Sentiment Analysis with Limited Training Data

Dissertation
zur Erlangung des Grades
Doktor der Ingenieurwissenschaften (Dr.-Ing.)
der Naturwissenschaftlich-Technischen Fakultät I
der Universität des Saarlandes

Lizhen Qu
Max-Planck-Institut für Informatik

Saarbrücken
2013

Dekan der Naturwissenschaftlich-Technischen Fakultät	Prof. Dr. Mark Groves
Vorsitzender der Prüfungskommission	Prof. Dietrich Klakow
Berichterstatter	Prof. Dr.-Ing. Gerhard Weikum
Berichterstatter	Dr.-Ing. Rainer Gemulla
Beisitzer	Dr.-Ing. Klaus Lorenz Berberich
Tag des Promotionskollquiums	04.12.2013

Abstract

Sentiments are positive and negative emotions, evaluations and stances. This dissertation focuses on learning based systems for automatic analysis of sentiments and comparisons in natural language text. The proposed approach consists of three contributions:

1. **Bag-of-opinions model**: For predicting document-level polarity and intensity, we proposed the bag-of-opinions model by modeling each document as a bag of sentiments, which can explore the syntactic structures of sentiment-bearing phrases for improved rating prediction of online reviews.

2. **Multi-experts model**: Due to the sparsity of manually-labeled training data, we designed the multi-experts model for sentence-level analysis of sentiment polarity and intensity by fully exploiting any available sentiment indicators, such as phrase-level predictors and sentence similarity measures.

3. **SENTI-LSSVM$_{\text{RAE}}$ model**: To understand the sentiments regarding entities, we proposed SENTI-LSSVM$_{\text{RAE}}$ model for extracting sentiments and comparisons of entities at both sentence and subsentential level.

Different granularity of analysis leads to different model complexity, the finer the more complex. All proposed models aim to minimize the use of hand-labeled data by maximizing the use of the freely available resources. These models explore also different feature representations to capture the compositional semantics inherent in sentiment-bearing expressions. Our experimental results on real-world data showed that all models significantly outperform the state-of-the-art methods on the respective tasks.

Kurzfassung

Sentiments sind positive und negative Gefühle, Bewertungen und Einstellungen. Die Dissertation beschäftigt sich mit lernbasierten Systemen zur automatischen Analyse von Sentiments und Vergleichen in Texten in natürlicher Sprache. Die vorliegende Abeit leistet dazu drei Beiträge:

1. **Bag-of-Opinions-Modell**: Zur Vorhersage der Polarität und Intensität auf Dokumentenebene haben wir das Bag-of-Opinions-Modell vorgeschlagen, bei dem jedes Dokument als ein Beutel Sentiments dargestellt wird. Das Modell kann die syntaktischen Strukturen von subjektiven Ausdrücken untersuchen, um eine verbesserte Bewertungsvorhersage von Online-Rezensionen zu erzielen.

2. **Multi-Experten-Modell**: Wegen des Mangels an manuell annotierten Trainingsdaten haben wir das Multi-Experten-Modell entworfen, um die Sentimentpolarität und -intensität auf Satzebene zu analysieren. Das Modell kann alle möglichen Sentiment-Indikatoren verwenden, wie Prädiktoren auf Phrasenebene und Ähnlichkeitsmaße von Sätzen.

3. **SENTI-LSSVM$_{\text{RAE}}$ -Modell**: Um Sentiments von Entitäten zu verstehen, wir haben wir das SENTI-LSSVM$_{\text{RAE}}$ -Modell zur Extraktion von Sentiments und Vergleichen von Entitäten auf Satz- und Ausdrucksebene vorgeschlagen.

Die unterschiedliche Granularität der Analyse führt zu unterschiedlicher Modellkomplexität; je feiner, desto komplexer. Alle vorgeschlagenen Modelle zielen darauf ab, möglichst wenige manuell annotierte Daten und möglichst viele frei verfügbare Ressourcen zu verwenden. Diese Modelle untersuchen auch verschiedene Merkmalsdarstellungen, um die Kompositionssemantik abzubilden, die subjektiven Ausdrücken inhärent ist. Die Ergebnisse unserer Experimente mit Realweltdaten haben gezeigt, dass alle Modelle für die

jeweiligen Aufgaben deutlich bessere Leistungen erzielen als die modernsten Methoden.

Summary

Sentiments are positive and negative emotions, evaluations and stances. This dissertation is concerned with automatic analysis of sentiments and comparisons in natural language text. The proposed approach consists of three contributions for sentiment analysis at different granularity levels:

Document-level analysis. The first contribution is the bag-of-opinions (BoO) model, which predicts *document-level* polarity and intensity of online reviews. The two quantities are *jointly* represented as numerical ratings, thus the task is also considered as document-rating prediction. The BoO model treats each document as a bag of sentiments, which correspond to a set of sentiment-bearing expressions. A sentiment-bearing expression consists of a root word with a known prior polarity (e.g. interesting), optionally a set of modifier words (e.g. very) and negation words (e.g. not). Although the composition rules regarding these components imply a non-linear function, the new feature representation of sentiment enables the learning with a linear model. Since the prior polarity of root words is obtained from domain-independent subjectivity lexicons, the linear model cannot capture the domain-dependent information. To circumvent the problem, we developed a two-stage approach: i) in the first stage, we learn a bag-of-opinions model on a large dataset of online reviews to obtain scores for domain-independent sentiments; ii) in the second stage, we combine the bag-of-opinions model with an unigram model trained on the domain-dependent corpus.

Sentence-level analysis. The second contribution is the weakly supervised multi-experts model (MEM) for sentence rating prediction. This model can exploit any available sentiment indicators, such as phrase-level predictors, language heuristics, co-occurrence counts and word order sensitive similarity measures of sentences. Instead of using them as features, these indicators

constitute the set of experts used in the model, which are similar to the ideas of ensemble learning.

Subsentential analysis. The third contribution is the SENTI-LSSVM$_{RAE}$ model, which is the first learning system to identify sentiments and comparisons of entities at both sentence and subsentential level. Unlike methods that rely on a full-fledged training corpus, we do not require any explicit annotation of sentiment-bearing expressions. To train such a model, we only need to annotate the entity mentions and their relationships, which are either sentiments, comparisons or other cases. Since sentiment-bearing expressions and their association to relationships do not exist in the training data, this model can detect the expressions and associate them to respective relationships in both training and testing. Moreover, to address the compositional semantics of sentiment-bearing expressions, we apply deep learning techniques to construct latent feature vectors to encode compositional patterns, which make the learning much easier.

Different granularity of analysis leads to different model complexity, the finer the more complex. All proposed models aim to minimize the use of hand-labeled data by maximizing the use of the freely available resources. These models explore also different feature representations to capture the compositional semantics inherent in sentiment-bearing expressions. A survey about sentiment analysis and semi-supervised learning completes the thesis.

Zusammenfassung

Sentiments sind positive und negative Gefühle, Bewertungen und Einstellungen. Diese Dissertation beschäftigt sich mit der automatischen Analyse von Sentiments und Vergleichen in Texten in natürlicher Sprache. Der vorliegende Ansatz liefert drei Beiträge zur Sentiment-Analyse mit unterschiedlicher Granularität:

Analyse auf Dokumentebene. Der erste Beitrag ist das Bag-of-Opinions-Modell (BoO), welches auf *Dokumentebene* die Polarität und Intensität von Online-Rezensionen vorhersagt. Die beiden Mengen werden *zusammen* als numerische Bewertungen dargestellt, weshalb diese Aufgabe auch als Dokumentbewertungsvorhersage betrachtet wird. Das BoO-Modell behandelt jedes Dokument wie einen Beutel Sentiments, welcher einer Menge subjektiver Ausdrücke entspricht. Ein subjektiver Ausdruck besteht aus einem Stammwort mit einer bekannten vorangestellten Polarität (z.B. interessant), einer optionalen Menge Modifikatoren (z.B. sehr) und Verneinungen (z.B. nicht). Obwohl die Zusammensetzungsregeln bei diesen Komponenten eine nichtlineare Funktion voraussetzen, ermöglicht die neue Merkmalsdarstelllung von Sentiments das Lernen mit einem linearen Modell. Da die vorangestellte Polarität der Stammwörter durch domain-unabhängige Subjektivitätslexika bestimmt wird, kann das lineare Modell nicht die domain-abhängigen Informationen berücksichtigen. Um dieses Problem zu umgehen, haben wir einen zweistufigen Ansatz entwickelt: i) im ersten Schritt trainieren wir ein Bag-of-Opinions-Modell anhand einer großen Datenmenge von Online-Rezensionen, um die Punkte für domain-unabhängige Sentiments zu erhalten; ii) im zweiten Schritt kombinieren wir das Bag-of-Opinions-Modell mit einem Unigramm-Modell, das anhand eines domain-abhängigen Korpus trainiert wurde.

Analyse auf Satzebene. Der zweite Beitrag besteht aus dem schwach überwachten Multi-Experten-Modell (MEM) zur Vorhersage von Satzbewertungen. Dieses Modell kann alle verfügbaren Sentiment-Indikatoren nutzen, wie Prädiktoren auf Phrasenebene, Sprachheuristiken, Kookkurrenzzählungen und wortstellungssensible Ähnlichkeitsmaße von Sätzen. Anstatt sie als Merkmale zu verwenden, bilden diese Indikatoren die Expertenmenge, die im Modell verwendet wird, was an die Ideen des Ensemble Learning erinnert.

Ausdrucksanalyse. Der dritte Beitrag ist das SENTI-LSSVM$_{RAE}$ -Modell, welches das erste Lernsystem ist, das Sentiments und Vergleiche von Entitäten auf Satz- und Ausdrucksebene identifizieren kann. Im Gegensatz zu Methoden, die sich auf einen voll ausgebildeten Trainingskorpus stützen, brauchen wir keine explizite Annotation subjektiver Ausdrücke. Um ein soches Modell zu trainieren, müssen wir nur die Erwähnungen der Entitäten und ihre Beziehungen annotieren, welche entweder Sentiments, Vergleiche o.Ä. sind. Da die subjektiven Ausdrücke und ihre Assoziationen mit Beziehungen nicht in den Trainingsdaten enthalten sind, kann dieses Modell die Ausdrücke erkennen und sie mit ihren entsprechenden Beziehungen assoziieren, sowohl im Training als auch in den Tests. Des Weiteren wenden wir im Hinblick auf die Kompositionssemantik subjektiver Ausdrücke Techniken des Deep Learning zur Erstellung latenter Merkmalsvektoren an, um Zusammensetzungsmuster codieren zu können, welche das Training stark erleichtern. Die unterschiedliche Granularität der Analyse führt zu unterschiedlicher Modellkomplexität; je feiner, desto komplexer. Alle vorgeschlagenen Modelle zielen darauf ab, möglichst wenige manuell annotierte Daten und möglichst viele frei verfügbare Ressourcen zu verwenden. Diese Modelle untersuchen auch verschiedene Merkmalsdarstellungen, um die Kompositionssemantik abzubilden, die subjektiven Ausdrücken inhärent ist. Die Dissertation schließt mit einer Umfrage über die Sentimentanalyse und halb überwachtes Lernen.

Acknowledgement

I would like to express my sincere gratitude to my advisor, Prof. Dr.-Ing. Gerhard Weikum. This work would not have been possible without his continuous support. I would also like to thank him for his scientific guidance, insigntful comments and giving me a lot of freedom in the choice of my research. I'm also deeply indebted to my co-advisor Dr. Rainer Gemulla for fruitful discussions and his invaluable help to this thesis. I would also like to offer my special thanks to Dr. Georgiana Ifrim with whom I collaborated on parts of this work. Furthermore, I thank all my friends and my colleagues in MPII for the memorable moments we enjoyed together. Last but not least, I owe my deepest gratitude to my parents, Bencheng Qu and Rulan He, for their endless love and unconditional support.

Contents

Chapter 1

Introduction

1.1 Movitation

An important kind of information conveyed in many online and traditional media is *sentiments or* opinions, which refer to positive and negative emotions, evaluations and stances (Wilson, 2008). The sentiments of other people often influence our decision-making process. Many of us asked our friends to recommend a digital camera or to share travel experiences on a tourist destination, requested reference letters regarding job applicants from colleagues, or consulted experts' reviews in magazines to decide which computer to buy. In that sense, the web provides a way to access the sentiments and experiences of a vast number of people and share our own opinions with them. As more and more people make their sentiments and experiences available online, this kind of information also assists our decision-making with growing importance in our daily life Pang and Lee (2007).

Sentiment analysis, also called opinion mining, has become an active rearch area since the year 2000. In this area, the main concern is to identify when a sentiment is being expressed and identifying attributes of the sentiment. Attributes of sentiments include who is expressing the sentiment, about what or whom the sentiment is being expressed, the polarity (positive, negative or neural) and intensity (degree to which a sentiment is positive or negative) of the sentiment (Pang and Lee, 2007). In addition, the identification of relative comparisons among entities has also gained attention (Jindal and Liu, 2006b;

Ganapathibhotla and Liu, 2008). Instead of targeting all these subtasks, this dissertation focuses on building learning systems to analyze the polarity and intensity of sentiments, as well as comparisons between entities expressed in natural language text.

Automatic sentiment analysis of linguistic expressions is challenging due to the inherent ambiguity of natural language. One source of ambiguity is the polysemy of words and potentially complex syntactic structures of sentences, which are common for most NLP tasks. Besides that, sentiment analysis is especially challenging in the following aspects:

Order sensitivity. Bag-of-words representations are often sufficient for topic-based text categorization or clustering, whereas sentiment and subjectivity are quite sensitive to word order and sentence order. For example, at the fine-grained level, "A is worse than B" conveys the opposite sentiment of "B is worse than A"; at the coarse-grained level, a document could start with a positive sentiment like "It sounds like a great plot." but concludes with a strong negative sentiment such as "However, the film can't hold up".

Compositional semantics. As pointed out by Yessenalina and Cardie (2011), it is important to capture the *compositionality of expressions* for fine-grained sentiment analysis. The meaning of a complex expression is a function of the meanings of its constituent expressions and the rules used to combine them. For example, the sentence "Tom Cruise does not inject any personality into the role." contains soly neutral words but these words jointly express a negative sentiment. An expression might use negators and intensifiers to modify a lower-level constituent with a different meaning, such as "not very good". Understanding the compositional patterns would significantly improve the generalization power of learning models.

Limited training data. It is labor-intensive to construct a hand-labeled training corpus for fine-grained sentiment analysis. Since sentiment-bearing expressions are often domain dependent, it is not practical to always build new training data for new domains. However, an intellegent system should be able to explore other freely available resources such as online review ratings and subjectivity lexicons, which provide indirect and noisy supervised information to compensate the sparsity of training data.

1.2 Contributions

To address the challenges outlined above, this dissertation presents three learning models that perform sentiment analysis at different granularity levels.

Bag-of-Opinions model. The first contribution is the bag-of-opinions (BoO) model, which predicts *document-level* polarity and intensity by treating each document as a bag of sentiments. Each sentiment-bearing expression consists of a root word with a known prior polarity (e.g. happy), optionally a set of modifier words (e.g. very) and negation words (e.g. not). Although the composition rules regarding these components imply a non-linear function, the new feature representation of sentiment enables the learning with a linear model. However, the prior polarity of root words are obtained from domain-independent subjectivity lexicons, which cannot capture the domain dependent information. To circumvent the problem, we developed a two-stage approach: i) in the first stage, we learn a bag-of-opinions model on a large dataset of online reviews to obtain scores for domain-independent sentiments; ii) in the second stage, we combine the bag-of-opinions model with an unigram model trained on the domain- dependent corpus. The results of BoO were presented in COLING 2010 (Qu et al., 2010).

Multi-Experts model. The second contribution is the weakly supervised multi-experts model (MEM) for *sentence-level* analysis of sentiment polarity and intensity, where the two quantities are *jointly* represented as numerical ratings. The multi-experts model can exploit any available sentiment indicators, such as phrase-level predictors, language heuristics, co-occurrence counts and word order sensitive similarity measures of sentences. Instead of using them as features, these indicators constitute the set of experts used in the model, which are similar to the ideas of ensemble learning. The results of MEM were presented in EMNLP 2012 (Qu et al., 2012).

SENTI-LSSVM$_{RAE}$ model. The third contribution is the SENTI-LSSVM$_{RAE}$ model, which is the first learning system to identify sentiments and comparisons of entities at both sentence and subsentential level. Unlike methods that rely on a full-fledged training corpus, we do not require any explicit annotation of sentiment-bearing expressions. To train such a model, we only need to annotate

the entity mentions and their relationships, which are either sentiments, comparisons or other cases. Since sentiment-bearing expressions and their association to relationships do not exist in the training data, this model can detect the expressions and associate them to respective relationships in both training and testing. Moreover, to address the compositional semantics of sentiment-bearing expressions, we apply deep learning techniques to construct latent feature vectors to encode compositional patterns, which make the learning much easier.

1.3 Organization

This dissertation is organized as follows. Chapter 2 establishes essential foundations for sentiment analysis as well as the semi-supervised techniques suitable for this application area. Chapter 3 presents the bag-of-opinions model for document rating prediction. Chapter 4 describes the multi-experts model that performs sentence rating prediction. Chapter 5 addresses the identification of sentiments and comparisons of entities at both sentence and subsentential level. Finally, Chapter 6 concludes this dissertation and points out appealing directions of future research.

Chapter 2

Background

This chapter gives a concise overview of the fundamentals for sentiment analysis as well as the semi-supervised learning methods (SSL) for automatic textual analysis with few training data. We start with summarizing the core tasks and works of sentiment analysis and opinion mining in Section 2.1, followed by introducing the linguistic concepts and observations with regard to sentiment in Section 2.2. Because sentiment analysis with a small amount of training data is the central topic investigated in this dissertation, we review the core techniques of semi-supervised learning as well as their applications in the area of sentiment analysis in Section 2.3.

2.1 Overview of Sentiment Analysis

Sentiment analysis, also called opinion mining, is concerned with the computational treatment of opinion, sentiment, and subjectivity in nature language text. It is the field of study that covers many tasks such as opinion extraction, opinion summarization, sentiment lexicon generation, and opinion spam detection etc.. Since the year 2000, the field has become a very active research area because it has a wide range of applications in a lot of domains like marketing and politics, and many research benefit directly from the huge volume of opinionated data in the social media on the Web (Pang and Lee, 2007; Liu, 2010).

The two core tasks of sentiment analysis are *polarity classification* and *rating*

prediction. The former one aims to classify whether a piece of text expresses a positive or negative sentiment. And it often introduces the third category *neutral* for factual expressions. The latter task considers additionally the intensity of sentiment (degree to which an expression is positive or negative). Both quantities can be analyzed jointly by mapping them to numerical ratings so that large positive/negative ratings indicate strong positive/negative sentiments. The two tasks are mainly investigated at three levels: document level, sentence level and expression level. At the document level, researchers often apply supervised models (Pang and Lee, 2005a; Pang et al., 2002a; Dave et al., 2003; Melville et al., 2009) or rule-based predictors (Turney, 2002; Taboada et al., 2011) for both tasks. In recent years, the idea of incorporating topic models with supervised models has also attracted much attention (Titov and McDonald, 2008a,b; Lin and He, 2009; Lin et al., 2012). At the sentence and expression levels, the rule-based predictors mainly rely on a domain independent subjectivity lexicon (Ding et al., 2008a; Taboada et al., 2011; Ding et al., 2009), while the supervised models often assumes the existence of sufficient training data, which are hard to obtain (Jin et al., 2009; Choi and Cardie, 2010; Yessenalina and Cardie, 2011; Wei and Gulla, 2010).

Another widely investigated task is *opinion extraction*, which aims to identify opinionated expressions, opinion targets and opinion holders from text. For a sentiment, the opinion targets are the topics or entities that the sentiment refers to, opinion holders are the sources or experiencers of the sentiment. For example, in "John likes the Canon 7D.", "John" is the opinion holder holding a positive sentiment toward the target "Canon 7D". Although most research work focus either on expression-level polarity classification, opinion target or holder extraction (Breck et al., 2007; Yessenalina and Cardie, 2011; Wilson et al., 2005), Choi et al. (2006) show that joint extraction of the three attributes can lead to significant performance improvement.

Apart from directly expressing positive or negative sentiments about an entity and its aspects, one can also express sentiments by comparing entities. Such sentiments are referred to as *comparative opinion* in (Jindal and Liu, 2006b). One example sentence carrying such opinions is "Canon 7D has a better image quality than Nikon D7000.". To understand which entities are preferred, current research work decomposes it into three subproblems: the detection of

sentences carrying comparative opinions (Jindal and Liu, 2006a), the extraction of entities participating in comparative relationships (Jindal and Liu, 2006b) and the identification of preferred entities given a comparative relationship (Ganapathibhotla and Liu, 2008). And these proposed techniques rely heavily on linguistic heuristics and variants of frequent itemset mining.

Opinion summarization is a way to aggregate and represent sentiment information drawn from a collection of documents. There are two main approaches to this task - extraction and abstraction (Carenini et al., 2006; Pang and Lee, 2007). Extraction involves concatenating the most representative extracts (e.g. sentences) from the corpus into a summary, whereas abstraction involves generating novel sentences from sentiment information extracted from the corpus. The former approach tends to drop the redundant information by leveraging existing topic-based multi-document summarization algorithms, while the latter one is concerned with the redundancy of information by showing e.g. how many web users express the same sentiment toward the same product. Thus the abstractive summaries can be regarded as the reports of the results of opinion extraction.

Most sentiment analysis methods employ sentiment lexions, which are lists of words and phrases labeled with either their prior polarity (positive, negative or neutral) or numerical ratings. There are three main approaches to build such lexicons: manual approach, dictionary-based approach, and corpus-based approach. The MPQA lexicon (Wilson et al., 2005) and SO-CAL lexicon (Taboada et al., 2011) are the examples of the manual approach, which requires manual assignment of polarities or numerical ratings to every word and phrase in the lexicons. The dictionary-based approach starts with a set of seed words and expands their labels to the remaining words and phrases in a dictionary (e.g. WordNet (Miller, 1995a)) based on the synonym and antonym structures of the dictionary (Baccianella et al., 2010; Hu and Liu, 2004a). Compared to the manually created ones, they often have higher coverage but lower precision (**?**). The corpus-based approach targets at building a domain dependent lexicon utilizing a domain corpus. Such a lexicon is built either from a seed list of known sentiment-bearing words (Kanayama and Nasukawa, 2006a; Hatzivassiloglou and McKeown, 1997) or by adapting a general-purpose sentiment lexicon (Choi and Cardie, 2009a).

Another concern of sentiment analysis is to assess the quality of sentiment-bearing documents such as reviews and blogs. This problem is either studied as a regression problem by predicting a quality score for a document (Ghose and Ipeirotis, 2007; Kim et al., 2006; Lu et al., 2010b) or as a classification problem to judge if a document is helpful for web users or not (Liu et al., 2007). Another related problem is opinion spam detection, which targets at the detection of fake opinions or reviews (Jindal and Liu, 2008, 2007).

2.2 Linguistic Perspective of Sentiment Analysis

In (Wilson, 2008), sentiment is defined as an attitude type of **private state**, which refers to the mental and emotional state of a writer or speaker or some other persons. The original definition of private states is attributed to (Quirk et al., 1985) (p. 1181): "A person may be observed to assert that God exists, but not to believe that God exists. Belief is in this sense 'private'.".

For the analysis of private states and their attributions, Wiebe (1994) developed a frame-style conceptual representation, which is adapted and expanded in (Wiebe, 2002; Wiebe et al., 2005; Wilson, 2008). In Section 2.2.1, we give a brief introduction of the representation framework, followed by discussing some interesting linguistic observations related to sentiment analysis in Section 2.2.2.

2.2.1 Conceptual Representation

In a basic representation (Wiebe et al., 2005), a private state is a state of an *experiencer*, holding an *attitude*, optionally toward a *target*. An agent frame is used to mark expressions that refer to sources or experiencers of private states; an attitude frame is used for the expressions of attitude and optionally a target frame represents the target or topic of the private state.

Example 2.1 *Analysts have complained that third-quarter corporate earnings haven't been very good, but the effect hit home particularly hard yesterday.*

In Example 2.1, the experiencer, marked by the agent frame "analysts", holds a negative sentiment about the target "third-quarter corporate earnings". The sentiment is expressed by the attitude frame "have complained that third-quarter corporate earnings haven't been very good".

At the coarse level (Wilson, 2008), there are six attitude types: sentiment, agreement, arguing, intention, speculation, and all other attitudes. Sentiment, agreement, arguing, intention can be further categorized into positive and negative variants. Since analyzing all attribute types is beyond the scope of this dissertation, we focus on sentiment.

Sentiments are positive and negative emotions, evaluations and stances. Sentiment analysis is the task of identifying when a sentiment is being expressed and which attributes the sentiment contains. Attributes of sentiment include who is expressing the sentiment, about whom or what the sentiment is being expressed, the polarity and intensity of the sentiment etc.

One of the major concerns of sentiment analysis is to determine its polarity, which can be either positive or negative. We usually consider an additional category *other* or *neural* to indicate whether a textual utterance conveys neither positive nor negative sentiment whenever polarity classification is the goal of an application. An expression can have a **prior polarity**, which refers to whether an expression evokes something positive or something negative when taken out of context. In contrast, the **contextual polarity** of an expression is the polarity of the expression considering the context in which it appears. The context can range from its context words to the sentence or the discourse.

Another important attribute of sentiment is **intensity**, which refers to the strength of the sentiment that is being expressed. As language users, we express sentiment in different intensity levels with diverse utterances. For example, "delighted" is more intensively positive than "pleased".

In addition to the frames characterizing the functional components of private states, Wiebe et al. (2005) also defined frames to categorize the various ways of expressing private states in language.

Private states may be explicitly mentioned. An example is "fear" in the following sentence.

Example 2.2 *Experts fear missile launch this weekend to mark founding father's birthday.*

They may also be expressed in speaking or writing events, as with the word "criticizes" in Example 2.3.

Example 2.3 *Malan criticizes the stance taken by multilateral institutions like the IMF.*

Wiebe et al. (2005) use the term **speech event** to refer to any speaking and writing event. Such an event has either a speaker or writer as well as a target, which is about whatever is said or written.

Private states may also be conveyed through **private state actions** such as applauding, laughing and frowning. Another example is "sighed" in the following sentence.

Example 2.4 *He sighed over the lost opportunity.*

If private states such as anger, frustration, positive sentiment etc. are expressed without explicitly stating that they are angry, frustrated, happy etc., these expressions are referred to as **expressive subjective elements** (Wiebe et al., 2005). One example of expressive subjective elements is the phrase "a pillar of support" in Example 2.5.

Example 2.5 *People are questioning corporate profits as a pillar of support for the equity market.*

In (Wiebe et al., 2005), the above different ways of expressing private states are represented using **direct subjective frames** and **expressive subjective element frames**. Direct subjective frames cover explicit mentions of private states, speech events stating private states and private state actions, while expressive subjective element frames represent expressive subjective elements.

2.2.2 Observations of Sentiment-bearing Expressions

Initially, one might think that there is a relatively small set of linguistic expressions to describe sentiment. In fact, after investigating a manually annotated corpus, Wiebe et al. (2005) found that there is a large variety of

words and diverse part-of-speech which appear in subjective expressions, among which the majority are used to express sentiment. Moreover, the number of expressive subjective elements is almost the same as the number of distinct words used to express them.

A sentiment is often conveyed by a complex expression with different granularities, ranging from one word to a whole sentence. In annotation studies, Wiebe et al. (2005); Wilson (2008) found that many words are ambiguous due to the fact that they appear in both subjective and objective expressions. Likewise, most sentences are mixtures of objectivity and subjectivity, and often contain subjective expressions of varying intensities. They were even surprised by the fact that human annotators are not sure about the subjectivity of a significant number of the direct subjective frames.

The contextual polarity of an expression deviates often from its prior polarity (Wilson, 2008). Negation is one of the most important influencers, which may be local (e.g. *not bad*) or involve long distance dependencies (e.g. *I don't think it works.*). Negation is not simple also because certain phrases contain negation words but do not change polarity (e.g. *Not only his friends but also his parents like his new song.*). Moreover, contextual polarity may be influenced by irrealis, which is the set of grammatical modalities that indicate a certain action or situation might not be reliable or does not happen. An example is the use of modal verbs in "I thought the new phone *would* be at least as good as its predecessors, but unfortunately, it wasn't.". In addition, contextual polarity often depends on domain and topic. For example, *long* conveys opposite polarities in "The battery life is *long*." and "It needs *long* charging time.".

Intensity is a more fine-grained attribute of sentiment than polarity and the differences of meanings conveyed by the corresponding expressions are often subtle. As the inter-annotator agreement study (Wilson, 2008) shows, it is more difficult for humans to achieve agreement on intensity than on polarity. A severe type of disagreement between annotators is the difference in intensity ordering for expressive subjective elements, which are the main way of expressing sentiment. This can be explained by more types of influencers for expressing intensity. In addition to the ones used for polarity, there are two

types of modifiers (Taboada et al., 2011): Amplifiers (e.g. very) increase the semantic intensity of a neighboring lexical item, while downtoners (e.g. little) decrease it. And the expressions for intensity are also diverse since high-intensity expression often contain words that are very infrequent such as "pre-historic" and "tyrannical" (Wilson, 2008).

2.3 Semi-supervised Learning Techniques for Sentiment Analysis

Semi-supervised learning (SSL) is halfway between supervised and unsupervised learning. In addition to unlabeled data, an SSL algorithm is provided with some supervision information in form of labels for a subset of available data. Formally, given a labeled data set $D_l = \{(\mathbf{x}_i, y_i)|(\mathbf{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y}, i = 1, ..., l\}$ and an unlabeled data set $D_u = \{\mathbf{x}_j|\mathbf{x}_j \in \mathcal{X}, j = l+1, ..., l+u\}$, where $\mathcal{X}$ denotes the input space of data points and $\mathcal{Y}$ is a label space, an SSL algorithm aims to learn a function $f : \mathcal{X} \to \mathcal{Y}$ in some function family $\mathcal{F}$. In general, it is assumed that the input-output pairs $(\mathbf{x}_i, y_i)$ are drawn *independently* and *identically distributed* (i.i.d.) from a distribution P on $\mathcal{X} \times \mathcal{Y}$ and the unlabeled examples are drawn i.i.d. from the marginal distribution $P_{\mathcal{X}}$ of P.

