Inferring aspect-specific opinion structure in product reviews

by

David H. Carter

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements
for the M.A.Sc. degree in
Electrical and Computer Engineering

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

Identifying differing opinions on a given topic as expressed by multiple people (as in a set of written reviews for a given product, for example) presents challenges. Opinions about a particular subject are often nuanced: a person may have both negative and positive opinions about different aspects of the subject of interest, and these aspect-specific opinions can be independent of the overall opinion on the subject. Being able to identify, collect, and count these nuanced opinions in a large set of data offers more insight into the strengths and weaknesses of competing products and services than does aggregating the overall ratings of such products and services.

I make two useful and useable contributions in working with opinionated text.

First, I present my implementation of a semi-supervised co-training machine classification method for identifying both product aspects (features of products) and sentiments expressed about such aspects. It offers better precision than fully-supervised methods while requiring much less text to be manually tagged (a time-consuming process). This algorithm can also be run in a fully supervised manner when more data is available.

Second, I apply this co-training approach to reviews of restaurants and various electronic devices; such text contains both factual statements and opinions about features/aspects of products. The algorithm automatically identifies the product aspects and the words that indicate aspect-specific opinion polarity, while largely avoiding the problem of misclassifying the products themselves as inherently positive or negative.

This method performs well compared to other approaches. When run on a set of reviews of five technology products collected from Amazon, the system performed with some demonstrated competence (with an average precision of 0.83) at the difficult task of simultaneously identifying aspects and sentiments, though comparison to contemporaries' simpler rules-based approaches was difficult. When run on a set of opinionated sentences about laptops and restaurants that formed the basis of a shared challenge in the SemEval-2014 Task 4 competition, it was able to classify the sentiments expressed about aspects of laptops better than any team that competed in the task (achieving 0.72 accuracy). It was above the mean in its ability to identify the aspects of restaurants about which people expressed opinions, even when co-training using only half of the labelled training data at the outset.

While the SemEval-2014 aspect-based sentiment extraction task considered only separately the tasks of identifying product aspects and determining their polarities, I take an extra step and evaluate sentences as a whole, inferring aspects and the aspect-specific sentiments expressed simultaneously, a more difficult task that seems more applicable to real-world tasks. I present first results of this sentence-level task.

The algorithm uses both lexical and syntactic information in a manner that is shown to be able to handle new words that it has never before seen. It offers some demonstrated ability to adapt to new subject domains for which it has no training data. The system is characterizable by very high precision and weak-to-average recall and it estimates its own confidence in its predictions; this characteristic should make the algorithm suitable for use on its own or for combination in a confidence-based voting ensemble. The software created for and described in the course of this dissertation is made available online.

Acknowledgements

I offer my sincerest and deepest gratitude to Dr. Diana Inkpen for her eminently capable supervision, fantastic feedback, creative ideas, gracious patience, and endless support. I feel truly blessed to have had such a hard-working mentor.

Many thanks to Dr. Stan Szpakowicz for passionate discussions about language and machine learning, and whom I credit for inspiring my pursuit of natural language processing in academia.

Thanks to Dr. Marina Sokolova and Dr. James Green for having served on my defence committee. My work is surely better as a result of the feedback provided, and I appreciate the time, energy, and enthusiasm they put into considering my work.

Thanks to the folks in The Cookie Group for their broad knowledge and keen insights into cutting-edge NLP research.

Many thanks to Joel Martin and all the natural language processing folks at National Research Council Canada for constant inspiration, encouragement, and support. Special thanks too to Elizabeth Scarlett for giving me a necessary push now and again.

Thanks to the folks in the Stanford NLP Group for building great tools for our community.

Thanks to the staff at Morisset library for maintaining a very robust set of licenses to NLP research, without which this work would have not been possible.

Finally, love and thanks to Sid Byrd, Liz Kim, Natalie Michelle Campbell, Tara Molloy, and Matt Connelly for their eminently capable moral support throughout my research.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

> All opinions are not equal. Some are a very great deal more robust, sophisticated and well supported in logic and argument than others.
> *Adams (2002)*

This dissertation describes a set of experiments that identify aspect-specific sentiments in English text: those where a writer has mentioned what aspects/features of a product they like and dislike, independent of whether they like or dislike the product itself.

1.1 Motivation

Humans are opinionated beings. Some opinions may be arbitrary, but a great number are nuanced and explicitly supported.[1]

As the amount of human-written text existing in the world increases, so too does the amount of opinionated text written and shared online. It is natural to be curious what people are opinionated about, and what nuances might be found in their opinions. A considerable amount of the accumulated writing of the human race is now available online; indeed, much human writing is now "born digital". People share their opinions online in great numbers. The deluge of available text makes these opinions accessible but, paradoxically, due to their sheer number, it becomes increasingly difficult to synthesize and generalize these opinions.

[1] One hopes that the arguments presented in this dissertation might fall in the latter category.

Contrast the world of opinions about movies: 25 years ago, one might have read a syndicated copy of Roger Ebert's column to glean information about the good and bad aspects of a particular film (e.g., great acting and poor cinematography); whereas today, one might find dozens of reviews on IMDB, hundreds on Metacritic, and thousands on Amazon (if a movie is available on disc). Reading one well-respected critic's aspect-specific opinions about a movie is feasible for an individual, but synthesizing a majority opinion on various aspects from numerous reviews is, at best, cumbersome, and quite possibly infeasible for most folks.

These kinds of aspect-specific opinions support decision-making. A film student interested in *nouvelle vague* cinema contemplating a movie night is not well-served by a five-star review of the latest Hollywood thriller; such a student's opinion of any given film is informed by different aspects than those in which the population at large is interested. Such a film student, if trying to find a film to watch, is more likely interested in the cinematography, the novelty of narrative devices used, and the degree to which the film is experimental; whereas the population at large is perhaps more interested in whether the actors are recognizable, whether a love story unfurls satisfactorily, whether there is a happy ending, and how large the explosions portrayed in action sequences might be. These particular *aspects* are key motivators for deciding whether a given individual might want to watch a particular movie; whereas, a movie is being determined atomically to be "good" is not, in itself, very useful. (As an extreme example, consider some low-budget art house cult films that are reviewed to be extraordinarily poor; for cult film aficionados, the lack of professionalism and polish of such films might indeed be a compelling selling point.)

