural Language Processing, 2003.

[26] R. Mihalcea, C. Banea, J. Wiebe, Learning multilingual subjective language via cross-lingual projections, in: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL'07), 2007.

[27] F. Su, K. Markert, From words to senses: A case study of subjectivity recognition, in: Proceedings of the 22nd International Conference on Computational Linguistics (COLING'08), 2008, pp. 825–832.

[28] B. Pang, L. Lee, A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts, in: ACL, 2004, pp. 271–278.

[29] M. Hu, B. Liu, Mining and summarizing customer reviews, in: Proceedings of 10th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2004.

[30] B. Liu, M. Hu, J. Cheng, Opinion observer: analyzing and comparing opinions on the web, in: Proceedings of 14th International World Wide Web Conference, 2005.

[31] T. T. Thet, J.-C. Na, C. S. G. Khoo, Aspect-based sentiment analysis of movie reviews on discussion boards, J. Information Science 36 (6) (2010) 823–848.

[32] J. Zhu, H. Wang, M. Zhu, B. K. Tsou, M. Y. Ma, Aspect-based opinion polling from customer reviews, IEEE Transactions on Affective Computing 2 (1) (2011) 37–49.

[33] S. Brody, N. Elhadad, An unsupervised aspect-sentiment model for online reviews, in: HLT-NAACL, The Association for Computational Linguistics, 2010, pp. 804–812.

[34] W. Wei, J. A. Gulla, Enhancing the hl-sot approach to sentiment analysis via a localized feature selection framework, in: Proceedings of 5th International Joint Conference on Natural Language Processing, 2011, pp. 327–335.

[35] Q. Mei, X. Ling, M. Wondra, H. Su, C. Zhai, Topic sentiment mixture: modeling facets and opinions in weblogs, in: Proceedings of the 16th International Conference on World Wide Web (WWW'07), 2007, pp. 171–180.

[36] Y. Lu, C. Zhai, Opinion integration through semi-supervised topic modeling, in: Proceedings of 17th International World Wide Web Conference, 2008.

[37] Y. Lu, C. Zhai, N. Sundaresan, Rated aspect summarization of short comments, in: Proceedings of 18th International World Wide Web Conference, 2009.

[38] I. Titov, R. T. McDonald, Modeling online reviews with multi-grain topic models, in: Proceedings of 17th International World Wide Web Conference, 2008.

[39] M. Taboada, J. Brooke, M. Tofiloski, K. D. Voll, M. Stede, Lexicon-based methods for sentiment analysis, Computational Linguistics 37 (2) (2011) 267–307.

[40] A. Andreevskaia, S. Bergler, Mining wordnet for a fuzzy sentiment: Sentiment tag extraction from wordnet glosses, in: Proceedings of 11th Conference of the European Chapter of the Association for Computational Linguistics, 2006.

[41] A. Esuli, F. Sebastiani, Determining the semantic orientation of terms through gloss classification, in: Proceedings of 14th ACM Conference on Information and Knowledge Management, 2005.

[42] A. Esuli, F. Sebastiani, Sentiwordnet: A publicly available lexical resource for opinion mining, in: Proceedings of 5th International Conference on Language Resources and Evaluation, 2006.

[43] J. Kamps, M. Marx, R. ort. Mokken, M. de Rijke, Using WordNet to measure semantic orientation of adjectives, in: Proceedings of 4th International Conference on Language Resources and Evaluation, 2004.

[44] A. Devitt, K. Ahmad, Sentiment polarity identification in financial news: A cohesion-based approach, in: Proceedings of 45th Annual Meeting of the Association for Computational Linguistics, 2007.

[45] C. Whitelaw, N. Garg, S. Argamon, Using appraisal taxonomies for sentiment analysis, in: Proceedings of 14th ACM Conference on Information and Knowledge Management, 2005.

[46] T. Wilson, J. Wiebe, P. Hoffmann, Recognizing contextual polarity in phrase-level sentiment analysis, in: Proceedings of Human Language Technology Conference and Empirical Methods in Natural Language Processing Conference, 2005.

[47] P. D. Turney, Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews, in: Proceedings of 40th Annual Meeting of the Association for Computational Linguistics, 2002.

[48] C. Lin, Y. He, Joint sentiment/topic model for sentiment analysis, in: Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM'11), 2009.

[49] B. Heerschop, F. Goossen, A. Hogenboom, F. Frasincar, U. Kaymak, F. de Jong, Polarity analysis of texts using discourse structure, in: Proceedings of the 20th ACM Conference on Information and Knowledge Management (CIKM'11), 2011.

[50] R. Mukras, N. Wiratunga, R. Lothian, Selecting bi-tags for sentiment analysis of text, in: Proceedings of AI-2007 Twenty-seventh SGAI International Conference on Artificial Intelligence, Springer, 2007, pp. 181–194.

[51] A. Duric, F. Song, Feature selection for sentiment analysis based on content and syntax models (2011).

[52] O. F. Zaidan, J. Eisner, Using "annotator rationales" to improve machine learning for text categorization, in: Proceedings of NAACL HLT 2007, 2007.

[53] J. Martineau, T. Finin, Delta TFIDF: An improved feature space for sentiment analysis, in: Proceedings of the Third International Conference on Weblogs and Social Media (ICWSM'09), 2009.

[54] M. Rushdi-Saleh, M. T. Martín-Valdivia, A. M. Ráez, L. A. U. López, Experiments with SVM to classify opinions in different domains, Expert Syst. Appl 38 (12) (2011) 14799–14804.

[55] G. Paltoglou, M. Thelwall, A study of information retrieval weighting schemes for sentiment analysis, in: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL'10), 2010, pp. 1386–1395.

[56] T. Mullen, N. Collier, Sentiment analysis using support vector machines with diverse information sources, in: Proceedings of EMNLP'04, pp. 412–418.

[57] I. K. Tim O Keefe, Feature selection and weighting methods in sentiment analysis, in: Proceedings of the 14th Australian Document Computing Symposium, 2009.

[58] Y. Mejova, P. Srinivasan, Exploring feature definition and selection for sentiment classifiers, in: Proceedings of the Fifth International Conference on Weblogs and Social Media, 2011.

[59] X. Bai, Predicting consumer sentiments from online text, Decision Support Systems 50 (4) (2011) 732–742.

[60] C. D. Manning, P. Raghavan, H. Schutze, Introduction to Information Retrieval, Cambridge University Press, 2008, Ch. 13, pp. 271–278.

[61] T. Mitchell, Machine Learning, McGraw-Hill, 1997.

[62] B. Selman, H. Levesque, D. Mitchell, A new method for solving hard satisfiability problems, in: Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92), San Jose, CA, 1992, pp. 440–446.

[63] B. Selman, H. A. Kautz, B. Cohen, Noise strategies for improving local search, in: Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94), 1994, pp. 337–343.

[64] P. W. Gu, J. Purdom, J. Franco, B. W. Wah, Satisfiability Problem: Theory and Applications, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, 1997, Ch. Algorithms for the Satisfiability SAT Problem: A Survey, pp. 19–152.

[65] H. H. Hoos, T. Stützle, Stochastic Local Search: Foundations and Applications, Morgan Kaufmann, San Francisco, 2005.

[66] K. Kask, R. Dechter, Stochastic local search for Bayesian networks, in: Proceedings Seventh International Workshop on Artificial Intelligence and Statistics, Morgan Kaufmann, Fort Lauderdale, FL, 1999.

[67] O. J. Mengshoel, Efficient Bayesian network inference: Genetic algorithms, stochastic local search, and abstraction, Ph.D. thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL (Apr. 1999).

[68] O. J. Mengshoel, D. Roth, D. C. Wilkins, Stochastic greedy search: Computing the most probable explanation in Bayesian networks, Tech. Rep. UIUCDCS-R-2000-2150, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL (Feb. 2000).

[69] J. D. Park, A. Darwiche, Approximating MAP using local search, in: Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI-01), Seattle, WA, 2001, pp. 403–410.

[70] J. D. Park, A. Darwiche, Complexity results and approximation strategies for MAP explanations, Journal of Artificial Intelligence Research (JAIR) 21 (2004) 101–133.

[71] M. F. Porter, An algorithm for suffix stripping, Program 14 (3) (1980) 130–137.

[72] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, The weka data mining software: an update, SIGKDD Explor. Newsl. 11 (1) (2009) 10–18.

[73] Z. Zhai, H. Xu, J. Li, P. Jia, Feature subsumption for sentiment classification in multiple languages, in: Proceedings of 14th Pacific-Asia Conference (PAKDD'10), Vol. 6119 of Lecture Notes in Computer Science, 2010, pp. 261–271.

[74] A. Kennedy, D. Inkpen, Sentiment classification of movie reviews using contextual valence shifters, Computational Intelligence 22 (2) (2006) 110–125.

[75] A. Pak, P. Paroubek, Text representation using dependency tree subgraphs for sentiment analysis, in: Proceedings of DASFAA 2011 Workshops, Vol. 6637 of Lecture Notes in Computer Science, Springer, 2011, pp. 323–332.

[76] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, C. Potts, Learning word vectors for sentiment analysis, in: Proceedings of The 49th Annual Meeting of the Association for Computational Linguistics (ACL'11), 2011, pp. 142–150.

Chapter 9

Paper Four

Wei Wei and Jon Atle Gulla, “Sentiment Learning on Product Reviews via Sentiment Ontology Tree", Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 404-413, 2010, Association for Computational Linguistics Stroudsburg, PA, USA, ISBN 978-1-932432-66-4,978-1-932432-67-1.

Sentiment Learning on Product Reviews via Sentiment Ontology Tree

Wei Wei
Department of Computer and Information Science
Norwegian University of Science and Technology
wwei@idi.ntnu.no

Jon Atle Gulla
Department of Computer and Information Science
Norwegian University of Science and Technology
jag@idi.ntnu.no

Abstract

Existing works on sentiment analysis on product reviews suffer from the following limitations: (1) The knowledge of hierarchical relationships of products attributes is not fully utilized. (2) Reviews or sentences mentioning several attributes associated with complicated sentiments are not dealt with very well. In this paper, we propose a novel HL-SOT approach to labeling a product's attributes and their associated sentiments in product reviews by a Hierarchical Learning (HL) process with a defined Sentiment Ontology Tree (SOT). The empirical analysis against a human-labeled data set demonstrates promising and reasonable performance of the proposed HL-SOT approach. While this paper is mainly on sentiment analysis on reviews of one product, our proposed HL-SOT approach is easily generalized to labeling a mix of reviews of more than one products.