There are two slightly different settings for SSL algorithms, namely inductive and transductive semi-supervised learning. In the former setting, a learned function is expected to predict the labels of a separate test sample, which is unseen during training, whereas in the latter setting, a learned function makes predictions only on the unlabeled data provided in the training phase. Likewise, we say that an algorithm performs inductive inference when it is used to infer labels of unseen data, while it performs transductive inference when using both labeled and unlabeled data.

In Section 2.3.1 and Section 2.3.2, we give an overview of graph-based semi-supervised learning methods and deep autoencoders, which provide technical foundation for Chapter 4 and Chapter 5, respectively. Further SSL techniques as well as their applications in the field of sentiment analysis are briefly introduced

in Section 2.3.3.

2.3.1 Graph-Based Semi-supervised Learning Methods

Graph-based semi-supervised learning methods (GSSL) propagate the labels of labeled data points to unlabeled ones based on the similarity of the data. This is represented by a similarity graph $G = \langle V, E \rangle$, where each vertex v_i from the vertex set V represents a data point $\mathbf{x}_i \in \mathcal{X}$ and each edge (v_i, v_j) from the edge set E is associated with a non-negative weight w_{ij} indicating the similarity between v_i and v_j. Moreover, $V = V_l \bigcup V_u$, where each vertex in V_l has an initial label $y \in \mathcal{Y}$ and all vertices in V_u are unlabeled.

Given a graph $G = \langle V, E \rangle$, the GSSL methods postulates that two vertices v_i, v_j should have the same label y with high probability if there is a path along edges with high similarity values between them in G. From a probabilistic point of view, they make the following smoothness assumption (Chapelle et al., 2006):

***Smoothness assumption**: If two data points $\mathbf{x}_i$ and $\mathbf{x}_j$ in a high density region are close, the corresponding function outputs y_i and y_j are likely to be the same.*

Based on these intuitions, Belkin et al. (2006a) introduced a general framework for a range of GSSL algorithms, which differ in empirical loss functions and regularizers. For a Mercer kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, there is an associated Reproducing Kernel Hibert Space (RKHS) H_K of functions $\mathcal{X} \to \mathbb{R}$ with the corresponding *norm* $\| \cdot \|_K$. In general, the kernel function is the same one used to compute the similarity values w_{ij} in the similarity graph. For a labeled dataset D_l and an unlabeled dataset D_u, the framework estimates a function by minimizing

$$f^* = \arg\min_{f \in H_K} \lambda_l \sum_{i=1}^{l} C(f(\mathbf{x}_i), y_i) + \lambda_K \|f\|_K^2 + \mathbf{f}^\top \mathbf{L} \mathbf{f} \tag{2.1}$$

where $C(f(\mathbf{x}_i), y_i)$ is a loss function such as squared loss $(y_i - f(\mathbf{x}_i))^2$ for regression or the hinge loss function $\max(0, 1 - y_i f(\mathbf{x}_i))$ for classification; the loss function is adjusted by the hyperparameter λ_l, which indicates the noise level of initial labels; the hyperparameter λ_K controls the complexity of the function by penalizing the RKHS norm $\|f\|_K^2$. The capability of label

propagation is provided by the graph regularizer $\mathbf{f}^\top \mathbf{L}\mathbf{f}$, where $\mathbf{f} = [f(\mathbf{x}_1), ..., f(\mathbf{x}_{l+u})]^\top$ and $\mathbf{L}$ is the graph Laplacian based on the similarity graph G that characterizes the marginal distribution $P_{\mathcal{X}}$. Moreover, the Representer Theorem (Belkin et al., 2006a) states that the solution to this optimization problem exists in H_K and takes the following form

$$f^*(\mathbf{x}) = \sum_{i=1}^{l+u} \alpha_i K(\mathbf{x}, \mathbf{x}_i) \tag{2.2}$$

which involves both labeled and unlabeled data points. Therefore, the problem is reduced to optimizing over the coefficients α_i. As we can see, the solution shares the similar form as support vector machines (SVM) and Ridge regression (Bishop et al., 2006) except the expansion to the unlabeled data. Thus, GSSL algorithms belong to the family of kernel machines.

Graph Laplacians

Two variants of graph Laplacian are widely used for semi-supervised learning.

- An unnormalized graph Laplacian is defined as $\mathbf{L}_u = \mathbf{D} - \mathbf{W}$, where the diagonal matrix $\mathbf{D}$ is given by $d_{ii} = \sum_{j=1}^{l+u} w_{ij}$. Then we have $\mathbf{f}^\top \mathbf{L}_u \mathbf{f} = \frac{1}{2} \sum_{i,j=1}^{l+u} w_{ij} (f(\mathbf{x}_i) - f(\mathbf{x}_j))^2$. As we can see, minimizing $\mathbf{f}^\top \mathbf{L}\mathbf{f}$ is the same as reducing discrepancies of function values between two data points proportionally to their similarity defined in the graph G.
- A normalized graph Laplacian takes the form $\mathbf{L}_n = \mathbf{D}^{-1/2} \mathbf{L}_u \mathbf{D}^{-1/2}$. For every f, we have $\mathbf{f}^\top \mathbf{L}_n \mathbf{f} = \frac{1}{2} \sum_{i,j=1}^{l+u} w_{ij} (\frac{f(\mathbf{x}_i)}{\sqrt{d_{ii}}} - \frac{f(\mathbf{x}_j)}{\sqrt{d_{jj}}})^2$. Compared to unnormalized graph Laplacian, the difference of two function values are further influenced by the sum of edge weights adjacent to the corresponding data points.

Besides enforcing the smoothness constraints on similar data points, the two graph Laplacian can also be augmented to incorporate dissimilarities between data points. Interested readers can find details in (Goldberg et al., 2007).

The differences between the two graph Laplacians are mainly studied in the context of clustering (von Luxburg, 2007). Since GSSL algorithms can be regarded as the methods of assigning labels to the data points in the existing

clusters, we can easily extend their findings to understand semi-supervised learning. More specifically, von Luxburg (2007) found that the two graph Laplacians lead to different graph partitioning strategies from a graph cut point of view. The graph cut problem can be stated as follows: given a similarity graph $G = \langle V, E \rangle$, we want to partition a graph such that the edges between different vertex groups have low weights while the edges within a vertex group have high weights. In practice we also expect that different vertex groups are reasonably large, which gives rise to the two commonly used objective functions ratio cut (RatioCut) (Hagen and Kahng, 1992) and normalized cut (Ncut) (Shi and Malik, 2000). Let the edge weights between two vertex groups V_a and V_b be denoted as

$$\text{cut}(V_a, V_b) = \sum_{v_i \in V_a, v_j \in V_b} w_{ij}$$

and $\overline{V_a}$ denote the complement of V_a, the two objective functions for partitioning a graph G into k vertex groups $V_1, ..., V_k$ are defined as:

$$\begin{aligned} \text{RatioCut}(V_1, ..., V_k) &= \frac{1}{2} \sum_{i=1}^{k} \frac{\text{cut}(V_i, \overline{V_i})}{|V_i|} \\ \text{Ncut}(V_1, ..., V_k) &= \frac{1}{2} \sum_{i=1}^{k} \frac{\text{cut}(V_i, \overline{V_i})}{\text{vol}(V_i)} \end{aligned} \tag{2.3}$$

where $|V_i|$ denote the number of vertices in V_i and $\text{vol}(V_i)$ denote the sum of weights of all edges attached to vertices in V_i. From the definitions we can see that, both RatioCut and Ncut aim to find a partition such that data points in different clusters are dissimilar to each other. However, both algorithms behave differently while maximizing within-cluster similarity. Ncut tends to maximize the sum of edge weights within clusters, while RatioCut prefers clusters with large numbers of vertices.

For partitioning a graph into two vertex groups, minimizing the two objective functions can be approximated by minimizing a graph regularizer $\mathbf{f}^\top \mathbf{L}\mathbf{f}$ subject to a set of constraints (von Luxburg, 2007). If $\mathbf{L}$ is an unnnormalized graph Laplacian, the optimization result is an approximation of ratio cut, whereas a normalized graph Laplacian leads to a normalized cut. This can also be

generalized for separating a graph into k groups. Therefore, the choice of graph Laplacian depends on which graph cut strategy is better suited for a specific application.

Solving the Optimization Problem

Due to the Representer Theorem (Belkin et al., 2006a), we can solve the optimization problem (2.1) in the dual space, which is a widely used approach for SVM. To illustrate the idea, we choose squared loss and normalized graph Laplacian for the problem (2.1), which becomes

$$f^* = \underset{\mathbf{f}\in H_K}{\arg\min}\, \lambda_l(\mathbf{Y}-\mathbf{J}\mathbf{f})^\top(\mathbf{Y}-\mathbf{J}\mathbf{f}) + \lambda_K\|\mathbf{f}\|_2^2 + \mathbf{f}^\top\mathbf{L}_n\mathbf{f} \tag{2.4}$$

where $\mathbf{Y} = [y_1, ..., y_l, 0, ..., 0]$ is a $(l+u)$ dimensional label vector and $\mathbf{J}$ is a $(l+u)\times(l+u)$ diagonal matrix with the first l diagnonal entries as 1 and the rest as 0. After replacing $\mathbf{f}$ with the solution form (2.2), we have

$$\boldsymbol{\alpha}^* = \underset{\boldsymbol{\alpha}\in\mathbb{R}^{l+u}}{\arg\min}\, \lambda_l(\mathbf{Y}-\mathbf{J}\mathbf{K}\boldsymbol{\alpha})^\top(\mathbf{Y}-\mathbf{J}\mathbf{K}\boldsymbol{\alpha}) + \lambda_K\boldsymbol{\alpha}^\top\mathbf{K}\boldsymbol{\alpha} + \boldsymbol{\alpha}^\top\mathbf{K}\mathbf{L}_n\mathbf{K}\boldsymbol{\alpha}$$

where $\mathbf{K} \in \mathbb{R}^{(l+u)\times(l+u)}$ is the positive semi-definite similarity matrix (Gram matrix) over labeled and unlabeled data points. Since the derived problem is a quadratic program, we can use an efficient convex optimization algorithm such as coordinate descent (Luo and Tseng, 1992) or L-BFGS (Liu and Nocedal, 1989) to obtain the minimizer $\boldsymbol{\alpha}^*$. Then $\boldsymbol{\alpha}^*$ can be substituted back into the function (2.2) to estimate the labels of unlabeled data points.

The alternative approach is to directly optimize over function values at each data point. More specifically, we consider $\mathbf{f}$ as a vector of function values, where f_i indicates the value of the data point $\mathbf{x}_i$. Then the problem (2.4) turns out to be the equivalent quadratic program in the Euclidean space:

$$f^* = \underset{\mathbf{f}\in\mathbb{R}^{l+u}}{\arg\min}\, \lambda_l(\mathbf{Y}-\mathbf{J}\mathbf{f})^\top(\mathbf{Y}-\mathbf{J}\mathbf{f}) + \lambda_K\|\mathbf{f}\|_2^2 + \mathbf{f}^\top\mathbf{L}_n\mathbf{f}$$

which is exactly the optimization problem for the local global consistency algorithm (LGC) (Zhou et al., 2004) so that we can apply either the LGC algorithm or a convex optimization algorithm to solve the problem. If the selected loss function is not differentiable such as hinge loss, we can employ the subgradient methods (Duchi and Singer, 2009b; Duchi et al., 2011).

Similarity Graph Construction

In the previous discussion, we have assumed that a similarity graph is given. Since a similarity graph represents the distribution $P_{\mathcal{X}}$, the construction of similarity graphs can significantly influence the classification results (Jebara et al., 2009). For a similarity measure $\text{sim}(\mathbf{x}_i, \mathbf{x}_j)$ with $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$, there are three popular ways to build a similarity graph (Maier et al., 2009):

- **ε-neighborhood graph:** $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected if $\text{sim}(\mathbf{x}_i, \mathbf{x}_j) \geq \varepsilon$.
- **symmetric k-nearest neighbor graph:** $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected if $\mathbf{x}_i \in \text{kNN}(\mathbf{x}_j)$ or $\mathbf{x}_j \in \text{kNN}(\mathbf{x}_i)$.
- **mutual k-nearest neighbor graph:** $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected if both $\mathbf{x}_i \in \text{kNN}(\mathbf{x}_j)$ and $\mathbf{x}_j \in \text{kNN}(\mathbf{x}_i)$.

where $\text{kNN}(\mathbf{x}_i)$ denotes the set of k nearest neighors of x_i.

From the graph definitions we can see that both ε-neighborhood graphs and mutual k-nearest neighbor graphs may contain disconnected graphs or even isolated singleton vertex, while symmetric k-nearest neighbor graph is always connected. In a symmetric k-nearest neighbor graph, data points in a low density region can connect to points in a high density region, while in a mutual k-nearest neighbor graph, data points tend to be connected within regions of constant density (von Luxburg, 2007). Moreover, theoretical analysis in (Maier et al., 2008) shows that normalized cut on a ε-neighborhood graph will produce systematically different clustering results than normalized cut on a symmetric k-nearest neighbor graph.

The same GSSL algorithms tend to perform better on a graph with similar vertex degree. Ozaki et al. (2011) show that the same GSSL algorithms perform significantly better on a mutual k-nearest neighbor graph than on a symmetric k-nearest neighbor graph for both word sense disambiguation and document classification. Jebara et al. (2009) also found that when b-matching method is used to enforce all vertices in a symmetric k-nearest neighbor graph to have the same number of edges, the same GSSL algorithms always gain a performance improvement over the one trained on a graph with diverse degree.

Limitations

Although GSSL algorithms are expected to learn from few labeled data, Bengio et al. (2005) show that, when the similarity function is local (see below), the number of required training points could grow exponentially with the dimensionality of the data for certain target functions. This problem is referred to as *curse of dimensionality* in (Bengio et al., 2005) and they show that it is a shared problem of all kernel machines with local kernel functions. We say that a kernel function $K(\cdot, \cdot)$ is **local**, if for all $\mathbf{x} \in X$, there is a neighborhood $N(\mathbf{x}) \subset \mathcal{X}$ such that

$$f(\mathbf{x}) \simeq \sum_{\mathbf{x}_i \in N(\mathbf{x})} \alpha_i K(\mathbf{x}, \mathbf{x}_i) \tag{2.5}$$

This means that $f(\mathbf{x})$ is mostly determined by the neighbors of $\mathbf{x}$. In that sense, a number of commonly used kernel functions such as Gaussian kernels and linear kernels are local. Bengio et al. (2005) further pointed out that for local kernel machines, curse of dimensionality is hard to avoid due to the bias-variance dilemma of statistical estimators, where **bias** of an estimator is the expected difference between the estimator and the target variable, and **variance** of an estimator is the expected squared difference between the estimator and its expected value. The dilemma states that the way to reduce variance is to decrease the size of neighborhood but this also increases bias. In contrast, increasing the size of neighborhood can reduce bias but increase variance. However, we expect to learn a function with both low bias and low variance.constant label

Another central argument they made is that local kernel machines require a large number of data points to learn highly varying target functions, whose outputs vary a lot across input space. One data point of such a complex function can be illustrated by Figure 2.1. We can see that near the decision surface, many data points with opposite labels are quite close to each other so that subtle changes of inputs will lead to completely differrent outputs. If we want a GSSL algorithm to learn such a function, a corresponding similarity graph would have many regions with identical label, where a *region with identical label* is defined as a connected maximal subgraph in which all vertices have the same label and no other vertex can be added while keeping these properties. Bengio et al. (2005)

claim that in each region with identical label there should be at least one correct labeled point, as indicated by the following proposition.

Proposition 2.1 *After running a GSSL algorithm to solve the optimization problem* (2.1)*, the number of regions with identical label is less than (or equal to) the number of labeled data points.*

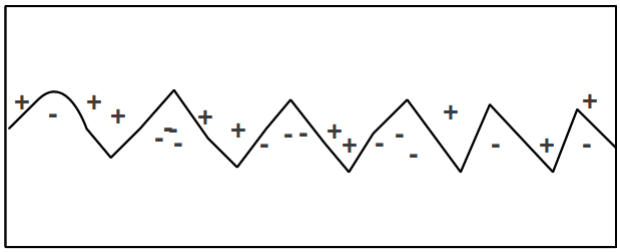


Figure 2.1: Two-dimensional manifold to demonstrate the curse of dimensionality problem. The curve in the middle is the decision boundary for the data points labeled by either + or -.

Due to the curse of dimensionality, the number of required labeled data could grow exponentially with the dimension of the input space (or manifold, if the data lie on an embedded low-dimensional manifold within the high-dimensional input space.). It means, if a labeling function varies highly along each dimension, we cannot achieve low error rates by using a small amount of labeled data.

Applications to Sentiment Analysis

In the area of opinion mining, almost all prior work using GSSL alogrithms focus on coarse-grained sentiment analysis. The tasks range from document polarity classification (Sandler et al., 2008; Sindhwani and Melville, 2008; Ren et al., 2011), document rating prediction (Goldberg and Zhu, 2006; Zhu and Goldberg, 2007) to the identification of authors' political affiliation (Lu et al., 2010a). Only Rao and Yarowsky (2009) applied GSSL algorithms to learn a sentiment polarity lexicon from a few seed words.

A suitable similarity measure for a specific task is the key to successful application of GSSL algorithms, because it determines the distance between

two data points which implies the similarity of their labels. In other words, GSSL algorithms work only if a proper similarity measure can be found to make their assumption hold. For document-level sentiment analysis, such a measure is non-trivial because the commonly used cosine similarity based on bag-of-words representation intends to favor the topic similarity rather than the sentiment similarity. Therefore, a high similarity value based on this measure often indicates that two documents share a lot of content words (e.g. "lens", "car") rather than similar sentiments. As shown in (Goldberg and Zhu, 2006), using the cosine similarity with bag-of-words representation, the GSSL algorithms performed even significantly worse than the ϵ-insensitive support vector regression (Joachims, 1999b) for rating prediction on movie reviews. If they change the measure to the PSP-based similarity (Pang and Lee, 2005a), which is determined by the percentage of positive sentences in a document, the GSSL algorithms outperform their supervised counterparts when the number of labeled documents is small.

Instead of defining a proper similarity measure for documents and constructing a similarity graph-based on the measure, Sandler et al. (2008); Sindhwani and Melville (2008) pursue different ways of contructing similarity graphs. Sandler et al. (2008) encode prior knowledge into a graph of word features $\mathbf{x}$, in which the vertices represent words and the edges represent similarities and dissimilarities between them. Graph regularization is applied to ensure that similar/dissimilar words should have similar/dissimilar weights $\boldsymbol{\beta}$, which constitute the function $f(\mathbf{x}) = \boldsymbol{\beta}^\top \mathbf{x}$ to predict document polarities. Compared to directly learning a linear function to predict document polarity, the advantage of their approach is to incorporate fine-grained lexical knowledge for function weights regularization. Sindhwani and Melville (2008) also assume a linear function for document polarity prediction but for regularization they construct a bipartite graph with two sets of vertices for documents and words respectively. In such a bipartite graph, an undirected edge exists between a document and a word if the document contains the word. Since the polarity of a sentiment-bearing word can be looked up in a subjectivity lexicon and the polarities of a subset of documents are known, the optimal labels of documents can be estimated by "spreading" the initial labels of both documents and words along the edges of the corresponding bipartite

graph. In this way, this approach can incorporate knowledge from both unlabeled documents and subjectivity lexicons.

While most prior work choose one of the previously mentioned graph Laplacian for regularization, Sandler et al. (2008) proposed a feature network penalty, which performs better on word feature similarity graph than normalized graph Laplacian. Suppose the word similarity is encoded into a matrix $\mathbf{W}$ and the out-degree of each vertex must sum to one, $\sum_j w_{ij} = 1$, for a word weight vector $\boldsymbol{\beta}$, the network penalty for d distinct words is defined as

$$\alpha \sum_{j=1}^{d} (\beta_j - \sum_k w_{ij}\beta_k)^2 \tag{2.6}$$

where the hyperparameter α ensures that each word weight receive a similar penalty. To see that the penalty fits into the GSSL framework, we can rewritte it as $\alpha\boldsymbol{\beta}^\top\mathbf{M}\boldsymbol{\beta}$ where $\mathbf{M} = (\mathbf{I} - \mathbf{W})^\top(\mathbf{I} - \mathbf{W})$. For a similarity graph that contains vertices with diverse degrees, the normalized and unnormalized graph Laplacians tend to focus most of the regularization cost on vertices with a large number of neighbors, while the network penalty distributes regularization cost equally for each vertex since the edge weights are normalized inside the squared loss. This property is desirable for word similarity graph since we do not expect a word receives low weight because of its large degree.

Although GSSL algorithms are successfully applied to coarse-grained sentiment analysis, expression-level analysis is extremely challenging for these algorithms because the outputs of the target functions vary highly due to tiny changes of inputs. For example, the polarity of "good image" is positive; "It is difficult to get a good image." turns to negative, but "It is not difficult to get a good image." is again positive. If the intensity of sentiment is taken into account, different types of influencers such as amplifiers and irrealis can substantially increase the complexity of target functions. Moreover, the dimensionality of data is also high due to the diversity of subjective expressions (Wiebe et al., 2005). Therefore, any GSSL algorithms with a local similarity measure such as cosine similarity and string kernels (Lodhi et al., 2000) cannot avoid the *curse of dimensionality*. To circumvent the problem, one way is to combine GSSL algorithms with multiple weak learners that perform differently in different regions with identical label so that joint estimation is more robust than using only one learner. In Chapter

4, we show that this idea works well for sentence-level analysis. For the more difficult expression-level analysis, the key idea discussed in Chapter 5 is to map semantic similar words to similar low-dimensional representations and use an encoder function to construct semantically consistent representations of phrases based on the compositional rules of expressions.

2.3.2 Deep Autoencoders

The purpose of this section is to give an overview of deep autoencoders as well as its applications in the area of sentiment analysis. Since deep autoencoders belong to the family of deep learning techniques, which use autoencoders as basic building blocks to build a deep multi-layer neural network, we start with introducing basic ideas and concepts of deep learning, followed by a brief comparison of two variants of autoencoders, and this section ends with recursive autoencoders (RAE) and its variants for sentiment analysis.

Basic Ideas and Concepts

Many artificial intelligence tasks, such as object recognition or semantic role labeling (SRL), are often decomposed into several sub-problems and people may use different levels of representation for different sub-problems. For example, a state-of-the-art pipeline for object recognition from natural images involves a sequence of modules that transform raw pixel representation into gradually more abstract representations such as edges and local shapes of the image. The identification of target objects could be a result of joint inference over all these information. Since humans often do not know how to design hand-crafted features to specify high level abstractions in terms of raw inputs, it is desirable to let learning algorithms to automatically discover abstractions, from the lowest level features to the highest level concepts.

The focus of deep learning methods is to learn feature hierarchies, in which features from higher levels of the hierarchy are formed by the composition of lower level features. Automatically learning features at multiple levels of abstraction can be regarded as learning complex functions. One example of

such a function is multilayer neural networks, which consists of several layers of non-linear operators for the composition of inputs from lower layers. As illustrated by Figure 2.2, we can represent a function with deep architecture as a graph, in which a vertex corresponds to a computational element. The depth of an architecture is the depth of the corresponding graph, i.e. the longest path from an input vertex to an output vertex. In practice, the computational elements in hidden layers can be regarded as "feature dectecors" for the top layer, which is an application dependent classifier or regression learner. The popular choices of the top layer could be a logistic regression classifier, a linear SVM or a ridge regression learner.

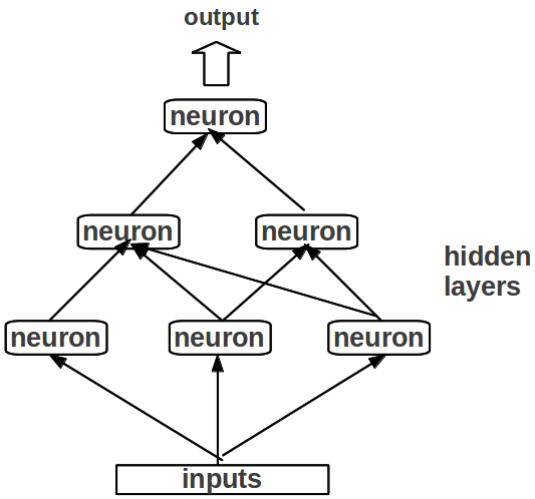


Figure 2.2: A multi-layer neural work of depth 3 as an example of a graph of computation. Each computational element is an artificial neuron implementing a function such as $f(\mathbf{x}) = \tanh(\mathbf{w}^\top \mathbf{x} + b)$ with parameters $(\mathbf{w}, b)$.

Theoretical results reveal that if a function can be compactly represented by a depth k architecture, it might need an exponential number of computational elements to be represented by a depth $k-1$ architecture (Bengio, 2009). Here we say that a function is *compact* when it has few computational elements. For a fixed number of training data points, we expect that compact representations of a target function would yield better generalization.

Although increasing the depth of architecture can significantly improve the expressive power, several commonly used learning algorithms have only

shallow architectures (Bengio et al., 2005). For example, logistic regression and linear regression has depth 1; kernel machines have two levels because the kernel function $K(\mathbf{x}, \mathbf{x}_i)$ for each selected representative training points $\mathbf{x}_i$ serves as the computational element in the first level and the second level performs an affine combination $b + \sum_i \alpha_i K(\mathbf{x}, \mathbf{x}_i)$ of the outputs from the first level. Therefore, these methods cannot represent a target function that requires exponentially many computational elements with a shallow archtecture.

As indicated above, a deep architecture can be viewed as the composition of a sequence of processing stages. **Distributed representation** is used by deep learning methods as the intermedia representation between stages. A distributed representation (Hinton, 1986; Rumelhart et al., 2002) contains a bag of features, which are in general not mutually exclusive. In practice, it often takes the form of a continous-valued vector. Distribution representation allows an abstract concept or an entity to be represented by a subset of correlated features. In contrast, a local representation uses a single feature to represent each distinct concept or entity. Compared to local representation, distributed representation is more compact. For example, a local representation requires a vector of N bits to represent one of N distinct words while a distributed representation for the same word could be a vector of $\log_2 N$ bits, because we can use all possible different combinations of bits to distinguish different words. Note that, distributed representation is a widely used representation form, which is designed not only for neural networks and deep learning but also for Principle Component Analysis (PCA) (Jolliffe, 1986) and matrix factorization (Lee and Seung, 1999) etc..

Deep architecture and distributed representation are in fact old ideas dating back to the last century, but researchers always failed to train a deep multi-layer neural network with more than three layers until 2006. In 2006, Hinton et al. (2006) discovered a pre-training algorithm, which makes the training of deep architectures practical. More specifically, they suggest that the training of a deep architecture with both labeled and unlabeled data takes two steps: i) unsupervised greedy layer-wise pre-training; ii) supervised fine-tuning of network parameters. In the first step, an unsupervised learning algorithm (Hinton et al., 2006) greadily trains one layer at a time, which aims to discover statistical regularities in data and provides a proper initialization of

the model parameters for the next step. In the second step, the initialized model parameters are fine-tuned by backpropagation based on the supervised information (Rumelhart et al., 2002). The two steps can also be carried out interchangeably, namely, we can randomly draw one or a few instances for unsupervised training, followed by another set of instances for supervised training, and repeat the process until certain stop criteria are met. Due to the incorporation of unlabeled data in the training process, we consider the deep learning techniques exploiting the two training steps as semi-supervised.

With deep architecture and distributed representation, deep learning methods can better learn highly varying functions than the algorithms with shallow architectures. Both deep architecture and distributed representation allows a system to represent functions in a compact manner. That means, we need fewer tunable parameters for a complex target function than algorithms with shallow architectures. Furthermore, the process of learning high level features can be seen as disentangling factors of variations (Bengio, 2009), where factors of variations are considered as different aspects of data that can vary separately and often independently (e.g. named entity types or syntactic properties of a phrase). If a task requires an abstraction from low level features, a conventional learner in the top layer should work better with the automatically discovered factors than directly with the low level features.

Autoencoders

Since the emergence of neural networks, various models are proposed to serve as basic building blocks for constructing deep neural networks. Here we restrict our attention to two types of autoencoders: classical autoencoders (Hinton and Zemel, 1994) and denoising autoencoders (Vincent et al., 2008), which serve as part of the model in Chapter 5. Both types of autoencoders include an encoder function and a decoder function in the same parameterized closed form. The main difference is that denoising autoencoder proactively corrupts an input data point before the point is fed into the encoder function.

Given an input vector $\mathbf{x} \in \mathbb{R}^n$, which could be a distributed representation, an encoder function $g_e : \mathbb{R}^n \rightarrow \mathbb{R}^m$ computing an output representation $\mathbf{h}$ takes the

following form

$$g_e(\mathbf{x}) = a(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{2.7}$$

where $a(\mathbf{z})$ is an activation function that usually performs a nonlinear transformation of the vector $\mathbf{z}$, which is computed by a linear transformation of $\mathbf{x}$ with the weight matrix $\mathbf{W} \in \mathbb{R}^{m \times n}$ and the bias vector $\mathbf{b} \in \mathbb{R}^m$. The two main activation functions used in current applications are hyperbolic tangent and the logistic function, which are defined as

$$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{2.8}$$

$$\text{logistic}(z) = \frac{1}{1 + e^{-z}} \tag{2.9}$$

The encoder function is also referred to as an artificial neuron, when it is used for building feedforward artificial neural networks.

A decoder function $g_d : \mathbb{R}^m \to \mathbb{R}^n$ maps an output representation $\mathbf{h}$ back to a reconstruction $\mathbf{r}$ in the input space, which is also parameterized with the weight matrix $\tilde{\mathbf{W}} \in \mathbb{R}^{m \times n}$ and the bias vector $\tilde{\mathbf{b}} \in \mathbb{R}^n$.

$$g_d(\mathbf{h}) = a(\tilde{\mathbf{W}}\mathbf{h} + \tilde{\mathbf{b}}) \tag{2.10}$$

If we constrain the weight matrix of the reverse mapping by $\tilde{\mathbf{W}} = \mathbf{W}^\top$, we say that the autoencoder has *tight* weights.

To measure the discrepancy between the original vector $\mathbf{x}$ and the reconstruction $\mathbf{r}$, the reconstruction error is defined as

$$L_{AE}(\mathbf{x}, \mathbf{r}) = \frac{1}{2}\|\mathbf{x} - \mathbf{r}\|_2^2 \tag{2.11}$$

Since minimizing reconstruction errors requires no labeled data, it is exactly the unsupervised training criterion to perform the greedy layer-wise initialization of model parameters. For a data set $D = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$, the model parameters $\theta = \{\mathbf{W}, \mathbf{b}, \tilde{\mathbf{W}}, \tilde{\mathbf{b}}\}$ are optimized to minimize the average reconstruction error:

$$\theta^* = \arg\min_\theta \frac{1}{n} \sum_{i=1}^{n} L_{AE}(\mathbf{x}_i, g_d(g_e(\mathbf{x}))) \tag{2.12}$$

Then the intermediate representations constructed by the optimized model are expected to retain a certain amount of information about its input, while at the same time try to capture the statistical regularities in the data.