The sheer amount of nuanced opinionated text available online makes it likely that, for any given consumer contemplating an everyday purchase, there may well exist a review[2] that addresses the aspects most salient to that particular consumer; and yet, because such opinionated text is now so prevalent, it makes it unlikely that the consumer will be able to discover such a well-suited review.

Computer software, of course, has no inherent difficulty in consuming such text en masse.

There are at least two reasonable approaches that could be undertaken to solve this particular data deluge dilemma. One solution could be an information retrieval approach: a search tool that "understands" product aspects and aspect-specific opinions and that could search through a vast number of reviews to find and rank a small set of well-suited reviews that mention a particular aspect (e.g., perform a search for the top ten movies with sedate narrative pace, excellent cinematography, and understated acting). Another solution would be for an aggregation tool that could discover aspects of products in a given domain and rank

[2]Amazon alone had well over 35 million product reviews as of March 2013 (McAuley and Leskovec, 2013).

them aspect-by-aspect based on a large corpus of stated opinions (e.g., present a list of cell phones sorted according to how many people liked its screen quality; or present a bar chart with competing products on the x axis and an aggregated screen quality score on the y axis). Underpinning both of these possible and plausible solutions is software that can recognize and extract aspect-specific opinions from human-written free-form text.

This dissertation describes work to improve the state of the art in identifying such aspect-specific opinions in text.

1.2 Thesis goals

The goal of this thesis is to develop usable and useful software that, given a set of casually written product reviews, identifies products' aspects (features) and infers writers' opinions about these aspects.

A second goal of this thesis is to investigate whether co-training, a semi-supervised machine learning approach, can improve the performance on such a product aspect sentiment inference task by taking advantage of a large set of unlabelled data at training time. Data annotated with aspects and fine-grained aspect-specific opinions is often not available; in such cases, it is easier to annotate a smaller data set and use co-training to glean some benefit from remaining non-annotated data than it is to annotate a full data set. A successful experiment will demonstrate that the results achieved with co-training approach those of a fully-supervised method on the same data; the latter can be reasonably thought of as an ideal performance ceiling for co-training.

In both cases, it is my goal that the software developed should behave in a manner that suggests adaptability in the face of imperfect language. It should handle regular, casual writing with aplomb, including misspellings and poor punctuation. It should be able to recognize and handle new, unseen language. It should, in at least some (perhaps rare) circumstances, have classification performance that can be argued to be better than human annotators (e.g., it should be able to find mistakes in the annotated data). It should incorporate some lexical and syntactic knowledge so that it can be argued that there is some language understanding occurring (as opposed to, say, simple bag-of-words models that simply count words, ignoring the beauty and complexity of the language they are meant to convey).

A final goal of this thesis is for the system developed to exceed the performance of contemporaries in some measure on some aspect-specific sentiment extraction task; the software developed should be better suited for at least some small subset of tasks than all other known sentiment classification systems on a particular data set.

1.3 Contributions

The unique and novel contributions of this dissertation are as follows:

1. The first use of co-training for aspect-specific sentiment extraction (simultaneously training or classifying both the aspects of the product/service and the sentiment expressions)

2. The first use of lexical information as one co-training view and syntactic information as another co-training view

3. A machine learning approach that is demonstrated to be able to correctly handle words that have never been seen before

4. Particularly high precision aspect-specific sentiment extraction (achieving higher precision than all 31 teams who participated in the SemEval 2014 aspect-specific sentiment extraction task)

Finally, this dissertation posits a set of criteria to use when selecting a data set for NLP experimentation.

1.4 Outline

This dissertation will unfold as follows. A succinct review of the fundamentals of natural language processing is offered in the next chapter. A thorough review of the literature in sentiment analysis in natural language processing is offered in section 3. The problem tackled and solved by this thesis is described in section 4. This is followed in section 5 by a description of the data used in the experiments, how they were selected, and how they were processed to be suitable for experimentation. Section 6 describes the aspect-specific sentiment identification experiments themselves, while section 7 presents and dissects the results of the experiments. Conclusions and ideas for further work are offered in the eighth (and final) section.

Chapter 2

A brief review of NLP and ML

This thesis seeks to use statistical natural language processing techniques and machine learning to find fine-grained sentiments expressed in text. A brief review of the fundamentals of the field is offered as a refresher.

2.1 Natural language processing

Natural language processing (or NLP) is the treatment of human language (text or sound) by computers, as implemented in some combination of hardware and/or software.

NLP is used for a plethora of tasks. Examples include:

- machine translation
- question answering (as exemplified by IBM's Watson (Ferrucci *et al.*, 2010) on the TV show *Jeopardy!*)
- named entity identification (tagging proper names of people and places)
- semantic searching (for example, trying to find news articles about *senior executives* who have been *convicted* of *indictable offences*; having the system realize that vice-presidents are senior executives, that pleading guilty is a type of conviction, and that mischief is an indictable offence; and returning a news article about RIM vice-presidents who notoriously caused an Air Canada flight to be diverted in 2011)
- speech-to-text dictation
- text-to-speech synthesis

- syntactic sentence parsing (deciding if a piece of text is grammatical and/or inferring syntactic structure from text)
- semantic sentence parsing (figuring out the structure of, for example, who did what to whom)
- information extraction (synthesizing facts expressed in human-written text and representing them in some sort of structure)
- spell- and grammar-checking
- sentiment analysis (deciding if a piece of text is positive, negative, or objective)
- emotion analysis (deciding if a piece of text indicates anger, surprise, happiness, and so on)
- natural language generation (trying to automatically describe what is depicted in a photograph or painting, for example)
- automatic text summarization

and so on.

There are two major schools of thought on NLP.

Classically, NLP applications have relied on sets of rules that guide the process at hand (for example, building a sentence parser using a nominally complete set of rules that define allowable words, their parts of speech, and allowable sequences of parts of speech; or building a machine translation system using rules-based finite state transducers that directly map input words to output words, perhaps with some rules for reordering words in the process). Such approaches, often termed computational linguistics, can be subject to criticism because it is difficult to model all language phenomena with a tractably finite set of rules, and because such rule sets must be labouriously updated over time. (By analogy, consider that dictionaries, which are essentially rule sets of allowable words, can never completely record all creative uses of unusual morphological variation, such as *verbifying* nouns; nor can they quickly keep up with new words like *selfie*).