1 Introduction

As the internet reaches almost every corner of this world, more and more people write reviews and share opinions on the World Wide Web. The user-generated opinion-rich reviews will not only help other users make better judgements but they are also useful resources for manufacturers of products to keep track and manage customer opinions. However, as the number of product reviews grows, it becomes difficult for a user to manually learn the panorama of an interesting topic from existing online information. Faced with this problem, research works, e.g., (Hu and Liu, 2004; Liu et al., 2005; Lu et al., 2009), of sentiment analysis on product reviews were proposed and have become a popular research topic at the crossroads of information retrieval and computational linguistics.

Carrying out sentiment analysis on product reviews is not a trivial task. Although there have already been a lot of publications investigating on similar issues, among which the representatives are (Turney, 2002; Dave et al., 2003; Hu and Liu, 2004; Liu et al., 2005; Popescu and Etzioni, 2005; Zhuang et al., 2006; Lu and Zhai, 2008; Titov and McDonald, 2008; Zhou and Chaovalit, 2008; Lu et al., 2009), there is still room for improvement on tackling this problem. When we look into the details of each example of product reviews, we find that there are some intrinsic properties that existing previous works have not addressed in much detail.

First of all, product reviews constitute domain-specific knowledge. The product's attributes mentioned in reviews might have some relationships between each other. For example, for a digital camera, comments on image quality are usually mentioned. However, a sentence like "40D handles noise very well up to ISO 800", also refers to image quality of the camera 40D. Here we say "noise" is a sub-attribute factor of "image quality". We argue that the hierarchical relationship between a product's attributes can be useful knowledge if it can be formulated and utilized in product reviews analysis. Secondly, Vocabularies used in product reviews tend to be highly overlapping. Especially, for same attribute, usually same words or synonyms are involved to refer to them and to describe sentiment on them. We believe that labeling existing product reviews with attributes and corresponding sentiment forms an effective training resource to perform sentiment analysis. Thirdly, sentiments expressed in a review or even in a sentence might be opposite on different attributes and not every attributes mentioned are with sentiments. For example, it is common to find a fragment of a review as follows:
Example 1: *"...I am very impressed with this camera except for its a bit heavy weight especially with*

Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 404–413,
Uppsala, Sweden, 11-16 July 2010.

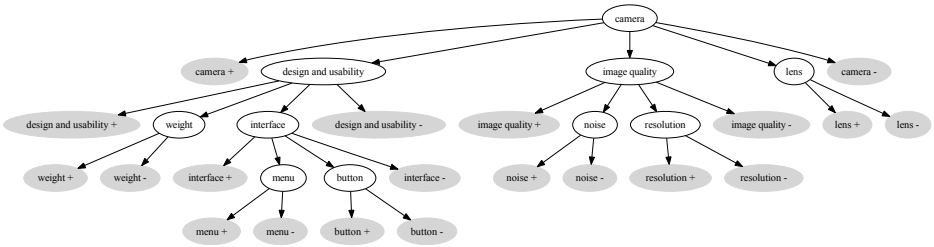


Figure 1: an example of part of a SOT for digital camera

extra lenses attached. It has many buttons and two main dials. The first dial is thumb dial, located near shutter button. The second one is the big round dial located at the back of the camera..."

In this example, the first sentence gives positive comment on the camera as well as a complaint on its heavy weight. Even if the words "lenses" appears in the review, it is not fair to say the customer expresses any sentiment on lens. The second sentence and the rest introduce the camera's buttons and dials. It's also not feasible to try to get any sentiment from these contents. We argue that when performing sentiment analysis on reviews, such as in the Example 1, more attention is needed to distinguish between attributes that are mentioned with and without sentiment.

In this paper, we study the problem of sentiment analysis on product reviews through a novel method, called the HL-SOT approach, namely Hierarchical Learning (HL) with Sentiment Ontology Tree (SOT). By sentiment analysis on product reviews we aim to fulfill two tasks, i.e., labeling a target text[1] with: 1) the product's attributes (attributes identification task), and 2) their corresponding sentiments mentioned therein (sentiment annotation task). The result of this kind of labeling process is quite useful because it makes it possible for a user to search reviews on particular attributes of a product. For example, when considering to buy a digital camera, a prospective user who cares more about image quality probably wants to find comments on the camera's image quality in other users' reviews. SOT is a tree-like ontology structure that formulates the relationships between a product's attributes. For example, Fig. 1 is a SOT for a digital camera[2]. The root node of the SOT is a camera itself. Each of the non-leaf nodes (white nodes) of the SOT represents an attribute of a camera[3]. All leaf nodes (gray nodes) of the SOT represent sentiment (positive/negative) nodes respectively associated with their parent nodes. A formal definition on SOT is presented in Section 3.1. With the proposed concept of SOT, we manage to formulate the two tasks of the sentiment analysis to be a hierarchical classification problem. We further propose a specific hierarchical learning algorithm, called HL-SOT algorithm, which is developed based on generalizing an online-learning algorithm H-RLS (Cesa-Bianchi et al., 2006). The HL-SOT algorithm has the same property as the H-RLS algorithm that allows multiple-path labeling (input target text can be labeled with nodes belonging to more than one path in the SOT) and partial-path labeling (the input target text can be labeled with nodes belonging to a path that does not end on a leaf). This property makes the approach well suited for the situation where complicated sentiments on different attributes are expressed in one target text. Unlike the H-RLS algorithm , the HL-SOT algorithm enables each classifier to separately learn its own specific threshold. The proposed HL-SOT approach is empirically analyzed against a human-labeled data set. The experimental results demonstrate promising and reasonable performance of our approach.

This paper makes the following contributions:

- To the best of our knowledge, with the proposed concept of SOT, the proposed HL-SOT approach is the first work to formulate the tasks of sentiment analysis to be a hierarchical classification problem.
- A specific hierarchical learning algorithm is

[1]Each product review to be analyzed is called target text in the following of this paper.

[2]Due to the space limitation, not all attributes of a digital camera are enumerated in this SOT; m+/m- means positive/negative sentiment associated with an attribute m.

[3]A product itself can be treated as an overall attribute of the product.

further proposed to achieve tasks of sentiment analysis in one hierarchical classification process.

- The proposed HL-SOT approach can be generalized to make it possible to perform sentiment analysis on target texts that are a mix of reviews of different products, whereas existing works mainly focus on analyzing reviews of only one type of product.

The remainder of the paper is organized as follows. In Section 2, we provide an overview of related work on sentiment analysis. Section 3 presents our work on sentiment analysis with HL-SOT approach. The empirical analysis and the results are presented in Section 4, followed by the conclusions, discussions, and future work in Section 5.

2 Related Work

The task of sentiment analysis on product reviews was originally performed to extract overall sentiment from the target texts. However, in (Turney, 2002), as the difficulty shown in the experiments, the whole sentiment of a document is not necessarily the sum of its parts. Then there came up with research works shifting focus from overall document sentiment to sentiment analysis based on product attributes (Hu and Liu, 2004; Popescu and Etzioni, 2005; Ding and Liu, 2007; Liu et al., 2005).

Document overall sentiment analysis is to summarize the overall sentiment in the document. Research works related to document overall sentiment analysis mainly rely on two finer levels sentiment annotation: *word-level sentiment annotation* and *phrase-level sentiment annotation*. The *word-level sentiment annotation* is to utilize the polarity annotation of words in each sentence and summarize the overall sentiment of each sentiment-bearing word to infer the overall sentiment within the text (Hatzivassiloglou and Wiebe, 2000; Andreevskaia and Bergler, 2006; Esuli and Sebastiani, 2005; Esuli and Sebastiani, 2006; Hatzivassiloglou and McKeown, 1997; Kamps et al., 2004; Devitt and Ahmad, 2007; Yu and Hatzivassiloglou, 2003). The *phrase-level sentiment annotation* focuses sentiment annotation on phrases not words with concerning that atomic units of expression is not individual words but rather appraisal groups (Whitelaw et al., 2005). In (Wilson et al., 2005), the concepts of *prior polarity* and *contextual polarity* were proposed. This paper presented a system that is able to automatically identify the *contextual polarity* for a large subset of sentiment expressions. In (Turney, 2002), an unsupervised learning algorithm was proposed to classify reviews as recommended or not recommended by averaging sentiment annotation of phrases in reviews that contain adjectives or adverbs. However, the performances of these works are not good enough for sentiment analysis on product reviews, where sentiment on each attribute of a product could be so complicated that it is unable to be expressed by overall document sentiment.

Attributes-based sentiment analysis is to analyze sentiment based on each attribute of a product. In (Hu and Liu, 2004), mining product features was proposed together with sentiment polarity annotation for each opinion sentence. In that work, sentiment analysis was performed on product attributes level. In (Liu et al., 2005), a system with framework for analyzing and comparing consumer opinions of competing products was proposed. The system made users be able to clearly see the strengths and weaknesses of each product in the minds of consumers in terms of various product features. In (Popescu and Etzioni, 2005), Popescu and Etzioni not only analyzed polarity of opinions regarding product features but also ranked opinions based on their strength. In (Liu et al., 2007), Liu et al. proposed Sentiment-PLSA that analyzed blog entries and viewed them as a document generated by a number of hidden sentiment factors. These sentiment factors may also be factors based on product attributes. In (Lu and Zhai, 2008), Lu et al. proposed a semi-supervised topic models to solve the problem of opinion integration based on the topic of a product's attributes. The work in (Titov and McDonald, 2008) presented a multi-grain topic model for extracting the ratable attributes from product reviews. In (Lu et al., 2009), the problem of rated attributes summary was studied with a goal of generating ratings for major aspects so that a user could gain different perspectives towards a target entity. All these research works concentrated on attribute-based sentiment analysis. However, the main difference with our work is that they did not sufficiently utilize the hierarchical relationships among a product attributes. Although a method of ontology-supported polarity mining, which also involved

ontology to tackle the sentiment analysis problem, was proposed in (Zhou and Chaovalit, 2008), that work studied polarity mining by machine learning techniques that still suffered from a problem of ignoring dependencies among attributes within an ontology's hierarchy. In the contrast, our work solves the sentiment analysis problem as a hierarchical classification problem that fully utilizes the hierarchy of the SOT during training and classification process.