If the activation function of the autoencoder is linear, the autoencoder is essentially equivalent to PCA, which additionally requires that all basis vectors are orthogonal (Kramer, 1991). Then each unit h_i in the output representation $\mathbf{h}$ can be referred to as a factor. From this point of view, an autoencoder with non-linear activation functions performs non-linear factor analysis in the unsupervised learning step.

Vincent et al. (2008) suggest that a good intermediate representation should be robust to partial destruction of the input. That means, good representations are expected to capture stable structures in form of dependencies and regularities of the distribution. They should be recoverable from partial observation and insensitive to outliers. Based on this idea, they proposed the denosing autoencoder. For an input vector $\mathbf{x}$, a corrupted version $\hat{\mathbf{x}}$ is created by means of a stochastic mapping $\hat{\mathbf{x}} \sim p_c(\hat{\mathbf{x}}|\mathbf{x})$. Instead of the original input, the encoder function (2.7) maps the corrupted $\hat{\mathbf{x}}$ to a new representation, which is used as the input for higher level computational components or as the input for computing reconstruction error with the same decoder function (2.10). In practice, the original input is corrupted by randomly choose a fixed number of units in $\mathbf{x}$ and force their values to 0. This procedure can also be viewed as randomly omitting these units of the input by applying the *dropout* operation (Hinton et al., 2012).

Recursive Autoencoders

Recursive autoencoders, which are also referred to as Recursive Auto-Associative Memory (RAAM) (Pollack, 1990), aim to construct a compact representation for data with recursive structures such as constituent parse trees. Given a fixed data structure like the constituent parse tree in Figure 2.3, an RAE repeatedly applies an autoencoder bottom-up to build the representations for each vertex. More precisely, in each step, the encoder function $g_e(\mathbf{x})$ is applied to construct a parent vertex representation $\mathbf{h} \in \mathbb{R}^n$ from the input $\mathbf{x} \in \mathbb{R}^{kn}$, which is the concatenation of the k children representations. And a parent vertex representation can be created if and only if all its children representations are available. During learning, it optimizes also its model parameters by minimizing reconstruction error. Otherwise, if no reconstruction

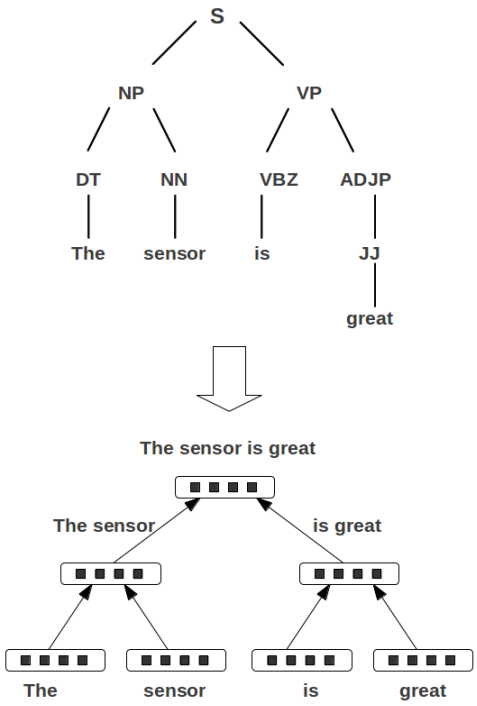


Figure 2.3: A binary representation tree built by a recursive autoencoder, given a simplied constituent parse tree.

error is minimized, the corresponding model is referred to as recursive neural networks (RNN) (Goller and Kuchler, 1996; Socher et al., 2011a).

Applications to Sentiment Analysis

Compared to GSSL algorithms, which are mainly applied to coarse-grained sentiment analysis, deep neural networks have been successfully applied to more fine-grained tasks such as sentence polarity classification (Socher et al., 2011b, 2012).

The central challenge of fine-grained sentiment analysis is to model highly varying functions for flexible sentiment-bearing expressions. As discussed before, the target functions vary a lot, because there are a large number of subjective expressions and subtle changes of the expressions often lead to different sentiments. Since the high volume of expressions are often composed by using a much smaller set of composition rules (e.g. negating an expression conveying positive sentiment leads to an expression carrying negative sentiment.), the intuitive idea is to focus on modeling the composition rules and map each word and phrase to a representation at the proper abstraction level required by the rules.

Based on this idea, Socher et al. (2011b) extend the classical RAE for sentence polarity prediction. They start with mapping each word to a distributed representation pre-trained by using a neural language model (Collobert and Weston, 2008). For a sentence with m words, they build a binary representation tree with an RAE bottom-up in a greedy unsupervised manner. In each step, an autoencoder is used to create a parent candidate by merging the representations of two children vertices. From all the parent candidates, the one with lowest reconstruction error will be selected to represent the corresponding phrase. Then the process is repeated until a tree root is created to represent the whole sentence. Because a phrase with more words tends to carry more information, the method modifies the reconstruction error of the classical autoencoder by using a weighted combination of L2 norms. Let $\mathbf{r}_i$ and $\mathbf{r}_j$ be the reconstructed vectors of children vertices $\mathbf{x}_i$ and $\mathbf{x}_j$ respectively, the

weighted reconstruction error is defined as

$$L_{RAE}(\mathbf{x}, \mathbf{r}) = \frac{n_i}{n_i + n_j}\|\mathbf{x}_i - \mathbf{r}_i\|_2^2 + \frac{n_j}{n_i + n_j}\|\mathbf{x}_j - \mathbf{r}_j\|_2^2 \tag{2.13}$$

where n_i and n_j are the number of words represented by the corresponding children. The coefficients of the weighted L2 norm can be regarded as a way of incorporating prior knowledge. Since the tree roots represent the corresponding sentences, they add a simple *softmax* layer on top of the tree roots while predicting sentence polarities. Let $\mathcal{Y}$ denote the polarity label set, the conditional probability of a label y given a tree root $\mathbf{x}_r$ is computed by

$$\text{softmax}(y, \mathbf{x}_r) = \frac{e^{(\beta_y^\top \mathbf{x}_r)}}{\sum_{y' \in \mathcal{Y}} e^{(\beta_{y'}^\top \mathbf{x}_r)}} \tag{2.14}$$

where $\{\boldsymbol{\beta}_y\}_{y \in \mathcal{Y}}$ are the weight vectors for respective polarity classes. Scheible and Schuetze (2013) further show that the representation trees can be significantly reduced without loss of classification accuracy. However, they apply this *softmax* function to every vertex of a representation tree during training, which indicates that if a sentence is positive/negative, all its constituents are positive/negative. This assumption can certainly not hold for most sentences.

Although the RAE proposed in (Socher et al., 2011b) can capture some compositional patterns of expressions, the experimental results in (Socher et al., 2012) show that it can hardly cope with negation. To address the issue, the matrix-vector recursive neural network (MV-RNN) is proposed, which is an RNN with a special encoder function applied on distributed representations of words learned by unlabeled data. To improve the expressive power of RAE, the method assumes that each constituent has a matrix operator in addition to its distributed representation vector. A matrix operator captures how it modifies the meaning of the other constituent that it combines with. Let two constituents be represented by $(\mathbf{x}_i, \mathbf{O}_i)$ and $(\mathbf{x}_j, \mathbf{O}_j)$, where $\mathbf{O}_i, \mathbf{O}_j \in \mathbb{R}^{n \times n}$ are their matrix operators, the encoder function is re-defined as

$$g_e'(\mathbf{x}_i, \mathbf{O}_i, \mathbf{x}_j, \mathbf{O}_j) = a\Big(\mathbf{W}\begin{bmatrix}\mathbf{O}_j\mathbf{x}_i \\ \mathbf{O}_i\mathbf{x}_j\end{bmatrix} + \mathbf{b}\Big) \tag{2.15}$$

where $\mathbf{W} \in \mathbb{R}^{n \times 2n}$ and $\mathbf{b} \in \mathbb{R}^n$. The function indicates that, if a constituent is modified by a negator, its representation will be transformed by the matrix

operator of the negator before it is fed into a classical encoder function. Experiments in (Socher et al., 2012) show that the additional expressive power of MV-RNN can indeed shift sentiment of a constituent when it is modified by a negator. However, the introduction of matrix operators dramatically increases the number of parameters. Although the method defined a function for creating a parent matrix operator from two children operators, they still need to initialize a matrix operator for each word, which are the leaves of a representation tree. Although the new parameters grow linearly with the number of dintinct words in a corpus, no unsupervised training criterion is provided to pre-train this model.

2.3.3 Other Semi-supervised Learning Methods

Self-training

Self-training is a simple wrapper method, which is characterized by the fact that the learning process uses its own predictions to teach itself. It starts with a supervised learner trained on available labeled data and works in iterations. In each round, it uses the learner to predict the labels of the unlabeled data and selects a subset of unlabeled data, together with their predicted labels to augment the training dataset. Typically, the subset contains the data points with the most confident predictions. Then the new training data is used to update or re-train the supervised learner for the next round. The iterative learning process implies that self-training works only if its predictions with highest confidences tend to be correct (Zhu and Goldberg, 2009). One successful application is to use the self-training algorithm AROW (Crammer et al., 2009) for large scale review polarity prediction. Haimovitch et al. (2012) show that this method can reduce the test error by more than half compared to the supervised classifier trained on the initial labeled dataset.

Co-training

Co-training (Blum and Mitchell, 1998) adopts a similar iterative learning process as self-training. Instead of using one supervised learner, it uses two

learners to teach each other. The two learners operate on different feature sets of a training data point, where each feature set is referred to as a *view* of the data. In each iteration, the two learners provide their most confident predictions of the unlabeled data for one another to enrich its training dataset. The process terminates if all unlabeled data are exhausted.

For the successful application of co-training, the following assumptions should hold (Zhu and Goldberg, 2009):

- *Each view alone is sufficient for good training, given enough labeled data.*
- *The views are conditionally independent given a class label.*

The assumptions can be used in practice to guide the selection of feature space partitioning strategies. For the cross-lingual document polarity classification (Wan, 2009), each view is a set of language dependent features. If the class distribution is imbalanced, Li et al. (2011) proposed a under-sampling method to generate balanced datasets of different views. Compared to self-training, the advantage of co-training is the diversity of the involved learners, which makes co-training perform better than self-training for subjectivity detection at the sentence level (Yu and Kübler, 2011).

Transductive Support Vector Machines

Transductive Support Vector Machines (TSVMs) (Vapnik, 1998; Joachims, 1999a) learn a large-margin hyperplane classifier using both labeled and unlabeled data. The critical difference to SVMs is that TSVMs force the decision surface to lie in a low-density region. That means, it is assumed that data points naturally form some clusters and data points from the same cluster are likely to be of the same class. To assign labels to all unlabeled data, TSVMs perform transductive inference on all available data and try to put the decision boundary not close to any labeled and unlabeled points.

TSVMs are used in (Dasgupta and Ng, 2009) as the last stage for review polarity classification. Their method first performs spectral clustering to find out *unambiguous* reviews, then exploits them to classify ambiguous reviews via active learning, and at the end the TSVMs and ensemble learning are combined

to assign labels to unlabeled data points.

Chapter 3

Document Rating Prediction

The main concern of sentiment analysis is analyzing polarity and intensity of sentiments expressed in natural language text. In this chapter, the two quantities are analyzed jointly as numerical ratings with the focus on the document level. To tackle the task, we present the bag-of-opinions model, which can explore the syntactic structures of sentiment-bearing phrases for improved rating prediction of online reviews.

3.1 Overview

3.1.1 Motivation

This chapter discusses the learning and prediction of numerical ratings from review texts, which is modeled as a metric *regression* problem over an appropriately defined feature space.

Formally, the input is a set of rated documents (i.e., reviews), $\{\mathbf{x}_i, y_i\}_{i=1}^N$, where $\mathbf{x}_i$ is a sequence of word-level unigrams $(w_1, ..., w_l)$ and $y_i \in \mathbb{R}$ is a rating. The goal is to learn a function $f(\mathbf{x})$ that maps the word vector $\mathbf{x}$ into a numerical rating $\hat{y}$, which indicates both the polarity and intensity of the sentiments expressed in a document.

Numerical review rating prediction is harder than classifying by polarity. Consider the following example from Amazon book reviews:

The organization of the book is hard to follow and the chapter titles are not very helpful, so going back and trying to find information is quite difficult.

We note that there are many subjective words (*hard, helpful, difficult*) modified by sentiment modifiers such as (*very, quite*) and negation words like (*not*). For rating prediction, considering sentiment modifiers is crucial; *very helpful* is a much stronger sentiment than *helpful*. Negation words also need attention. As pointed out by Liu and Seneff (2009) we cannot simply reverse the polarity. For example, if we assign a higher positive score to *very helpful* than to *helpful*, simply reversing the sign of the scores would incorrectly suggest that *not helpful* is less negative than *not very helpful*.

The widely used unigram (bag-of-words) model (Pang and Lee, 2005b; Snyder and Barzilay, 2007; Goldberg and Zhu, 2006; Ganu et al., 2009) cannot properly capture phrase patterns. Consider the following example: *not so helpful* vs. *not so bad*. In a unigram-based regression model each unigram gets a weight indicating its polarity and intensity. High positive/negative weights are strongly positive/negative clues. It is reasonable to assign a positive weight to *helpful* and a negative weight to *bad*. The fundamental problem of unigrams arises when assigning a weight to *not*. If *not* had a strongly negative weight, the positive weight of *helpful* would be strongly reduced while the negative weight of *bad* would be amplified (by combining weights). The same problem holds for *so*, which is an amplifier that should keep the same sign as the word it modifies. We refer to this limitation of the unigram model as **polarity incoherence**.

A promising way of overcoming this weakness is to include *n-grams*, generalizing the bag-of-words model into a bag-of-phrases model (Baccianella et al., 2009; Pang and Lee, 2008). However, regression models over the feature space of all n-grams (for either fixed maximal n or variable-length phrases) are computationally expensive in their training phase. Moreover and most importantly for our setting, including n-grams in the model results in a very high dimensional feature space: many features will then occur only very rarely in the training data. Therefore, it is difficult if not impossible to reliably learn n-gram weights from limited-size training sets. We refer to this problem as the **n-gram sparsity bottleneck**. In our experiments we investigate the effect of using bigrams and variable-length ngrams for improving review rating

prediction.

3.1.2 Contribution

To overcome the above limitations of unigram and n-gram features, we have developed a novel kind of *bag-of-opinions* model, which exploits domain-independent corpora of sentiments (e.g., all Amazon reviews), but is finally applied for learning predictors on domain-specific reviews (e.g., movies as rated in IMDB or Rottentomatoes). A document is represented as a bag of sentiments each of which has three components: a root word, a set of modifier words and one or more negation words. In the phrase *not very helpful*, the sentiment root is *helpful*, one (of potentially many) sentiment modifier(s) is *very*, and a negation word is *not*. We enforce polarity coherence by the design of a learnable function that assigns a score to a sentiment.

Our approach generalizes the cumulative linear offset model (CLO) presented in (Liu and Seneff, 2009). The CLO model makes several restrictive assumptions, most notably, that all sentiment scores within one document are the same as the overall document rating. This assumption does not hold in practice, not even in reviews with extremely positive/negative ratings. For example, in a 5-star Amazon review the phrases *most impressive book* and *it helps explain* should receive different scores. Otherwise, the later transfer step to different domains would yield poor predictions. Due to this restriction, CLO works well on particular types of reviews that have pro/con entries listing characteristic major sentiments about the object under review. For settings with individual reviews whose texts do not exhibit any specific structure, the CLO model faces its limitations.

In our bag-of-opinions method, we address the learning of sentiment scores as a constrained ridge regression problem. We consider the sentiment scores in a given review to be drawn from an unknown probability distribution (so they do not have to be the same within a document). We estimate the review rating based on a set of statistics (e.g., expectation, variance, etc.) derived from the scores of sentiments in a document. Thus, our method has a sound statistical foundation and can be applied to arbitrary reviews with mixed sentiment

polarities and intensities. We avoid the n-gram sparsity problem by the limited-size structured feature space of *(root,modifiers,negators)* sentiments.

We treat domain-independent and domain-dependent sentiments differently in our system. In the first step we learn a bag-of-opinions model on a large dataset of online reviews to obtain scores for domain-independent sentiments. Since the polarity of sentiments is not bound to a topic, one can learn sentiment scores from a pooled corpus of reviews for various categories, e.g., movies, books, etc., and then use these scored sentiments for predicting the ratings of reviews belonging to a particular category. In order to also capture domain-dependent information (possibly complementary to the sentiment lexicon used for learning domain-independent sentiments), we combine the bag-of-opinions model with an unigram model trained on the domain-dependent corpus. Since domain-dependent training is typically limited, we model it using unigram models rather than bag-of-opinions. By combining the two models, even if a sentiment does not occur in the domain-dependent training set but it occurs in a test review, we can still accurately predict the review rating based on the globally learned sentiment score. In some sense our combined learning scheme is similar to smoothing in standard learning techniques, where the estimate based on a limited training set is smoothed using a large background corpus (Zhai and Lafferty, 2004).

In summary, the contributions of this chapter are the following:

1. We introduce the bag-of-opinions model, for capturing the influence of n-grams, but in a structured way with root words, modifiers, and negators, to avoid the explosion of the feature space caused by explicit n-gram models.

2. We develop a constrained ridge regression method for learning scores of sentiments from domain-independent corpora of rated reviews.

3. For transferring the regression model to newly given domain-dependent applications, we derive a set of statistics over sentiment scores in documents and use these as features, together with standard unigrams, for predicting the rating of a review.

4. Our experiments with Amazon reviews from different categories (books, movies, music) show that the bag-of-opinions method outperforms prior

state-of-the-art techniques.

3.1.3 Related Work

Rating Prediction

Document rating prediction is modeled either as an ordinal regression problem (Pang and Lee, 2005b; Goldberg and Zhu, 2006; Snyder and Barzilay, 2007) or as a metric regression problem (Goldberg and Zhu, 2006). These methods simply use the bag-of-words representation with regression algorithms. Titov and McDonald (2008a); Lu et al. (2011) extend the topic models for multi-aspect rating prediction. As seen previously, either the bag-of-words representation or latent features induced from sets of words cannot exploit the syntactic structures of phrases. Baccianella et al. (2009) restrict the n-grams to the ones having certain POS patterns. However, the long n-grams matching the patterns still suffer from sparsity. The same seems to hold for sparse n-gram models (BCR in this chapter) in the spirit of Ifrim et al. (2008). Although sparse n-gram models can explore arbitrarily large n-gram feature spaces, they can be of little help if the n-grams of interests occur sparsely in the datasets.

Sentiment Lexicon Learning

Since our approach can be regarded as learning a domain-independent sentiment lexicon, it is related to the area of automatically building domain-independent sentiment lexicons (Esuli and Sebastiani, 2006; Godbole et al., 2007b; Kim and Hovy, 2004). However, this prior work focused mainly on the polarity of sentiment words, neglecting the sentiment intensity. Recently, the lexicon based approaches were extended to learn domain-dependent lexicons (Kanayama and Nasukawa, 2006b; Qiu et al., 2009; Choi and Cardie, 2009a), but these approaches also neglect the aspect of sentiment intensity. Our method requires only the prior polarity of sentiment roots and can thus be used on top of those methods for learning the scores of domain-dependent sentiment components. The methods proposed in (Hu and Liu, 2004b; Popescu

and Etzioni, 2005a) can also be categorized into the lexicon based framework because their procedure starts with a set of seed words whose polarities are propagated to other sentiment-bearing words.

3.2 Bag-of-Opinions Model

In this section we first introduce the bag-of-opinions model, followed by the method for learning (domain-independent) model parameters. Then we show how we annotate sentiments and how we adapt the model to domain-dependent data.

3.2.1 Model Representation

We model each document as a bag-of-opinions $\{op_k\}_{k=1}^{K}$, where the number of sentiments K varies among documents. Each sentiment op_k consists of an sentiment root w_r, $r \in S_R$, a set of sentiment modifiers $\{w_m\}_{m=1}^{M}$, $m \in S_M$ and a set of negation words $\{w_z\}_{z=1}^{Z}$, $z \in S_Z$, where the sets S_R, S_M, S_Z are component index sets of sentiment roots, sentiment modifiers and negation words respectively. The union of these sets forms a global component index set $S \in \mathbb{N}^d$, where d is the dimension of the index space. The sentiment root determines the prior polarity of the sentiment. Modifiers intensify or weaken the intensity of the prior polarity. Negation words strongly reduce or reverse the prior polarity. For each sentiment, the set of negation words consists of at most a negation valence shifter like *not* (Kennedy and Inkpen, 2006) and its modifiers like capitalization of the valence shifter. Each sentiment component is associated with a score. We assemble the scores of sentiment elements into a sentiment-score by using a score function. For example, in the sentiment *not very helpful*, the sentiment root *helpful* determines the prior polarity positive say with a score 0.9, the modifier *very* intensifies the polarity say with a score 0.5. The prior polarity is further strongly reduced by the negation word *not* with e.g., a score -1.2. Then we sum up the scores to get a score of 0.2 for the sentiment *not very helpful*.

Formally, we define the function score(op) as a linear function of sentiment components, which takes the form

$$\text{score}(\text{op}) = \text{sgn}(r)\beta_r x_r + \sum_{m=1}^{M} \text{sgn}(r)\beta_m x_m + \sum_{z=1}^{Z} \text{sgn}(r)\beta_z x_z \tag{3.1}$$

where $\{x_z, x_m, x_r\}$ are binary variables denoting the presence or absence of negation words, modifiers and sentiment root. $\{\beta_z, \beta_m, \beta_r\}$ are weights of each sentiment elements. $\text{sgn}(r) : w_r \rightarrow \{-1, 1\}$ is the sentiment polarity function of the sentiment root w_r. It assigns a value 1/-1 if a sentiment root is positive/negative. Due to the semantics of sentiment elements, we have constraints that $\beta_r \geq 0$ and $\beta_z \leq 0$. The sign of β_m is determined in the learning phase, since we have no prior knowledge whether it intensifies or weakens the prior polarity.

Since a document is modeled as a bag-of-opinions, we can simply consider the expectation of sentiment scores as the document rating. If we assume the scores are uniformly distributed, the prediction function is then $f(\mathbf{x}) = \frac{1}{K}\sum_{k=1}^{K} \text{score}(\text{op}_k)$ which assigns the average of sentiment scores to the document $\mathbf{x}$.

3.2.2 Learning Regression Parameters

We assume that we can identify the sentiment roots and negation words from a subjectivity lexicon. In this work we use MPQA (Wilson et al., 2005). In addition, the lexicon provides the prior polarity of the sentiment roots. In the training phase, we are given a set of documents with ratings $\{\mathbf{x}_i, y_i\}_{i=1}^{N}$, and our goal is to find an optimal function f^* whose predictions $\{\hat{y}_i\}_{i=1}^{N}$ are as close as possibile to the original ratings $\{y_i\}_{i=1}^{N}$. Formally, we aim to minimize the following loss function:

$$L = \frac{1}{2N}\sum_{i=1}^{N} (f(\mathbf{x}_i) - y_i)^2 \tag{3.2}$$

where $f(\mathbf{x}_i)$ is modeled as the average score of sentiments in review $\mathbf{x}_i$.

First, we rewrite score(op) as the dot product $\langle \boldsymbol{\beta}, \mathbf{p} \rangle$ between a weight vector $\boldsymbol{\beta} = [\boldsymbol{\beta}_z, \boldsymbol{\beta}_m, \beta_r]$ and a feature vector $\mathbf{p} = [\text{sgn}(r)\mathbf{x}_z, \text{sgn}(r)\mathbf{x}_m, \text{sgn}(r)x_r]$. In order

to normalize the vectors, we rewrite the weight and feature vectors in the d dimensional vector space of all root words, modifiers and negation words. Then $\boldsymbol{\beta} = [.., \boldsymbol{\beta}_z, 0, .., \boldsymbol{\beta}_m, 0, .., \beta_r, 0..] \in R^d$ and $\mathbf{p} = [\text{sgn}(r)\mathbf{x}_z, 0, .., \text{sgn}(r)\mathbf{x}_m, 0, .., \text{sgn}(r)x_r, ...] \in R^d$. The function $f(\mathbf{x_i})$ can then be written as the dot product $\langle \boldsymbol{\beta}, \mathbf{v_i} \rangle$, where $\mathbf{v_i} = \frac{1}{K_i} \sum_{k=1}^{K_i} \mathbf{p}_k$, with K_i the number of sentiments in review $\mathbf{x_i}$. By using this feature representation, the learning problem is equivalent to:

$$\min_{\beta} \; L(\boldsymbol{\beta}) = \frac{1}{2N} \sum_{i=1}^{N} (\langle \boldsymbol{\beta}, \mathbf{v}_i \rangle + \beta_0 - y_i)^2$$

s.t.

$$\begin{aligned} \beta_z &\leq 0 \quad z \in S_Z \\ \beta_r &\geq 0 \quad r \in S_R \end{aligned} \tag{3.3}$$

where $\boldsymbol{\beta} \in R^d$, $\boldsymbol{\beta} = [\boldsymbol{\beta}_z, \boldsymbol{\beta}_m, \boldsymbol{\beta}_r]$. β_0 is the intercept of the regression function, which is estimated as the mean of the ratings in the training set. We define a new variable $\tilde{y}_i = y_i - \beta_0$.

In order to avoid overfitting, we add an l2 norm regularizer to the loss function with the parameter $\lambda > 0$.

$$LR(\boldsymbol{\beta}) = \frac{1}{2N} \sum_{i=1}^{N} (\langle \boldsymbol{\beta}, \mathbf{v}_i \rangle - \tilde{y}_i)^2 + \frac{\lambda}{2} \parallel \boldsymbol{\beta} \parallel_2^2$$

s.t.

$$\begin{aligned} \beta_z &\leq 0 \quad z \in S_Z \\ \beta_r &\geq 0 \quad r \in S_R \end{aligned} \tag{3.4}$$

We solve the above optimization problem by Algorithm 1 using coordinate descent. The procedure starts with $\boldsymbol{\beta}^0 = 0$, $\boldsymbol{\beta}^0 \in R^d$. Then it updates iteratively every coordinate of the vector $\boldsymbol{\beta}$ until convergence. Algorithm 1 updates every coordinate $\beta_j, j \in \{1, 2, ..., d\}$ of $\boldsymbol{\beta}$ by solving the following one-variable sub-problem:

$$\min_{l_j \leq \beta_j \leq c_j} LR(\beta_1, ..., \beta_j, ..., \beta_d)$$

where l_j and c_j denote the lower and upper bounds of β_j. If $j \in S_Z$, $l_j = -\infty$ and $c_j = 0$. If $j \in S_R$, $l_j = 0$ and $c_j = \infty$. Otherwise both bounds are infinity.

According to (Luo and Tseng, 1992), the solution of this one-variable sub-problem is

$$\hat{\beta}_j = \max\{l_j, \min\{c_j, g_j\}\}$$

where

$$g_j = \frac{\frac{1}{N}\sum_{i=1}^{N} v_{ij}(\tilde{y}_i - \sum_{l \neq j} \beta_l v_l)}{\frac{1}{N}\sum_{i=1}^{N} v_{ij}^2 + \lambda}$$

Here g_j is the close form solution of standard ridge regression at coordinate j (for details see (Friedman et al., 2010)). We prove the convergence of Algorithm 1, by the following theorem using techniques in (Luo and Tseng, 1992).

Theorem 1 *A sequence of* $\boldsymbol{\beta}$ *generated by Algorithm 1 globally converges to an optimal solution* $\beta^* \in \chi^*$ *of problem (3.4), where* χ^* *is the set of optimal solutions.*

Proof: *Luo and Tseng (1992) show that coordinate descent for constrained quadratic functions in the following form converges to one of its global optimal solutions.*

$$\begin{aligned} \min_{\boldsymbol{\beta}} \quad & h(\boldsymbol{\beta}) = \langle \boldsymbol{\beta}, \mathbf{Q}\boldsymbol{\beta} \rangle / 2 + \langle \mathbf{q}, \boldsymbol{\beta} \rangle \\ \text{s.t.} \quad & \mathbf{E}^\top \boldsymbol{\beta} \geq \mathbf{b} \end{aligned}$$

where $\mathbf{Q}$ *is a* $d \times d$ *symmetric positive-definite matrix,* $\mathbf{E}$ *is a* $d \times d$ *matrix having no zero column,* $\mathbf{q}$ *is a* d*-vector and* $\mathbf{b}$ *is a* d*-vector.*

We rewrite LR *in matrix form as*

$$\begin{aligned} & \frac{1}{2N}(\tilde{\mathbf{y}} - \mathbf{V}\boldsymbol{\beta})^\top(\tilde{\mathbf{y}} - \mathbf{V}\boldsymbol{\beta}) + \frac{\lambda}{2}\boldsymbol{\beta}^\top\boldsymbol{\beta} \\ = \; & \frac{1}{2N}(\mathbf{V}\boldsymbol{\beta})^\top(\mathbf{V}\boldsymbol{\beta}) + \frac{\lambda}{2}\boldsymbol{\beta}^\top\boldsymbol{\beta} - \frac{1}{2N}((\mathbf{V}\boldsymbol{\beta})^\top\tilde{\mathbf{y}} \\ & -\frac{1}{2N}\tilde{\mathbf{y}}^\top(\mathbf{V}\boldsymbol{\beta})) + \frac{1}{2N}\tilde{\mathbf{y}}^\top\tilde{\mathbf{y}} \\ = \; & \langle \boldsymbol{\beta}, \mathbf{Q}\boldsymbol{\beta} \rangle / 2 + \langle \mathbf{q}, \boldsymbol{\beta} \rangle + \text{constant} \end{aligned}$$

where

$$\mathbf{Q} = \mathbf{B}^\top\mathbf{B}, \mathbf{B} = \begin{bmatrix} \sqrt{\frac{1}{N}}\mathbf{V} \\ \sqrt{\lambda}\mathbf{I}^{d \times d} \end{bmatrix}, \mathbf{q} = \frac{-1}{N}(\mathbf{V}^\top\tilde{\mathbf{y}})$$

where $\mathbf{I}^{d \times d}$ *is the identity matrix. Because* $\lambda > 0$*, all columns of* $\mathbf{B}$ *are linearly independent. As* $\mathbf{Q} = \mathbf{B}^\top\mathbf{B}$ *and symmetric,* $\mathbf{Q}$ *is positive definite.*

Algorithm 1 Constrained Ridge Regression

1: Input: λ and $\{\mathbf{v}_n, \tilde{y}_n\}_{n=1}^N$
2: Output: optimal $\boldsymbol{\beta}$
3: **repeat**
4: **for** $j = 1, ..., d$ **do**
5: $g_j = \frac{\frac{1}{N}\sum_{i=1}^N v_{ij}(\tilde{y}_i - \sum_{l \neq j} \beta_l v_l)}{\frac{1}{N}\sum_{i=1}^N v_{ij}^2 + \lambda}$
6:

$$\hat{\beta}_j = \begin{cases} 0, & \text{if } j \in S_R \text{ and } g_j < 0 \\ 0, & \text{if } j \in S_Z \text{ and } g_j > 0 \\ g_j, & \text{else} \end{cases}$$

7: **end for**
8: **until** Convergence condition is satisfied

We define $\mathbf{E}$ *as a* $d \times d$ *diagonal matrix with all entries on the main diagonal equal to 1 except* $e_{ii} = -1, i \in S_Z$ *and* $\mathbf{b}$ *is a* d*-vector with all entries equal to* $-\infty$ *except* $b_i = 0$*, for* $i \in S_Z$ *or* $i \in S_R$*.*

Because the almost cyclic rule is applied to generate the sequence $\{\boldsymbol{\beta}^t\}$*, the algorithm converges to a solution* $\boldsymbol{\beta}^* \in \chi^*$*.*

3.2.3 Annotating Opinions

The MPQA lexicon contains separate lexicons for subjectivity clues, modifiers and valence shifters (Wilson et al., 2005), which are used for identifying sentiment roots, modifiers and negation words. Sentiment roots are identified as the positive and negative subjectivity clues in the subjectivity lexicon. In the same manner, modifiers and valence shifters of the type {negation, shiftneg} are mapped to modifiers and negation words. Other modifier candidates are adverbs, conjunctions and modal verbs around sentiment roots. We consider non-words modifiers as well, e.g., punctuations, capitalization and repetition of sentiment roots. If the sentiment root is a noun, adjectives are also included into modifier sets.