Statistical NLP, on the other hand, consists of quantitative (often probabilistic) approaches to dealing with language, modelling language implicitly (by counting words or short sequences of words called *n-grams*; or by using large sets of aligned parallel text to accomplish machine translation) rather than by using explicit rules. These, too, can be subject to criticism, as the statistical assumptions that underlie these techniques may not match our

intuition of how language works (or "should" work), and corpus-based applications can be criticized for having insufficient data (for example, if parallel English and Khmer texts are used to build a machine translation system, there is a good chance that such corpora might not yet contain the word *selfie* either).

An inquiring reader might be well-advised to consult Jurafsky and Martin (2000), a comprehensive treatise of natural language processing; or perhaps Manning and Schütze (1999), the seminal work on statistical approaches to NLP. The major tasks in NLP are also introduced by Russell and Norvig (2003, Chapter 22).

2.2 Machine learning

Machine learning is a branch of the field of artificial intelligence, which is in turn a branch of computer science, engineering, cognitive science, and perhaps even philosophy to an extent.

Machine learning seeks to make decisions about new and unseen data without having been explicitly instructed about how to do so; that is, there is no written code/rules for dealing with new and unseen data. A machine learning system either learns by example (supervised and semi-supervised learning) or by trying to use fitting functions (clustering, regression).

In supervised learning, the software is given examples of known data that it can use to try to infer something about new instances of unknown data.

In semi-supervised learning (e.g., bootstrapping), the system learns in a supervised fashion from a small set of labelled examples to begin with, and then predicts labels for some unlabelled examples and adds them to the list of "known" data, iteratively building a larger and larger set of data with which to train itself.

Unsupervised learning can be used, as an example, to cluster documents that are similar by some metric, perhaps by topic. In some algorithms the user might specify the desired number of clusters ahead of time or might specify a threshold for how similar documents must be in order to be included in a cluster. Data are not labelled ahead of time; the system learns exclusively by inspection.

Machine learning is not inherently tied to natural language processing; it is possible to do NLP without machine learning, and there is a wide range of applications of machine learning that have nothing to do with NLP.

Machine learning techniques can be combined to accomplish a goal (ensemble learning). One might use clustering to build a knowledge base used to train a supervised learning system, for example; or different systems can be used to analyze the same data in parallel and vote on the result.

A reader might wish to consult the work of Witten and Frank (2005) for thorough and well-written coverage of machine learning algorithms and techniques. Similar material is covered by Russell and Norvig (2003, Chapters 18-21).

2.3 Machine classification

Machine classification is a type of supervised (or, occasionally, semi-supervised) machine learning. The goal of machine classification is to be able to sort new and previously unseen data into a discrete number of bins/categories. This can also be thought of as labelling data with a pre-determined, controlled, and finite set of labels.

Classification models are built ("trained") ahead of time using a set of labelled data. Once the classification model is built, the classification algorithm uses these models and, given a new piece of data, tries to match it into the category that is most similar, as determined by some measure.

Designing a good representation of the data that is tractable and appropriate for the problem domain is one of the challenges of building a good machine classification system. This measure by which data are compared is typically implemented as a set of *features*. For instance, one could build a classifier to determine whether a motor vehicle will fit into a small residential garage. The classifier would be binary: for new piece of data, it would guess whether the vehicle *will fit* or *will not fit*. A comprehensive set of features might be: manufacturer, model, year of production, and trim line. If it is known, for example, that all Ford trucks made between 1950 and 2012 do not fit in the garage, and that no trucks made by Ford are known to fit, and a new unknown model of Ford truck is introduced, it seems a reasonable bet that, unless given new knowledge, the new unknown truck will not fit. Or perhaps older, smaller Ford trucks made before 1950 are known to fit, but Chevrolet trucks of that age do not, so upon seeing a new unseen Chevrolet truck the classifier might conclude that it will not fit, but that the Ford truck will. A classifier, during training, builds a model that can generalize combinations of features for each example so as to give the best hint about how to assign new exemplars into the established categories.

Features can be thought of as dimensions of a single vector in a highly dimensional space; this interpretation allows some algorithms to use vector algebra to determine similarity (e.g., cosine similarity) between two examples' sets of features.

There are several major algorithms that perform classification, distinguished by how the learned model is built from the data and how new data are compared to the learned knowledge.

Some commonly used machine classification algorithms are: Naïve Bayes; support vector machines (SVM); neural networks (and deep learning, an evolving variation thereof); expectation maximization (EM); k-nearest neighbour (k-NN); and decision trees. Each of these is based on a fundamentally different mathematical interpretation of the data and a different intuition about how to select or evolve a model that generalizes the data, and each has its own strengths and weaknesses. All, fundamentally, try to optimize the model learned at training time by attempting to minimize the aggregate error in the data.

This dissertation describes work that uses support vector machine classifiers (Cortes and Vapnik, 1995). These treat each labelled or unlabelled example as a vector in a highly dimensional space; an example with three features can be thought of in a three-dimensional space for convenience. Given a set of examples sorted into two (or sometimes more) classes, a SVM attempts to more-or-less optimally fit a splitting plane in the dimensional space, trying to minimize the number of labelled examples that fall on the incorrect side of the plane. Once trained, classification of new examples is relatively trivial: it is necessary only to determine on which side(s) of the plane(s) the new example falls. Variations of the SVM approach involve flat planes (the simplest form), quadratic planes, or highly-contoured planes that tightly wrap around all examples on which it was trained (an approach that often leads to *over-fitting*, that is, a lack of generalization that makes it unlikely that the model will be able to make good guesses about unseen data).

Justification of the choice of support vector machines is offered in Section 6.3.1.

2.4 Features and aspects that aren't features and aspects

It is worth noting two matters of nomenclature.

The word *feature* has at least two precise meanings (one avoided) in this dissertation. *Feature* is taken herein to mean a machine learning feature; that is, a dimension of a vector used by a machine learning classifier to represent a part of an example being either classified or used to train the classifier. For example, if trying to build a classifier that could identify a person, the machine learning features used might be age, hair colour, height, nationality, fashion style, and so on. By contrast, in the problem domain, one could say that products about which specific opinions are expressed have features; that is, an iPhone has the features of screen size, battery life, weight, size, and so on. To avoid confusion, these product features will be called *aspects* instead; this also conforms to literature on the domain.