3 The HL-SOT Approach

In this section, we first propose a formal definition on SOT. Then we formulate the HL-SOT approach. In this novel approach, tasks of sentiment analysis are to be achieved in a hierarchical classification process.

3.1 Sentiment Ontology Tree

As we discussed in Section 1, the hierarchial relationships among a product's attributes might help improve the performance of attribute-based sentiment analysis. We propose to use a tree-like ontology structure SOT, i.e., Sentiment Ontology Tree, to formulate relationships among a product's attributes. Here,we give a formal definition on what a SOT is.

Definition 1 *[SOT] SOT is an abbreviation for Sentiment Ontology Tree that is a tree-like ontology structure* $T(v, v^+, v^-, \mathbb{T})$. v *is the root node of* T *which represents an attribute of a given product.* v^+ *is a positive sentiment leaf node associated with the attribute* v. v^- *is a negative sentiment leaf node associated with the attribute* v. $\mathbb{T}$ *is a set of subtrees. Each element of* $\mathbb{T}$ *is also a SOT* $T'(v', v'^+, v'^-, \mathbb{T}')$ *which represents a sub-attribute of its parent attribute node.*

By the Definition 1, we define a root of a SOT to represent an attribute of a product. The SOT's two leaf child nodes are sentiment (positive/negative) nodes associated with the root attribute. The SOT recursively contains a set of sub-SOTs where each root of a sub-SOT is a non-leaf child node of the root of the SOT and represent a sub-attribute belonging to its parent attribute. This definition successfully describes the hierarchical relationships among all the attributes of a product. For example, in Fig. 1 the root node of the SOT for a digital camera is its general overview attribute. Comments on a digital camera's general overview attribute appearing in a review might be like "this camera is great". The "camera" SOT has two sentiment leaf child nodes as well as three non-leaf child nodes which are respectively root nodes of sub-SOTs for sub-attributes "design and usability", "image quality", and "lens". These sub-attributes SOTs recursively repeat until each node in the SOT does not have any more non-leaf child node, which means the corresponding attributes do not have any sub-attributes, e.g., the attribute node "button" in Fig. 1.

3.2 Sentiment Analysis with SOT

In this subsection, we present the HL-SOT approach. With the defined SOT, the problem of sentiment analysis is able to be formulated to be a hierarchial classification problem. Then a specific hierarchical learning algorithm is further proposed to solve the formulated problem.

3.2.1 Problem Formulation

In the proposed HL-SOT approach, each target text is to be indexed by a unit-norm vector $x \in \mathcal{X}, \mathcal{X} = \mathbb{R}^d$. Let $\mathscr{Y} = \{1, ..., N\}$ denote the finite set of nodes in SOT. Let $y = \{y_1, ..., y_N\} \in \{0, 1\}^N$ be a label vector to a target text x, where $\forall i \in \mathscr{Y}$:

$$y_i = \begin{cases} 1, & \text{if } x \text{ is labeled by the classifier of node } i, \\ 0, & \text{if } x \text{ is not labeled by the classifier of node } i. \end{cases}$$

A label vector $y \in \{0, 1\}^N$ is said to respect SOT if and only if y satisfies $\forall i \in \mathscr{Y}, \forall j \in \mathcal{A}(i)$: if $y_i = 1$ then $y_j = 1$, where $\mathcal{A}(i)$ represents a set ancestor nodes of i, i.e.,$\mathcal{A}(i) = \{x | ancestor(i, x)\}$. Let $\mathcal{Y}$ denote a set of label vectors that respect SOT. Then the tasks of sentiment analysis can be formulated to be the goal of a hierarchical classification that is to learn a function $f : \mathcal{X} \rightarrow \mathcal{Y}$, that is able to label each target text $x \in \mathcal{X}$ with classifier of each node and generating with x a label vector $y \in \mathcal{Y}$ that respects SOT. The requirement of a generated label vector $y \in \mathcal{Y}$ ensures that a target text is to be labeled with a node only if its parent attribute node is labeled with the target text. For example, in Fig. 1 a review is to be labeled with "image quality +" requires that the review should be successively labeled as related to "camera" and "image quality". This is reasonable and consistent with intuition, because if a review cannot be identified to be related to a camera, it is not safe to infer that the review is commenting a camera's image quality with positive sentiment.

3.2.2 HL-SOT Algorithm

The algorithm H-RLS studied in (Cesa-Bianchi et al., 2006) solved a similar hierarchical classification problem as we formulated above. However, the H-RLS algorithm was designed as an online-learning algorithm which is not suitable to be applied directly in our problem setting. Moreover, the algorithm H-RLS defined the same value as the threshold of each node classifier. We argue that if the threshold values could be learned separately for each classifiers, the performance of classification process would be improved. Therefore we propose a specific hierarchical learning algorithm, named HL-SOT algorithm, that is able to train each node classifier in a batch-learning setting and allows separately learning for the threshold of each node classifier.

Defining the f function Let $w_1, ..., w_N$ be weight vectors that define linear-threshold classifiers of each node in SOT. Let $W = (w_1, ..., w_N)^\top$ be an $N \times d$ matrix called weight matrix. Here we generalize the work in (Cesa-Bianchi et al., 2006) and define the hierarchical classification function f as:

$$\hat{y} = f(x) = g(W \cdot x),$$

where $x \in \mathcal{X}, \hat{y} \in \mathcal{Y}$. Let $z = W \cdot x$. Then the function $\hat{y} = g(z)$ on an N-dimensional vector z defines:
$\forall i = 1, ..., N$:

$$\hat{y}_i = \begin{cases} \mathfrak{B}(z_i \geq \theta_i), & \text{if } i \text{ is a root node in SOT} \\ & \text{or } y_j = 1 \text{ for } j = \mathcal{P}(i), \\ 0, & \text{else} \end{cases}$$

where $\mathcal{P}(i)$ is the parent node of i in SOT and $\mathfrak{B}(S)$ is a boolean function which is 1 if and only if the statement S is true. Then the hierarchical classification function f is parameterized by the weight matrix $W = (w_1, ..., w_N)^\top$ and threshold vector $\theta = (\theta_1, ..., \theta_N)^\top$. The hierarchical learning algorithm HL-SOT is proposed for learning the parameters of W and θ.

Parameters Learning for f function Let D denote the training data set: $D = \{(r, l) | r \in \mathcal{X}, l \in \mathcal{Y}\}$. In the HL-SOT learning process, the weight matrix W is firstly initialized to be a 0 matrix, where each row vector w_i is a 0 vector. The threshold vector is initialized to be a 0 vector. Each instance in the training set D goes into the training process. When a new instance r_t is observed, each row vector $w_{i,t}$ of the weight matrix W_t is updated by a regularized least squares estimator given by:

$$w_{i,t} = (I + S_{i,Q(i,t-1)} S_{i,Q(i,t-1)}^\top + r_t r_t^\top)^{-1} \times S_{i,Q(i,t-1)} (l_{i,i_1}, l_{i,i_2}, ..., l_{i,i_{Q(i,t-1)}})^\top \quad (1)$$

where I is a $d \times d$ identity matrix, $Q(i, t-1)$ denotes the number of times the parent of node i observes a positive label before observing the instance r_t, $S_{i,Q(i,t-1)} = [r_{i_1}, ..., r_{i_{Q(i,t-1)}}]$ is a $d \times Q(i, t-1)$ matrix whose columns are the instances $r_{i_1}, ..., r_{i_{Q(i,t-1)}}$, and $(l_{i,i_1}, l_{i,i_2}, ..., l_{i,i_{Q(i,t-1)}})^\top$ is a $Q(i, t-1)$-dimensional vector of the corresponding labels observed by node i. The Formula 1 restricts that the weight vector $w_{i,t}$ of the classifier i is only updated on the examples that are positive for its parent node. Then the label vector $\hat{y}_{r_t}$ is computed for the instance r_t, before the real label vector l_{r_t} is observed. Then the current threshold vector θ_t is updated by:

$$\theta_{t+1} = \theta_t + \epsilon(\hat{y}_{r_t} - l_{r_t}), \quad (2)$$

where ϵ is a small positive real number that denotes a corrective step for correcting the current threshold vector θ_t. To illustrate the idea behind the Formula 2, let $y'_t = \hat{y}_{r_t} - l_{r_t}$. Let $y'_{i,t}$ denote an element of the vector y'_t. The Formula 2 correct the current threshold $\theta_{i,t}$ for the classifier i in the following way:

- If $y'_{i,t} = 0$, it means the classifier i made a proper classification for the current instance r_t. Then the current threshold θ_i does not need to be adjusted.

- If $y'_{i,t} = 1$, it means the classifier i made an improper classification by mistakenly identifying the attribute i of the training instance r_t that should have not been identified. This indicates the value of θ_i is not big enough to serve as a threshold so that the attribute i in this case can be filtered out by the classifier i. Therefore, the current threshold θ_i will be adjusted to be larger by ϵ.

- If $y'_{i,t} = -1$, it means the classifier i made an improper classification by failing to identify the attribute i of the training instance r_t that should have been identified. This indicates the value of θ_i is not small enough to serve as a threshold so that the attribute i in this case

Algorithm 1 Hierarchical Learning Algorithm HL-SOT

INITIALIZATION:
Each vector $w_{i,1}, i = 1, ..., N$ of weight matrix W_1 is set to be 0 vector
Threshold vector θ_1 is set to be 0 vector
BEGIN
for $t = 1, ..., |D|$ **do**
  Observe instance $r_t \in \mathcal{X}$
  for $i = 1, ...N$ **do**
    Update each row $w_{i,t}$ of weight matrix W_t by Formula 1
  end for
  Compute $\hat{y}_{r_t} = f(r_t) = g(W_t \cdot r_t)$
  Observe label vector $l_{r_t} \in \mathcal{Y}$ of the instance r_t
  Update threshold vector θ_t by Formula 2
end for
END

can be recognized by the classifier i. Therefore, the current threshold θ_i will be adjusted to be smaller by ϵ.