The automatic sentiment annotation starts with locating the continous subjectivity clue sequence. Once we find such a sequence and at least one of the

subjectivity clue is positive or negative, we search to the left up to 4 words for negation words and modifier candidates, and stop if encountering another sentiment root. Similarly, we search to the right up to 3 unigrams for modifiers and stop if we find negation words or any other sentiment roots. The prior polarity of the subjectivity sequence is determined by the polarity of the last subjectivity clue with either positive or negative polarity in the sequence. The other subjectivity clues in the same sequence are treated as modifiers.

3.2.4 Adaptation to Domain-Dependent Data

The adaptation of the learned (domain-independent) sentiment scores to the target domain and the integration of domain-dependent unigrams is done in a second ridge-regression task. Note that this is a simpler problem than typical domain-adaptation, since we already know from the sentiment lexicon which are the domain-independent features. Additionally, its relatively easy to obtain a large mixed-domain corpus for reliable estimation of domain-independent sentiment scores (e.g., use all Amazon product reviews). Furthermore, we need a domain-adaptation step since domain-dependent and domain-independent data have generally different rating distributions. The differences are mainly reflected in the intercept of the regression function (estimated as the mean of the ratings). This means that we need to scale the positive/negative mean of the sentiment scores differently before using it for prediction on domain-dependent reviews. Moreover, other statistics further characterize the sentiment score distribution. We use the variance of sentiment scores to capture the reliability of the mean, multiplied by the negative sign of the mean to show how much it strengthens/weakens the estimation of the mean. The mean score of the dominant polarity (*major exp*) is also used to reduce the influence of outliers. Because positive and negative means should be scaled differently, we represent positive and negative values of the mean and *major exp* as 4 different features. Together with variance, they are the 5 statistics of the sentiment score distribution. The second learning step on sentiment score statistics and domain-dependent unigrams as features, re-weights the importance of domain-independent and domain-dependent information according to the target domain bias.

3.3 Experimental Setup

feature models		uni	uni+bi	n-gram	CLO	**CRR-BoO**
DD	book	1.004	0.961	0.997	1.469	0.942
	dvd	1.062	1.018	1.054	1.554	0.946
	music	0.686	0.672	0.683	0.870	0.638
DD+MD	book	1.649	1.403	1.611	1.032	**0.884**
	dvd	1.592	1.389	1.533	1.086	**0.928**
	music	1.471	1.281	1.398	0.698	**0.627**
γDD+$(1-\gamma)$MD	book	0.996	0.944	0.986	n/a	n/a
	dvd	1.061	1.011	1.054	n/a	n/a
	music	0.695	0.673	0.690	n/a	n/a

Table 3.1: Mean squared error for rating prediction methods on Amazon reviews.

We performed experiments on three target domains of Amazon reviews: books, movies (DVDs), and music (CDs). For each domain, we use ca. 8000 Amazon reviews for evaluation; an additional set of ca. 4000 reviews are withheld for parameter tuning (regularization parameter, etc.). For learning weights for domain-independent sentiments, we use a mixed-domain corpus of ca. 350,000 reviews from Amazon (electronics, books, dvds, etc.); this data is disjoint from the test sets and contains no reviews from the music domain. In order to learn unbiased scores, we select about the same number of positive and negative reviews (where reviews with more/less than 3 stars are regarded as positive/negative). The regularization parameters used for this corpus are tuned on withheld data with ca. 6000 thematically mixed reviews.[1].

We compare our method, subsequently referred to as *CRR-BoO* (Constrained Ridge Regression for Bag-of-opinions), to a number of alternative state-of-the-art methods. These competitors are varied along two dimensions: 1) feature space, and 2) training set. Along the first dimension, we consider a) unigrams coined *uni*, b) unigrams and bigrams together, coined *uni+bi*, c) variable-length n-grams coined *n-gram*, d) the sentiment model by (Liu and

[1] All datasets are available from `http://www.mpi-inf.mpg.de/~lqu`

Seneff, 2009) coined *CLO* (cumulative linear offset model). As learning procedure, we use ridge regression for a), b), and d), and bounded cyclic regression, coined *BCR*, for c). Along the second - orthogonal - dimension, we consider 3 different training sets: i) domain-dependent training set coined *DD*, ii) domain-dependent training set and the large mixed-domain set coined *DD+MD*, iii) weighted combination $\gamma\text{DD} + (1-\gamma)\text{MD},\ \gamma \in (0,1)$. For the *DD+MD* training set, we apply our two stage approach for *CRR-BoO* and *CLO*, i.e., we use the mixed-domain corpus for learning the sentiment scores in the first stage, and integrate unigrams from *DD* in a second domain-adaptation stage. We train the remaining feature models directly on the combination of the whole mixed-domain corpus and the training part of *DD*. For the weighted set $\gamma\text{DD} + (1-\gamma)\text{MD}$ and the *uni*, *uni+bi* and *n-gram* models, we show experiments (Table 3.1) for the best $\gamma = 0.998$ (tuned by line search).

The CLO model is adapted as follows. Since bags-of-opinions generalize CLO, adjectives and adverbs are mapped to sentiment roots and modifiers, respectively; negation words are treated the same as CLO. Subsequently we use our regression technique. As Amazon reviews do not contain pro and con entries, we learn from the entire review.

For BCR, we adapt the variable-length n-grams method of (Ifrim et al., 2008) to elastic-net-regression (Friedman et al., 2010) in order to obtain a fast regularized regression algorithm for variable-length n-grams. We search for significant n-grams by incremental expansion in backward direction (e.g., expand *bad* to *not bad*). BCR pursues a dense solution for unigrams and a sparse solution for n-grams. Further details on the BCR learning algorithm will be found on a subsequent technical report.

As for the regression techniques, we show only results with ridge regression (for all feature and training options except BCR). It outperformed ϵ-*support vector regression* (SVR) of libsvm (Chang and Lin, 2001), lasso (Tibshirani, 1996), and elastic net (Zou and Hastie, 2005) in our experiments.

3.4 Results and Discussion

Table 3.1 shows the mean square error (MSE) from each of the three domain-specific test sets. The error is defined as $\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(f(\mathbf{x}_i) - y_i)^2$. The right most two columns of the table show results for the full-fledge two-stage learning for our method and CLO, with domain-dependent weight learning and the domain adaptation step. The other models are trained directly on the given training sets. For all methods we use five-fold cross-validation on the domain-specific sets.

Table 3.1 clearly shows that our *CRR-BoO* method outperforms all alternative methods by a significant margin. Most noteworthy is the music domain, which is not covered by the mixed-domain corpus. As expected, unigrams only perform poorly, and adding bigrams leads only to marginal improvements. BCR pursues a dense solution for unigrams and a sparse solution for variable-length n-grams, but due to the sparsity of occurence of long n-grams, it filters out many interesting-but-infrequent ngrams and therefore performs worse than the dense solution of the *uni+bi* model. The CLO method of (Liu and Seneff, 2009) shows unexpectedly poor performance. Its main limitation is the assumption that sentiment scores are identical within one document. This does not hold in documents with mixed sentiment polarities. It also results in conflicts for sentiment components that occur in both positive and negative documents. In contrast, *CRR-BoO* naturally captures the mixture of sentiments as a bag of positive/negative scores. We only require that the mean of sentiment scores equals the overall document rating.

The right most column of Table 1 shows that our method can be improved by learning sentiment scores from the large mixed-domain corpus. However, the high error rates of the models learned directly on the DD+MD corpus show that direct training on the DD+MD combination can introduce a significant amount of noise into the prediction models. Although the noise can be reduced by downweighting the influence of documents from MD, the performance is still comparable to directly learning only from the domain-dependent corpora. Thus, the two stages of our method (learning domain-independent sentiment scores plus domain-adaptation) are decisive for a good performance, and the sentiment-lexicon-based BoO model leads to robust learning of

sentiment	score
good	0.18
recommend	1.64
most difficult	-1.66
but it gets very good!	2.37
would highly recommend	2.73
would not recommend	-1.93

Table 3.2: Example sentiments learned from the Amazon mixed-domain corpus.

domain-independent sentiment scores.

Another useful property of BoO is its high interpretability. Table 3.2 shows example sentiment scores learned from the mixed-domain corpus. We observe that the scores corelate well with our intuitive interpretation of sentiments.

Our *CRR-BoO* method is highly scalable. Excluding the preprocessing steps (same for all methods), the learning of sentiment component weights from the ca. 350,000 domain-independent reviews takes only 11 seconds.

3.5 Conclusion

In this chapter we show that the bag-of-opinions (BoO) representation is better suited for capturing the expressive power of n-grams while at the same time overcoming their sparsity bottleneck. Although in this chapter we use the BoO representation to model domain-independent sentiments, we believe the same framework can be extended to domain-dependent sentiments and other NLP applications which can benefit from modelling n-grams (given that the n-grams are decomposable in some way). Moreover, the learned model can be regarded as a domain-independent sentiment lexicon with each entry in the lexicon having an associated score indicating its polarity and intensity. This in turn has potential applications in sentiment summarization, opinionated information retrieval and opinion extraction.

Chapter 4

Sentence Rating Prediction

The previous chapter introduced the bag-of-opinions model, which provides coarse-grained information to understand users' preference to entities (such as products or movies) mentioned in online reviews. To understand *why* users like or dislike certain items, however, we need to perform more fine-grained analysis of the review text itself. Therefore, this chapter presents the weakly supervised *multi-experts model* (MEM), which analyzes the polarity and intensity of sentiments expressed in sentences of online reviews.

4.1 Overview

4.1.1 Motivation

This chapter discusses the task of identifying polarity and intensity of sentiments expressed at the sentence level. The two quantities are jointly analyzed in the form of numerical ratings. A key challenge in fine-grained rating prediction is that fine-grained training data for both polarity and intensity is hard to obtain. We thus focus on a weakly supervised setting in which only coarse-level training data (such as document ratings and subjectivity lexicons) and, optionally, a small amount of fine-grained training data (such as sentence polarities) is available.

A number of lexicon-based approaches for phrase-level rating prediction has been proposed in the literature (Taboada et al., 2011; Qu et al., 2010). These

methods utilize a subjectivity lexicon of words along with information about their polarity and intensity; they focus on phrases that contain words from the lexicon. A key advantage of sentence-level methods is that they are able to cover all sentences in a review and that phrase identification is avoided. To the best of our knowledge, the problem of rating prediction at the sentence level has not been addressed in the literature. A naive approach would be to simply average phrase-level ratings. Such an approach performs poorly, however, since (1) phrases are analyzed out of context (e.g., modal verbs or conditional clauses), (2) domain-dependent information about polarity and intensity is not captured in the lexicons, (3) only phrases that contain lexicon words are covered. Here (1) and (2) lead to low precision, (3) to low recall.

4.1.2 Contribution

To address the challenges outlined above, we propose the weakly supervised *multi-experts model* (MEM) for sentence-level rating prediction. MEM starts with a set of potentially noisy indicators of sentiments including phrase-level predictions, language heuristics, and co-occurrence counts. We refer to these indicators as *base predictors*; they constitute the set of experts used in our model. MEM is designed such that new base predictors can be easily integrated. Since the information provided by the base predictors can be contradicting, we use ideas from ensemble learning (Dietterichl, 2002) to learn the most confident indicators and to exploit domain-dependent information revealed by document ratings. Thus, instead of averaging base predictors, MEM integrates their features along with the available coarse-grained training data into a unified probabilistic model.

The integrated model can be regarded as a Gaussian process (GP) model (Rasmussen, 2004) with a novel *multi-expert prior*. The multi-expert prior decomposes into two component distributions. The first component distribution integrates sentence-local information obtained from the base predictors. It forms a special realization of stacking (Dzeroski and Zenko, 2004) but uses the features from the base predictors instead of the actual predictions. The second component distribution propagates sentiment information across similar sentences using techniques from graph-based semi-supervised learning

(GSSL) (Zhu et al., 2003; Belkin et al., 2006b). It aims to improve the predictions on sentences that are not covered well enough by our base predictors.

In summay, the key constributions of MEM are the following:

- Among the probabilistic models applied for sentiment analysis, MEM allows integrating the richest resources to tackle the target task.
- As traditional GSSL algorithms support either discrete labels (classification) or numerical labels (regression), MEM extends these techniques to support both types of labels simultaneously.
- Our model uses a novel variant of word sequence kernels (Cancedda et al., 2003) to measure sentence similarity. Our kernel takes the relative positions of words but also their sentiment scores and synonymity into account.
- Our experiments indicate that MEM significantly outperforms prior work in both sentence-level rating prediction and sentence-level polarity classification.

4.1.3 Related Work

Weakly Supervised Learning

Supervised approaches for sentiment analysis focus mainly on opinion mining at the document level (Pang and Lee, 2004; Pang et al., 2002b; Pang and Lee, 2005b; Goldberg and Zhu, 2006), but have also been applied to sentence-level polarity classification in specific domains (Mao and Lebanon, 2006; Pang and Lee, 2004; McDonald et al., 2007b). In these settings, a sufficient amount of training data is available. In contrast, we focus on opinion mining tasks with little or no fine-grained training data.

The weakly supervised HCRF model (Täckström and McDonald, 2011a,b) for sentence-level polarity classification is perhaps closest to our work in spirit. Similar to MEM, HCRF uses coarse-grained training data and, when available, a small amount of fine-grained sentence polarities. In contrast to MEM, HCRF

does not predict the sentiment intensity and ignores the order of words within sentences.

Lexicon-based Methods

There exists a large number of lexicon-based methods for polarity classification (Ding et al., 2008b; Choi and Cardie, 2009b; Hu and Liu, 2004b; Zhuang et al., 2006b; Fu and Wang, 2010; Ku et al., 2008; Wang et al., 2013). Most of them rely on linguistically-motivatedd rules so that they fail to detect many domain dependent textual patterns of sentiments. The lexicon-based methods of (Taboada et al., 2011; Qu et al., 2010) also predict ratings at the phrase level; these methods are used as experts in our model.

Ensemble and GSSL Methods

MEM leverages ideas from ensemble learning (Dietterichl, 2002; Bishop et al., 2006) and GSSL methods (Zhu et al., 2003; Zhu and Ghahramani, 2002; Chapelle et al., 2006; Belkin et al., 2006b). The ensemble methods such as stacking (Dietterichl, 2002) and bagging (Breiman, 1996) combine multiple weak learners in an attempt to produce a strong predictor, whereas GSSL methods aim to build a robust model by making use of both labeled and unlabeled data. We extend GSSL with support for multiple, heterogenous labels. This allows us to integrate our base predictors as well as the available training data into a unified model that exploits that strengths of algorithms from both families.

4.2 Base Predictors

Each of our *base predictors* predicts the polarity or the rating of a single phrase. As indicated above, we do not use these predictions directly in MEM but instead integrate the features of the base predictors (see Sec. 4.3.4). MEM is designed such that new base predictors can be integrated easily.

Our base predictors use a diverse set of available web and linguistic resources. The hope is that this diversity increases overall prediction performance (Dietterichl, 2002): The *statistical polarity predictor* focuses on local syntactic patterns; it is based on corpus statistics for sentiment-bearing words and sentiment topic words. The *heuristic polarity predictor* uses manually constructed rules to achieve high precision but low recall. Both the *bag-of-opinions rating predictor* and the *SO-CAL rating predictor* are based on lexicons. The BoO predictor uses a lexicon trained from a large generic-domain corpus and is recall-oriented; the SO-CAL predictor uses a different lexicon with manually assigned weights and is precision-oriented.

4.2.1 Statistical Polarity Predictor

The polarity of an sentiment-bearing word strongly depends on its target word. For example, consider the phrase "I began this novel with the *greatest* of *hopes* [...]". Here, "greatest" has a positive polarity in all subjectivity lexicons, but the combination "greatest of hopes" often indicates a negative sentiment. We refer to a pair of sentiment-bearing word ("greatest") and a target word ("hopes") as an *sentiment-target pair*. Our statistical polarity predictor learns the polarity of sentiments and targets jointly, which increases the robustness of its predictions.

Syntactic dependency relations of the form $A \xrightarrow{R} B$ are a strong indicator for sentiment-target pairs (Qiu et al., 2009; Zhuang et al., 2006b); e.g., "great" $\xrightarrow{\text{nmod}}$ "product". To achieve high precision, we only consider pairs connected by the following predefined set of shortest dependency paths: verb $\xleftarrow{\text{subj}}$ noun, verb $\xleftarrow{\text{obj}}$ noun, adj $\xrightarrow{\text{nmod}}$ noun, adj $\xrightarrow{\text{prd}}$ verb $\xleftarrow{\text{subj}}$ noun. We only retain sentiment-target pairs that are sufficiently frequent.

For each extracted pair z, we count how often it co-occurs with each document polarity $y \in \mathcal{Y}$, where $\mathcal{Y} = \{positive, negative, other\}$ denotes the set of polarities. If z occurs in a document but is preceded by a negator, we treat it as a co-occurrence of opposite document polarity. If z occurs in a document with polarity *other*, we count the occurrence with only half weight, i.e., we increase both $\#z$ and $\#(other, z)$ by 0.5. These documents are typically a mixture of positive and negative sentiments so that we want to reduce their impact. The

marginal distribution of polarity label y given that z occurs in a sentence is estimated as $P(y \mid z) = \#(y, z)/\#z$. The predictor is trained using the text and ratings of the reviews in the training data, i.e., without relying on fine-grained annotations.

The statistical polarity predictor can be used to predict sentence-level polarities by averaging the phrase-level predictions. As discussed previously, such an approach is problematic; we use it as a baseline approach in our experimental study. We also employ phrase-level averaging to estimate the variance of base predictors; see Sec. 4.3.3. Denote by $Z(\mathbf{x})$ the set of sentiment-target pairs in sentence $\mathbf{x}$. To predict the sentence polarity $y \in \mathcal{Y}$, we take the Bayesian average of the phrase-level predictors: $P(y \mid Z(\mathbf{x})) = \sum_{z \in Z(\mathbf{x})} P(y \mid z) P(z) = \sum_{z \in Z(\mathbf{x})} P(y, z)$. Thus the most likely polarity is the one with the highest co-occurrence count.

4.2.2 Heuristic Polarity Predictor

Heuristic patterns can also serve as base predictors. In particular, we found that some authors list positive and negative aspects separately after keywords such as "pros" and "cons". A heuristic that exploits such patterns achieved a high precision ($> 90\%$) but low recall ($< 5\%$) in our experiments.

4.2.3 Bag-of-Opinions Rating Predictor

We leverage the bag-of-opinion (BoO) model of Qu et al. (2010) as a base predictor for phrase-level ratings. As mentioned in the previous chapter, a sentiment-bearing expression consists of a sentiment root (e.g., "good") contained in the MPQA lexicon Wilson et al. (2005), optionally a set of modifiers (e.g., "very") and negators (e.g., "not"). After learning the weights for each component of such expressions from a mixed-domain corpus, the phrase scores are computed by the function (3.1).

4.2.4 SO-CAL Rating Predictor

The Semantic Orientation Calculator (SO-CAL) of Taboada et al. (2011) also predicts phrase-level ratings via a scoring function similar to the one of BoO. The SO-CAL predictor uses a manually created lexicon, in which each word is classified as either an sentiment-bearing word (associated with a numerical score), a modifier (also associated with a numerical score), or a negator. SO-CAL employs various heuristics to detect irrealis and to correct for the positive bias inherent in most lexicon-based classifiers. Compared to BoO, SO-CAL has lower recall but higher precision.

4.3 Multi-Experts Model

Our multi-experts model incorporates features from the individual base predictors, coarse-grained labels (i.e., document ratings or polarities), similarities between sentences, and optionally a small amount of sentence polarity labels into an unified probabilistic model. We first give an overview of MEM, and then describe its components in detail.

4.3.1 Model Overview

Denote by $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ a set of sentences. We associate each sentence $\mathbf{x}_i$ with a set of *initial labels* $\hat{\mathbf{y}}_i$, which are strong indicators of sentiment polarity and intensity: the coarse-grained rating of the corresponding document, the polarity label of our heuristic polarity predictor, the phrase-level ratings from the SO-CAL predictor, and optionally a manual polarity label. Note that the number of initial labels may vary from sentence to sentence and that initial labels are heterogeneous in that they refer to either polarities or ratings. Let $\hat{\mathbf{Y}} = \{\hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_N\}$. Our goal is to predict the unobserved ratings $\mathbf{r} = \{r_1, \ldots, r_N\}$ of each sentence.

Our multi-experts model is a probabilistic model for $\mathbf{X}$, $\hat{\mathbf{Y}}$, and $\mathbf{r}$. In particular, we model the rating vector $\mathbf{r}$ via a multi-experts prior $P_E(\mathbf{r} \mid \mathbf{X}, \boldsymbol{\beta})$ with parameter $\boldsymbol{\beta}$ (Sec. 4.3.2). P_E integrates both features from the base predictors

and sentence similarities. We correlate ratings to initial labels via a set of conditional distributions $P_b(\hat{\mathbf{y}}^b \mid \mathbf{r})$, where b denotes the type of initial label (Sec. 4.3.3). The posterior of $\mathbf{r}$ is then given by

$$P(\mathbf{r} \mid \mathbf{X}, \hat{\mathbf{Y}}, \boldsymbol{\beta}) \propto \prod_b P_b(\hat{\mathbf{y}}^b \mid \mathbf{r}) P_E(\mathbf{r} \mid \mathbf{X}, \boldsymbol{\beta}).$$

Note that the posterior is influenced by both the multi-experts prior and the set of initial labels.

We use MAP inference to obtain the most likely rating of each sentence, i.e., we solve

$$\arg\min_{\mathbf{r},\boldsymbol{\beta}} - \sum_b \log(P_b(\hat{\mathbf{y}}^b \mid \mathbf{r})) - \log(P_E(\mathbf{r} \mid \mathbf{X}, \boldsymbol{\beta})),$$

where as before $\boldsymbol{\beta}$ denotes the model parameters. We solve the above optimization problem using cyclic coordinate descent (Friedman et al., 2010).

4.3.2 Multi-Experts Prior

The multi-experts prior $P_E(\mathbf{r} \mid \mathbf{X}, \boldsymbol{\beta})$ consists of two component distributions $\mathcal{N}_1$ and $\mathcal{N}_2$. Distribution $\mathcal{N}_1$ integrates features from the base predictors, $\mathcal{N}_2$ incorporates sentence similarities to propagate information across sentences.

In a slight abuse of notation, denote by $\mathbf{x}_i$ the set of features for the i-th sentence. Vector $\mathbf{x}_i$ contains the features of all the base predictors but also includes bigram features for increased coverage of syntactic patterns; see Sec. 4.3.4 for details about the feature design. Let $m(\mathbf{x}_i) = \boldsymbol{\beta}^T \mathbf{x}_i$ be a linear predictor for r_i, where $\boldsymbol{\beta}$ is a real weight vector. Assuming Gaussian noise, r_i follows a Gaussian distribution $\mathcal{N}_1(r_i \mid m_i, \sigma^2)$ with mean $m_i = m(\mathbf{x}_i)$ and variance σ^2. Note that predictor m can be regarded as a linear combination of base predictors because both m and each of the base predictors are linear functions. By integrating all features into a single function, the base predictors are trained jointly so that weight vector $\boldsymbol{\beta}$ automatically adapts to domain-dependent properties of the data. This integrated approach significantly outperformed the alternative approach of using a weighted vote of the individual predictions made by the base predictors. We regularize the

weight vector $\boldsymbol{\beta}$ using a Laplace prior $P(\boldsymbol{\beta} \mid \alpha)$ with parameter α to encourage sparsity.

Note that the bigram features in $\mathbf{x}_i$ partially capture sentence similarity. However, such features cannot be extended to longer subsequences such as trigrams due to data sparsity: useful features become as infrequent as noisy terms. Moreover, we would like to capture sentence similarity using gapped (i.e., non-consecutive) subsequences. For example, the sentences "The book is an easy read." and "It is easy to read." are similar but do not share any consecutive bigrams. They do share the subsequence "easy read", however. To capture this similarity, we make use of a novel sentiment-augmented variant of word sequence kernels (Cancedda et al., 2003). Our kernel is used to construct a similarity matrix $\mathbf{W}$ among sentences and the corresponding regularized Laplacian $\widetilde{\mathbf{L}}$. To capture the intuition that similar sentences should have similar ratings, we introduce a Gaussian prior $\mathcal{N}_2(\mathbf{r} \mid 0, \widetilde{\mathbf{L}}^{-1})$ as a component into our multi-experts prior; see Sec. 4.3.5 for details and a discussion of why this prior encourages similar ratings for similar sentences.

Since the two component distributions feature different expertise, we take their product and obtain the multi-experts prior

$$P_E(\mathbf{r} \mid \mathbf{X}, \boldsymbol{\beta}) \propto \mathcal{N}_1(\mathbf{r} \mid \mathbf{m}, \mathbf{I}\sigma^2)\mathcal{N}_2(\mathbf{r} \mid 0, \widetilde{\mathbf{L}}^{-1})P(\boldsymbol{\beta} \mid \alpha),$$

where $\mathbf{m} = (m_1, \ldots, m_N)$. Note that the normalizing constant of P_E can be ignored during MAP inference since it does not depend on $\boldsymbol{\beta}$.

4.3.3 Incorporating Initial Labels

Recall that the initial labels $\hat{\mathbf{Y}}$ provide strong clues about sentiments expressed in each sentence; they correspond to either discrete polarity labels or to continuous rating labels. This heterogeneity constitutes the main difficulty for incorporating the initial labels via the conditional distributions $P_b(\hat{\mathbf{y}}^b \mid \mathbf{r})$. We assume independence throughout so that $P_b(\hat{\mathbf{y}}^b \mid \mathbf{r}) = \prod_i P_b(\hat{y}_i^b \mid r_i)$.

Rating Labels For continuous labels, we assume Gaussian noise and set $P_b(\hat{y}_i^b \mid r_i) = \mathcal{N}(\hat{y}_i^b \mid r_i, \eta_i^b)$, where variance η_i^b is a type- and

sentence-dependent.

For SO-CAL labels, we simply set $\eta_i^{\text{SO-CAL}} = \eta^{\text{SO-CAL}}$, where $\eta^{\text{SO-CAL}}$ is a hyperparameter. The SO-CAL scores have limited influence in our overall model; we found that more complex designs lead to little improvement. We proceed differently for document ratings. Our experiment suggests that document ratings constitute the most important indicator of the rating of a sentence. Thus sentence ratings should be close to document ratings unless strong evidence to the contrary exists. In other words, we want variance η_i^{Doc} to be small.

When no manually created sentence-level polarity labels are available, we set the value of η_i^{Doc} depending on the polarity class. In particular, we set $\eta_i^{\text{Doc}} = 1$ for both positive and negative documents, and $\eta_i^{\text{Doc}} = 2$ for neutral documents. The reasoning behind this choice is that sentence ratings in neutral documents express higher variance because these documents often contain a mixture of positive and negative sentences.

When a small set of manually created sentence polarity labels is available, we train a classifier that predicts whether the sentence polarity coincides with the document polarity. If so, we set the corresponding variance η_i^{Doc} to a small value; otherwise, we choose a larger value. In particular, we train a logistic regression classifier (Bishop et al., 2006) using the following binary features: (1) an indicator variable for each document polarity, and (2) an indicator variable for each triple of base predictor, predicted polarity, and document polarity (set to 1 if the polarities match). We then set $\eta_i^{\text{Doc}} = (\tau p_i)^{-1}$, where p_i is the probability of matching polarities obtained from the classifier and τ is a hyperparameter that ensures correct scaling.