The word *aspect* itself has at least two meanings that could be confused in this dissertation. *Aspect* is taken to mean features of products (or restaurants) about which people might

express specific opinions. There is also a linguistic/grammatical notion of aspect, used to describe the degree to which a verb is used in a finite or continuous sense. Grammatical aspect is not considered explicitly in this dissertation, so this sense is not used.

2.5 Putting it all together

This thesis uses *supervised* and *semi-supervised machine learning* to accomplish a *[statistical] natural language processing* task: given text in reviews of products and restaurants, trying to find the sentiments expressed about the specific *aspects* of said products/restaurants. It uses two *support vector machine classifiers* to predict whether a given word in a sentence is inside of a phrase that expresses sentiment, inside a phrase that expresses a product aspect, or outside of either; and it tries to infer from these predicted words a correct and complete list of aspect-specific sentiments expressed in a sentence.

Chapter 3

Related work

There have been attempts at inferring the sentiment of sentences using computers for twenty years, with some approaches based on manually coded rules based on observed linguistic phenomena, and some using machine learning and other forms of artificial intelligence. My work draws on both approaches.

This chapter highlights seminal work in sentiment analysis in natural language processing, and offers an in-depth survey of work extracting sentiment from informal (and sometimes poorly-written) product reviews, movie reviews, financial stock analyses, and Twitter posts.

As my sentiment extraction algorithm is based on machine learning, an overview of relevant research on classifier selection and semi-supervised learning is offered.

Finally, while this dissertation does not make any particular contributions to the linguistics community, research into the nature of subjective text is covered briefly, as a useful and necessary precursor to sentiment analysis work.

3.1 Seminal and salient results in sentiment mining

There has been much work in sentiment analysis and opinion mining.

There are several commendable works that survey the state-of-the-art in sentiment analysis. Liu and Zhang (2012) offer a compelling overview of various sentiment classification tasks: aspect-specific and document-level sentiment classification, subjectivity classification, clustering, lexical approaches, all in a thoroughly linguistically-grounded manner. Liu (2012) expands upon this overview in greater detail. Pang and Lee (2008) offer a similarly good overview, and while they don't mention the most recent natural language processing work in the field, they do delve into the fundamental reasons why sentiment analysis is useful; they

summarize results of surveys that conclude that reviews of restaurants, hotels, and the like have a significant influence on consumers' purchasing decisions, and that, depending on the type of product or service, 20% to 99% of consumers will pay more for an item that is rated five stars out of five than a competing item ranked four stars out of five. They proceed to survey papers in three areas: papers that tackle the problem of classifying subjective versus objective material; papers that seek to classify the sentiment expressed in a document or to classify aspect-specific sentiments; and papers that describe systems that can summarize and present opinionated text in a useful way to an end user.

There are several highly-cited and highly-respected papers that made great leaps in sentiment analysis. Hatzivassiloglou and McKeown (1997) presented an algorithm for determining the semantic orientation of adjectives (a word class that is particularly strong at conveying opinion). Turney (2002) sought to classify the semantic orientation (positive/negative) of reviews of cars, banks, movies, and travel destinations. The basis for this work was comparing given phrases to a fixed positive word and a fixed negative word and assigning a mutual information score. Nigam and Hurst (2004) propose a collocation assumption that if a sentence contains topical information and polar sentiment language, that the two are related, and they offer several metrics for aggregating opinions about topics. They posit several useful evaluation metrics for sentiment classification tasks. Critically, they delve into the performance of humans on sentiment polarity classification tasks at both the message and the sentence level; human agreement on message-level polarity had precision of approximately 80% and recall of approximately 76%, while at the sentence level, precision was 88% and recall was 70%. Note that these tests involved classifying a whole message or whole sentence, and were therefore not aspect-specific.

One compelling approach to sentiment analysis consists of two phases: first, separating passages of text that contain opinions from those that don't, and then trying to determine whether the opinionated passages indicate a positive or negative sentiment. An early effort in this vein was presented by Yu and Hatzivassiloglou (2003), who classified documents in a news corpus as either largely subjective (editorials) or objective (regular news articles). They also sought to classify sentences as objective or subjective, and classified nominally subjective sentences as positive, negative, or neutral, with fairly impressive results. Pang and Lee (2004) aim to determine whether a given movie review is positive or negative. They first label sentences as either subjective or objective, then discard the latter; they then use only the subjective sentences to classify the document as positive or negative. Similarly, Wilson (2005) separates the tasks of classifying sentences as opinionated or neutral from the task of determining the polarity of phrases within the sentences. She approaches the problem as one

of adapting a lexicon of sentiment terms and phrases, aiming to infer the sentiment of such terms in context, even when such words' polarity in a given context is different from their usual polarity. She uses machine learning with a variety of features, including dependency trees. The task of determining whether text is objective or subjective is of interest even by itself; Joshi and Penstein-Rosé (2009) use dependency relation features with a form of back-off (that is, the ability to use less information or simpler information when ideal information is not present) to try to improve performance in classifying text as subjective or objective. While the task of determining whether text is opinionated is related to the task of determining whether opinionated text is positive or negative, these tasks can also be treated independently.

There is a body of work that seeks to connect opinions expressed in text with the person who holds them. This task is not directly related to the work in this dissertation, but shares a few of the same challenges of identifying opinionated text. For a taste of the task, one might examine the work of Kim and Hovy (2006b), who take an opinion frame/semantic frame approach to infer connections between opinions and those who hold them.

3.2 Classifier selection and feature selection

Some machine learning techniques are used in the experiments detailed in this dissertation. It was useful to select, implement, and tune classifiers in such a manner that is supported by previous work in the field.

Several papers have sought to answer a simple question: what is the best machine learning classifier? While there is not yet agreement on a single answer, the papers that pursue an answer offer a great deal of insight into the strengths and weaknesses of various available classifiers. Pang *et al.* (2002) compared Naïve Bayes, Maximum Entropy, and support vector machine (SVM) classifiers on a task of classifying movie reviews as positive or negative; they found that SVM performed best in their experiment. Wang and Manning (2012) examine the performance of Naïve Bayes and SVMs on sentiment and topic classification tasks, concluding that each has its benefits, and positing that using Naïve Bayes log-count ratios as features in a support vector machine outperforms their other experiments.