The hierarchial learning algorithm HL-SOT is presented as in Algorithm 1. The HL-SOT algorithm enables each classifier to have its own specific threshold value and allows this threshold value can be separately learned and corrected through the training process. It is not only a batch-learning setting of the H-RLS algorithm but also a generalization to the latter. If we set the algorithm HL-SOT's parameter ϵ to be 0, the HL-SOT becomes the H-RLS algorithm in a batch-learning setting.

4 Empirical Analysis

In this section, we conduct systematic experiments to perform empirical analysis on our proposed HL-SOT approach against a human-labeled data set. In order to encode each text in the data set by a d-dimensional vector $x \in \mathbb{R}^d$, we first remove all the stop words and then select the top d frequency terms appearing in the data set to construct the index term space. Our experiments are intended to address the following questions:(1) whether utilizing the hierarchical relationships among labels help to improve the accuracy of the classification? (2) whether the introduction of separately learning threshold for each classifier help to improve the accuracy of the classification? (3) how does the corrective step ϵ impact the performance of the proposed approach?(4)how does the dimensionality d of index terms space impact the proposed approach's computing efficiency and accuracy?

4.1 Data Set Preparation

The data set contains 1446 snippets of customer reviews on digital cameras that are collected from a customer review website[4]. We manually construct a SOT for the product of digital cameras. The constructed SOT (e.g., Fig. 1) contains 105 nodes that include 35 non-leaf nodes representing attributes of the digital camera and 70 leaf nodes representing associated sentiments with attribute nodes. Then we label all the snippets with corresponding labels of nodes in the constructed SOT complying with the rule that a target text is to be labeled with a node only if its parent attribute node is labeled with the target text. We randomly divide the labeled data set into five folds so that each fold at least contains one example snippets labeled by each node in the SOT. For each experiment setting, we run 5 experiments to perform cross-fold evaluation by randomly picking three folds as the training set and the other two folds as the testing set. All the testing results are averages over 5 running of experiments.

4.2 Evaluation Metrics

Since the proposed HL-SOT approach is a hierarchical classification process, we use three classic loss functions for measuring classification performance. They are the One-error Loss (O-Loss) function, the Symmetric Loss (S-Loss) function, and the Hierarchical Loss (H-Loss) function:

- One-error loss (O-Loss) function is defined as:
$$L_O(\hat{y}, l) = \mathfrak{B}(\exists i : \hat{y}_i \neq l_i),$$
where $\hat{y}$ is the prediction label vector and l is the true label vector; $\mathfrak{B}$ is the boolean function as defined in Section 3.2.2.

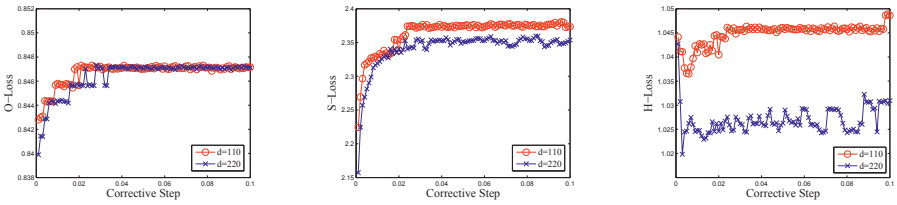
- Symmetric loss (S-Loss) function is defined as:
$$L_S(\hat{y}, l) = \sum_{i=1}^{N} \mathfrak{B}(\hat{y}_i \neq l_i),$$

- Hierarchical loss (H-Loss) function is defined as:
$$L_H(\hat{y}, l) = \sum_{i=1}^{N} \mathfrak{B}(\hat{y}_i \neq l_i \wedge \forall j \in \mathcal{A}(i), \hat{y}_j = l_j),$$

[4] http://www.consumerreview.com/

Table 1: Performance Comparisons (A Smaller Loss Value Means a Better Performance)

Metrics	Dimensinality=110			Dimensinality=220		
	H-RLS	HL-flat	HL-SOT	H-RLS	HL-flat	HL-SOT
O-Loss	0.9812	0.8772	**0.8443**	0.9783	0.8591	**0.8428**
S-Loss	8.5516	2.8921	**2.3190**	7.8623	2.8449	**2.2812**
H-Loss	3.2479	1.1383	**1.0366**	3.1029	1.1298	**1.0247**



(a) O-Loss (b) S-Loss (c) H-Loss

Figure 2: Impact of Corrective Step ϵ

where $\mathcal{A}$ denotes a set of nodes that are ancestors of node i in SOT.

Unlike the O-Loss function and the S-Loss function, the H-Loss function captures the intuition that loss should only be charged on a node whenever a classification mistake is made on a node of SOT but no more should be charged for any additional mistake occurring in the subtree of that node. It measures the discrepancy between the prediction labels and the true labels with consideration on the SOT structure defined over the labels. In our experiments, the recorded loss function values for each experiment running are computed by averaging the loss function values of each testing snippets in the testing set.

4.3 Performance Comparison

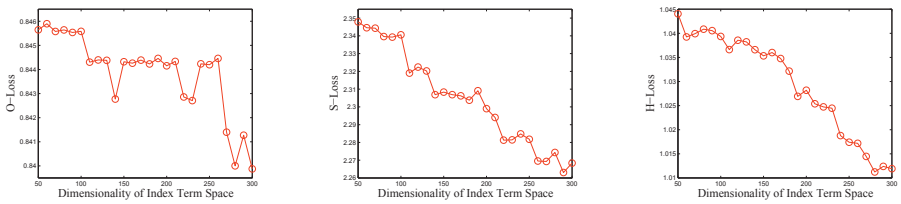
In order to answer the questions (1), (2) in the beginning of this section, we compare our HL-SOT approach with the following two baseline approaches:

- HL-flat: The HL-flat approach involves an algorithm that is a "flat" version of HL-SOT algorithm by ignoring the hierarchical relationships among labels when each classifier is trained. In the training process of HL-flat, the algorithm reflexes the restriction in the HL-SOT algorithm that requires the weight vector $w_{i,t}$ of the classifier i is only updated on the examples that are positive for its parent node.
- H-RLS: The H-RLS approach is implemented by applying the H-RLS algorithm studied in (Cesa-Bianchi et al., 2006). Unlike our proposed HL-SOT algorithm that enables the threshold values to be learned separately for each classifiers in the training process, the H-RLS algorithm only uses an identical threshold values for each classifiers in the classification process.

Experiments are conducted on the performance comparison between the proposed HL-SOT approach with HL-flat approach and the H-RLS approach. The dimensionality d of the index term space is set to be 110 and 220. The corrective step ϵ is set to be 0.005. The experimental results are summarized in Table 1. From Table 1, we can observe that the HL-SOT approach generally beats the H-RLS approach and HL-flat approach on O-Loss, S-Loss, and H-Loss respectively. The H-RLS performs worse than the HL-flat and the HL-SOT, which indicates that the introduction of separately learning threshold for each classifier did improve the accuracy of the classification. The HL-SOT approach performs better than the HL-flat, which demonstrates the effectiveness of utilizing the hierarchical relationships among labels.

4.4 Impact of Corrective Step ϵ

The parameter ϵ in the proposed HL-SOT approach controls the corrective step of the classifiers' thresholds when any mistake is observed in the training process. If the corrective step ϵ is set too large, it might cause the algorithm to be too



(a) O-Loss (b) S-Loss (c) H-Loss

Figure 3: Impact of Dimensionality d of Index Term Space ($\epsilon = 0.005$)

sensitive to each observed mistake. On the contrary, if the corrective step is set too small, it might cause the algorithm not sensitive enough to the observed mistakes. Hence, the corrective step ϵ is a factor that might impact the performance of the proposed approach. Fig. 2 demonstrates the impact of ϵ on O-Loss, S-Loss, and H-Loss. The dimensionality of index term space d is set to be 110 and 220. The value of ϵ is set to vary from 0.001 to 0.1 with each step of 0.001. Fig. 2 shows that the parameter ϵ impacts the classification performance significantly. As the value of ϵ increase, the O-Loss, S-Loss, and H-Loss generally increase (performance decrease). In Fig. 2c it is obviously detected that the H-Loss decreases a little (performance increase) at first before it increases (performance decrease) with further increase of the value of ϵ. This indicates that a finer-grained value of ϵ will not necessarily result in a better performance on the H-loss. However, a fine-grained corrective step generally makes a better performance than a coarse-grained corrective step.

4.5 Impact of Dimensionality d of Index Term Space

In the proposed HL-SOT approach, the dimensionality d of the index term space controls the number of terms to be indexed. If d is set too small, important useful terms will be missed that will limit the performance of the approach. However, if d is set too large, the computing efficiency will be decreased. Fig. 3 shows the impacts of the parameter d respectively on O-Loss, S-Loss, and H-Loss, where d varies from 50 to 300 with each step of 10 and the ϵ is set to be 0.005. From Fig. 3, we observe that as the d increases the O-Loss, S-Loss, and H-Loss generally decrease (performance increase). This means that when more terms are indexed better performance can be achieved by the HL-SOT approach. However,

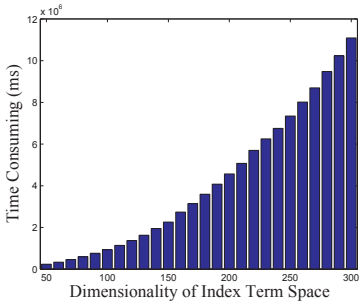


Figure 4: Time Consuming Impacted by d

considering the computing efficiency impacted by d, Fig. 4 shows that the computational complexity of our approach is non-linear increased with d's growing, which indicates that indexing more terms will improve the accuracy of our proposed approach although this is paid by decreasing the computing efficiency.

5 Conclusions, Discussions and Future Work

In this paper, we propose a novel and effective approach to sentiment analysis on product reviews. In our proposed HL-SOT approach, we define SOT to formulate the knowledge of hierarchical relationships among a product's attributes and tackle the problem of sentiment analysis in a hierarchical classification process with the proposed algorithm. The empirical analysis on a human-labeled data set demonstrates the promising results of our proposed approach. The performance comparison shows that the proposed HL-SOT approach outperforms two baselines: the HL-flat and the H-RLS approach. This confirms two intuitive motivations based on which our approach is proposed: 1) separately learning threshold values for

each classifier improve the classification accuracy; 2) knowledge of hierarchical relationships of labels improve the approach's performance. The experiments on analyzing the impact of parameter ϵ indicate that a fine-grained corrective step generally makes a better performance than a coarse-grained corrective step. The experiments on analyzing the impact of the dimensionality d show that indexing more terms will improve the accuracy of our proposed approach while the computing efficiency will be greatly decreased.