Polarity Labels We now describe how to model the correlation between the polarity of a sentence and its rating. An simple and effective approach is to partition the range of ratings into three consecutive partitions, one for each polarity class. We thus considering the polarity classes $\{positive, other, negative\}$ as ordered and formulate polarity classification as an ordinal regression

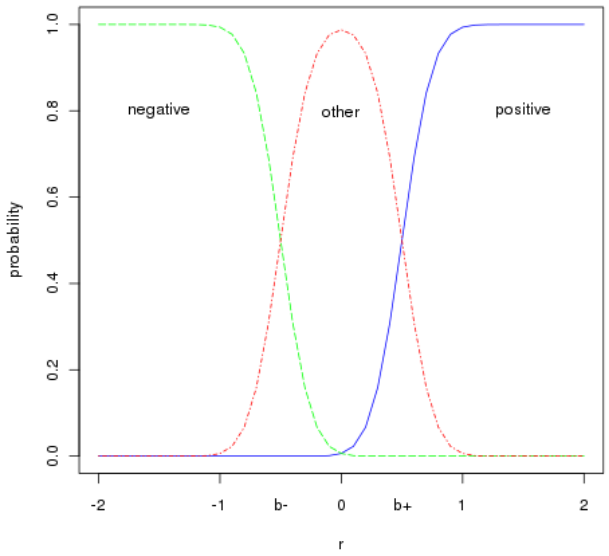


Figure 4.1: Distribution of polarity given rating.

problem (Chu and Ghahramani, 2006). We immediately obtain the distribution

$$
\begin{aligned}
P_b(\hat{y}_i^b = \text{pos} \mid r_i) &= \Phi\left(\frac{r_i - b^+}{\sqrt{\eta^b}}\right) \\
P_b(\hat{y}_i^b = \text{oth} \mid r_i) &= \Phi\left(\frac{b^+ - r_i}{\sqrt{\eta^b}}\right) - \Phi\left(\frac{b^- - r_i}{\sqrt{\eta^b}}\right) \\
P_b(\hat{y}_i^b = \text{neg} \mid r_i) &= \Phi\left(\frac{b^- - r_i}{\sqrt{\eta^b}}\right),
\end{aligned}
$$

where b^+ and b^- are the partition boundaries between *positive*/*other* and *other*/*negative*, respectively,[1] $\Phi(x)$ denotes the cumulative distribution function of the Gaussian distribution, and variance η^b is a hyperparameter. It is easy to verify that $\sum_{\hat{y}_i^b \in \mathcal{Y}} p(\hat{y}_i^b \mid r_i) = 1$. The resulting distribution is shown in Fig. 4.1. We can use the same distribution to use MEM for sentence-level polarity classification; in this case, we pick the polarity with the highest probability.

[1] We set $b^+ = 0.3$ and $b^- = -0.3$ to calibrate to SO-CAL, which treats ratings in $[-0.3, 0, 3]$ as polarity *other*.

4.3.4 Incorporating Base Predictors

Base predictors are integrated into MEM via component $\mathcal{N}_1(r_i \mid m_i, \sigma^2)$ of the multi-experts prior (see Sec. 4.3.2). Recall that m_i is a linear function of the features $\mathbf{x}_i$ of each sentence. In this section, we discuss how $\mathbf{x}_i$ is constructed from the features of the base predictors. New base predictors can be integrated easily by exposing their features to MEM.

Most base predictors operate on the phrase level; our goal is to construct features for the entire sentence. Denote by n_i^b the number of phrases in the i-th sentence covered by base predictor b, and let $\mathbf{o}_{ij}^b$ denote a set of associated features. Features $\mathbf{o}_{ij}^b$ may or may not correspond directly to the features of base predictor b; see the discussion below. A straightforward strategy is to set $\mathbf{x}_i^b = (n_i^b)^{-1} \sum_j \mathbf{o}_{ij}^b$. We proceed slightly differently and average the features associated with phrases of positive prior polarity separately from those of phrases with negative prior polarity (Taboada et al., 2011). We then concatenate the averaged feature vectors, i.e., we set $\mathbf{x}_i^b = (\bar{\mathbf{o}}_{ij}^{b,pos} \;\; \bar{\mathbf{o}}_{ij}^{b,neg})$, where $\bar{\mathbf{o}}_{ij}^{b,p}$ denotes the average of the feature vectors $\mathbf{o}_{ij}^b$ associated with phrases of prior polarity p. This procedure allows us to learn a different weight for each feature depending on its context (e.g., the weight of intensifier "very" may differ for positive and negative phrases). We construct $\mathbf{x}_i$ by concatenating the sentence-level features $\mathbf{x}_i^b$ of each base predictor and a feature vector of bigrams.

To integrate a base predictor, we only need to specify the relevant features and, if applicable, prior phrase polarities. For our choice of base predictors, we use the following features:

SO-CAL predictor. The prior polarity of a SO-CAL phrase is given by the polarity of its sentiment-bearing word in the SO-CAL lexicon. The feature vector $\mathbf{o}_{ij}^{\text{SO-CAL}}$ consists of the weight of the sentiment-bearing word from the lexicon as well the set of negator words, irrealis marker words, and modifier words in the phrase. Moreover, we add the first two words preceding the sentiment-bearing word as context features (skipping nouns, negators, irrealis markers, and intensifiers, and stopping at clause boundaries). All words are encoded as binary indicator features.

BoO predictor. Similar to SO-CAL, we determine the prior polarity of a phrase

based on the BoO dictionary. In contrast to SO-CAL, we directly use the BoO score as a feature because the BoO predictor weights have been trained on a very large corpus and are thus reliable. We also add irrealis marker words in the form of indicator features.

Statistical polarity predictor. Recall that the statistical polarity predictor is based on co-occurrence counts of sentiment-topic pairs and document polarities. We treat each sentiment-topic pair as a phrase and use the most frequently co-occurring polarity as the phrase's prior polarity. We use the logarithm of the co-occurrence counts with positive, negative, and other polarity as features; this set of features performed better than using the co-occurrence counts or estimated class probabilities directly. We also add the same type of context features as for SO-CAL, but rescale each binary feature by the logarithm of the occurrence count #z of the sentiment-topic pair (i.e., the features take values in $\{0, \log \#z\}$).

4.3.5 Incorporating Sentence Similarities

The component distribution $\mathcal{N}_2(\mathbf{r} \mid 0, \widetilde{\mathbf{L}}^{-1})$ in the multi-experts prior encourages similar sentences to have similar ratings. The main purpose of $\mathcal{N}_2$ is to propagate information from sentences on which the base predictors perform well to sentences for which base prediction is unreliable or unavailable (e.g., because they do not contain sentiment-bearing words). To obtain this distribution, we first construct an $N \times N$ sentence similarity matrix $\mathbf{W}$ using a sentiment-augmented word sequence kernel (see below). We then compute the regularized graph Laplacian $\widetilde{\mathbf{L}} = \mathbf{L} + \mathbf{I}/\lambda^2$ based on the unnormalized graph Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{W}$ (Chapelle et al., 2006), where $\mathbf{D}$ be a diagonal matrix with $d_{ii} = \sum_j w_{ij}$ and hyperparameter λ^2 controls the scale of sentence ratings.

To gain insight into distribution $\mathcal{N}_2$, observe that

$$\mathcal{N}_2(\mathbf{r} \mid 0, \widetilde{\mathbf{L}}^{-1}) \propto \exp\Big(-\frac{1}{2}\sum_{i,j} w_{ij}(r_i - r_j)^2 - \|\mathbf{r}\|_2^2/\lambda^2\Big).$$

The left term in the exponent forces the ratings of similar sentences to be similar: the larger the sentence similarity w_{ij}, the more penalty is paid for dissimilar

ratings. For this reason, $\mathcal{N}_2$ has a smoothing effect. The right term is an L2 regularizer and encourages small ratings; it is controlled by hyperparameter λ^2.

The entries w_{ij} in the sentence similarity matrix determine the degree of smoothing for each pair of sentence ratings. We compute these values by a novel sentiment-augmented word sequence kernel, which extends the well-known word sequence kernel of Cancedda et al. (2003) by (1) BoO weights to strengthen the correlation of sentence similarity and rating similarity and (2) synonym resolution based on WordNet (Miller, 1995b).

In general, a word sequence kernel computes a similarity score of two sequences based on their shared subsequences. In more detail, we first define a score function for a pair of shared subsequences, and then sum up these scores to obtain the overall similarity score. Consider for example the two sentences "The book is an easy read." (s_1) and "It is easy to read." (s_2) along with the shared subsequence "is easy read" (u). Observe that the words "an" and "to" serve as gaps as they are not part of the subsequence. We represent subsequence u in sentence s via a real-valued projection function $\phi_u(s)$. In our example, $\phi_u(s_1) = v_{is} v^g_{an} v_{easy} v_{read}$ and $\phi_u(s_2) = v_{is} v_{easy} v^g_{to} v_{read}$. The decay factors $v_w \in (0, 1]$ for matching words characterize the importance of a word (large values for significant words). On the contrary, decay factors $v^g_w \in (0, 1]$ for gap words are penalty terms for mismatches (small values for significant words). The score of subsequence u is defined as $\phi_u(s_1)\phi_u(s_2)$. Thus two shared subsequences have high similarity if they share significant words and few gaps. Following Cancedda et al. (2003), we define the similarity between two sequences as

$$k_n(s_i, s_j) = \sum_{u \in \Omega^n} \phi_u(s_i)\phi_u(s_j),$$

where Ω is a finite set of words and n denotes the length of the considered subsequences. This similarity function can be computed efficiently using dynamic programming.

To apply the word sequence kernel, we need to specify the decay factors. A traditional choice is $v_w = \log(\frac{N}{N_w})/\log(N)$, where N_w is the document frequency of the word w and N is the total number of documents. This IDF decay factor is not well-suited to our setting: Important sentiment-bearing words such as "great" have a low IDF value due to their high document

frequency. To overcome this problem, we incorporate additional weights for sentiment-bearing words using the BoO lexicon. To do so, we first rescale the BoO weights into $[0, 1]$ using the sigmoid $g(w) = (1 + \exp(-a\omega_w + b))^{-1}$, where ω_w denotes the BoO weight of word w.[2] We then set $\upsilon_w = \min(\log(\frac{N}{N_w})/\log(N) + g(w), 0.9)$. The decay factor for gaps is given by $\upsilon_w^g = 1 - \upsilon_w$. Thus we strongly penalize gaps that consist of infrequent words or sentiment-bearing words.

To address data sparsity, we incorporate synonyms and hypernyms from WordNet into our kernel. In particular, we represent words found in WordNet by their first two synset names (for verbs, adjectives, nouns) and their direct hypernym (nouns only). Two words are considered the same when their synsets overlap. Thus, for example, "writer" has the same representation as "author".

To build the similarity matrix $\mathbf{W}$, we construct a k-nearest-neighbor graph for all sentences.[3] We consider subsequences consisting of three words (i.e., $w_{ij} = k_3(s_i, s_j)$); longer subsequences are overly sparse, shorter subsequences are covered by the bigrams features in $\mathcal{N}_1$.

4.4 Experiments

We evaluated both MEM and a number of alternative approaches for both sentence-level polarity classification and sentence-level intensity prediction across a number of domains. We found that MEM outperforms state-of-the-art approaches by a significant margin.

4.4.1 Experimental Setup

We implemented MEM as well as the HCRF classifier of (Täckström and McDonald, 2011b,a), which is the best-performing estimator of sentence-level polarity in the weakly-supervised setting reported in the literature. We train

[2] We set $a = 2$ and $b = 1$ in our experiments.

[3] We use $k = 15$ and only consider neighbors with a similarity above 0.001.

both methods using (1) only coarse labels (MEM-Coarse, HCRF-Coarse) and (2) additionally a small number of sentence polarities (MEM-Fine, HCRF-Fine[4]). We also implemented a number of baselines for both polarity classification and intensity prediction: a document oracle (DocOracle) that simply uses the document label for each sentence, the BoO rating predictor (Base_{BoO}), and the SO-CAL rating predictor ($\text{Base}_{\text{SO-CAL}}$). For polarity classification, we compare our methods also to the statistical polarity predictor ($\text{Base}_{\text{polarity}}$). To judge on the effectiveness of our multi-export prior for combining base predictors, we take the majority vote of all base predictors and document polarity as an additional baseline (Majority-Vote). Similarly, for intensity prediction, we take the arithmetic mean of the document rating and the phrase-level predictions of Base_{BoO} and $\text{Base}_{\text{SO-CAL}}$ as a baseline (Mean-Rating). We use the same hyperparameter setting for MEM across all our experiments.

We evaluated all methods on Amazon reviews from different domains using the corpus of Ding et al. (2008b) and the test set of Täckström and McDonald (2011b). For each domain, we constructed a large balanced dataset by randomly sampling 33,000 reviews from the corpus of Ding et al. (2008b). We chose the books, electronics, and music domains for our experiments; the dvd domain was used for development. For sentence polarity classification, we use the test set of Täckström and McDonald (2011b), which contains roughly 60 reviews per domain (20 for each polarity). For intensity evaluation, we created a test set of 300 pairs of sentences per domain from the polarity test set. Each pair consisted of two sentences of the same polarity; we manually determined which of the sentences is more positive. We chose this pairwise approach because (1) we wanted the evaluation to be invariant to the scale of the predicted ratings, and (2) it much easier for human annotators to rank a pair of sentences than to rank a large collection of sentences.

We followed Täckström and McDonald (2011a) and used 3-fold cross-validation, where each fold consisted of a set of roughly 20 documents from the test set. In each fold, we merged the test set with the reviews from the corresponding domain. For MEM-Fine and HCRF-Fine, we use the data from the other two folds as fine-grained polarity annotations. For our experiments

[4]We used the best-performing model that fuses HCRF-Coarse and the supervised model (McDonald et al., 2007b) by interpolation.

on polarity classification, we converted the predicted ratings of MEM, Base_{BoO}, and $\text{Base}_{\text{SO-CAL}}$ into polarities by the method described in Sec. 4.3.3. We compare the performance of each method in terms of accuracy, which is defined as the fraction of correct predictions on the test set (correct label for polarity / correct ranking for intensity). All reported numbers are averages over the three folds. In our tables, boldface numbers are statistically significant against all other methods (t-test, p-value 0.05).

4.4.2 Results for Polarity Classification

Table 4.1 summarizes the results of our experiments for sentence polarity classification. The base predictors perform poorly across all domains, mainly due to the aforementioned problems associated with averaging phrase-level predictions. In fact, DocOracle performs almost always better than any of the base predictors. However, accurracy increases when we combine base predictors and DocOracle using majority voting, which indicates that ensemble methods work well.

When no fine-grained annotations are available (HCRF-Coarse, MEM-Coarse), both MEM-Coarse and Majority-Vote outperformed HCRF-Coarse, which in turn has been shown to outperform a number of lexicon-based methods as well as classifiers trained on document labels (Täckström and McDonald, 2011b). MEM-Coarse also performs better than Majority-Vote. This is because MEM propagates evidence across similar sentences, which is especially useful when no explicit sentiment-bearing words exist. Also, MEM learns weights of features of base predictors, which leads to a more adaptive integration, and our ordinal regression formulation for polarity prediction allows direct competition among positive and negative evidence for improved accuracy.

When we incorporate a small amount of sentence polarity labels (HCRF-Fine, MEM-Fine), the accuracy of all models greatly improves. HCRF-Fine has been shown to outperform the strongest supervised method on the same dataset (McDonald et al., 2007b; Täckström and McDonald, 2011a). MEM-Fine falls short of HCRF-Fine only in the electronics domain but performs better on all other domains. In the book and music domains, where MEM-Fine is

	Book	Electronics	Music	Avg
$\text{Base}_{\text{polarity}}$	43.7	40.3	43.8	42.6
Base_{BoO}	50.9	48.9	52.6	50.8
$\text{Base}_{\text{SO-CAL}}$	44.6	50.2	45.0	46.6
DocOracle	51.9	49.6	59.3	53.6
Majority-Vote	53.7	53.4	58.7	55.2
HCRF-Coarse	52.2	53.4	57.2	54.3
MEM-Coarse	54.4	54.9	**64.5**	57.9
HCRF-Fine	55.9	**61.0**	58.7	58.5
MEM-Fine	**59.7**	59.6	63.8	**61.0**

Table 4.1: Accuracy of polarity classification per domain and averaged across domains.

	Book		Electronics		Music	
	op	fact	op	fact	op	fact
HCRF-Fine	55.7	55.9	**63.3**	54.6	59.0	57.4
MEM-Fine	**58.9**	**62.4**	60.7	**56.7**	**64.5**	**60.8**

Table 4.2: Accuracy of polarity classification for sentences with sentiment words (op) and without sentiment words (fact).

particularly effective, many sentences feature complex syntactic structure and sentiment-bearing words are often used without reference to the quality of the product (but to describe contents, e.g., "a love story" or "a horrible accident").

Our models perform especially well when they are applied to sentences containing no or few sentiment words from lexicons. Table 4.2 reports the evaluation results for both sentences containing sentiment-bearing words from either MPQA or SO-CAL lexicons and for sentences containing no such words. The results explain why our model falls short of HCRF-Fine in the electronics domain: reviews of electronic products contain many sentiment-bearing words, which almost always express sentiments. Nevertheless, MEM-Fine handles sentences without explicit sentiment-bearing words well across all domains; here the propagation of information across sentences helps to learn the facts implying evaluation (such as "short battery life").

We found that for all methods, most of the errors are caused by misclassifying positive/negative sentences as *other* and vice versa. Moreover, sentences with polarity opposite to the document polarity are hard cases if they do not feature frequent strong patterns. Another difficulty lies in off-topic sentences, which may contain explicit sentiment-bearing words but are not related to the item under review. This is one of the main reasons for the poor performance of the lexicon-based methods.

Overall, we found that MEM-Fine is the method of choice. Thus our multi-experts model can indeed balance the intensity of the individual experts to obtain better estimation accuracy.

4.4.3 Results for Strength Prediction

Table 4.3 shows the accuracy results for intensity prediction. Here our models outperformed all baselines by a large margin. Although document ratings are strong indicators in the polarity classification task, they lead to worse performance than lexicon-based methods. The main reason for this drop in accuracy is that the document oracle assigns the same rating to all sentences within a review. Thus DocOracle cannot rank sentences from the same review, which is a severe limitation. This shortage can be partly compensated by averaging the base predictions and document rating (Mean-Rating). Note that it is non-trivial to apply existing ensemble methods for the weights of individual base predictors because of the absence of the sentence ratings as training labels. In contrast, our MEM models use indirect supervision to adaptively assign weights to the features from base predictors. Similar to polarity classification, a small amount of sentence polarity labels often improved the performance of MEM.

4.5 Conclusion

We proposed the Multi-Experts Model for analyzing both sentiment polarity and intensity at the sentence level. MEM is weakly supervised; it can run without any fine-grained annotations but is also able to leverage such annotations when

	Book	Electronics	Music	Avg
Base_{BoO}	58.3	51.6	53.5	54.5
$\text{Base}_{\text{SO-CAL}}$	60.6	57.1	47.6	55.1
DocOracle	45.1	36.2	41.4	40.9
Mean-Rating	70.3	57.0	60.8	62.7
MEM-Coarse	68.7	60.5	**69.5**	66.2
MEM-Fine	**72.4**	**63.3**	67.2	**67.6**

Table 4.3: Accuracy of intensity prediction.

available. MEM is driven by a novel multi-experts prior, which integrates a number of diverse base predictors and propagates information across sentences using a sentiment-augmented word sequence kernel. Our experiments indicate that MEM achieves better overall accuracy than alternative methods.

Chapter 5

Subsentential Relationship Extraction

Previous chapters discussed document-level and sentence-level sentiment analysis regardless of the entities mentioned in text. However, if we want to identify sentiments with respect to entities, we need both syntactic and semantic analysis at both sentence and expression level. We refer to this task as subsentential relationship extraction. This chapter presents SENTI-LSSVM$_{RAE}$, a method capable of extracting both binary and ternary sentiment-oriented relationships from sentences. It can learn from training datasets that do not contain explicit annotations of sentiment-bearing expressions. The empirical evaluation shows that SENTI-LSSVM$_{RAE}$ significantly outperforms the state-of-the-art baselines across domains (camera and movie) and across genres (reviews and forum posts).

5.1 Overview

5.1.1 Motivation

Sentiment-oriented relationship extraction is concerned with recognizing sentiments and comparisons between entities with respect to their attributes from natural language text. A critical challenge of this task is that a significant amount of sentences contain more than one type of relationships and most

sentences are mixtures of both subjective and objective expressions (Wilson, 2008; Wiebe et al., 2005). However, most prior work of sentiment analysis focus on an individual task, either subjective sentence detection (Yu and Kübler, 2011), polarity classification (Johansson and Moschitti, 2011; Wilson et al., 2005), or comparative relationship identification (Jindal and Liu, 2006b; Ganapathibhotla and Liu, 2008). Therefore, our goal is to identify both comparative relationships and the sentiment polarity for entities of interest *simultaneously* from online reviews and forums, with an assumption that all the mentions of (disambiguated) entities and attributes are given. More specifically, we focus on extracting two types of relationships from text:

- *Sentiment*. We are interested in positive and negative sentiments regarding an entity and its attribute. Thus, a sentiment is a binary relationship in the form of *polarity*(entity, attribute) (e.g. *positive*(`Nikon D7000`, `lens`)).
- *Comparative relationship*. This type of relationship expresses the preference between two entities concerning an attribute, which takes the form of *better*(entity A, entity B, attribute), such as *better*(`Nikon D7000`, `Canon 7D`, `price`). These relationships are also referred to as *non-equal gradable* comparative relations in (Jindal and Liu, 2006b).

For example, a forum post might contain the following sentence

- Example 5.1 : *"[The Canon SD880i, which you mention], [is probably better than the SD770i and the SD790is,] [and also has a wide angle view (28mm)]."*

Then a relationship extraction system aims to extract the following relationships

better(`Canon SD880i`, `Canon SD770i`, `product`)
better(`Canon SD880i`, `Canon SD790is`, `product`)
positive(`Canon SD880i`, `lens`)

where `product` is augmented as the implied attribute.

To the best of our knowledge, the only existing system capable of extracting both comparisons and sentiments is a rule-based system proposed by Ding et al. (2009). We argue that it is better to formulate the task as a *learning problem* with *structured outputs*, because we can exploit the correlation among the sentiment-oriented relationships expressed in a sentence to improve performance. As we can see from the above example sentence, we can reduce

ambiguity by encoding constraints to ensure that "wide angle view" as the mention of `lens` participates either in comparative relationships or in sentiments. We can also infer that `lens` is more likely to be associated with a positive sentiment than a negative one given an obvious preference of `Canon SD880i` conveyed by the keyword "better". As a result, a learning based system should be able to extract a set of correlated relationships from a given sentence by performing joint inference.

However, constructing a fully annotated training corpus for this task is labor-intensive and time-consuming. We found that this overhead can be largely reduced by applying an output-oriented annotation scheme, in which annotators only mark entity mentions, disambiguate the entities and label the relationships for each sentence. As the annotations based on our scheme provide already full semantic information, the annotations of sentiment-bearing expressions are in fact optional. Based on this scheme, we have created a small Sentiment Relationship Graph (SRG) corpus, which is significantly different from the corpora used in prior work (Wei and Gulla, 2010; Kessler et al., 2010; Toprak et al., 2010; Wiebe et al., 2005; Zhuang et al., 2006a; Hu and Liu, 2004a) in the following perspectives: i) both sentiments and comparative relationships are annotated; ii) all mentioned entities are disambiguated; and iii) no subjective expressions are annotated, unless they are part of entity mentions.

The light weight annotation scheme raises a new challenge for learning algorithms that they need to automatically find words, phrases or sentences as *textual evidences* for each relationship in the training phase. In Example 5.1, there are the following 12 relationship candidates with unknown relationships but much less expressions (a set of possible expressions are marked by brackets in Example 5.1) potentially conveying certain relationships. As the relationships between entities and attributes are mainly determined by the linguistic expressions around them, these three expressions are shared by different relationship candidates, which can be of different relationship types. The gold standard training data do not contain i) the annotations of sentiment-bearing expressions; ii) the ones indicating associations between expressions and relationship candidates. Hence one approach would be to take all expressions around a relationship candidate as its feature representation.

(Canon SD880i, Canon SD770i, product)
(Canon SD880i, Canon SD790is, product)
(Canon SD770i, Canon SD790is, product)
(Canon SD880i, Canon SD770i, lens)
(Canon SD880i, Canon SD790is, lens)
(Canon SD770i, Canon SD790is, lens)
(Canon SD880i, lens)
(Canon SD770i, lens)
(Canon SD790is, lens)
(Canon SD880i, product)
(Canon SD770i, product)
(Canon SD790is, product)

Table 5.1: Relationship candidates of Example 5.1.

However, it leads to overly similar representations for candidates of different types. For example, the relationships *better*(Canon SD770i, product) and *other*(Canon SD880i, Canon SD770i, product) can include the keyword "better" into their feature representation despite of different relationship types. As shown in our experiments below, this problem can lead to poor learning performance. Therefore, a textual fragment should be assigned to a relationship, only if it conveys the relationship.

Even if the textual evidences are assigned properly to relationships, it is beneficial to capture the *compositionality of expressions* for understanding the sentiment-bearing expressions (Choi and Cardie, 2008; Yessenalina and Cardie, 2011), since the meanings of higher level expressions are often constructed based on the low level constituents (words and phrases), which may indicate different sentiments or relationships. Considering the following example sentences:

- Example 5.2 : *It is very difficult to get good images at the long end without a tripod.*
- Example 5.3 : *Tom Cruise does not even attempt a German accent and does not really inject any personality into the role.*

In Example 5.2, the negative expression "It is very difficult to get good images" contains a positive phrase "good images". Example 5.3 consists soly of neutral words, however, these words jointly express a negative sentiment. Learning the compositional patterns instead of treating them as different word sets or sequences could make models much easier generalized to unseen data. In our problem setting, it is more challenging since the SRG corpus contains only a few hundred documents without any annotated relation-bearing expressions.

5.1.2 Contribution

This chapter presents SENTI-LSSVM$_{RAE}$, a latent structural SVM (Yu and Joachims, 2009) based on recursive autoencoder (RAE) (Socher et al., 2011b) for sentiment oriented multi-relationship extraction. SENTI-LSSVM$_{RAE}$ is applied to find the most likely set of the relationships expressed in a given sentence, where the latent variables are used to assign the most appropriate textual evidences to the respective relationships. To capture the compositionality of expressions, we model each textual evidence as a latent feature vector and apply RAE to construct the latent feature vectors of phrases and sentences from the latent features of words, which are initialized with unlabeled data.

In summary, the contributions of this chapter are the following:

- To our best knowledge, SENTI-LSSVM$_{RAE}$ is the first learning system with the capability of extracting both binary and ternary sentiment-oriented relationships.
- We designed a task-specific integer linear programming (ILP) formulation for inference over the latent structural SVM method.
- The newly constructed SRG corpus is a valuable asset for the evaluation of sentiment-oriented relationship extraction.
- Our experiments on the online reviews and forum posts show that SENTI-LSSVM$_{RAE}$ model can i) effectively learn from a training corpus without explicitly annotated subjective expressions and ii) it significantly outperforms the state-of-the-art systems.

5.1.3 Related Work

Fine-grained Sentiment Analysis

There is ample work on analyzing the sentiment polarities and product comparisons, but the majority of them studied the two tasks in isolation.

Most prior approaches for fine-grained sentiment analysis focus on polarity classification. Supervised approaches on expression-level analysis require the annotations of sentiment-bearing expressions as training data (Choi et al., 2006; Jin et al., 2009; Choi and Cardie, 2010; Yessenalina and Cardie, 2011; Wei and Gulla, 2010). However, the corresponding annotation process is time-consuming. Although sentence-level annotations are easier to obtain, the analysis at this level cannot cope with sentences conveying relationships of multiple types (McDonald et al., 2007a; Täckström and McDonald, 2011; Socher et al., 2012). Moreover, lexicon-based approaches require no training data (Ku et al., 2006; Kim and Hovy, 2006; Godbole et al., 2007a; Ding et al., 2008a; Popescu and Etzioni, 2005b; Liu et al., 2005) but suffer from inferior performance (Wilson et al., 2005; Qu et al., 2012). In contrast, our method requires no annotations of sentiment-bearing expressions for training and can predict both sentiment polarities and comparative relationships.

Sentiment-oriented comparative relationships have been studied in the context of user-generated discourse (Jindal and Liu, 2006a,b; Ganapathibhotla and Liu, 2008). Their approaches rely on linguistically-motivated rules and assume the existence of independent keywords in sentences which indicate comparative relationships. Therefore, their methods fall short of extracting the comparative relationships based on domain dependent information.

Learning for Structured Outputs

Structural SVM is a widely used model for the NLP problems with structured outputs, such as POS tagging and parsing (Tsochantaridis et al., 2004). This model aims to learn a mapping from input data points $x \in \mathcal{X}$ to structured outputs $y \in \mathcal{Y}$ based on a set of input-output pairs (x_1, y_1),...,$(x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$. The output $y \in \mathcal{Y}$ could be, for instance, a sequence of POS tags or a constituent

parse tree for a sentence. And the mapping is obtained by learning a so called *discriminant function* $F : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$, which predicts a score for each input-output pair to reflect their probabilities. Hence we can derive a prediction by maximizing F over the response variable for a given input x, as shown by the following equation.

$$F^* = \arg\max_{y \in \mathcal{Y}} F(x, y; \theta) \tag{5.1}$$

where θ denotes the function parameters. If we further assume the function is linear, we obtain the linear variant of the model

$$F^* = \arg\max_{y \in \mathcal{Y}} \beta \Phi(x, y) \tag{5.2}$$

where $\Phi(x, y)$ is a feature function characterizing input-output pairs and β is the corresponding weight vector. In most cases, the feature function is in the form of a feature vector, which stores the numerical values mapped from application dependent feature-label pairs. To ensure the tractability of inference, the whole output structure is decomposed into a bag of correlated substructures, which is the same as factorizing $\Phi(x, y)$ into a set of local feature functions, capturing local patterns of data points. For example, given a sentence x, the linear model for dependency parsing can be decomposed as

$$\Phi(x, y) = \sum_{e \in E} \Phi_e(x, y_e)$$

where $\Phi_e(x_i, y_i)$ is a local feature function representing an edge and E is the edge set of a possible parse tree. The feature function could be a vector storing the value of an indicator function for the co-occurrence of a surface word and a certain dependency edge label. It could also be a composite function serving as a automatic feature detector, which are learned by using deep learning techniques (Socher et al., 2011a). To apply this model to our relationship extraction task, we can only associate a fixed local feature function with each relationship candidate, which leads to the overly similar feature representations for relationships of different types (cf. Section 5.1.1). In contrast, our model is based on latent structural SVM, which provides the flexibility to automatically select textual evidence for each relationship and ensures that a textual evidence conveys at most one relationship of interest.

As indicated by the function (5.1), the inference problems of structural SVM are solved by finding the most likely output structures given the model

parameters. Since the models are factorized into a bag of correlated substructures, this kind of correlations can often be encoded as a set of soft and hard constraints. Thus it is convenient to formulate the inference as integer linear programs (Roth and Yih, 2005; Punyakanok et al., 2004; Riedel and Clarke, 2006). In the field of sentiment analysis, Choi and Cardie (2009a) applied ILP to adapt an existing subjectivity lexicon into a new one to reflect the domain dependent information; Somasundaran and Wiebe (2009) incorporate constraints in the form of ILP to recognize which stance a person is taking in an online debate; and Wu et al. (2011) use ILP to infer the most likely graph representations of sentiments as well as their conditions. In contrast, our task requires a task-specific ILP reformulation due to 1) the absence of annotated sentiment expressions and 2) the joint extraction of both sentiments and comparative relationships.