Using machine learning to tackle sentiment analysis necessitates choosing some machine learning features. While some such machine learning features are common in natural language processing and are not unique to sentiment analysis (such as using a "bag of words" approach that counts or marks the presence of single words to accomplish a task), some more complex features can draw inspiration from linguistics. To some degree, there has been a gulf between linguists' work in analyzing sentiment structure in text and the work of those

in natural language processing. There is, however, some work in natural language processing that draws heavily upon notions of relations between syntax and sentiment. Such work is generally predicated on using parse trees of sentences (hierarchical trees that break down sentences into noun phrases and verb phrases, and such phrases into their constituent nouns and verbs, for example) and dependency features in sentences (subjects and objects of verbs, for example) to help indicate phrases that express opinions. Gamon (2004) experimented with various types of linguistic features, including phrase structure patterns (like observing that a sentence might be composed of a noun phrase + a verb + another noun phrase) and dependency relations (taking note of the nominal subjects of verbs, as an example), and concluded that the more complex linguistic features increased classification performance. Matsumoto *et al.* (2005) took a similar approach, translating dependency sub-trees into machine learning features to help classify the sentiment orientation of movie reviews, and reported that their system was quite accurate at this task as a result. Wilson *et al.* (2004) classified the strength of opinions using syntactic tree features. Joshi and Penstein-Rosé (2009), mentioned previously, use rather complex dependency tree features to try to classify text as objective or subjective. Ng *et al.* (2006) experiment with using simpler subject-verb and verb-object relations in trying to classify text as objective versus subjective and classify subjective text as positive or negative, though their work does not derive much benefit from these relations. The bulk of work in this field, however, seems to suggest that using linguistic features offers better classification performance; which also seems in line with an intuition that systems that use more linguistic features might offer better language understanding.

Finally, some general feature selection guidelines for sentiment classification tasks are opined by Abbasi *et al.* (2008).

While it might be tempting to do a pre-filtering pass to remove objective sentences (those that are not opinionated), experiments by Cui *et al.* (2006) demonstrated that this was neither necessary nor helpful. They also note that high order n-grams improve performance of classifiers in analyzing text fragments for both polarity and strength. It should be noted that this disagrees with earlier work by Pang *et al.* (2002); the latter suggested (non-intuitively) that unigrams were best for sentiment classification, while the former demonstrates an ability to capture increasing levels of nuance in text, including some ironic uses of words.

3.3 Co-training

A co-training algorithm is developed in this dissertation. Co-training is a semi-supervised learning approach that uses both labelled and unlabelled data. Two (or more) classifiers try

to classify the same data into the same classes using different and uncorrelated sets of features ("views", in co-training parlance). The algorithm iteratively builds larger and larger sets of training data when it finds unlabelled examples that at least one classifier can classify with high (estimated) confidence.

Co-training was introduced by Blum and Mitchell (1998). They present an approach for using a small set of labelled data and a large set of unlabelled data to iteratively build a more complete classifier model. Classification features are divided into two views. The main example they provided was a task to classify web pages by topic, where one view was the textual content of the pages, and the other view was composed of the URLs used to access the pages. Two assumptions are made; that each view is sufficient to classify the data, and that the views are conditionally independent given the class label. At approximately the same time, Brin (1999) described an approach similar to co-training, without explicitly calling it such. It has a slant towards pattern matching. It does not seem to be credited in any NLP literature.

Many researchers saw promise in Blum and Mitchell's proposed co-training algorithm, but sought to alleviate some concern about the two assumptions it made about the co-training views. Collins and Singer (1999) build on the work of Blum and Mitchell (1998) by using decision lists combined with co-training to classify named entities. They discuss the sufficiency and conditional independence assumptions in the latter in depth and develop their method using a less strict set of assumptions. Goldman and Zhou (2000) make a significant advance over Blum and Mitchell by relaxing the requirement that the two views be conditionally independent. Their co-training approach, while quite computationally intensive, is able to achieve good results with relatively few iterations. They use different supervised learning algorithms for the two views, positing that each algorithm might "notice" different phenomena in the data. Dasgupta *et al.* (2002) offer more mathematical theoretical support to the work in the Blum and Mitchell paper and demonstrate that one can relax some of the strong assumptions therein. Abney (2002) demonstrates further that the independence assumption of co-training can be relaxed, and that co-training is still effective under a weaker independence assumption. An ability to further relax the independence assumptions in Blum's work was demonstrated by Wang and Zhou (2013); usefully, the authors suggest that, even if the views in a co-training approach are themselves insufficient to classify the data, co-training is still a valid approach. Balcan *et al.* (2004) also offer an ability to relax the strong assumptions of Blum and Mitchell; critically, they propose an *expansion* assumption of the data and offer a set of proofs that data meeting this assumption will derive benefit from a co-training approach (also assuming that the underlying machine learning classifiers are never confident in their classifications in cases when they are incorrect). This expansion assumption presumes that both the positive

and the negative class are composed of several clusters of data – that each class is a union of smaller classes – and that through iteration the classifier can expand its learned model to include more of these lesser clusters. They also assume that the views are at most "weakly dependent", rather than assuming conditional independence as in the Blum and Mitchell paper; and are, in fact, quite explicit in stating that this assumption is the "right" assumption compared to the earlier assumption. It is worth noting that Blum is a co-author of this paper, so the contradiction of the seminal paper should be reliable. This paper also introduces "one-shot" co-training, avoiding the computational cost of iteration. A very practical analysis of the assumptions underlying co-training is offered by Du *et al.* (2011). These authors offer a pragmatic approach to applying co-training to a domain that has a single natural view, requiring the feature set to thus be split into two views for co-training. They offer approaches for splitting the data into two (non-obvious) views and an approach for verifying that the two views satisfy the sufficiency and independence assumptions of the original Blum and Mitchell paper.

The aforementioned work in co-training assumed that the underlying classifiers had high confidence; so high as to be assumed to be perfect. Real-world classifiers can offer excellent performance in some domains, but in domains where classifiers might have more difficulty, having classifiers estimate their own performance can offer an advantage. Confidence-based co-training (where a classifier generates class probabilities, and only those with high estimated probabilities are added to training data in subsequent iterations) was verified to work by Nigam and Ghani (2000). Confidence-based co-training was also used by Huang *et al.* (2012); as opposed to taking a random sample as in Blum and Mitchell's approach, they sample the data where the two views' classifiers agree the most, which is an intriguing approach. Using a Naïve Bayes classifier as the heart of their co-training approach, they saw significantly better results in six of sixteen data sets that they analyzed, compared to the Blum and Mitchell algorithm, and saw significantly worse results in only one of the sixteen data sets that they used.