The focus of this paper is on analyzing review texts of one product. However, the framework of our proposed approach can be generalized to deal with a mix of review texts of more than one products. In this generalization for sentiment analysis on multiple products reviews, a "big" SOT is constructed and the SOT for each product reviews is a sub-tree of the "big" SOT. The sentiment analysis on multiple products reviews can be performed the same way the HL-SOT approach is applied on single product reviews and can be tackled in a hierarchical classification process with the "big" SOT.

This paper is motivated by the fact that the relationships among a product's attributes could be a useful knowledge for mining product review texts. The SOT is defined to formulate this knowledge in the proposed approach. However, what attributes to be included in a product's SOT and how to structure these attributes in the SOT is an effort of human beings. The sizes and structures of SOTs constructed by different individuals may vary. How the classification performance will be affected by variances of the generated SOTs is worthy of study. In addition, an automatic method to learn a product's attributes and the structure of SOT from existing product review texts will greatly benefit the efficiency of the proposed approach. We plan to investigate on these issues in our future work.

Acknowledgments

The authors would like to thank the anonymous reviewers for many helpful comments on the manuscript. This work is funded by the Research Council of Norway under the VERDIKT research programme (Project No.: 183337).

References

Alina Andreevskaia and Sabine Bergler. 2006. Mining wordnet for a fuzzy sentiment: Sentiment tag extraction from wordnet glosses. In *Proceedings of 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL'06)*, Trento, Italy.

Nicolò Cesa-Bianchi, Claudio Gentile, and Luca Zaniboni. 2006. Incremental algorithms for hierarchical classification. *Journal of Machine Learning Research (JMLR)*, 7:31–54.

Kushal Dave, Steve Lawrence, and David M. Pennock. 2003. Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In *Proceedings of 12nd International World Wide Web Conference (WWW'03)*, Budapest, Hungary.

Ann Devitt and Khurshid Ahmad. 2007. Sentiment polarity identification in financial news: A cohesion-based approach. In *Proceedings of 45th Annual Meeting of the Association for Computational Linguistics (ACL'07)*, Prague, Czech Republic.

Xiaowen Ding and Bing Liu. 2007. The utility of linguistic rules in opinion mining. In *Proceedings of 30th Annual International ACM Special Interest Group on Information Retrieval Conference (SIGIR'07)*, Amsterdam, The Netherlands.

Andrea Esuli and Fabrizio Sebastiani. 2005. Determining the semantic orientation of terms through gloss classification. In *Proceedings of 14th ACM Conference on Information and Knowledge Management (CIKM'05)*, Bremen, Germany.

Andrea Esuli and Fabrizio Sebastiani. 2006. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of 5th International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy.

Vasileios Hatzivassiloglou and Kathleen R. McKeown. 1997. Predicting the semantic orientation of adjectives. In *Proceedings of 35th Annual Meeting of the Association for Computational Linguistics (ACL'97)*, Madrid, Spain.

Vasileios Hatzivassiloglou and Janyce M. Wiebe. 2000. Effects of adjective orientation and gradability on sentence subjectivity. In *Proceedings of 18th International Conference on Computational Linguistics (COLING'00)*, Saarbrüken, Germany.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of 10th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'04)*, Seattle, USA.

Jaap Kamps, Maarten Marx, R. ort. Mokken, and Maarten de Rijke. 2004. Using WordNet to measure semantic orientation of adjectives. In *Proceedings of 4th International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal.

Bing Liu, Minqing Hu, and Junsheng Cheng. 2005. Opinion observer: analyzing and comparing opinions on the web. In *Proceedings of 14th International World Wide Web Conference (WWW'05)*, Chiba, Japan.

Yang Liu, Xiangji Huang, Aijun An, and Xiaohui Yu. 2007. ARSA: a sentiment-aware model for predicting sales performance using blogs. In *Proceedings of the 30th Annual International ACM Special Interest Group on Information Retrieval Conference (SIGIR'07)*, Amsterdam, The Netherlands.

Yue Lu and Chengxiang Zhai. 2008. Opinion integration through semi-supervised topic modeling. In *Proceedings of 17th International World Wide Web Conference (WWW'08)*, Beijing, China.

Yue Lu, ChengXiang Zhai, and Neel Sundaresan. 2009. Rated aspect summarization of short comments. In *Proceedings of 18th International World Wide Web Conference (WWW'09)*, Madrid, Spain.

Ana-Maria Popescu and Oren Etzioni. 2005. Extracting product features and opinions from reviews. In *Proceedings of Human Language Technology Conference and Empirical Methods in Natural Language Processing Conference (HLT/EMNLP'05)*, Vancouver, Canada.

Ivan Titov and Ryan T. McDonald. 2008. Modeling online reviews with multi-grain topic models. In *Proceedings of 17th International World Wide Web Conference (WWW'08)*, Beijing, China.

Peter D. Turney. 2002. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, Philadelphia, USA.

Casey Whitelaw, Navendu Garg, and Shlomo Argamon. 2005. Using appraisal taxonomies for sentiment analysis. In *Proceedings of 14th ACM Conference on Information and Knowledge Management (CIKM'05)*, Bremen, Germany.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of Human Language Technology Conference and Empirical Methods in Natural Language Processing Conference (HLT/EMNLP'05)*, Vancouver, Canada.

Hong Yu and Vasileios Hatzivassiloglou. 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of 8th Conference on Empirical Methods in Natural Language Processing (EMNLP'03)*, Sapporo, Japan.

Lina Zhou and Pimwadee Chaovalit. 2008. Ontology-supported polarity mining. *Journal of the American Society for Information Science and Technology (JASIST)*, 59(1):98–110.

Li Zhuang, Feng Jing, and Xiao-Yan Zhu. 2006. Movie review mining and summarization. In *Proceedings of the 15th ACM International Conference on Information and knowledge management (CIKM'06)*, Arlington, USA.

Chapter 10

Paper Five

Wei Wei, “Analyzing Text Data for Opinion Mining", Proceedings of the 16th International Conference on Applications of Natural Language to Information Systems, Lecture Notes in Computer Science Volume 6716, 2011, pp 330-335, Springer-Verlag Berlin, Heidelberg, ISBN 978-3-642-22326-6.

Is not included due to copyright

Chapter 11

Paper Six

Wei Wei and Jon Atle Gulla, “Enhancing the HL-SOT Approach to Sentiment Analysis via a Localized Feature Selection Framework", Proceedings of the 5th International Joint Conference on Natural Language Processing, pp. 327-335, Asian Federation of Natural Language Processing, ISBN 978-974-466-564-5.

Enhancing the HL-SOT Approach to Sentiment Analysis via a Localized Feature Selection Framework

Wei Wei
Department of Computer and Information Science
Norwegian University of Science and Technology
wwei@idi.ntnu.no

Jon Atle Gulla
Department of Computer and Information Science
Norwegian University of Science and Technology
jag@idi.ntnu.no

Abstract

In this paper, we propose a Localized Feature Selection (LFS) framework tailored to the HL-SOT approach to sentiment analysis. Within the proposed LFS framework, each node classifier of the HL-SOT approach is able to perform classification on target texts in a locally customized index term space. Extensive empirical analysis against a human-labeled data set demonstrates that with the proposed LFS framework the classification performance of the HL-SOT approach is enhanced with computational efficiency being greatly gained. To find the best feature selection algorithm that caters to the proposed LFS framework, five classic feature selection algorithms are comparatively studied, which indicates that the TS, DF, and MI algorithms achieve generally better performances than the CHI and IG algorithms. Among the five studied algorithms, the T-S algorithm is best to be employed by the proposed LFS framework.

1 Introduction

With tens and thousands of review texts being generated online, it becomes increasingly challenging for an individual to exhaustively collect and study the online reviews. Therefore, research on automatic sentiment analysis on review texts has emerged as a popular topic at the crossroads of information retrieval and computational linguistics.

Sentiment analysis on product reviews aims at extracting sentiment information from texts. It includes two tasks, i.e., labeling a target text[1] with 1) the product's attributes it mentions (attributes identification task), and 2) the corresponding sentiments mentioned therein (sentiment annotation task). Recently, Wei and Gulla proposed the HL-SOT approach (Wei and Gulla, 2010), i.e., Hierarchical Learning (HL) with Sentiment Ontology Tree (SOT), that is able to achieve the two tasks in one hierarchical classification process. In the HL-SOT approach, each target text is encoded by a vector in a globally unified d-dimensional index term space and is respectively labeled by different nodes[2] of SOT in a hierarchical manner. Although the HL-SOT approach is reported with promising classification performance on tasks of sentiment analysis, its computational efficiency, especially as d increases, becomes very low. Furthermore, as d increases it will have more chance to index noisy term into the globally unified index term space so that the classification performance of the HL-SOT approach might be depressed. Hence, we argue that if a locally customized index term space could be constructed for each node respectively, both the computational efficiency and the classification performance of the HL-SOT approach would be improved.

In this paper, we propose a Localized Feature Selection (LFS) framework tailored to the HL-SOT approach. The rationale of the proposed LFS framework draws on the following two observations. Firstly, a feature term that is relevant to a node is usually irrelevant to nodes which stay at another branch of SOT. For example, "ergonomics" might be a feature term for the node "design and usability" (see Fig. 1) but it is irrelevant to the node "image quality". Secondly, a feature ter-

[1] Each product review to be analyzed is called target text in the following of this paper.

[2] If specified otherwise in the following of this paper the term "node" refers to the classifier of the node.

Proceedings of the 5th International Joint Conference on Natural Language Processing, pages 327–335,
Chiang Mai, Thailand, November 8 – 13, 2011.