Both Johansson and Moschitti (2011) and Wu et al. (2011) applied structural SVM for fine-grained sentiment analysis. Johansson and Moschitti (2011) proposed a joint model to identify the boundaries and polarities of sentiment-bearing expressions, while Wu et al. (2011) consider not only the polarity but also the conditions or scopes associated with sentiments, e.g. the positive sentiment expressed in "This book is excellent for beginners." is restricted only to "beginners". However, both works require the annotations of sentiment-bearing expressions and they focus only on polarity classification.

Compositional Semantics

Due to the fact that the meaning of an sentiment-bearing expression can often be inferred from its parts, there is an emerging interest on modeling compositionality of subjective expressions. Choi and Cardie (2008) exploit the idea of computational semantics for expression-level polarity classification by applying the heuristic rules on the automatically inferred semantic categories of individual words. Yessenalina and Cardie (2011) represent each word as a matrix and capture the compositional effects by multiplying these word matrices. For an expression, these matrix-space representations serve as features for predicting its ordinal sentiment score, which reflects both its polarity and strength. In both works, the boundaries of subjective expressions

are assumed to be given. In our work, we employ the RAE (Socher et al., 2011b) as an automatic feature extractor of our model, and augment it with L1 regularizers for predicting both sentiment polarity and comparative relationships. We consider RAE because it provides a principled way to initialize the vector representations of each word with unlabeled data and these vectors can be further updated based on supervised information.

5.1.4 Organization

The remainder of this chapter is organized as follows. Section 5.2 gives an overview of the whole system. After introducing the SENTI-LSSVM$_{RAE}$ model in Section 5.3, we give the details of its feature space in Section 5.4, its ILP-based structural inference in Section 5.5 and the learning of the model parameters in Section 5.6. Moreover, Section 5.7 covers the details of our SRG corpus. The experimental evaluation and its results based on this corpus are described in Section 5.8. Finally, we draw the conclusion in Section 5.9.

5.2 System Overview

This section gives an overview of the whole system for extracting sentiment-oriented relationships. Prior to presenting the system architecture, we introduce first the essential concepts and the definitions of two kinds of directed hypergraphs as the representation of correlated relationships extracted from sentences.

5.2.1 Concepts and Definitions

Entities. An entity is a abstract or concrete thing, which needs not be of material existence. An entity in this chapter is either a product, product brand or a product attribute (e.g. camera lens, movie title).

Attribute. An attribute refers to an object closely associated with or belonging to an entity, such as the lens of a camera.

Entity Types. We denote by *entity types*, sets of entities with similar properties. For example, "Canon 7D" is of the type *product* and "Christopher Nolan" has the type *director*.

Relationships. A relationship is a tuple of entities with a certain association or connection. In this chapter, we are interested in the binary and ternary relationships involving at least a product or a product brand and exactly one product attribute. For unary relationships such as the one from a sentence like "Canon 7D is excellent.", we normalize it as *positive*(Canon 7D, *product*) by augmenting with the implied attribute *product*.

Sentiment-Oriented Relationships. A sentiment-oriented relationship is either a sentiment or a comparative relationship, which are defined in Section 5.1.1. Therefore, a sentiment-oriented relationship is either a binary or a ternary relationship between entities, such as *better*(`Nikon D7000`, `Canon 7D`, `price`) and *positive*(`Canon 7D`, `sensor`).

Sentiment Relationship Graphs. A sentiment relationship graph (**SRG**) represents all sentiment-oriented relationships conveyed in a sentence, as illustrated in Figure 5.1. Formally, it is a directed hypergraph $R = \langle V, A \rangle$, where each vertex $v \in V$ denotes an entity and an edge $e \in A$ denotes a relationship between entities. A binary relationship in an SRG is in the form of an ordinary binary edge pointing to an attribute. A ternary comparative relationship is denoted by a special T-shaped directed hyperedge linking two products/brands in comparison and an attribute, where the direction between entities indicates the preference, e.g. *better*(Nikon D7000, Canon 7D, *price*) in Figure 5.3.

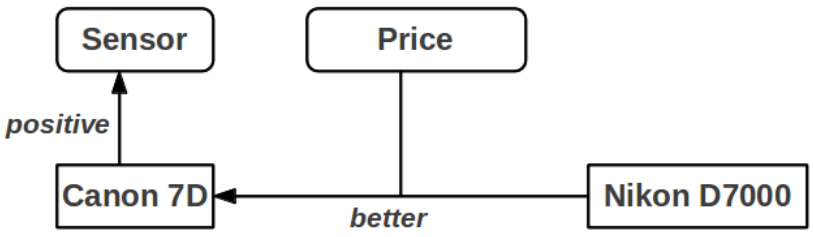


Figure 5.1: The sentiment relationship graph of the sentence *"The sensor is great, but the price is higher than Nikon D7000."* about Canon 7D.

Mention-Based Relationships. A mention-based relationship refers to a tuple

of entity mentions with a certain relation. Such kind of relationships are introduced as the representations of relationships in a sentence by replacing entities with the corresponding entity mentions. For example, *positive*("Canon SD880i", "wide angle view") is the mention-based relationship of the relationship *positive*(`Canon SD880i`, `lens`) in Example 5.1.

Relationship Types. A relationship type denotes a set of relationships sharing the same characteristics between entities. For the mention-based relationships converted from sentiment oriented relationships, we consider the following types:

- *positve*: A relationship of this type refers to a positive sentiment about an entity and its attribute.
- *negative*: An instance of this type is a negative sentiment about an entity and its attribute.
- *better*: A relationship of this type indicates a preference between two entities with respect to an attribute.
- *worse*: This type is introduced to indicate the opposite direction of *better*.

For any binary relationships of interest between two entities, which are not sentiment-oriented relationships, we denote them with the type *other*. Consequently, all relationships of interest are of one of the five relationship types: *positive, negative, better, worse* and *other*.

Mention-Based Relationship Graphs. A mention-based relationship graph (or **MRG**) represents a collection of mention-based relationships expressed in a sentence. As illustrated in Figure 5.2, an MRG is a directed hypergraph $G = \langle M, E \rangle$ with a vertex set M and an edge set E. A vertex $m_i \in M$ denotes an entity mention occurring either within the sentence or in its context. We say that a mention is from the context if it is mentioned in the previous sentence or is an attribute implied in the current sentence. An edge $e_l \in E$ is either binary or ternary. A binary edge e_l is denoted by a tuple $e = (m_i, m_j)$ and a label l, where m_i and m_j are entity mentions. If both mentions m_i and m_j are product mentions, $l \in \{better, worse, other\}$. Otherwise, if one of the two mentions refers to an attribute, the edge is assigned a label $l \in \{positive, negative, other\}$ instead. A ternary edge e_l includes a tuple $e = (m_i, m_j, m_a)$ and a label l. Herein, two

product mentions m_i and m_j are compared with respect to the attribute mention m_a, and the relationship type $l \in \{better, worse\}$. Since the comparative relationships are directional, we assume m_i occurs before m_j.

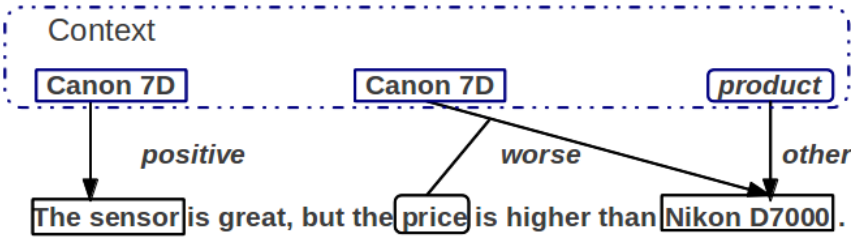


Figure 5.2: A mention-based relationship graph.

Textual Evidences. A textual evidence is a word, a phrase or a sentence conveying a relationship of interest. It is regarded as the support of a particular relationship assertion.

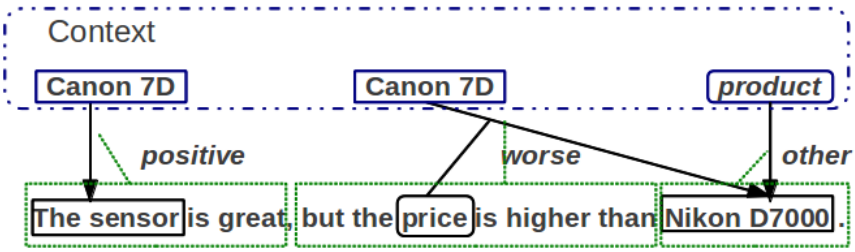


Figure 5.3: An evidentiary mention-based relationship graph. The textual evidences are wrapped by green dashed boxes.

Evidentiary Mention-Based Relationship Graphs. An *evidentiary* mention-based relationship graph, coined **EMRG** , extends an MRG by associating each edge with a textual evidence to support the corresponding relationship assertions (see Figure 5.3). Formally, an EMRG is a tuple (G, C, r), where G is an MRG, C is the textual evidence set of G and the function $r : E \rightarrow C$ maps an edge of G to its textual evidence. Consequently, an edge in an EMRG is denoted by a pair (e_l, c), where e_l represents a mention-based relationship and c is the associated textual evidence.

5.2.2 System Architecture

As illustrated by Figure 5.4, at the core of our system is the SENTI-LSSVM$_{RAE}$ model, which extracts sets of mention-based relationships in the form of EMRGs from sentences. For a given sentence, the model starts with constructing a latent feature vector for each constituent as well as the whole sentence. Section 5.4.2 will show that the latent feature vectors are created in a recursive way by using RAE so that each of these vectors corresponds to a vertex in a binary tree. As entity mentions are given, we select all possible mention sets as *relationship candidates*, where each set includes at least one product mention. Then we associate each relationship candidate with a set of constituents or the whole sentence as the textual evidence candidates and the corresponding latent feature vectors are obtained from the built representation tree (cf. Section 5.5.1). Subsequently, the inference component aims to find the most likely emrg from all possible combinations of mention-based relationships and their textual evidences (cf. Section 5.5). The representation form EMRG is chosen because it characterizes exactly the model outputs by letting each edge correspond to a mention-based relationship and the associated textual evidence. Finally, the model parameters of RAE and the inference module are learned by an online algorithm, whose details are given in Section 5.6.

The same set of sentiment-oriented relationships can be represented by different EMRGs due to the correference of entity mentions and the existence of paraphrases for conveying the same relationship. Therefore, we need to transform entity mention-based EMRGs into entity-based SRGs as the final outputs, which is carried out by using Algorithm 2. The algorithm essentially maps the mentions from an EMRG into entities in an SRG and converts the mention-to-mention edges from an EMRG into entity-to-entity edges in the corresponding SRG. Only two exceptional cases require special attention: i) the binary edges indicating direct comparisons between two products are normalized by augmenting with the implied attribute *product* (line 5-7); ii) a comparative relationship labeled with *worse* is rewritten as an equivalent relationship labeled with *better*, which requires reversing the order of the products in comparison (line 19-22).

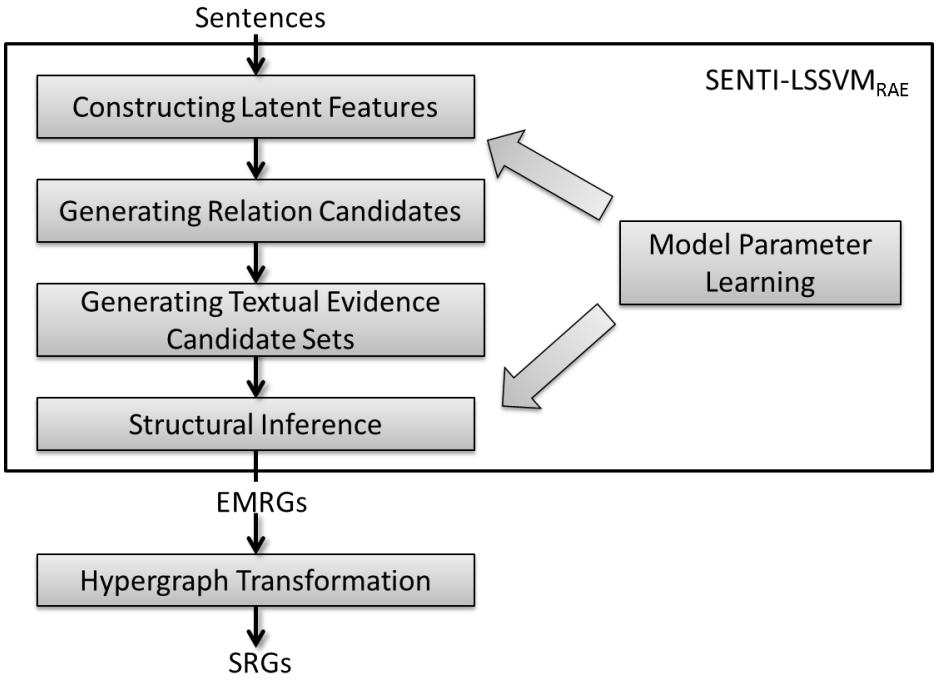


Figure 5.4: System architecture.

Algorithm 2 Graph Transformation Algorithm

Input: an EMRG $G = \langle M, E \rangle$
Output: an SRG $R = \langle V, A \rangle$
$V = \emptyset$ and $A = \emptyset$
for $e_l \in E$ **do**
  if $e = (m_i, m_j)$ and $l \in \{better, worse\}$ **then**
    $e = (m_i, m_j, \mathtt{product})$
  end if
  $\forall m_k$ in e, map entities in m_k to a set $V_k \subseteq V$
  if $e = (m_i, m_j)$ **then**
    {Normalizing sentiments.}
    $S = V_i \times V_j$
    if m_i is an attribute mention **then**
      $S = V_j \times V_i$
    end if
    for $a = (v_i, v_j) \in S$ **do**
      $a_l \rightarrow A$
    end for
  else
    {Normalizing comparative relationships.}
    $S = V_i \times V_j \times V_k$
    if $l = worse$ **then**
      $S = V_j \times V_i \times V_k$
      $l = better$
    end if

5.3 SENTI-LSSVM$_{RAE}$ Model

Given entity mentions and disambiguated entities, SENTI-LSSVM$_{RAE}$ aims to extract the most likely set of mention-based relationships from each sentence. After generating relationship and textual evidence candidates, this task can be conceptually divided into two subtasks : i) identifying the most likely set of mention-based relationships; ii) assigning a proper textual evidence to each relationship to support their relationship assertions. The latter is a special challenge of this task, ignoring this step will lead to overly similar feature representation for relationships with different types, as discussed in Section 5.1.1 and 5.1.3.

The prediction of both mention-based relationships and their textual evidences is equivalent to selecting the optimal EMRGs for sentences, since each edge in an EMRG is defined as a mention-based relationship associated with a textual evidence. In particular, the set of EMRGs are created by attaching every valid MRG with various assignments of textual evidences. It is also desirable to carry out the two subtasks *jointly* as these two subtasks could enhance each other. First, the identification of relationships requires proper textual evidences; second, the soft and hard constraints imposed by the correlated relationships facilitate the recognition of the relationships of the textual evidences. Because the assignment of textual evidences is not observed in both training and testing, it is treated as latent variables and thus *latent structural SVM* instead of structural SVM is employed as the core component of this model.

As discussed in Section 5.1.1, although we can already use bag-of-words to represent textual evidences, RAE is selected as the feature detectors to construct more expressive features for textual evidences in order to capture the underlying compositional semantics. Figure 5.5 illustrates the relation between latent structural SVM and RAE. RAE creates the latent feature vectors of textual evidences for latent structural SVM, whereas in training, latent structural SVM backpropagate errors based on supervised information to optimize the parameters of RAE as well as the latent feature vectors.

Formally, let $\mathcal{X}$ denote the set of all available sentences, and we define $y \in \mathcal{Y}(x)$ as the set of labeled edges of an MRG for a sentence $x \in \mathcal{X}$ and $\mathcal{Y} = \cup_{x \in \mathcal{X}} \mathcal{Y}(x)$.

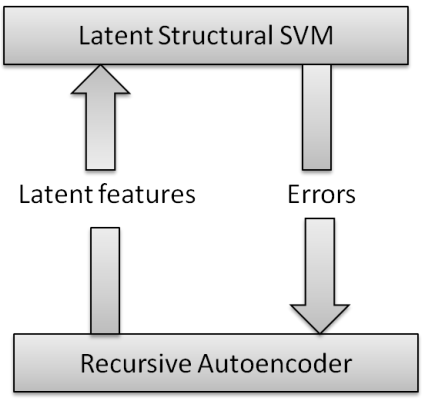


Figure 5.5: Dependencies between latent structural SVM and RAE.

The corresponding EMRG is denoted by (y, h), where $h \in \mathcal{H}(x)$ is a latent variable indicating a mapping from edges in y to their textual evidences, and $\mathcal{H} = \cup_{x \in \mathcal{X}} \mathcal{H}(x)$. Then $(a, c) \in (y, h)$ denotes an edge a attached with a textual evidence c. Given a labeled dataset $D = \{(x_1, y_1), ..., (x_n, y_n)\} \in (\mathcal{X} \times \mathcal{Y})^n$, latent structural SVM aims to learn a discriminant function $F : \mathcal{X} \times \mathcal{Y} \times \mathcal{H} \rightarrow \mathbb{R}$ of the form

$$F(x, y, h) = \boldsymbol{\beta}^\top \Phi(x, y, h) \tag{5.3}$$

where $\Phi(x, y, h)$ is the feature function of an EMRG and $\boldsymbol{\beta}$ is the corresponding weight vector. Section 5.5 will show that $\Phi(x, y, h)$ includes both the conventional non-latent feature vectors and the latent features in the form of composite functions. As the outputs of $F(x, y, h)$ are the scores reflecting the probabilities of EMRGs, the prediction is derived by maximizing F over all possible EMRGs, which expands the label space by the latent space for textual evidence assignment.

$$f(x) = \text{argmax}_{(y,h) \in \mathcal{Y}(x) \times \mathcal{H}(x)} \boldsymbol{\beta}^\top \Phi(x, y, h) \tag{5.4}$$

Note that, the main difference to structural SVM is the inclusion of latent variables, which enables the selection of the most likely textual evidences for respective relationships. Given $\boldsymbol{\beta}$ and the parameters of RAE, the computation of $f(x)$ is referred to as the inference problem, which will be covered in Section 5.5. And the details about learning model parameters are given in Section 5.6.

To ensure tractability, we employ edge-based factorization for our model. Let M_p denote the set of product mentions and $y_r(m_i)$ be a set of edges labeled with

sentiment-oriented relationships incident to m_i, the factorization of $\Phi(x, y, h)$ is given as

$$\Phi(x, y, h) = \sum_{(a,c)\in(y,h)} \Phi_e(x, a, c) + \sum_{m_i\in M_p} \sum_{a,a'\in y_r(m_i),a\neq a'} \Phi_c(a, a') \quad (5.5)$$

where $\Phi_e(x, a, c)$ is a local edge feature function characterizing a labeled edge a attached with a textual evidence c and $\Phi_c(a, a')$ is a feature function capturing co-occurrence of two labeled edges a_{m_i} and a'_{m_i} incident to a product mention m_i. In the following section, we give more details about these feature functions.

5.4 Feature Space

Due to the manner in which we factorize the model, we specify the features at the edge level, which are the realization of the local edge function $\Phi_e(x, a, c)$ and the edge co-occurrence function $\Phi_c(a, a')$ in Eq.(5.5). More specifically, $\Phi_e(x, a, c)$ denotes the conjunction of both the non-latent features of relationship candidates and the latent features of textual evidences, whose details are given below. In that sense, $\Phi_c(a, a')$ is viewed as the non-latent ones characterizing pairs of edges.

5.4.1 Non-latent Features

This section summarizes the following non-latent features characterizing edges in EMRGs. For each edge, the values of non-latent features are stored in a feature vector as part of $\Phi_e(x, a, c)$.

POS Tags. As mentioned before, a textual evidence could be either a word, phrase or sentence. We consider all POS tags in the textual evidence as lexical features.

Context. Since users often express related sentiments about the same product across sentence boundaries, we describe the sentiment flow using a set of contextual binary features. For example, if product A concerned with a relationship candidate is mentioned in the previous sentence, the contextual

binary features indicate that the mentioned sentiment-oriented relationships in this sentence are also related to product A.

Co-occurrence. We have mentioned the co-occurrence feature in Equation 5.5, indicated by $\Phi_c(a, a')$. It captures the co-occurrence of two labeled edges incident to the same product mention. Note that the co-occurrence feature function is considered only if there is a contrast conjunction such as *"but"* between the non-shared entity mentions incident to the two labeled edges.

Senti-predictors. Following the idea of (Qu et al., 2012), we encode the prediction results from the rule-based multi-relationship predictor (Ding et al., 2009) and the bag-of-opinions predictor (cf. Section 4.2.3) as features based on the textual evidence. The former predictor uses rules and subjectivity lexicons to classify phrases into one of the four categorizies: "positive", "negative", "better" or "worse", which are encoded as four boolean features. The output of the latter predictor is a numerical scores capturing both sentiment polarity and intensity. We use two numerical features to represent the positive and negative scores respectively.

Edge type. For an edge candidate, a set of binary features are used to denote the types of the edge and its entity mentions. For instance, a binary feature indicates whether an edge is a binary edge related to an entity mentioned in context.

Syntactic path and word distance measures. To characterize the syntactic dependencies between two adjacent entity mentions, we use the path through the dependency tree between the heads of the corresponding constituents, the number of words and other mentions in-between as features. Additionally, if the textual evidence is a constituent, its feature with respect to an edge is the dependency path to the closest mention of the edge that is not overlapped with this constituent.

5.4.2 Constructing Latent Features from Text

In this section we show how to construct latent features for text evidence to capture the compositionality of expressions.

We observed that the high volume of sentiment-bearing expressions are often

composed by using a much smaller set of composition rules, such as negating an expression conveying positive sentiment leads to an expression carrying negative sentiment. Most of these rules require that the related words and phrases are represented at the proper abstraction level, such as indicating the polarity of an expression. Instead of using explicit hand-crafted rules, which often lack in coverage, we apply RAE to automatically detect the corresponding compositional patterns, and learn the representations of constituents and sentences *jointly*.

As discussed in Section 2.3.2, deep learning techniques such as RAE are well suited for learning abstract feature representations. Therefore, we choose **distributed representation** (Hinton, 1986) as the latent features for each word, phrase and sentence. In our work, a distributed representation takes the form of a continuous-valued vector $\mathbf{v}$ with $\mathbf{v} \in \mathcal{R}^k$. It is called distributed because a piece of information such as "*A word X is polar*." can be encoded jointly by a set of potentially dependent components from the vector. Distributed representation can be compact in the sense that it can use a low dimensional vector to represent the same information of a conventional high dimensional feature vector because it uses different subsets of features to indicate different information. Compared to conventional hand-crafted feature vectors, another property of distributed representation is that the representation vectors can be learned from unlabeled data for prior knowledge of the data and can be fine tuned by labeled data for application dependent tasks. The latter is a property that is not featured by the widely used topic models (Blei et al., 2003; Hofmann, 1999). Without this property, a topic model can only learn topic-oriented representations insensitive to word order and sentiments. Then it is very likely that "very good" and "very bad" share similar representations since they share globally similar context words.

In the following, we show how to construct representations for words, phrases and sentences.

Word Representation

Our word representation consists of a distributed representation and a conventional feature vector. The distributed representations can be initialized with unlabeled data to encode distributional similarity. In our experimental settings, they are 100-dimensional continuous-valued vectors, where the first 50-dimensions are initialized by the word vectors from (Collobert et al., 2011) by using the text of the english Wikipedia, the last 50-dimensions are randomly initialized to increase model capability. These word vectors, pre-trained with the Wikipedia text, can capture coarse-grained syntactic and semantic information because they are trained with a special neural language model capable of encoding distributional similarity. As a result, a supervised model trained on these word vectors can be generalized to unseen words if there are distributionally similar words in the training data.

In the conventional feature vector, we store POS and polarity scores from the SO-CAL lexicon (Taboada et al., 2011). In addition, if a word is part of an entity mention, a binary feature is used to indicate the type of the entity. In the movie domain, if the names of a director, actor or role are not manually annotated due to coreference, we use the jaccard similarity between the entity name and the longest matched surface string containing the word as the value of the entity type feature.

Composition Model for Two Words

Once the word representations are initialized, we can construct a new representation representing two consecutive words by using the RAE.

RAE consists of an encoder and a decoder functions. The encoder function takes the vector representations $\mathbf{v_i}, \mathbf{v_j} \in \mathbf{R}^\kappa$ of two words i and j as input. The corresponding function is defined as

$$g_e(\mathbf{v}_i, \mathbf{v}_j) = a\Big(\mathbf{W} \begin{bmatrix} \mathbf{v}_i \\ \mathbf{v}_j \end{bmatrix} + \mathbf{b}\Big) \tag{5.6}$$

where $\mathbf{W} \in \mathbb{R}^{\kappa \times 2\kappa}$ is the weight matrix for the input vectors and $\mathbf{b} \in \mathbb{R}^\kappa$ is the bias vector. $a(\mathbf{u})$ is ususally a nonlinear activation function, which introduces

nonlinearity into the model. In our experiments, we employ the following hyperbolic tangent function to make it compatible to the word vectors learned with the neural language model (Collobert et al., 2011).

$$a(u) = \frac{e^u - e^{-u}}{e^u + e^{-u}}$$

To measure how well the output $\bar{\mathbf{v}}$ of the encoder function represents the original inputs, a decoder function is used to reconstruct the original inputs from $\hat{\mathbf{v}}$.

$$g_d(\bar{\mathbf{v}}) = a(\mathbf{W}^\top \hat{\mathbf{v}} + \tilde{\mathbf{b}}) \tag{5.7}$$

where $\tilde{\mathbf{b}} \in \mathbb{R}^{2\kappa}$ is the bias vector. Then the quality of the representation $\bar{\mathbf{v}}$ is measured by the Euclidean distance between the original input and its reconstruction. Let $(\hat{\mathbf{v}}_i, \hat{\mathbf{v}}_j) = g_d(g_e(\mathbf{v}_i, \mathbf{v}_j))$, the reconstruction error function is defined as

$$\Delta(\hat{\mathbf{v}}_i, \hat{\mathbf{v}}_j, \mathbf{v}_i, \mathbf{v}_j) = \frac{1}{2}\left[\frac{\omega_i}{n_i + n_j}\|\hat{\mathbf{v}}_i - \mathbf{v}_i\|^2 + \frac{\omega_j}{n_i + n_j}\|\hat{\mathbf{v}}_j - \mathbf{v}_j\|^2\right] + \gamma_1|\theta_{rae}| + \gamma_2|\mathbf{v}| \tag{5.8}$$

where ω_k is a weight characterizing informativeness of the corresponding input vectors, which will be referred to as I-weight later for ease of discussion; n_k is the number of words in the corresponding constituent; γ_1 and γ_2 are the hyperparameters for the L1 regularization on the model parameters $\theta_{rae} = (\mathbf{W}, \mathbf{b}, \tilde{\mathbf{b}})$ and $\mathbf{v} = \begin{bmatrix} \mathbf{v}_i \\ \mathbf{v}_j \end{bmatrix}$ respectively.

I-weights replace the numerators of the weights $\frac{n_i}{n_i+n_j}$ in Eq.(2.13), which are soly based on the number of words. We found that $\frac{\omega_i}{n_i+n_j}$ works better than $\frac{\omega_i}{\omega_i+\omega_j}$ in our preliminary experiments because $\frac{\omega_i}{n_i+n_j}$ not only provides a way of normalization but also indicates the average informativeness of all words in a constituent. For sentiment analysis, a good choice for a word-level I-weight is a combination of IDF and sentiment word scores from a subjectivity lexicon (Qu et al., 2012). For IDF, we use $\log(\frac{N}{N_i})/\log(N)$ (Cancedda et al., 2003), where N_i is the document frequency of the word i and N is the total number of documents. For sentiment words, we normalize the word scores from SO-CAL lexicon into $[0, 1]$ by $|\iota_i|/5$, where ι_i is the score of word i ranging from -5 to 5. Then the I-weight of the word i is defined as

$$\omega_i = \max(\log(\frac{N}{N_w})/\log(N), |\iota_i|/5)$$

To show that RAE can be regarded as a feature detector that can detect factors of variations, we rewrite the encoder function as $g_e(\mathbf{v}_i, \mathbf{v}_j) = a(\mathbf{u})$, where

$$\mathbf{u} = \mathbf{W} \begin{bmatrix} \mathbf{v}_i \\ \mathbf{v}_j \end{bmatrix} + \mathbf{b}$$

Then each u_m shares the same form as the factors produced by principle component analysis (Jolliffe, 1986). If there are correlated features scattered in some input vectors and the weight matrix is properly trained, we can represent a group of correlated features by one or few factors, which are easier to learn by using a linear classifier than the original features. In addition, we expect that each factor u_m captures different aspects of the underlying text, thus it is desirable to make $\mathbf{W}$ sparse so that different u_m encode different subsets of input units.

Building Representations for Phrases and Sentences

Considering the output of the encoder function represents the composited representation of input vectors, we can apply the encoder function of RAE *recursively* to construct a binary represenation tree (e.g. Figure 5.6) in a bottom-up manner. The order of composition is guided by constituent parse trees produced by the Stanford parser (Klein and Manning, 2003) so that each constituent are mapped to a vertex in the corresponding representation tree. More precisely, the construction of a representation tree starts with word representations. In each step, RAE creates a new parent vertex representing two child constituents by merging their representations using the encoder function. As the representation of the new parent vertex is now available for building representations of longer word sequences, the construction step is repeated until it reaches the root of the tree, which represents the whole sentence. If a constituent has more than two children, we merge the children from right to left so that the syntactic heads are merged into the tree often before their dependents. Moreover, for each phrase, its I-weight is the sum of the I-weights of all its children.