Limitations of co-training were posited by Pierce and Cardie (2001), who suggest that co-training improves the performance of classifiers to a certain threshold (as high-confidence data are added to the training models in early iterations), and then as more examples are added, performance declines slightly. Similarly, Wang and Zhou (2007) offer several mathematical analyses that extend and validate some of the mathematical axioms provided in Blum and Mitchell (1998). They, too, conclude that the usefulness of co-training depends largely on (and is roughly proportional to) the difference between the two views. Ng and Cardie (2003b) criticize the performance of co-training relative to self-training and expectation maximiza-

tion.

Co-training has been applied to several natural language processing tasks. Wan (2011) uses Blum and Mitchell's co-training algorithm to do sentiment classification on reviews, using Chinese data as one view and English data as a second view, and using an SVM classifier. Machine translation appears to be used to create the views, the output of which may not be perfectly independent from its input; and Chinese and English form a language pair that is considered relatively challenging in machine translation circles. The same author published a similar cross-lingual sentiment classification two years earlier (Wan, 2009). Only two examples of applying co-training to unilingual sentiment analysis tasks were identified. Liu *et al.* (2013a) and Liu *et al.* (2013b) aim to classify the sentiments of Twitter tweets using co-training and support vector machines, while Biyani *et al.* (2013) uses co-training to identify sentiment in an online healthcare-related community. In a similar vein, Li *et al.* (2010b) uses co-training and ensemble learning to classify writers' stated views about objects as either personal or impersonal. Co-training has been applied to other natural language processing tasks including email classification (Kiritchenko and Matwin, 2001), sentence parsing (Sarkar, 2001), word sense disambiguation (Mihalcea, 2004), co-reference resolution (Ng and Cardie, 2003a), and part-of-speech tagging (Clark *et al.*, 2003).

The aforementioned Clark *et al.* (2003) paper uses co-training to train a part-of-speech tagger. Their co-training algorithm aims to maximize agreement between the two classifiers. They conclude that a naïve co-training process that does not explicitly seek to maximize agreement on unlabelled data can lead to similar performance as one that does, at a much lower computational cost.

A few algorithms have been proposed that are similar in principle to co-training. A views-based algorithm is proposed by Amini and Goutte (2010) and contrasted against Blum and Mitchell's algorithm. Riloff and Jones (1999) present a bootstrapping approach they use to build both a semantic lexicon and a set of extraction patterns for use in an information extraction system. Using a small set of examples to begin with, they iteratively use an instance in the lexicon to generate an extraction pattern and vice versa. While this approach is not explicitly co-training, it shares key properties, and the algorithm is similar in practice. A fundamentally similar two-view pattern-learning system is described by Craven *et al.* (1998), and is used to build a knowledge base of relationships (e.g., employee-of, student-of, advisor-of) between people profiled on the web.

3.4 Applications of sentiment analysis

Sentiment analysis experiments in the natural language processing field tend to need human-written text that contains opinions. There are bodies of research working on product reviews (as one might find on Amazon, for example); movie reviews (as on IMDB or Rotten Tomatoes); investor sentiment; political sentiment; and, of course, opinionated tweets on Twitter. Each of these application domains is reviewed in turn.

3.4.1 Product reviews

Product reviews are inherently opinionated, and tend to contain both subjective and objective language, and thus are reasonably challenging and useful to analyze. In particular, reviewers often tend to mention specific aspects of the product they are reviewing and the sentiments they associate with such aspects; a reviewer of a particular cell phone might like its battery life but dislike its screen, for example, and might state so in a single sentence. A system that can extract these product aspects and the sentiments associated with them, rather than classifying whole sentences or whole reviews as positive or negative (an interesting and useful task in itself) could be reasonably described as having a primitive Turing-like understanding of the text.

Analyzing sentiments in product reviews is a useful task. Ghose and Ipeirotis (2011) discuss abstractly why product reviews are useful, while Archak *et al.* (2007) describe how reviews impact pricing power of an item. Pang and Lee (2008) offer (among many other contributions) a survey of research on why product reviews are useful, and also enumerate several compelling commercial applications of sentiment analysis.

Various natural language processing applications have been created that seek to identify sentiment(s) expressed in product reviews.

Reviews that appear on Amazon have been used in some interesting ways. Hu and Liu (2004) annotated one of the first sets of Amazon data and then created a system to try to predict aspect-specific sentiments expressed in the data (although they simplified their evaluation somewhat by considering only whether their system could predict the majority opinion in a sentence; this is discussed in greater detail in Section 7.2). Pontiki *et al.* (2014) describe a shared SemEval-2014 challenge in which participants seek to identify aspects of laptops and restaurants in unlabelled sentences and identify opinions about aspects that have been tagged. Opinions about aspects of products as stated in Amazon reviews were analyzed by Blitzer *et al.* (2007), using reviews of books, DVDs, electronics, kitchen appliances; impressive domain adaptation results were achieved. Their key observation is that positive domain-

specific terminology tends to correlate highly with words like "excellent" and has little or no correlation with "awful"; a property that seems to hold true across domains. They then try to align domain-specific terms across domains, much like one might try to align corresponding terms in two different languages in a machine translation system. Ghose and Ipeirotis (2007) experimented with Amazon reviews, but collected them ad-hoc, making the work difficult to compare to others'. Notwithstanding, their work aims to rank the subjectivity of reviews, and has a human survey about whether reviews are informative and/or whether they influenced readers' decisions. The system predicts the usefulness of reviews. Ding *et al.* (2008) created a system called Opinion Observer that tries to look beyond the mere presence of opinion-bearing words; they try to solve the problems associated with contextual opinion words (one person might like his or her phone to be *small* and their car *big*, while another might prefer a *big* phablet[1] and a *small* sporty car), negation, and idioms. The system works on Amazon product reviews and is based on manually defined linguistic rules. Popescu and Etzioni (2007) work with data of Hu and Liu (2004), performing a similar task. Their OPINE unsupervised information extraction system identifies product aspects, opinions relating to aspects, and determines the polarity of opinions; and they also rank opinions based on strength. Theirs is a rule-based approach (based on parse trees) that uses point-wise mutual information (PMI) relative to a general-purpose web corpus to find domain-specific terms. Scaffidi *et al.* (2007) predict review scores (on a one- to five-star system) of Amazon reviews, a document-level task that at least tries to learn some gradation of opinions. Finally, Kim *et al.* (2006) seek to classify reviews according to their helpfulness, using Amazon reviews and SVM classifiers. This task is not one of assessing sentiment, per se; and the work itself relies largely on a bag-of-words approach plus the length of the review.