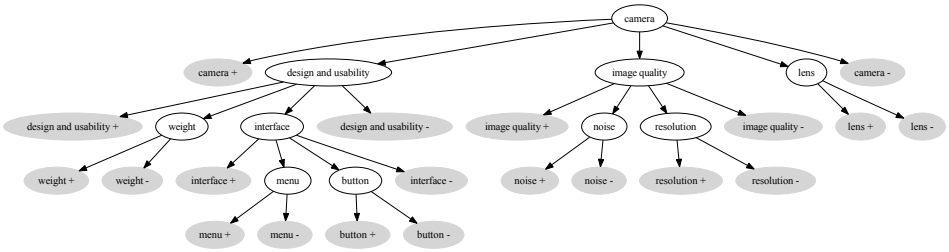


Figure 1: an example of part of a SOT for digital camera

m might become insignificant for child nodes of i even if the feature term is significant to i. For example, for a sentence commenting on a digital camera like "40D handles noise very well", terms such as "noise" and "well" are significant feature terms for the node "noise". However, the term "noise" becomes insignificant for its child nodes "noise +" and "noise -", since the hierarchical classification characteristic of the HL-SOT approach that a node only processes target texts which are labeled as true by its parent node ensures that each target text handled by the nodes "noise +" and "noise -" is already classified as related to "noise".

In the proposed LFS framework, the concept of "local hierarchy" is defined and introduced as delimitation of local scope of nodes. The localized feature selection process is conducted for each node within its local scope to generate the customized index term space for the node. The proposed LFS framework is empirically analyzed on a human-labeled data set. The experimental results show that with the proposed LFS framework the classification performance of the HL-SOT approach is enhanced and the computational efficiency is significantly improved. To test which is the best to be employed by the proposed LFS framework, we further comparatively study five classic feature selection algorithms respectively based on document frequency (DF) (Manning et al., 2008), mutual information (MI) (Manning et al., 2008; Church and Hanks, 1990), χ^2-statistic (CHI) (Manning et al., 2008), information gain (IG) (Mitchell, 1997), and term strength (TS) (Wilbur and Sirotkin, 1992). The comparatively experimental results suggest that the TS, DF, and MI algorithms achieve generally better performance than the CHI and IG algorithms. Among the five employed algorithms, the TS algorithm is the best to be employed by the proposed LFS framework. This paper makes the following contributions:

- We propose a LFS framework to enhance the classification performance and improve the computational efficiency of the HL-SOT approach;
- We conduct a comparative study on five feature selection algorithms that can be employed in the proposed LFS.

The remainder of the paper is organized as follows. In section 2, we discuss an overview of related work on sentiment analysis. In section 3, we review the HL-SOT approach proposed in (Wei and Gulla, 2010). In section 4, we present the proposed LFS framework. The empirical analysis and the results are presented in section 5. Finally, we conclude the paper and discuss the future work in section 6.

2 Related Work

Research on sentiment analysis was originally performed to extract overall sentiments from target texts. However, as shown in the experiments in (Turney, 2002), the whole sentiment of a document is not necessarily the sum of its parts. Recent work has shifted the focus from overall document sentiment to sentiment analysis based on product attributes (Hu and Liu, 2004; Popescu and Etzioni, 2005; Ding and Liu, 2007; Liu et al., 2005).

Document overall sentiment analysis is to summarize the overall sentiment in the document, which relies on two finer levels of sentiment annotation: *word-level sentiment annotation* and *phrase-level sentiment annotation*. The *word-level sentiment annotation* is to utilize the polarity annotation of words in each sentence and summarize the overall sentiment of each sentiment-bearing word to infer the overall sentiment within

the text (Hatzivassiloglou and Wiebe, 2000; Andreevskaia and Bergler, 2006; Esuli and Sebastiani, 2005; Esuli and Sebastiani, 2006; Hatzivassiloglou and McKeown, 1997; Kamps et al., 2004; Devitt and Ahmad, 2007; Yu and Hatzivassiloglou, 2003). The *phrase-level sentiment annotation* focuses sentiment annotation on phrases not words with concerning that atomic units of expression is not individual words but rather appraisal groups (Whitelaw et al., 2005). In (Wilson et al., 2005), the concepts of *prior polarity* and *contextual polarity* were proposed. This paper presented a system that is able to automatically identify the *contextual polarity* for a large subset of sentiment expressions. In (Turney, 2002), an unsupervised learning algorithm was proposed to classify reviews as recommended or not recommended by averaging sentiment annotation of phrases in reviews that contain adjectives or adverbs. However, the performances of these approaches are not satisfactory for sentiment analysis on product reviews, where sentiment on each attribute of a product could be so complicated that it is unable to be expressed by overall document sentiment.

Attributes-based sentiment analysis is to analyze sentiment based on each attribute of a product. In (Hu and Liu, 2004), mining product features was proposed together with sentiment polarity annotation for each opinion sentence. In that work, sentiment analysis was performed at the product attributes level. In (Liu et al., 2005), a system with framework for analyzing and comparing consumer opinions of competing products was proposed. The system made users be able to clearly see the strengths and weaknesses of each product in the minds of consumers in terms of various product features. In (Popescu and Etzioni, 2005), Popescu and Etzioni not only analyzed polarity of opinions regarding product features but also ranked opinions based on their strength. In (Liu et al., 2007), Liu et al. proposed Sentiment-PLSA that analyzed blog entries and viewed them as a document generated by a number of hidden sentiment factors. These sentiment factors may also be factors based on product attributes. In (Lu and Zhai, 2008), Lu et al. proposed a semi-supervised topic models to solve the problem of opinion integration based on the topic of a product's attributes. The work in (Titov and McDonald, 2008) presented a multi-grain topic model for extracting the ratable attributes from product reviews. In (Lu et al., 2009), the problem of rated attributes summary was studied with a goal of generating ratings for major aspects so that a user could gain different perspectives towards a target entity. In a most recent research work (Wei and Gulla, 2010), Wei and Gulla proposed the HL-SOT approach that sufficiently utilizes the hierarchical relationships among a product attributes and solves the sentiment analysis problem in a hierarchical classification process. However, the HL-SOT approach proposed in (Wei and Gulla, 2010) uses a globally unified index term space to encode target texts for different nodes which is deemed to limit the performance of the HL-SOT approach. Therefore, the LFS framework proposed in this paper aims at overcoming the weakness of the HL-SOT approach and consequently improving its performance by generating a locally customized index term space for each node.

3 The HL-SOT Approach Review

In the HL-SOT approach (Wei and Gulla, 2010), each target text is indexed by a vector $x \in \mathcal{X}, \mathcal{X} = \mathbb{R}^d$. Weight vectors $w_i (1 \leq i \leq N)$ define linear-threshold classifiers of each node i in SOT so that the target text x is labeled true by node i if x is labeled true by i's parent node and $w_i \cdot x \geq \theta_i$. The parameters w_i and θ_i are learned from the training data set: $D = \{(r, l) | r \in \mathcal{X}, l \in \mathcal{Y}\}$, where $\mathcal{Y}$ denotes the set of label vectors. In the training process, when a new instance r_t is observed, each row vector $w_{i,t}$ is updated by a regularized least squares estimator given by:

$$w_{i,t} = (I + S_{i,Q(i,t-1)} S_{i,Q(i,t-1)}^{\top} + r_t r_t^{\top})^{-1} \\ S_{i,Q(i,t-1)} (l_{i,i_1}, l_{i,i_2}, ..., l_{i,i_{Q(i,t-1)}})^{\top} \quad (1)$$

where I is a $d \times d$ identity matrix, $Q(i, t-1)$ denotes the number of times the parent of node i observes a positive label before observing the instance r_t, $S_{i,Q(i,t-1)} = [r_{i_1}, ..., r_{i_{Q(i,t-1)}}]$ is a $d \times Q(i, t-1)$ matrix whose columns are the instances $r_{i_1}, ..., r_{i_{Q(i,t-1)}}$, and $(l_{i,i_1}, l_{i,i_2}, ..., l_{i,i_{Q(i,t-1)}})^{\top}$ is a $Q(i, t-1)$-dimensional vector of the corresponding labels observed by node i. The Formula 1 restricts that the weight vector $w_{i,t}$ of the classifier i is only updated on the examples that are positive for its parent node. Then the label vector $\hat{y}_{r_t}$ is computed for the instance r_t, before the real label vector l_{r_t} is observed. Then the current threshold vector θ_t is updated by:

$$\theta_{t+1} = \theta_t + \epsilon(\hat{y}_{r_t} - l_{r_t}), \quad (2)$$

where ϵ is a small positive real number that denotes a corrective step for the current threshold vector θ_t. After the training process for each node of SOT, each target text is to be labeled by each node i parameterized by the weight vector w_i and the threshold θ_i in the hierarchical classification process.

4 The Localized Feature Selection

In this section, we propose the LFS framework to generate a locally customized index term space for each node of SOT respectively. We first discuss why localized feature selection is needed for the HL-SOT approach. Then we define the concept of local hierarchy of SOT to introduce the local feature selection scope of a node, followed by a presentation on the local hierarchy based feature selection process.

4.1 Why Localized Feature Selection for the HL-SOT

One deficiency of the HL-SOT approach is that it uses a globally unified index term space to index target texts, which cannot efficiently encode feature information required by each local individual node of SOT. When we look into the detailed classification process of each node of SOT, we observe the following two types of phenomena. Firstly, SOT organizes domain knowledge in a tree like structure. Within a particular domain knowledge represented by SOT, nodes that stay in different branches of SOT represent independent different attributes in that domain. In this way, feature terms (e.g., the term "ergonomics") that are relevant to a node (e.g., the node "design and usability") might be irrelevant to other nodes (e.g., the node "image quality") that stay at another branches of SOT; Secondly, the HL-SOT approach labels each target text in a hierarchical order which ensures that each target text that comes to be handled by a node has already been labeled as true by its parent node. Due to this characteristic, feature terms (e.g., the term "noise") that are significant to a node i (e.g., the node "noise") might become a trivial term for i's child nodes (e.g., the nodes "noise +" and "noise -"). Therefore, the purpose of the localized feature selection is to filter out irrelevant terms that are insignificant to each individual node and build a locally customized index term space for the node so that the performance of the node can be improved.

4.2 Local Feature Selection Scope for a Node

In order to select locally customized feature terms for each individual node, we need to define a suitable scope, called local feature selection scope[3], within which the feature selection process can be effectively conducted for the node. Since the HL-SOT approach is a hierarchical classification process, before we introduce the local scope for a node we first give a formal definition on local hierarchy of SOT.