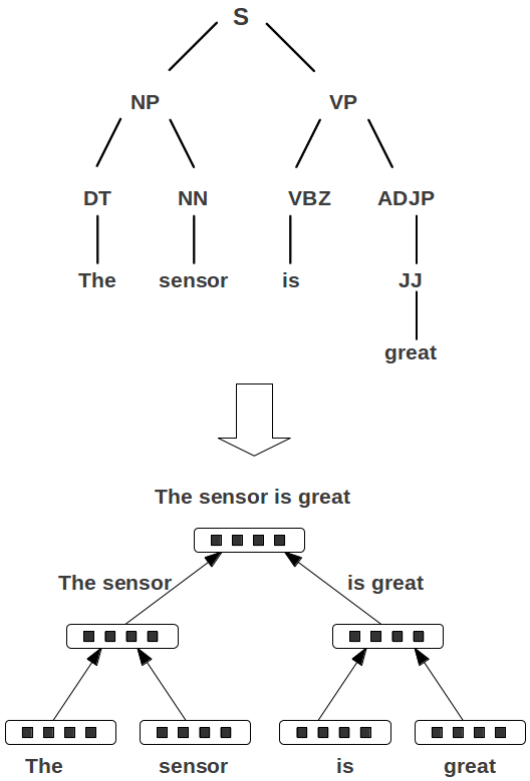


Figure 5.6: Building representation tree by using recursive autoencoder

5.5 Structural Inference

This section presents the ILP formulation to infer the most likely EMRG for a sentence. As EMRGs are generated out of relationship candidates and the candidate sets for textual evidence, we give first details about selecting candidates of textual evidence from the constructed representation trees.

5.5.1 Generating Candidate Sets for Textual Evidence

Textual evidences are selected based on the binary representation trees and entity mentions. For each mention in a sentence, we first locate a vertex in the tree with the maximal overlap by Jaccard similarity. If a set of entity mentions are linked by the conjunction "and" or "or", we merge them into a super vertex. Starting from this vertex, we consider two types of candidates: *type I* candidates are vertices at the highest level which contain neither any word of

another mention nor any contrast conjunctions such as *"but"*; *type II* candidates are vertices at the highest level which cover exactly two mentions of an edge and do not overlap with any other mentions. For a binary edge connecting a product mention and an attribute mention, we consider a *type I* candidate starting from the attribute mention. For a binary edge connecting two product mentions, we consider *type I* candidates starting from both mentions. Moreover, for a comparative ternary edge, we consider both *type I* and *type II* candidates starting from the attribute mention. Such strategy is based on our observation that these candidates often cover the most important information with respect to the covered entity mentions. Since one candidate usually conveys only one relationship, it is possible to coarsely segment a sentence based on potential relationships.

5.5.2 ILP Formulation

We formulate the inference problem of finding the most likely EMRG as an ILP problem due to its convenient integration of both soft and hard constraints.

Linear Programming (LP) is a technique for optimizing a linear objective function subject to a set of linear constraints, which takes the following form

$$\begin{aligned} \max_{\mathbf{z}\in\mathbb{R}^d} \quad & \mathbf{s}^\top \mathbf{z} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{z} \leq \mathbf{d} \end{aligned}$$

where $\mathbf{z}$ represents the vector of variables, $\mathbf{s}$ is a vector of coefficients, $\mathbf{A}$ and $\mathbf{d}$ are a matrix and a vector for constraints respectively. The function to be maximized is referred to as the *objective function*. ILP is an extension of LP where $\mathbf{z}$ must take integer values.

As shown in Section 5.3, the most likely EMRG is predicted by the function (5.4). From it we can derive the ILP program for inference. Since the function to maximize is the discriminant function (5.3), we obtain the objective function of

the ILP program as follows.

$$\begin{aligned}\boldsymbol{\beta}^\top \Phi(x, y, h) &= \sum_{(a,c)\in(y,h)} \boldsymbol{\beta}^\top \Phi_e(x, a, c) + \sum_{m_i \in M_p} \sum_{a,a' \in y_r(m_i), a \neq a'} \boldsymbol{\beta}^\top \Phi_c(a, a') \\ &= \sum_{(a,c)\in(y,h)} s_{ac} z_{ac} + \sum_{m_i \in M_p} \sum_{a,a' \in y_r(m_i), a \neq a'} s_{aa'} z_{aa'} \\ &= \mathbf{s}^\top \mathbf{z}\end{aligned}$$

where $s_{ac} = \boldsymbol{\beta}^\top \Phi_e(x, a, c)$ denotes the score of a labeled edge a attached with a textual evidence c; $s_{aa'} = \boldsymbol{\beta}^\top \Phi_c(a, a')$ is the edge co-occurrence score. As $\Phi_e(x, a, c)$ represents a mention-based relationship attached with a textual evidence, the binary variables z_{ac} are introduced to indicate the presence or absence of the corresponding relationships. The same for $\Phi_c(a, a')$, $z_{aa'}$ denotes if two mention-based relationships co-occurr in the same sentence. As a result, different assignment of these binary variables leads to different selection of EMRGs.

Since not every relationship candidate set can constitute an EMRG, we define a valid EMRG by introducing a set of linear constraints, which form our *constraint space*. Consequently, the function (5.4) is equivalent to

$$\begin{aligned}\max_{\mathbf{z}\in\mathbb{B}} \quad & \mathbf{s}^\top \mathbf{z} \\ \text{s.t.} \quad & \mathbf{A} \begin{bmatrix} \mathbf{z} \\ \boldsymbol{\eta} \\ \boldsymbol{\tau} \end{bmatrix} \leq \mathbf{d} \\ & \mathbf{z}, \boldsymbol{\eta}, \boldsymbol{\tau} \in \mathbb{B}\end{aligned}$$

where $\mathbb{B} = 2^S$ with $S = \{0, 1\}$; $\boldsymbol{\eta}$ and $\boldsymbol{\tau}$ are auxiliary binary variables that help defining the constraint space.

In the following, we consider two types of constraint space, 1) an EMRG with only binary edges and 2) an EMRG with both binary and ternary edges.

EMRG with only Binary Edges An EMRG has only binary edges if a sentence contains no attribute mention or at most one product mention.

Before describing the constraint space in detail, we introduce first the constraints for conjunction and disjunction of binary variables, which will be used repeatly

in this section. Let $z^c_{a_1,...,a_K} = z_{a_1} \wedge ... \wedge z_{a_K}$, the logical relation is ensured by a set of agreement constraints (Martins et al., 2009; Nemhauser and Wolsey, 1988)

$$\begin{aligned} z^c_{a_1,...,a_K} &\leq z_{a_i}, i = 1, ..., K \\ z^c_{a_1,...,a_K} &\geq \sum_{i=1}^{K} z_{a_i} - K + 1 \end{aligned}$$

Then the aggreement constraints for $z^d_{a_1,...,a_K} = z_{a_1} \vee ... \vee z_{a_K}$ are expressed as:

$$\begin{aligned} z^d_{a_1,...,a_K} &\geq z_{a_i}, i = 1, ..., K \\ z^d_{a_1,...,a_K} &\leq \sum_{i=1}^{K} z_{a_i} \end{aligned}$$

We expect that each mention-based relationship has only one relationship label and is supported by a single textual evidence. To facilitate the formulation of constraints, we introduce η_{e_l} to denote the presence or absence of a mention-based relationship e_l, and η_{ec} to indicate if a textual evidence c is assigned to a relationship candidate e. It is trivial to see that $z_{e_lc} = \eta_{ec} \wedge \eta_{e_l}$, where z_{e_lc} denotes an edge in an EMRG.

Let C_e denote the candidate set for textual evidence of a relationship candidate e, the constraint of *at most one textual evidence per relationship candidate* is formulated as:

$$\sum_{c \in C_e} \eta_{ec} \leq 1 \tag{5.9}$$

Furthermore, we assume that a textual evidence c conveys at most one relationship so that an evidence will not be assigned to the relationships of different types, which is the main problem for the structural SVM based model. Let η_{cl} indicate that the textual evidence c is labeled by the relationship type l. The corresponding constraints are expressed as

$$\sum_{l \in L} \eta_{cl} \leq 1$$

where L is the set of all relationship types (cf. Section 5.2.1).

Once a textual evidence is assigned to a mention-based relationship, their relationship labels should match and the number of relationships must agree

with the number of attached textual evidences:

$$z_{e_l c} \leq \eta_{cl}; \qquad \sum_{l \in L_e} \eta_{e_l} = \sum_{c \in C_e} \eta_{ec}$$

where L_e denotes the set of all possible labels for a relationship candidate e.

Another kind of agreement must be made among the textual evidences overlapping each other. Since a textual evidence corresponds to a vertex in a binary representation tree, the overlapped textual evidences are the vertices on the same path in the tree. For a set of such vertices, we require that any two nearest ones are labeled by either the same relationship type or the type *other* unless there is a negator existing only in the higher level vertex. Thus the label agreement of a set of textual evidences can be decomposed into a set of pair-wise constraints. In particular, for two vertices c and c', which are the nearest neighbours in a path and there is no negation involved between them, we consider the following constraint for each *non-other* relationship type l:

$$\eta_{cl} + \eta_{co} = \eta_{c'l} + \eta_{c'o}$$

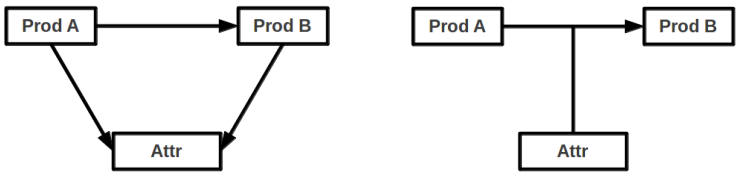
where o denotes the relationship type *other*.

In order to avoid a textual evidence being overly reused by multiple relationships, we first penalize the assignment of a textual evidence c to a relationship candidate e by associating the corresponding z_{ec} with a fixed negative cost $-\mu$ in the objective function. Then selecting at least one textual evidence per relationship candidate e is encouraged by associating μ to z_e^d in the objective function, where $z_e^d = \bigvee_{e \in S_c} \eta_{ec}$ and S_c is the set of relationship candidates that have the textual evidence c as a candidate. This soft constraint encourages an injective mapping from a textual evidence to a relationship, but also keeps it open for many-to-one mappings.

For any two mention-based relationships a and a' incident to the same product mention, the relationship co-occurrence is described by $z^c_{a,a'} = z_a \wedge z_{a'}$.

EMRG with both Binary and Ternary Edges If there are more than one product mentions and at least one attribute mention in a sentence, an EMRG can potentially have both binary and ternary edges. In this case, we assume

that a mention of attributes can participate either in binary relationships or in ternary relationships. The assumption is held in more than 99.9% of the sentences in our SRG corpus (cf. Section 5.7), thus we describe it as a set of hard constraints. Geometrically, the assumption can be visualized as the selection between two alternative structures incident to the same attribute mention, as shown in Figure 5.7. Note that, in the binary edge structure, we include not only the edges incident to the attribute mention but also the edge between the two product mentions.



(a) Binary edges structure (b) Ternary edges structure

Figure 5.7: Alternative structures associating to an attribute mention.

Let $S^b_{m_i}$ be the set of all possible labeled edges in a binary edge structure of an attribute mention m_i. Variable $\tau^b_{m_i} = \bigvee_{e_l \in S^b_{m_i}} \eta_{e_l}$ indicates whether the attribute mention is associated with a binary edge structure or not. In the same manner, we use $\tau^t_{m_i} = \bigvee_{e_l \in S^t_{m_i}} \eta_{e_l}$ to indicate the association of the an attribute mention m_i with an ternary edge structure from the set of all incident ternary edges $S^t_{m_i}$. The selection between two alternative structures is formulated as

$$\tau^b_{m_i} + \tau^t_{m_i} = 1$$

As this influences only the edges incident to an attribute mention, we keep all the constraints introduced in the previous section unchanged except for constraint (5.9), which is modified as,

$$\sum_{c \in C_e} \eta_{ec} \leq \tau^b_{m_i}; \qquad \sum_{c \in C_e} \eta_{ec} \leq \tau^t_{m_i}$$

Therefore, we can have either binary edges or ternary edges for an attribute mention.

5.6 Learning Model Parameters

In the previous section, we assume that the model parameters are given. In this section, we present the online learning algorithm for SENTI-LSSVM$_{\text{RAE}}$. In order to understand the algorithm, we need to view the model from a different angle. If the latent structural SVM is treated as the top-level classifier and the RAE is considered as the underlying hidden layers, SENTI-LSSVM$_{\text{RAE}}$ is a neural network. Thus the discriminant function (5.3) is a composite function in the following form

$$f(\cdot) = f^K(f^{K-1}(...f^1(\cdot)...)) \tag{5.10}$$

where $f^K(\cdot)$ corresponds to the latent structural SVM and the functions for lower layers indicate the composition of children vectors for each parent vertex in a representation tree. As the number of composition is proportional to the number of words in a sentence, the neural network has a deep architecture, in which all hidden layers share the same parameters.

As SENTI-LSSVM$_{\text{RAE}}$ can be viewed as a deep neural network, we can apply the corresponding deep learning algorithms to learn the model (cf. Section 2.3.2). Given a set of training sentences $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, we apply the unsupervised learning step and the supervised fine-tuning step interchangeably to learn the model parameters $\theta = (\boldsymbol{\beta}, \theta_{\text{rae}}, V)$, where $\boldsymbol{\beta}$ is the weight vector in Eq.(5.3), $\theta_{\text{rae}} = (\mathbf{W}, \mathbf{b}, \tilde{\mathbf{b}})$ denotes the parameters of RAE and V is the set of latent feature vectors for all words. For a training sentence, the unsupervised learning step is carried out by minimizing the reconstruction errors (5.8) after creating each new vertex in the representation tree. To gain robustness against noise, we proactively corrupt the distributed representation of each leaf by randomly omitting 20% units, which can be regarded as applying the denoising autoencoder (Vincent et al., 2008). In the supervised fine-tuning step, we aim to solve the following optimization problem:

$$\theta^* = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} [\max_{(\hat{y},\hat{h})\in\mathcal{Y}(x)\times\mathcal{H}(x)} (\boldsymbol{\beta}^\top \Phi(x, \hat{y}, \hat{h}) + \delta(\hat{y}, \hat{h}, y)) - \max_{\bar{h}\in\mathcal{H}(x)} \boldsymbol{\beta}^\top \Phi(x, y, \bar{h}) + \gamma_1|\theta_{\text{rae}}| + \sum_{\mathbf{v}\in V} \gamma_2|\mathbf{v}| + \gamma_3|\boldsymbol{\beta}|] \tag{5.11}$$

where $\delta(\hat{h}, \hat{y}, y)$ is a *loss function* measuring the discrepancies between an

EMRG $(\bar{h}, y)$ with gold standard edge labels y and an EMRG $(\hat{h}, \hat{y})$ with inferred labeled edges $\hat{y}$ and textual evidences $\hat{h}$. Since we expect sparse parameterization of RAE (cf. Section 5.4.2) and certain sparse non-latent features such as dependency paths, we apply L1 regularizers to θ_{rae}, the word vectors and the weight vector $\boldsymbol{\beta}$, where the degree of sparsity is controlled by the hyperparameter γ_1, γ_2 and γ_3 respectively.

Because both training criteria involve non-differentiable L1 norms, we apply the online *forward-backward splitting* (FOBOS) algorithm (Duchi and Singer, 2009a), which can be viewed as a combination of the stochastic gradient descent (SGD) (Bottou, 2003) and the projected subgradient method (Calamai and Moré, 1987). In particular, the online FOBOS is applied in a mini-batch setting that it considers k training instances each time to compute gradients for updating the model parameters θ. On iteration t, two steps are required to update the parameters:

$$\theta_{t+\frac{1}{2}} = \theta_t - \varepsilon_t \nabla_t \tag{5.12}$$

$$\theta_{t+1} = \arg\min_{\theta} \frac{1}{2}\|\theta - \theta_t\|^2 + \varepsilon_t \gamma |\theta| \tag{5.13}$$

where ∇_t is the gradient computed for k instances without considering the L1 regularizers and ε_t is the learning rate. The step (5.12) takes exactly the same form as the weight updating formula of SGD and ∇_t is computed by using backpropagation (Rumelhart et al., 2002). Since the unsupervised and the supervised steps are taken interchangeably, the gradient are computed based on both training criterions. In addition, the projection step (5.13) performs L1 regularization and finds a sparse solution of model parameters. Then the two steps are repeated through the training data several times until the convergence condition is met.

Minimizing reconstruction errors requires only direct application of the FOBOS algorithm, whereas the supervised fine-tuning involves two inference problems. For a labeled sentence x, the gradient ∇_t of Eq.(5.11) in the step (5.12) takes the form

$$\nabla_t = \frac{\partial \boldsymbol{\beta}^\top \Phi(x, \hat{y}^*, \hat{h}^*)}{\partial \theta} - \frac{\partial \boldsymbol{\beta}^\top \Phi(x, y, \bar{h}^*)}{\partial \theta}$$

where the feature functions of the corresponding EMRGs are inferred by solving

$$(\hat{y}^*, \hat{h}^*) = \underset{(\hat{h},\hat{y})\in\mathcal{H}(x)\times\mathcal{Y}(x)}{\arg\max} \; [\boldsymbol{\beta}^\top \Phi(x, \hat{y}, \hat{h}) + \delta(\hat{y}, \hat{h}, y)]$$

and

$$(y, \bar{h}^*) = \underset{\bar{h}\in\mathcal{H}(x)}{\arg\max}\, \boldsymbol{\beta}^\top \Phi(x, y, \bar{h})$$

as indicated in the optimization problem (5.11).

The former inference problem is similar to the one we considered in the previous section except the inclusion of the loss function. It in fact finds the most error-prone EMRG according to the loss between an EMRG (h, y) and a gold standard EMRG. Then the objective function of the ILP program becomes

$$\max_{\mathbf{z}\in\mathbb{B}} \quad \mathbf{s}^\top \mathbf{z} + \delta(\hat{h}, \hat{y}, y)$$

and the loss is the sum of per-relationship costs for the sake of easy computation.

$$\delta(\hat{h}, \hat{y}, y) = \sum_{e\in E'} \varphi_e z_e$$

where E' is the set of all candidates of mention-based relationship and φ_e is the misclassification cost of the corresponding relationship e. Since we aim to maximize the F-Measure of SRG, which involves edges with non-other relationship types, the errors by classifying *other* as non-other relationship types should be weighted smaller than misclassifying non-other relationship types. Therefore, φ_e could be one of the two costs φ_{fp} and φ_{fn}, which are fixed for misclassifying *other* and non-other relationship types respectively.

In addition, since the non-positive weights of relationship labels in the initial learning phase often lead to EMRGs with few edges, which results in too few error cases, we fix it by adding a constraint for the minimal number of edges in an EMRG,

$$\sum_{e\in A} \sum_{e\in C_e} \eta_{ec} \geq \zeta \tag{5.14}$$

where A is the set of all relationship candidates, C_e is the candidate set for textual evidence of the relationship e, and ζ denotes the lower bound of edges.

Empirically, we find the best way to determine ζ is to make it equal to the maximal number of edges in an EMRG with the restriction that a textual

evidence can be assigned to at most one relationship candidate. Hence we represent all the relationship candidates A and all the textual evidence candidates C as two vertex sets in a bipartite graph $\hat{G} = \langle V = (A, C), E \rangle$ (with edges in E indicating which textual evidence can be assigned to which relationship candidate). Then ζ corresponds to exactly the size of a maximum matching of the bipartite graph, which is computed by the Hopcroft-Karp algorithm (Hopcroft and Karp, 1973) in our implementation.

To find the optimal EMRG $(\bar{h}^*, y)$, we consider the following set of constraints for inference since the labels of the edges are known for the training data. For an edge e with the gold label k, we have

$$\sum_{c \in C_e} \eta_{ec} \leq 1; \qquad \eta_{ec} \leq l_{ck}$$

$$\sum_{\hat{k} \in L} l_{c\hat{k}} \leq 1; \qquad \sum_{e \in B_c} \eta_{ec} \leq 1$$

We include also the soft constraints to avoid a textual evidence being overly reused by multiple relationships. In addition, we found it useful to assume a minimal number of edges labeled with *non-other* relationship types by

$$\sum_{a \in A_r} \sum_{c \in C_a} \eta_{ac} \geq \zeta_r \tag{5.15}$$

where A_r is the set of all *non-other* relationships and ζ_r denotes the minimal number of such edges, computed in the same way as for the constraint (5.14).

5.7 Experimental Corpus

For evaluation we constructed the SRG corpus, which in total consists of 1686 manually annotated online reviews and forum posts in the digital camera and movie domains[1]. The numbers of sentences of each subset are summarized in Table 5.2. For each domain, we maintain a set of entity types and a list of product names for the sake of entity disambiguation.

[1]The 107 camera reviews are from *bestbuy.com* and *Amazon.com*; the 667 camera forum posts are downloaded from *forum.digitalcamerareview.com*; the 138 movie reviews and 774 forum posts are from *imdb.com* and *boards.ie* respectively

	Camera		Movie	
	Reviews	Forums	Reviews	Forums
on-topic sentences	488	1588	1629	1634
all sentences	590	3867	1880	3203

Table 5.2: Number of sentences in SRG corpus

The annotation scheme for the sentiment representation asserts minimal linguistic knowledge from our annotators. By focusing on the meanings of the sentences, the annotators make decisions based on their language intuition, not restricted by specific syntactic structures. As illustrated by Figure 5.8, the annotators only need to mark the underlined entity mentions and label the relationships, whereas in prior work, people have annotated the sentiment-bearing expressions such as "better" and link them to the respective relationships as well. This also enables them to annotate both sentiment polarities and comparisons, which are conveyed by not only explicit sentiment-bearing expressions like *"excellent performance"*, but also factual expressions implying evaluations such as *"The 7V has 10x optical zoom and the 9V has 16x."*

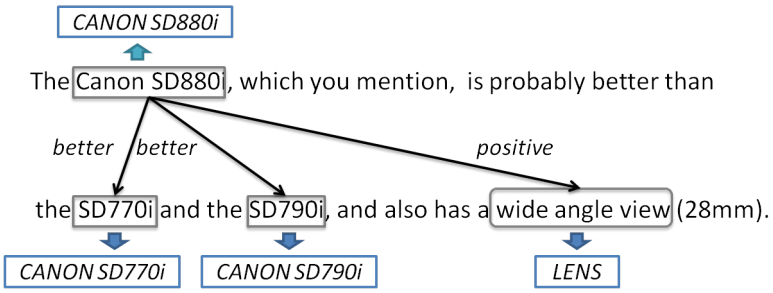


Figure 5.8: An annotation example.

14 annotators, all university students with various majors, participated in the annotation project. After a short training period, annotator work on randomly assigned documents one at a time. For reviews, the system lists all relevant information about the product and the predefined entity types. For forum posts, the system shows only the entity types. For each sentence in a document, the annotator first determines if it refers to a product of interest. If not, the sentence is marked as *off-topic*. Otherwise, the annotator will identify the most

obvious mentions of relevant entities and disambiguate them by selecting the product name or entity type provided by the system. Then they select one of the five possible relationships for each pair of semantically related mentions (see Table 5.3 for the corpus distribution). Camera forum posts contain the largest proportion of comparisons because the they are mainly about the recommendation of digital cameras. In contrast, web users are much less interested in comparing movies, in both reviews and forums. In all subsets, positive relationships play a dominant role since web users intend to express more positive attitudes online than negative ones.

	Camera		Movie	
	Reviews	Forums	Reviews	Forums
positive	386	1539	879	905
negative	165	363	529	331
comparison	30	480	39	35

Table 5.3: Distribution of relationships in SRG corpus

5.8 Experiments

This section describes the empirical evaluation of SENTI-LSSVM$_{RAE}$ together with two competitive baselines on the SRG corpus and found that Senti-LSSVM outperforms state-of-the-art approaches by a significant margin.

5.8.1 Experimental Setup

We implemented a rule-based baseline (RULE-BASED) and a structural SVM (Tsochantaridis et al., 2004) baseline (STRUCT-SVM) for comparison.

- **RULE-BASED** extends the work of Ding et al. (2009), which designed several linguistically-motivated rules based on a sentiment polarity lexicon for relationship identification and assumes there is only one type of sentiment relationship in a sentence. In our implementation, we keep all the rules of (Ding et al., 2009) and add one phrase-level rule when there

were more than one mention in a sentence. The additional rule assigns sentiment-bearing words and negators to its nearest relationship candidates based on the absolute surface distance between the words and the corresponding mentions. In this case, the phrase-level sentiments and comparisons depend only on the assigned sentiment words and negators.

- **STRUCT-SVM** is based on a structural SVM and does not consider the assignment of textual evidences to relationships. The textual features of a relationship candidate are all unigrams, POS and senti-predictor features within a surface distance of four words from the mentions of the candidate. Thus, it does not need the inference constraints of SENTI-LSSVM$_{\text{RAE}}$ for the selection of textual evidences.

In addition, we also evaluate the variant of SENTI-LSSVM$_{\text{RAE}}$, coined SENTI-LSSVM$_{\text{BOW}}$, which replace the latent feature vectors by the bag of words within the span of textual evidences.

For each domain and text genre, we withheld 15% documents for development and use the remaining for evaluation. The hyperparameters of all systems are tuned on the development datasets. For all experiments of SENTI-LSSVM$_{\text{BOW}}$, we use $\gamma_3 = 0.0001$ for the L1 regularizer in Eq.(5.11) and $\varphi_{\text{fp}} = \varphi_{\text{fn}} = 0.05$ for the loss function; for STRUCT-SVM, $\gamma_3 = 0.0001$ and $\varphi_{\text{fp}} = \varphi_{\text{fn}} = 0.01$; and for SENTI-LSSVM$_{\text{RAE}}$, $\gamma_1 = 0.3$, $\gamma_2 = 0.01$, $\gamma_3 = 0.01$, $\varphi_{\text{fp}} = 0.01$ and $\varphi_{\text{fn}} = 0.1$. We let $\varphi_{\text{fp}} = \varphi_{\text{fn}}$ for STRUCT-SVM and SENTI-LSSVM$_{\text{BOW}}$ because the configurations with $\varphi_{\text{fp}} \neq \varphi_{\text{fn}}$ led to worse performance. Since the relationship type of *off-topic* sentences is certainly *other*, we evaluate all systems with 5-fold cross-validation only on the *on-topic* sentences in the evaluation dataset. Since the same SRG can have several equivalent EMRGs and the relationship type *other* is not of our interest, we evaluate the *non-other* relationships of SRGs in terms of precision, recall and F-measure. All reported numbers are averaged over the 5 folds. Boldface figures are statistically significantly better than all others in the same comparison group under t-test $p = 0.05$.

5.8.2 Results

Table 5.4 shows the complete results of all systems. Here both SENTI-LSSVM$_{RAE}$ and SENTI-LSSVM$_{BOW}$ outperformed all baselines in terms of the average F-Measures and recalls by a large margin. The F-Measure on movie reviews is about 14% over the best baseline. The rule-based system has higher precisions than recalls in most cases. However, simply increasing the coverage of the domain independent sentiment polarity lexicon might lead to worse performance (Taboada et al., 2011) because many sentiment oriented relationships are conveyed by domain dependent expressions and factual expressions implying evaluations, such as *"This camera does not have manual control."* Compared to RULE-BASED, STRUCT-SVM performs better in the camera domain but worse for the movies due to many misclassification of negative relationships as *other*. It also wrongly predicted more positive relationships as *other* than latent structural SVM based systems. We found that the recalls of these relationships are low because they often have overly similar features with the relationships of the type *other* linking to the same mentions. The problem gets worse in the movie domain since i) many sentences contain no explicit sentiment-bearing words; ii) the prior polarity of the sentiment-bearing words do not agree with their contextual polarity in the sentences. E.g., the following sentence from a forum post about the movie *"Superman Returns"*: *"Have a look at Superman: the Animated Series or Justice League Unlimited ...that is how the characters of Superman and Lex Luthor should be."*. In contrast, latent structural SVM based models minimize the overlapped features by assigning them to the most likely relationship candidates. This leads to significantly better performance. Although STRUCT-SVM has low recalls for both positive and negative relationships, it achieves the highest recall for the comparative relationship among all systems in the movie domain. However, the test dataset of each fold in these document sets contains less than 10 comparative relationships. This advantage disappears on the camera forum posts, where there are about 100 comparative relationships in each test set.

SENTI-LSSVM$_{RAE}$ has an edge over SENTI-LSSVM$_{BOW}$ in terms of recall and F-Measure on all datasets except movie reviews. Since SENTI-LSSVM$_{RAE}$ represents every constituent or sentence as a low-dimensional

		Positive			Negative			Comparison			Micro-average		
		P	R	F	P	R	F	P	R	F	P	R	F
Camera Forum	RULE-BASED	0.560	0.392	0.461	0.459	0.240	0.315	0.431	0.107	0.171	0.534	0.302	0.386
	STRUCT-SVM	0.602	0.356	0.448	0.442	0.385	0.412	0.280	0.401	0.329	0.437	0.367	0.399
	SENTI-LSSVM$_{bow}$	**0.692**	0.389	0.498	**0.508**	0.393	**0.443**	**0.426**	0.351	0.385	**0.565**	0.380	0.454
	SENTI-LSSVM$_{RAE}$	0.618	**0.550**	**0.582**	0.424	**0.410**	0.417	0.416	**0.437**	**0.426**	0.525	**0.497**	**0.510**
Camera Review	RULE-BASED	**0.836**	0.693	0.758	0.686	0.388	0.496	0.300	0.169	0.216	**0.812**	0.589	0.682
	STRUCT-SVM	0.726	0.754	0.740	0.639	0.625	0.632	0.280	0.389	0.325	0.681	0.704	0.693
	SENTI-LSSVM$_{bow}$	0.773	0.854	0.812	**0.689**	0.613	**0.649**	0.223	0.207	0.216	0.731	0.734	0.737
	SENTI-LSSVM$_{RAE}$	0.763	**0.876**	0.816	0.629	0.621	0.625	**0.433**	**0.429**	**0.431**	0.727	**0.784**	**0.754**
Movie Forum	RULE-BASED	0.634	0.374	0.470	0.276	0.343	0.306	0.000	0.000	0.000	0.480	0.357	0.409
	STRUCT-SVM	**0.662**	0.301	0.413	0.256	0.173	0.207	0.442	**0.567**	**0.497**	**0.533**	0.279	0.366
	SENTI-LSSVM$_{bow}$	0.633	0.442	0.521	**0.297**	0.456	**0.360**	**0.401**	0.450	0.424	0.497	0.446	0.470
	SENTI-LSSVM$_{RAE}$	0.564	**0.525**	**0.544**	0.256	**0.568**	0.353	0.390	0.550	0.456	0.434	**0.537**	**0.482**
Movie Review	RULE-BASED	**0.665**	0.472	0.552	0.421	0.394	0.407	0.314	0.120	0.174	0.562	0.441	0.494
	STRUCT-SVM	0.613	0.540	0.574	0.452	0.137	0.211	0.245	**0.633**	0.353	0.546	0.392	0.457
	SENTI-LSSVM$_{bow}$	0.590	0.791	0.676	0.533	0.514	0.523	0.283	0.340	0.309	**0.579**	0.688	**0.629**
	SENTI-LSSVM$_{RAE}$	0.576	**0.809**	0.672	0.530	0.517	0.523	0.283	0.567	**0.377**	0.558	**0.698**	0.620

Table 5.4: Evaluation results of all systems.

continuous-valued vector, it provides a more compact way to encode the textual patterns involving multiple words than the high-dimensional bag-of-words representations. As a result, SENTI-LSSVM$_{RAE}$ can correctly extract relationships from a sentence like "I would go for the FX55.", which SENTI-LSSVM$_{BOW}$ fails to achieve. To gain more insight into the model performance, we summarize the experimental results in Table 5.5 and Table 5.6, which are categorized according to whether the gold standard relationships are from the sentences with multiple non-other relationships or not. As we can see, when the syntactic structures of subjective expressions become more complex, such as the ones for comparative relationships and the sentences containing multiple non-other relationships, SENTI-LSSVM$_{RAE}$ performed significantly better than SENTI-LSSVM$_{BOW}$. The slight drop of precision is due to the fact that the learned distributed representations are more close to each other in the

low-dimensional space than their high-dimensional counterparts, it is a way of trading specificity for greater generality.