Various researchers have tackled the task of finding aspect-specific sentiments expressed in text. Nasukawa and Yi (2003) extract aspect-opinion pairs at the sentence level from mixed web pages, camera reviews, and news articles. They manually create a lexicon of sentiment expressions and some associated simple semantic frames (whether a term takes an object or has a subject), then use dependency parsing and use simple rules to look for instances of the contents of their lexicon. Titov and McDonald (2008a) identify product aspects in reviews and then match tokens from the reviews' sentences that correspond to each product aspect; they use latent discourse analysis (LDA). The resulting system is somewhat limited in that it extracts exactly three aspects per product type; it is a clever clustering model with a high-pass filter, in essence. The nominally sentiment-bearing words extracted that relate to each aspect are dubious. The same researchers also detail similar work trying to extract rateable

[1] A portmanteau of *phone* and *tablet*; a very large smartphone.

aspects of objects using hotel review data (Titov and McDonald, 2008b). Jo and Oh (2011) take an approach to aspect-specific sentiment extraction that uses LDA and a generative model that does not perform as well as supervised models, but also doesn't need sentiment-bearing words to be labelled in the input data. They work with reviews of restaurants and electronic devices. Jin *et al.* (2009) created a system that uses lexical hidden Markov models (claimed to be neither statistical nor rules-based, though it expressly uses a great number of rules and heuristics, and so seems to be largely the latter) to extract and summarize aspect-specific opinions from product reviews of cameras. Their handling of negation is rules-based. They use synonyms and antonyms to augment their training set of product aspects; they search for clauses matching similar patterns with synonyms substituted, for example. They also use regular expressions to group numbers and product part numbers. The main task seems to use only part-of-speech tags. Finding topics and associated opinions, a task not unlike that of aspect-specific sentiment extraction, is pursued by Mei *et al.* (2007) on a set of review-like opinionated text concerning laptops, movies, universities, airlines, and cities gathered from a blog information retrieval system called Opinmind. In the process, they learn the language of product aspects, although this seems to be more of a by-product than a goal of theirs. They model how topic-sentiment pairs change over time. Nigam and Hurst (2004) use a document-level classifier to classify sentences in product reviews, with the goal of extracting strongly opinionated and topical sentences; interestingly, however, their system seems to extract mostly aspect-specific sentences. Their precision was impressive but their system suffered from poor recall (43% recall for positive sentences, and 16% recall for negative sentences). Glance *et al.* (2005) detail a case study of extracting product aspect-sentiment pairs from online message boards to generate marketing intelligence summaries. Brody and Elhadad (2010) use unsupervised methods to mine aspects and sentiments for restaurants and netbooks. Interestingly, they found that the extracted aspects were more representative than a manually-constructed list on the same data, avoiding problems of over-generalization or over-representation (being too granular or too fine-grained in combining similar aspects). On the other hand, their ranking of sentiment adjectives includes a ranking of ethnicities of food, which could be criticized; according to their system, the word *Mexican* as it applies to cuisine is inherently better than *Cuban* (i.e., the very word *Mexican* has a more positive orientation than *Cuban*), which seems specious. The work of Dave *et al.* (2003) tries to be aspect-specific, first mining the product aspect and then the opinion polarities of each aspect; in practice, the only experiment in which they have reasonable results is classifying the polarity of the review (i.e., at the document level), whereas their other experiments appear somewhat ineffective, by their admission (for example, "camera" being a top aspect and one

with an inherently positive orientation; both of these inferences are clearly false). They work with CNet and Amazon reviews. They do note that working with individual sentences caused some problems due to noise and ambiguity. Gamon *et al.* (2005) mine topics and sentiment orientations in car reviews (900 000 sentences total) using clustering techniques to mine, in a simple manner, unigrams mentioned in positive and negative contexts for different makes and models of cars. While this does extract aspect-specific sentiments (VW Golfs have poor "service" and very good "handling", in an example provided), there is no effort to go beyond clustering, so non-sentiment words like "feel" and "lot" are also tagged as prominent aspects with positive sentiment.

There are a few efforts to simply classify reviews as positive or negative, with no particular attempt to find product aspects. Morinaga *et al.* (2002) gather product reviews using a search engine and use linguistic rules to try to classify the reviews as positive/neutral/negative (with no attempt to extract aspects). Cui *et al.* (2006) classify product reviews as positive or negative, ignoring aspects entirely. Usefully, however, they find that using trigram features offers better performance than unigrams or bigrams. They note that parse trees and part-of-speech features did not improve their results noticeably. A system to predict the sentiment (on a scale of one to five) of TripAdvisor hotel reviews is offered by Wachsmuth *et al.* (2014). These authors use argumentation structure (a particular form of discourse structure, wherein an argument might begin with background, then offer a statement, followed by one or more sections of elaboration, perhaps followed by some comparative contrasting statements). They evaluate their work on movie reviews, too. The authors make a compelling argument about how better sentiment analysis systems provide explainability: inferring how the user picked the final hotel rating, rather than just guessing it from a bag of words. Their results were roughly on par with other simpler work on the same data.

Other work aims only to separate the opinionated text in reviews from the objective text. Morinaga *et al.* (2002) created a rules-based system to classify statements as being opinionated (and the opinion orientation, positive or negative) or neutral, with the goal of determining products' reputations. Jindal and Liu (2006) classify whether sentences are comparative or not (a more fine-grained distinction than merely opinionated versus objective).