Definition 1 *[Local Hierarchy] A local hierarchy Δ_u of SOT is defined to be formed by all the child nodes of u in SOT, where the node u must be a non-leaf node of the SOT.*

By the Definition 1, we say all the child nodes of u are on the same local hierarchy under u which is denoted by Δ_u. For examples, in Fig. 2 nodes "camera +", "design and usability", "image quality", "lens", "camera -" are deemed on the same local hierarchy under the node "camera" and nodes "weight +", "weight -" are deemed on the same local hierarchy under the node "weight", etc. In the hierarchical labeling process of the HL-SOT approach, after a target text is labeled as true by a node i it will go further to the local hierarchy under i and is to be labeled by all nodes on the local hierarchy Δ_i. For a target text the labeling processes of nodes on Δ_i locally can be considered as a multi-label classification process where each node is a local label. Therefore, the measurement for selecting terms as features should be calculated among nodes on the same hierarchy. Hence, the local scope for a node is defined within the local hierarchy which the node is on.

4.3 Local Hierarchy Based Feature Selection

In the proposed LFS framework, local feature selection for a node i of SOT is performed within the *local scope* of the node i. Since nodes on the same local hierarchy share the same local scope, local feature selection process for all nodes of SOT is achieved in local hierarchy based manner. Specifically, for the feature selection process on a local hierarchy Δ, let $c_1, c_2, ..., c_K$ denote the K nodes on Δ. Let D denote the training data set for the HL-SOT approach. Let D_{c_k} denote the set of instances in D that contains the label of the node $c_k (1 \leqslant k \leqslant K)$. Let D_Δ denote the training corpus for the local hierarchy Δ which is the set of

[3] In this paper, we also call it "local scope" for short.

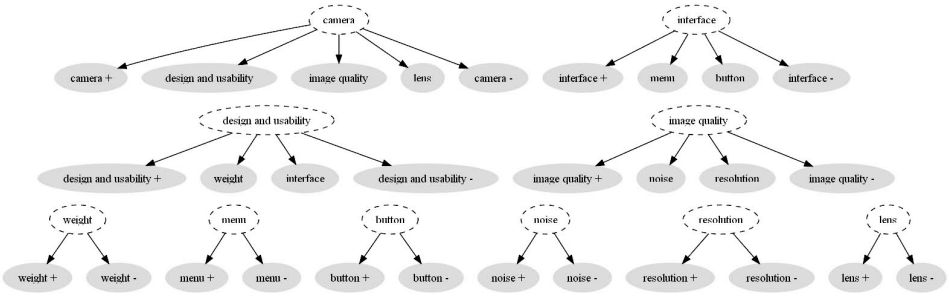


Figure 2: All local hierarchies of the example SOT: the grey nodes sharing the same parent node in dashed line are called on the same local hierarchy under the parent node

all instances in the training data set D that contain any label of nodes on the local hierarchy Δ: $D_\Delta = \bigcup_{k=1}^{K} D_{c_k}$. Let V_{c_k} denote the set of all the vocabularies that appears in D_{c_k}. Let $s_{c_k}(w)$ denote the term score that measures the suitability of w as a feature for node c_k. Let F_{c_k} denote the set of feature terms selected for c_k. Let d_{c_k} denote the number of features to be selected in F_{c_k}. A local feature selection process for nodes on the local hierarchy Δ is described in Algorithm 1.

Algorithm 1 Localized Feature Selection Algorithm

DATA INITIALIZATION:
for each node c_k on Δ **do**
  Establish D_{c_k} containing instances being labeled true by c_k;
  Establish the vocabulary set V_{c_k};
  Remove stop words from V_{c_k};
end for
Establish the training corpus: $D_\Delta = \bigcup_{k=1}^{K} D_{c_k}$;
BEGIN
for each node c_k on Δ **do**
  for each term $w \in V_{c_k}$ **do**
    with training corpus D_Δ and data instance set D_{c_k}:
    Calculate $s_{c_k}(w)$ with a specified feature selection algorithm;
  end for
  Establish feature space F_{c_k} with top d_{c_k} terms from V_{c_k};
end for
END

In the data initialization phase of the Algorithm 1, the data instance set D_{c_k} and vocabulary set V_{c_k} for each node on the local hierarchy Δ as well as the training corpus D_Δ are established. In a local feature selection process, a term score $s_{c_k}(w)$ for each term $w \in V_{c_k}$ can be calculated by a specified feature selection algorithm, taking D_Δ as the training corpus and D_{c_k} as the data instance set in the class c_k. The local feature selection process can employ any specific feature selection algorithm to calculate the term scores. After all terms in V_{c_k} are calculated, those terms with top d_{c_k} scores are selected to establish the feature space F_{c_k} for the node c_k. Since the number of terms in V_{c_k} varies from node to node, in order to produce a rational dimensionality d_{c_k} for the established feature space F_{c_k}, we introduce a feature selection rate, denoted by γ, to control d_{c_k} for each node c_k, i.e., $d_{c_k} = \ulcorner |V_{c_k}| \times \gamma \urcorner$.

After local feature selection processes for all the nods of SOT are accomplished, a locally customized index term space F_{c_k} for each node c_k is established. Each target text will be respectively indexed by a customized vector $x_{c_k} \in \mathcal{X}_{c_k} (\mathcal{X}_{c_k} = \mathbb{R}^{d_{c_k}})$ when it goes through the hierarchical classification process of the HL-SOT approach. In next section, we will present the empirical analysis on evaluating the proposed LFS framework.

5 Empirical Analysis

In this section, we conduct extensive experiments to empirically analyze the proposed LFS framework. Our experiments are intended to address the following questions: (1) can the classification performance of the HL-SOT approach be improved with the LFS framework; (2) how much computational efficiency can be gained for the HL-SOT to be implemented in the LFS framework; (3) how are the comparative performances produced by different feature selection algorithms when employed in the proposed LFS framework.

5.1 Data Set Preparation

We construct our data set based on the digital camera review data set used in the HL-SOT approach (Wei and Gulla, 2010). In total, the constructed data set contains 1500 snippets of customer reviews on digital cameras, where 35 attributes of a digital camera are mentioned in the

review data. We build an ontology structure to organize the mentioned attributes and label each review text with correspondent attributes as well as sentiments, which complying the rule that if a review text is assigned with a label of a node then it is assigned with a label of the parent node. We randomly divide the labeled data set into five folds so that each fold at least contains one example instance labeled by each attribute node. To catch the statistical significance of experimental results, we perform 5 cross-fold evaluation by using four folds as training data and the other one fold as testing data. All the experimental results presented in this section are averaged over 5 runs of each experiment.

5.2 Evaluation Metrics

Since the existing HL-SOT approach is a hierarchical classification process, we use the same three classic loss functions (Wei and Gulla, 2010) for measuring classification performance. They are respectively the One-error Loss (O-Loss) function, the Symmetric Loss (S-Loss) function, and the Hierarchical Loss (H-Loss) function[4]:

One-error loss (O-Loss) function is defined as:

$$L_O(\hat{y}, l) = \mathfrak{B}(\ i : \hat{y}_i = l_i), \tag{3}$$

where $\hat{y}$ is the prediction label vector and l is the true label vector; $\mathfrak{B}(S)$ is a boolean function which is 1 if and only if the statement S is true, otherwise it is 0.

Symmetric loss (S-Loss) function is defined as:

$$L_S(\hat{y}, l) = \sum_{i=1}^{N} \mathfrak{B}(\hat{y}_i = l_i), \tag{4}$$

Hierarchical loss (H-Loss) function is defined as:

$$L_H(\hat{y}, l) = \sum_{i=1}^{N} \mathfrak{B}(\hat{y}_i = l_i \quad j \quad (i), \hat{y}_j = l_j), \tag{5}$$

where denotes a set of nodes that are ancestors of node i in SOT.

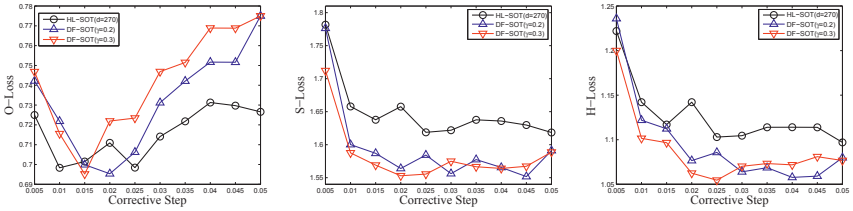
5.3 Performance Comparison

In this section, we conduct experiments to show performance improvement from the proposed LFS framework. The performance considered here include both classification performance and computational efficiency. We use the existing HL-SOT approach as a baseline. Since the HL-SOT approach used terms' document frequencies (DF) (Manning et al., 2008) algorithm to select features to build the globally unified index term space, employing the same DF feature selection algorithm we apply the proposed LFS framework on the HL-SOT approach and call the implemented method "DF-SOT". The only difference between HL-SOT and DF-SOT is the index term space for each node of SOT, i.e., in the HL-SOT all the nodes using the globally unified index term space while in the DF-SOT each node respectively using a locally customized index term space. In this way, the performance difference between the two methods will indicate the effect of the proposed LFS framework.

5.3.1 Comparison on Classification Performance

We conduct experiments to investigate whether the classification performance of the HL-SOT can be improved when it is implemented with the LFS framework. Fig. 3 presents the experimental results of classification accuracies between HL-SOT and DF-SOT. In the experiments, the dimensionality d of the globally unified index term space of the HL-SOT approach is set to 270, which is large enough for the HL-SOT approach to reach its best performance level. The feature selection rate γ for the locally customized index term space of the DF-SOT approach is set to 0.2 and 0.3, which brings respectively 80% and 70% vocabulary reduction. The value of the corrective step ϵ is set to varying from 0.005 to 0.05 with each step of 0.005 so that each running approach can achieve its best performance with a certain value of ϵ. From Fig. 3, we can observe that when $\gamma = 0.2$ the DF-SOT approach reaches its best performance with 0.6953 ($\epsilon = 0.02$) on O-Loss, 1.5516 ($\epsilon = 0.045$) on S-Loss, and 1.0578 ($\epsilon = 0.04$) on H-Loss, and that when $\gamma = 0.3$ the DF-SOT approach reaches its best performance with 0.6953 ($\epsilon = 0.015$) on O-Loss, 1.5531 ($\epsilon = 0.02$) on S-Loss, and 1.0547 ($\epsilon = 0.025$) on H-Loss, which outperforms the best performance of the HL-SOT approach on O-Loss 0.6984 ($\epsilon = 0.025$), on S-Loss 1.6188 ($\epsilon = 0.025$), and on H-Loss 1.0969 ($\epsilon = 0.05$). This indicates that with the proposed LFS framework, compared with the HL-SOT approach, the DF-SOT approach generally improves the classification performance.