		Positive			Negative			Comparison			Micro-average		
		P	R	F	P	R	F	P	R	F	P	R	F
Camera Forum	SENTI-LSSVM$_{bow}$	**0.639**	0.202	0.307	0.450	**0.203**	0.280	0.487	0.348	0.406	0.488	0.256	0.336
	SENTI-LSSVM$_{RAE}$	0.590	**0.258**	**0.359**	**0.692**	0.174	0.278	0.493	**0.413**	**0.450**	**0.508**	**0.299**	**0.377**
Camera Review	SENTI-LSSVM$_{bow}$	0.743	0.569	0.645	0.700	0.676	0.688	**0.167**	0.133	**0.148**	**0.750**	0.624	0.681
	SENTI-LSSVM$_{RAE}$	0.743	**0.636**	**0.685**	**0.743**	0.676	**0.708**	0.107	0.133	0.119	0.733	**0.674**	**0.702**
Movie Forum	SENTI-LSSVM$_{bow}$	**0.693**	0.365	0.478	0.576	0.192	0.288	**0.200**	0.100	**0.133**	**0.674**	0.277	0.392
	SENTI-LSSVM$_{RAE}$	0.560	**0.455**	**0.502**	**0.670**	**0.302**	**0.417**	0.067	0.100	0.080	0.630	**0.382**	**0.476**
Movie Review	SENTI-LSSVM$_{bow}$	0.482	0.400	0.437	0.530	**0.437**	**0.479**	0.050	0.100	0.067	0.418	0.397	0.407
	SENTI-LSSVM$_{RAE}$	**0.567**	**0.410**	**0.475**	**0.553**	0.397	0.462	**0.150**	**0.300**	**0.200**	**0.451**	0.398	**0.423**

Table 5.5: Evaluation results of SENTI-LSSVM$_{bow}$ and SENTI-LSSVM$_{RAE}$ for sentences with multiple relationships.

All systems perform better in predicting positive relationships than the negative ones. This corresponds well to the empirical findings in (Wilson, 2008) that people intend to use more complex expressions for negative sentiments than their affirmative counterparts. It is also in accordance with the distribution of these relationships in our SRG corpus which is randomly sampled from the online documents. For learning systems, it can also be explained by the fact that the training data for positive relationships are considerably more than those for negative ones. The comparative relationship is the hardest one to process since we found that many corresponding expressions do not contain explicit keywords for comparison.

5.9 Conclusion

We proposed SENTI-LSSVM$_{RAE}$ model for extracting both sentiments and comparisons at and below the sentence level. SENTI-LSSVM$_{RAE}$ can effectively learn from our SRG corpus, created by using the output-oriented annotation scheme. Due to the use of latent structural SVM, our model can automatically

		Positive			Negative			Comparison			Micro-average		
		P	R	F	P	R	F	P	R	F	P	R	F
Camera Forum	SENTI-LSSVM$_{bow}$	**0.811**	0.415	0.549	**0.659**	0.459	**0.541**	**0.465**	0.372	0.414	**0.702**	0.414	0.521
	SENTI-LSSVM$_{RAE}$	0.804	**0.601**	**0.688**	0.554	**0.484**	0.516	0.441	**0.472**	**0.456**	0.680	**0.557**	**0.612**
Camera Review	SENTI-LSSVM$_{bow}$	**0.825**	0.865	0.845	**0.781**	0.586	**0.670**	0.175	0.156	0.165	0.793	0.751	0.772
	SENTI-LSSVM$_{RAE}$	0.820	**0.884**	**0.851**	0.731	**0.597**	0.657	**0.394**	**0.411**	**0.402**	**0.801**	**0.794**	**0.798**
Movie Forum	SENTI-LSSVM$_{bow}$	**0.886**	0.447	0.595	**0.508**	0.513	0.510	0.442	0.510	0.473	**0.748**	0.461	0.570
	SENTI-LSSVM$_{RAE}$	0.856	**0.529**	**0.653**	0.445	**0.625**	**0.520**	**0.447**	**0.610**	**0.516**	0.696	**0.551**	**0.615**
Movie Review	SENTI-LSSVM$_{bow}$	**0.743**	0.805	0.773	**0.749**	0.524	0.617	**0.375**	0.318	0.344	**0.755**	0.703	**0.728**
	SENTI-LSSVM$_{RAE}$	0.729	**0.824**	0.773	0.731	**0.532**	0.616	0.348	**0.571**	**0.433**	0.726	**0.714**	0.720

Table 5.6: Evaluation results of SENTI-LSSVM$_{bow}$ and SENTI-LSSVM$_{RAE}$ for the sentences with at most one relationships.

find textual evidences to support its relationship predictions and achieves significantly better F-Measures than alternative state-of-the-art methods.

Chapter 6

Conclusion

In this dissertation, we have presented three learning models for sentiment analysis at different granularity levels: the bag-of-opinions model for document rating prediction (introduced in Chapter 3), the multi-experts model for sentence rating prediction (presented in Chapter 4), the SENTI-LSSVM$_{RAE}$ model for identifying sentiment-oriented relationships at both sentence and expression level (in Chapter 5). Different granularity of analysis leads to different model complexity, the finer the more complex. All models explore different feature representations to capture the compositional semantics inherent in sentiment-bearing expressions. Due to the lack of sufficient hand-labeled training data, these models maximize the use of freely available resources such as subjectivity lexicons and document ratings to circumvent the problem. Since the multi-experts model and SENTI-LSSVM$_{RAE}$ also use unlabeled data to improve the generalization power, they can be viewed as semi-supervised models.

There are several opportunities to extend and improve this work. At the coarse-grained level, as sentiments are often related across sentences or even documents, we can analyze these correlations to discover patterns for sentiment-flow. At the fine-grained level, it is straightforward to adapt SENTI-LSSVM$_{RAE}$ to extract facts of any given binary or n-ary relations. In this manner, our model allows also joint extraction of both facts and sentiments so that the two traditional tasks can be tackled by the same model. A more ambitious goal is to extract sentiment-oriented relationships with respect to certain facts, which can be regarded as higher order relationship extraction.

Bibliography

S. Baccianella, A. Esuli, and F. Sebastiani. Multi-facet rating of product reviews. In *Proceedings of the annual European Conference on Information Retrieval*. Springer, 2009.

S. Baccianella, A. Esuli, and F. Sebastiani. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proceedings of the International Conference on Language Resources and Evaluation*, 2010.

M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006a.

M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *The Journal of Machine Learning Research*, 7:2399–2434, 2006b.

Y. Bengio. Learning deep architectures for ai. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009.

Y. Bengio, O. Delalleau, and N. L. Roux. The curse of highly variable functions for local kernel machines. In *Advances in Neural Information Processing Systems*, 2005.

C. M. Bishop et al. *Pattern recognition and machine learning*, volume 1. springer New York, 2006.

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

A. Blum and T. M. Mitchell. Combining labeled and unlabeled sata with co-training. In *Proceedings of the Conference on Learning Theory*, pages 92–100, 1998.

L. Bottou. Stochastic learning. In *Advanced Lectures on Machine Learning*, pages 146–168, 2003.

E. Breck, Y. Choi, and C. Cardie. Identifying expressions of opinion in context. In *Proceedings of the 20th international joint conference on Artifical intelligence*, IJCAI'07, pages 2683–2688, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

L. Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

P. H. Calamai and J. J. Moré. Projected gradient methods for linearly constrained problems. *Mathematical Programming*, 39(1):93–116, 1987.

N. Cancedda, É. Gaussier, C. Goutte, and J.-M. Renders. Word-sequence kernels. *Journal of Machine Learning Research*, 3:1059–1082, 2003.

G. Carenini, R. T. Ng, and A. Pauls. Multi-document summarization of evaluative text. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*, 2006.

C. Chang and C. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006. URL `http://www.kyb.tuebingen.mpg.de/ssl-book`.

Y. Choi and C. Cardie. Learning with compositional semantics as structural inference for subsentential sentiment analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 793–801, 2008.

Y. Choi and C. Cardie. Adapting a polarity lexicon using integer linear programming for domain-specific sentiment classification. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, EMNLP '09, pages 590–598, Stroudsburg, PA, USA, 2009a. Association for Computational Linguistics. ISBN 978-1-932432-62-6.

Y. Choi and C. Cardie. Adapting a polarity lexicon using integer linear programming for domain-specific sentiment classification. In *Proceedings of*

the Conference on Empirical Methods in Natural Language Processing, volume 2, pages 590–598, 2009b.

Y. Choi and C. Cardie. Hierarchical sequential learning for extracting opinions and their attributes. In *Proceedings of the Annual meeting of the Association for Computational Linguistics*, pages 269–274. Association for Computational Linguistics, 2010.

Y. Choi, E. Breck, and C. Cardie. Joint extraction of entities and relations for opinion recognition. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 431–439, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

W. Chu and Z. Ghahramani. Gaussian processes for ordinal regression. *Journal of Machine Learning Research*, 6(1):1019, 2006.

R. Collobert and J. Weston. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of the International Conference on Machine Learning*, pages 160–167, 2008.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. P. Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011.

K. Crammer, A. Kulesza, and M. Dredze. Adaptive regularization of weight vectors. In *Advances in Neural Information Processing Systems*, pages 414–422, 2009.

S. Dasgupta and V. Ng. Mine the easy, classify the hard: A semi-supervised approach to automatic sentiment classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, volume 2, pages 701–709. Association for Computational Linguistics, 2009.

K. Dave, S. Lawrence, and D. M. Pennock. Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In *Proceedings of the 12th international conference on World Wide Web*, pages 519–528. ACM, 2003.

T. G. Dietterichl. Ensemble learning. *The Handbook of Brain Theory and Neural Networks*, pages 405–408, 2002.

X. Ding, B. Liu, and P. S. Yu. A holistic lexicon-based approach to opinion mining. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, pages 231–240, New York, NY, USA, 2008a. ACM. ISBN 978-1-59593-927-2.

X. Ding, B. Liu, and P. S. Yu. A holistic lexicon-based approach to opinion mining. In *Proceedings of the International Conference on Web Search and Data Mining*, pages 231–240, 2008b.

X. Ding, B. Liu, and L. Zhang. Entity discovery and assignment for opinion mining applications. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1125–1134, 2009.

J. Duchi and Y. Singer. Efficient online and batch learning using forward backward splitting. *The Journal of Machine Learning Research*, 10:2899–2934, 2009a.

J. C. Duchi and Y. Singer. Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research*, 10:2899–2934, 2009b.

J. C. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12: 2121–2159, 2011.

S. Dzeroski and B. Zenko. Is combining classifiers with stacking better than selecting the best one? *Machine Learning*, 54(3):255–273, 2004.

A. Esuli and F. Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of the International Conference on Language Resources and Evaluation*, pages 417–422, 2006. URL `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.61.7217`.

J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33:1, 2010.

G. Fu and X. Wang. Chinese sentence-level sentiment classification based on fuzzy sets. In *Proceedings of the International Conference on Computational Linguistics*, pages 312–319. Association for Computational Linguistics, 2010.

M. Ganapathibhotla and B. Liu. Mining opinions in comparative sentences. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, pages 241–248, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. ISBN 978-1-905593-44-6.

G. Ganu, N. Elhadad, and A. Marian. Beyond the stars: Improving rating predictions using review text content. In 12th *International Workshop on the Web and Databases*, 2009.

A. Ghose and P. G. Ipeirotis. Designing novel review ranking systems: Predicting the usefulness and impact of reviews. In *Proceedings of the ninth international conference on Electronic commerce*, pages 303–310. ACM, 2007.

N. Godbole, M. Srinivasaiah, and S. Skiena. Large-scale sentiment analysis for news and blogs (system demonstration). In *Proceedings of the International AAAI Conference on Weblogs and Social Media*, 2007a.

N. Godbole, M. Srinivasaiah, and S. Skiena. Large-scale sentiment analysis for news and blogs. In *Proceedings of the International AAAI Conference on Weblogs and Social Media*, 2007b.

A. B. Goldberg and X. Zhu. Seeing stars when there aren't many stars: graph-based semi-supervised learning for sentiment categorization. In *Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing*, pages 45–52. Association for Computational Linguistics, 2006.

A. B. Goldberg, X. Zhu, and S. J. Wright. Dissimilarity in graph-based semi-supervised classification. *Journal of Machine Learning Research - Proceedings Track*, 2:155–162, 2007.

C. Goller and A. Kuchler. Learning task-dependent distributed representations by backpropagation through structure. In *Neural Networks, 1996., IEEE International Conference on*, volume 1, pages 347–352. IEEE, 1996.

L. W. Hagen and A. B. Kahng. New spectral methods for ratio cut partitioning and clustering. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 11(9): 1074–1085, 1992.

Y. Haimovitch, K. Crammer, and S. Mannor. More is better: Large scale partially-supervised sentiment classification. In *Proceedings of the 4th Asian Conference on Machine Learning*, Singapore, November 2012.

V. Hatzivassiloglou and K. McKeown. Predicting the semantic orientation of adjectives. In *Proceedings of the Annual meeting of the Association for Computational Linguistics*, pages 174–181, 1997.

G. E. Hinton. Learning distributed representations of concepts. In *Proceedings of the eighth annual conference of the cognitive science society*, pages 1–12. Amherst, MA, 1986.

G. E. Hinton and R. S. Zemel. Autoencoders, minimum description length, and helmholtz free energy. *Advances in neural information processing systems*, pages 3–3, 1994.

G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *Computing Research Repository*, abs/1207.0580, 2012.

T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.

J. E. Hopcroft and R. M. Karp. An nˆ5/2 algorithm for maximum matchings in bipartite graphs. *SIAM Journal on computing*, 2(4):225–231, 1973.

M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pages 168–177, New York, NY, USA, 2004a. ACM. ISBN 1-58113-888-1.

M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 168–177, 2004b.

G. Ifrim, G. Bakir, and G. Weikum. Fast logistic regression for text categorization with variable-length n-grams. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 354–362, New York,USA, 2008. ACM. ISBN 978-1-60558-193-4.

T. Jebara, J. Wang, and S.-F. Chang. Graph construction and *b*-matching for semi-supervised learning. In *Proceedings of the International Conference on Machine Learning*, page 56, 2009.

W. Jin, H. H. Ho, and R. K. Srihari. Opinionminer: a novel machine learning system for web opinion mining and extraction. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1195–1204, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-495-9.

N. Jindal and B. Liu. Identifying comparative sentences in text documents. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '06, pages 244–251, New York, NY, USA, 2006a. ACM. ISBN 1-59593-369-7.

N. Jindal and B. Liu. Mining comparative sentences and relations. In *Proceedings of the 21st International Conference on Artificial Intelligence - Volume 2*, AAAI'06, pages 1331–1336. AAAI Press, 2006b. ISBN 978-1-57735-281-5.

N. Jindal and B. Liu. Review spam detection. In *Proceedings of the 16th international conference on World Wide Web*, pages 1189–1190. ACM, 2007.

N. Jindal and B. Liu. Opinion spam and analysis. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, pages 219–230, 2008.

T. Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of the International Conference on Machine Learning*, pages 200–209, 1999a.

T. Joachims. Making large scale svm learning practical. In *Advances in Kernel Methods-Support Vector Learning*. MIT Press, 1999b.

R. Johansson and A. Moschitti. Extracting opinion expressions and their polarities–exploration of pipelines and joint models. In *Proceedings of the Annual meeting of the Association for Computational Linguistics*, volume 11, pages 101–106, 2011.

I. T. Jolliffe. *Principal component analysis*, volume 487. Springer-Verlag New York, 1986.

H. Kanayama and T. Nasukawa. Fully automatic lexicon expansion for domain-oriented sentiment analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 355–363, 2006a.

H. Kanayama and T. Nasukawa. Fully automatic lexicon expansion for domain-oriented sentiment analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 355–363, 2006b.

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

J. S. Kessler, M. Eckert, L. Clark, and N. Nicolov. The 2010 icwsm jdpa sentment corpus for the automotive domain. In *4th International AAAI Conference on Weblogs and Social Media Data Workshop Challenge (ICWSM-DWC 2010)*, 2010.

S. Kim and E. Hovy. Determining the sentiment of opinions. In *Proceedings of the International Conference on Computational Linguistics*, pages 1367–1373, 2004. URL `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.68.1034`.

S.-M. Kim and E. Hovy. Extracting opinions, opinion holders, and topics expressed in online news media text. In *Proceedings of the Workshop on Sentiment and Subjectivity in Text*, SST '06, pages 1–8, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics. ISBN 1-932432-75-2.

S.-M. Kim, P. Pantel, T. Chklovski, and M. Pennacchiotti. Automatically assessing review helpfulness. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 423–430. Association for Computational Linguistics, 2006.

D. Klein and C. D. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 423–430, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. doi: 10.3115/1075096.1075150. URL `http://dx.doi.org/10.3115/1075096.1075150`.

M. A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, 37(2):233–243, 1991.

L.-W. Ku, Y.-T. Liang, and H.-H. Chen. Opinion extraction, summarization and tracking in news and blog corpora. In *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, pages 100–107, 2006.

L.-W. Ku, I.-C. Liu, C.-Y. Lee, K. hua Chen, and H.-H. Chen. Sentence-level opinion analysis by copeopi in ntcir-7. In *Proceedings of NTCIR-7 Workshop Meeting*, 2008.

D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

S. Li, Z. Wang, G. Zhou, and S. Y. M. Lee. Semi-supervised learning for imbalanced sentiment classification. In *Proceedings of the International Conference on Artificial Intelligence*, pages 1826–1831, 2011.

C. Lin and Y. He. Joint sentiment/topic model for sentiment analysis. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 375–384. ACM, 2009.

C. Lin, Y. He, R. Everson, and S. Ruger. Weakly supervised joint sentiment-topic detection from text. *Knowledge and Data Engineering, IEEE Transactions on*, 24 (6):1134–1145, 2012.

B. Liu. Sentiment analysis and subjectivity. *Handbook of natural language processing*, 2:568, 2010.

B. Liu, M. Hu, and J. Cheng. Opinion observer: analyzing and comparing opinions on the web. In *Proceedings of the 14th international conference on World Wide Web*, pages 342–351, New York, NY, USA, 2005. ACM. ISBN 1-59593-046-9.

D. C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

J. Liu and S. Seneff. Review Sentiment Scoring via a Parse-and-Paraphrase Paradigm. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 161–169. ACL, 2009.

J. Liu, Y. Cao, C.-Y. Lin, Y. Huang, and M. Zhou. Low-quality product review detection in opinion summarization. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 334–342, 2007.

H. Lodhi, J. Shawe-Taylor, N. Cristianini, and C. J. C. H. Watkins. Text classification using string kernels. In *Advances in Neural Information Processing Systems*, pages 563–569, 2000.

B. Lu, M. Ott, C. Cardie, and B. K. Tsou. Multi-aspect sentiment analysis with topic models. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*, pages 81–88. IEEE, 2011.

Y. Lu, P. Tsaparas, A. Ntoulas, and L. Polanyi. Exploiting social context for review quality prediction. In *Proceedings of the International World Wide Web Conference*, pages 691–700, 2010a.

Y. Lu, P. Tsaparas, A. Ntoulas, and L. Polanyi. Exploiting social context for review quality prediction. In *Proceedings of the 19th international conference on World wide web*, pages 691–700. ACM, 2010b.

Z. Luo and Q. Tseng. On the convergence of the coordinate descent method for convex differentiable minimization. *Journal of Optimization Theory and Applications*, 72(1):7–35, 1992.

M. Maier, U. von Luxburg, and M. Hein. Influence of graph construction on graph-based clustering measures. In *Advances in Neural Information Processing Systems*, pages 1025–1032, 2008.

M. Maier, M. Hein, and U. von Luxburg. Optimal construction of k-nearest-neighbor graphs for identifying noisy clusters. *Theor. Comput. Sci.*, 410(19): 1749–1764, 2009.

Y. Mao and G. Lebanon. Isotonic Conditional Random Fields and Local Sentiment Flow. *Advances in Neural Information Processing Systems*, pages 961–968, 2006.

A. L. Martins, N. A. Smith, and E. P. Xing. Concise integer linear programming formulations for dependency parsing. In *Proceedings of the Annual meeting of the Association for Computational Linguistics*, pages 342–350, 2009.

R. T. McDonald, K. Hannan, T. Neylon, M. Wells, and J. C. Reynar. Structured models for fine-to-coarse sentiment analysis. In *Proceedings of the Annual meeting of the Association for Computational Linguistics*, 2007a.

R. T. McDonald, K. Hannan, T. Neylon, M. Wells, and J. C. Reynar. Structured models for fine-to-coarse sentiment analysis. In *Proceedings of the Annual Meeting on Association for Computational Linguistics*, volume 45, page 432, 2007b.

P. Melville, W. Gryc, and R. D. Lawrence. Sentiment analysis of blogs by combining lexical knowledge with text classification. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1275–1284. ACM, 2009.

G. A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11): 39–41, 1995a.

G. A. Miller. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995b.

G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*, volume 18. Wiley New York, 1988.

K. Ozaki, M. Shimbo, M. Komachi, and Y. Matsumoto. Using the mutual k-nearest neighbor graphs for semi-supervised classification of natural language data. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning, CoNLL*, volume 11, pages 154–162, 2011.

B. Pang and L. Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the*

Annual Meeting on Association for Computational Linguistics, pages 271–278, 2004.

B. Pang and L. Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the Annual meeting of the Association for Computational Linguistics*, 2005a.

B. Pang and L. Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 124–131, 2005b.

B. Pang and L. Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2007.

B. Pang and L. Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2008.

B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. *CoRR*, cs.CL/0205070:79–86, 2002a.

B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 79–86, 2002b.

J. B. Pollack. Recursive distributed representations. *Artificial Intelligence*, 46(1-2): 77–105, 1990.

A. Popescu and O. Etzioni. Extracting product features and opinions from reviews. In *Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing*, volume 5, pages 339–346. Springer, 2005a.

A.-M. Popescu and O. Etzioni. Extracting product features and opinions from reviews. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 339–346, Stroudsburg, PA, USA, 2005b. Association for Computational Linguistics.

V. Punyakanok, D. Roth, W.-t. Yih, and D. Zimak. Semantic role labeling via integer linear programming inference. In *Proceedings of the 20th*

international conference on Computational Linguistics, page 1346. Association for Computational Linguistics, 2004.

G. Qiu, B. Liu, J. Bu, and C. Chen. Expanding Domain Sentiment Lexicon through Double Propagation. In *International Joint Conference on Artificial Intelligence*, pages 1199–1204, 2009.

L. Qu, G. Ifrim, and G. Weikum. The bag-of-opinions method for review rating prediction from sparse text patterns. In C.-R. Huang and D. Jurafsky, editors, *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, ACL Anthology, pages 913–921, Beijing, China, 2010. Tsinghua University Press.

L. Qu, R. Gemulla, and G. Weikum. A weakly supervised model for sentence-level semantic orientation analysis with multiple experts. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 149–159, Jeju Island, Korea, July 2012. Proceedings of the Annual meeting of the Association for Computational Linguistics. ISBN 978-1-937284-43-5.

R. Quirk, S. Greenbaum, G. Leech, J. Svartvik, and D. Crystal. *A Comprehensive Grammar of the English Language*, volume 397. Cambridge Univ Press, 1985.

D. Rao and D. Yarowsky. Ranking and semi-supervised classification on large scale graphs using map-reduce. In *Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing*, pages 58–65. Association for Computational Linguistics, 2009.

C. E. Rasmussen. *Gaussian processes in machine learning*. Springer, 2004.

Y. Ren, N. Kaji, N. Yoshinaga, M. Toyoda, and M. Kitsuregawa. Sentiment classification in resource-scarce languages by using label propagation. In *PACLIC*, pages 420–429, 2011.

S. Riedel and J. Clarke. Incremental integer linear programming for non-projective dependency parsing. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 129–137. Association for Computational Linguistics, 2006.

D. Roth and W.-t. Yih. Integer linear programming inference for conditional random fields. In *Proceedings of the 22nd international conference on Machine learning*, pages 736–743. ACM, 2005.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 1:213, 2002.

T. Sandler, J. Blitzer, P. P. Talukdar, and L. H. Ungar. Regularized learning with networks of features. In *Advances in Neural Information Processing Systems*, pages 1401–1408, 2008.

C. Scheible and H. Schuetze. Cutting recursive autoencoder trees. In *Proceedings of the 1st International Conference on Learning Representations*, Scottsdale, Arizona, USA, 2013.

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, 2000.

V. Sindhwani and P. Melville. Document-word co-regularization for semi-supervised sentiment analysis. In *Proceedings of the IEEE International Conference on Data Mining series*, pages 1025–1030, 2008.

B. Snyder and R. Barzilay. Multiple Aspect Ranking using the Good Grief Algorithm. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 300–307, 2007.

R. Socher, C. C. Lin, A. Y. Ng, and C. D. Manning. Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In *Proceedings of the 26th International Conference on Machine Learning*, volume 2, 2011a.

R. Socher, J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 151–161, 2011b.

R. Socher, B. Huval, C. D. Manning, and A. Y. Ng. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1201–1211, 2012.

S. Somasundaran and J. Wiebe. Recognizing stances in online debates. In *Proceedings of the Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, pages 226–234, 2009.

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

O. Täckström and R. McDonald. Discovering fine-grained sentiment with latent variable structured prediction models. In *Proceedings of the 33rd European conference on Advances in information retrieval*, ECIR'11, pages 368–374, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 978-3-642-20160-8.

O. Täckström and R. T. McDonald. Semi-supervised latent variable models for sentence-level sentiment analysis. *In Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 569–574, 2011a.

O. Täckström and R. T. McDonald. Discovering Fine-Grained Sentiment with Latent Variable Structured Prediction Models. In *Proceedings of the European Conference on Information Retrieval*, pages 368–374, 2011b.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

I. Titov and R. McDonald. A joint model of text and aspect ratings for sentiment summarization. In *Proceedings of the Annual meeting of the Association for Computational Linguistics*, pages 308–316, 2008a.

I. Titov and R. T. McDonald. Modeling online reviews with multi-grain topic models. In *Proceedings of the 17th international conference on World Wide Web*, pages 111–120, 2008b.

C. Toprak, N. Jakob, and I. Gurevych. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 575–584, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the International Conference on Machine Learning*, 2004.

P. D. Turney. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 417–424. Association for Computational Linguistics, 2002.

V. Vapnik. *Statistical learning theory*. Wiley, 1998. ISBN 978-0-471-03003-4.

P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the International Conference on Machine Learning*, pages 1096–1103, 2008.

U. von Luxburg. A tutorial on spectral clustering. *CoRR*, abs/0711.0189, 2007.

X. Wan. Co-training for cross-lingual sentiment classification. In *Proceedings of the Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, pages 235–243, 2009.

J. Wang, D. Song, L. Liao, W. Zou, X. Yan, and Y. Su. The chinese bag-of-opinions method for hot-topic-oriented sentiment analysis on weibo. In *Semantic Web and Web Science*, pages 357–367. Springer, 2013.

W. Wei and J. A. Gulla. Sentiment learning on product reviews via sentiment ontology tree. In *Proceedings of the Annual meeting of the Association for Computational Linguistics*, pages 404–413, 2010.

J. Wiebe. Tracking point of view in narrative. *Computational Linguistics*, 20(2): 233–287, 1994.

J. Wiebe. Instructions for annotating opinions in newspaper articles. Technical Report TR-02-101, University of Pittsburg, 2002.

J. Wiebe, T. Wilson, and C. Cardie. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210, 2005.

T. Wilson, J. Wiebe, and P. Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 347–354, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

T. A. Wilson. *Fine-grained subjectivity and sentiment analysis: recognizing the intensity, polarity, and attitudes of private states*. PhD thesis, UNIVERSITY OF PITTSBURGH, 2008.

Y. Wu, Q. Zhang, X. Huang, and L. Wu. Structural opinion mining for graph-based sentiment representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1332–1341, 2011.

A. Yessenalina and C. Cardie. Compositional matrix-space models for sentiment analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 172–182, 2011.

C.-N. J. Yu and T. Joachims. Learning structural svms with latent variables. In *Proceedings of the International Conference on Machine Learning*, page 147, 2009.

N. Yu and S. Kübler. Filling the gap: Semi-supervised learning for opinion detection across domains. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, pages 200–209. Association for Computational Linguistics, 2011.

C. X. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems*, 22 (2):179–214, 2004. ISSN 1046-8188. doi: http://doi.acm.org/10.1145/984321.984322.

D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *Advances in neural information processing systems*, volume 16, page 284, 2004.

X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, 2002.

X. Zhu and A. B. Goldberg. Kernel regression with order preferences. In *Proceedings of the Conference on Artificial Intelligence*, pages 681–687, 2007.

X. Zhu and A. B. Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009.

X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised Learning using Gaussian Fields and Harmonic Functions. In *Proceedings of the International Conference on Machine Learning*, pages 912–919, 2003.

L. Zhuang, F. Jing, and X. Zhu. Movie review mining and summarization. In *Proceedings of the ACM International Conference on Information and Knowledge Management*, pages 43–50, 2006a.

L. Zhuang, F. Jing, and X. Zhu. Movie review mining and summarization. In *Proceedings of the ACM international conference on Information and knowledge management*, pages 43–50, 2006b.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B(Statistical Methodology)*, 67(2):301–320, 2005.

List of Figures

List of Tables