[4]Since the three loss functions are respectively well-defined by each formula and self-explained by their names, due to the space limitation, we do not present more explanation.



(a) O-Loss (b) S-Loss (c) H-Loss

Figure 3: Classification Performance (A Smaller Loss Value Means Better Classification Performance)

5.3.2 Comparison on Computational Efficiency

We conduct further experiments to analyze computational efficiency gained through the proposed LFS framework. All the experiments are conducted on a normal personal computer containing an Intel Pentium D CPU (2.4 GHz, Dual Core) and 4G memory. Fig. 4 summarizes the computational time consumed by experiment runs respectively for HL-SOT ($d = 270$) and DF-SOT ($\gamma = 0.2$ and $\gamma = 0.3$). From Fig. 4, we can observe that the HL-SOT approach consumes 15917695 ms to finish an experimental run, although the DF-SOT approach only takes respectively 2.29% (with $\gamma = 0.2$) and 4.91% (with $\gamma = 0.2$) of computational time as the existing HL-SOT approach consumes and achieves even better classification performance than the HL-SOT approach (see Fig.3). This confirms that much computational efficiency can be gained for the HL-SOT approach to be implemented in the LFS framework while better classification performance is ensured. Since the computational complexity of each node classifier of DF-SOT is the same as HL-SOT, the computational efficiency gained from the proposed LFS framework should be attributed to the dimension reduction of the index term space.

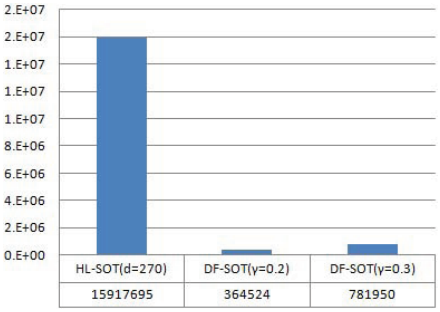


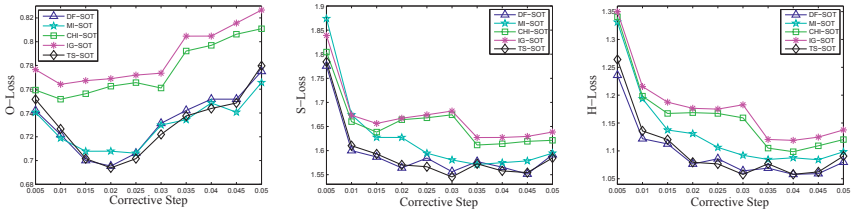
Figure 4: Time Consuming (ms)

5.4 Comparative Study on Feature Selection Algorithms

The proposed LFS framework for the HL-SOT approach can employ various feature selection algorithms to select local features for each individual node. In this section, we conduct intensive experiments to comparatively study five classic feature selection algorithms employed within the LFS framework. The five employed feature selection algorithms are respectively document frequency (DF) (Manning et al., 2008) based feature selection algorithm, mutual information (MI) (Manning et al., 2008; Church and Hanks, 1990) based feature selection algorithm, χ^2-statistic (CHI) (Manning et al., 2008) based feature selection algorithm, information gain (IG) (Mitchell, 1997) based feature selection algorithm as well as term strength (TS) (Wilbur and Sirotkin, 1992) based feature selection algorithm[5].

In the experiments, the feature selection rate γ is set to 0.2 and 0.3 respectively. The value of the corrective step ϵ varies from 0.005 to 0.05 with each step of 0.005. The experimental results are summarized in Fig. 5. From Fig. 5 it is observed that DF, MI, and TS feature selection algorithms achieve generally better performances than CHI and IG feature selection algorithms when they are employed in the proposed LFS framework. Specifically, the TS algorithm is generally the best among the five employed algorithms while the DF algorithm can also achieve as

[5]Due to the space limitation, details of the studied feature selection algorithms are not reviewed here. The mechanism of each algorithm can be read in the related references.



(a) O-Loss ($\gamma = 0.2$) (b) S-Loss ($\gamma = 0.2$) (c) H-Loss ($\gamma = 0.2$)

(d) O-Loss ($\gamma = 0.3$) (e) S-Loss ($\gamma = 0.3$) (f) H-Loss ($\gamma = 0.3$)

Figure 5: Comparative Performances on the Employed Feature Selection Algorithms

comparable good performance as the TS algorithm does. This is due to that both the TS and the DF algorithms favor high frequency terms and vocabularies used in customer reviews on a specific product are usually overlapping. When $\gamma = 0.3$, it can be also observed that the MI algorithm achieves as comparable good performance as the TS algorithm does. This is because, in customer reviews, although some vocabularies are rarely used they always occur as significant features in some specific categories. For example, "ergonomics" is a rare term but almost always appears in the class of "design and usability". Therefore, the MI algorithm can also achieve relatively better performance through favoring rare terms that always co-occur with specific classes.

6 Conclusions and Future Work

In this paper, we propose a LFS framework tailored to the HL-SOT approach to sentiment analysis. In the proposed LFS framework, significant feature terms of each node can be selected to construct the locally customized index term space for the node so that the classification performance and computational efficiency of the existing HL-SOT approach are improved. The effectiveness of the proposed LFS is validated against a human-labeled data set. Further comparative study on five employed feature selection algorithms within the proposed LFS framework indicates that the TS, DF, and MI algorithms achieve generally better performance than the CHI and IG algorithms. Among the five employed algorithms, the TS algorithm is the best to be employed by the proposed LFS framework.

Although the proposed LFS framework shows its effectiveness of improving on the HL-SOT approach, its improvement on the classification performance is not so obvious compared with its much improvement on computational efficiency. Due to the limited number of instances in the training data set, the classification performance still suffers from the problem that unobserved terms appear in testing cases. This problem is inherently raised by the bag-of-word model. A concept-based indexing scheme that can infer concepts of unobserved terms might alleviate the problem. We plan to investigate on this issue in the future work.

Acknowledgments

The authors would like to thank the anonymous reviewers for the helpful comments on the manuscript. This work is funded by the Research Council of Norway under the VERDIKT research programme (Project No.: 183337).

References

Alina Andreevskaia and Sabine Bergler. 2006. Mining wordnet for a fuzzy sentiment: Sentiment tag extraction from wordnet glosses. In *Proceedings of 11th Conference of the European Chapter of the Association for Computational Linguistics*.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29.

Ann Devitt and Khurshid Ahmad. 2007. Sentiment polarity identification in financial news: A cohesion-based approach. In *Proceedings of 45th Annual Meeting of the Association for Computational Linguistics*.

Xiaowen Ding and Bing Liu. 2007. The utility of linguistic rules in opinion mining. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and development in Information Retrieval*.

Andrea Esuli and Fabrizio Sebastiani. 2005. Determining the semantic orientation of terms through gloss classification. In *Proceedings of 14th ACM Conference on Information and Knowledge Management*.

Andrea Esuli and Fabrizio Sebastiani. 2006. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of 5th International Conference on Language Resources and Evaluation*.

Vasileios Hatzivassiloglou and Kathleen R. McKeown. 1997. Predicting the semantic orientation of adjectives. In *Proceedings of 35th Annual Meeting of the Association for Computational Linguistics*.

Vasileios Hatzivassiloglou and Janyce M. Wiebe. 2000. Effects of adjective orientation and gradability on sentence subjectivity. In *Proceedings of 18th International Conference on Computational Linguistics*.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of 10th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Jaap Kamps, Maarten Marx, R. ort. Mokken, and Maarten de Rijke. 2004. Using WordNet to measure semantic orientation of adjectives. In *Proceedings of 4th International Conference on Language Resources and Evaluation*.

Bing Liu, Minqing Hu, and Junsheng Cheng. 2005. Opinion observer: analyzing and comparing opinions on the web. In *Proceedings of 14th International World Wide Web Conference*.

Yang Liu, Xiangji Huang, Aijun An, and Xiaohui Yu. 2007. ARSA: a sentiment-aware model for predicting sales performance using blogs. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and development in Information Retrieval*.

Yue Lu and Chengxiang Zhai. 2008. Opinion integration through semi-supervised topic modeling. In *Proceedings of 17th International World Wide Web Conference*.

Yue Lu, ChengXiang Zhai, and Neel Sundaresan. 2009. Rated aspect summarization of short comments. In *Proceedings of 18th International World Wide Web Conference*.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schutze, 2008. *Introduction to Information Retrieval*, chapter 13, pages 271–278. Cambridge University Press.

Tom Mitchell. 1997. *Machine Learning*. McGraw-Hill.

Ana-Maria Popescu and Oren Etzioni. 2005. Extracting product features and opinions from reviews. In *Proceedings of Human Language Technology Conference and Empirical Methods in Natural Language Processing Conference*.

Ivan Titov and Ryan T. McDonald. 2008. Modeling online reviews with multi-grain topic models. In *Proceedings of 17th International World Wide Web Conference*.

Peter D. Turney. 2002. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*.

Wei Wei and Jon Atle Gulla. 2010. Sentiment learning on product reviews via sentiment ontology tree. In *Proceedings of 48th Annual Meeting of the Association for Computational Linguistics*.

Casey Whitelaw, Navendu Garg, and Shlomo Argamon. 2005. Using appraisal taxonomies for sentiment analysis. In *Proceedings of 14th ACM Conference on Information and Knowledge Management*.

J. W. Wilbur and K. Sirotkin. 1992. The automatic identification of stop words. *Journal of the American Society for Information Science*, 18:45–55.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of Human Language Technology Conference and Empirical Methods in Natural Language Processing Conference*.

Hong Yu and Vasileios Hatzivassiloglou. 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of 8th Conference on Empirical Methods in Natural Language Processing*.

Chapter 12

Paper Seven

Wei Wei and Jon Atle Gulla, “Sentiment Analysis in a Hybrid Hierarchical Classification Process", Proceedings of the 7th International Conference on Digital Information Management, pp. 47-55, IEEE 2012, ISBN 978-1-4673-2428-1.

Is not included due to copyright