By contrast, there has been some work that seeks only to identify product aspects, and in some cases, hierarchies thereof. A process of learning a hierarchy of aspects of a single product (building a hierarchical aspect ontology where lower levels in the ontology are increasingly fine-grained) is described by Wei and Gulla (2010). For example, their work might try to determine that a car has stereo; a car stereo has a screen; and such a screen has brightness and resolution. Zhang *et al.* (2010) seek to extract product aspects (though not sentiments)

from online consumer-written reviews of cars and mattresses and online discussion forum posts about phones and LCD screens. They use certain phrase and sentence patterns. Su *et al.* (2006) detail work that, given a set of predefined product aspects, finds other implicit references to those features, using point-wise mutual information (a technique that tries to measure how often two things appear together compared to how often they appear separately). Experiments to identify the "pro and con" reasons that underlie product reviews is described by Kim and Hovy (2006a). The authors use a maximum entropy model (with 66% precision and 76% recall) to classify opinions. This could implicitly identify some mentions of aspects, even though finding aspects is not their stated goal. Zhan *et al.* (2009) use extractive summarization to group phrases mentioning product aspects; in many ways, this is a different way of getting to a similar goal of extracting aspects and sometimes aspect-specific sentiments.

Some emerging technologies have been applied to the task of analyzing sentiments expressed in reviews. For example, Li *et al.* (2010a) apply "skip tree" conditional random fields (CRF) to both movie reviews and product reviews. The authors conclude that skip tree CRF seems better than rule-based, lexicon-based, hidden Markov model, and maximum entropy approaches; puzzlingly, they didn't mention Naïve Bayes nor support vector machines, two very common solutions.

There is some fascinating and creative work on the fringes of the work that analyzes sentiment in product reviews. Rohrdantz *et al.* (2012) analyze customer feedback (that one might gather in an online chat with warranty/service personnel, for example) looking for text that contains the most sentiment, reckoning that analysts could synthesize such information en masse to detect common issues that would need to be addressed to maintain a brand's reputation. In a roundabout manner, this is aspect-based sentiment classification. Although not analyzing product reviews per se, Mostafa (2013) aims to mine social media for brand sentiment. Similarly, Ghiassi *et al.* (2013) mine Twitter for brand sentiment using n-grams and neural networks. (It is worth noting that I try to explicitly exclude brand sentiment in trying to find aspect-specific sentiments.) Tangential work by Jindal and Liu (2007) proposes three separate categories of review spam: false opinions, reviews on brands only (rather than specific products of that brand), and non-reviews (advertisements for the same product or for a competing product; users' questions about the product; or random text). Later, Jindal and Liu (2008) take the work a bit further, trying to classify a large number of reviews and reviewers on Amazon. Similarly, some work on "opinion searching" appears in work by Liu *et al.* (2006), wherein the authors perform searches in a corpus of opinionated text about specific aspects of a specific product; in essence, an information extraction task where the

end goal is aspect-related sentiments. They use only comparative reviews, where two or more products are evaluated against each other in the same sentence.

Finally, a survey paper discussing sentiment detection and opinion summarization in reviews that compares results on four opinion data sets (though not one on which I test my method) is offered by Tang *et al.* (2009).

3.4.2 Movie reviews

Movie reviews present an interesting challenge, in that a typical review discusses aspects of the plot, technical aspects of the given movie (cinematography, score, and so on), and aspects of the persons involved in the film (individual actors, the director, etc.). A compelling system working with movie reviews would thus be able to distinguish these categories so that, for example, the bigram *Tom Tykwer* (a film director) is not learned to be an inherent indicator of a positive review.

As with product reviews, most of the work in analyzing the sentiments expressed in movie reviews tries to classify the documents (the reviews themselves) as positive or negative, rather than trying to identify aspect-specific sentiments. Pang *et al.* (2002) presented some of the first notable work in the field, classifying the sentiment of entire movie reviews (i.e., at the document level) using Naïve Bayes, maximum entropy, and SVM classifiers. Ng *et al.* (2006) classify the polarity of movie reviews. They make a notable contribution in their approach to feature selection, demonstrating that adding bigrams, trigrams, dependency relations, and the polarity of adjectives – while discarding objective phrases – influences results. Goldberg and Zhu (2006) apply a semi-supervised approach to sentiment categorization of movie reviews. Kennedy and Inkpen (2006) use term counting with synonyms, negations, intensifiers, and diminishers to classify movie reviews, then combine these as features in a SVM classifier for better results. Martineau *et al.* (2009) use a bag-of-words system and SVM classifiers for doing document-level sentiment analysis of movie reviews; subjectivity detection in movie reviews; and, parenthetically, for analyzing US Congress debates to classify whether a speaker agrees with or disagrees with a bill under consideration. They use a variation of TFIDF (term frequency/inverse document frequency, a relative measure of how often a particular word or phrase is used in various documents) to upweight sentiment words, which the authors argue are usually infrequent in any given document that they analyzed, despite being used frequently across each corpus with which they worked.

Experiments in identifying aspect-specific sentiments expressed in movie reviews are few and far between. Zhuang *et al.* (2006) argue that mining movie reviews is more challenging

than mining product reviews because individual actors and directors are named; this is not well-supported. Nonetheless, they extract aspect-sentiment pairs, where aspects are defined as elements (screenplay, music, lighting) and people (actors, directors). They manually create an ontology of movie aspects using high-frequency terms (so that if "story", "script", or "screenplay" appears in the review, it is associated with a "screenplay" class). Thus, the salient part of their work is attaching sentiment words to aspects that are extracted by simple rules. The work of Mei *et al.* (2007), mentioned in the previous section, finds topics and associated opinions in movie reviews (among other corpora), a task not unlike that of aspect-specific sentiment extraction.

3.4.3 Other application domains

Product reviews and movie reviews seem to be the two fields where there is a reasonable body of work in trying to identify aspect-specific sentiments. More general sentiment analysis techniques have been applied to other fields as well.

Investor sentiment

A wealth of research seeks to mine the web for sentiments (or changes therein) related to companies and stocks. There seem to be a lot of infrequently-cited singleton papers in this field; perhaps the authors are all now wildly wealthy. As an example, Das *et al.* (2001) mined sentiment on stock message boards using five voting SVM classifiers. Ultimately, however, they find that the sentiment they mine correlates better to sector indices than stocks themselves.

Political discourse

Major elections in the last several years have spawned papers mining sentiment on Twitter (posited as an alternative to polling); such papers go beyond merely counting mentions of politicians and parties, and consider only opinions expressed about those running for election, counting positive and negative mentions to try to predict election results. Sang and Bos (2012) tried to predict the results of a federal election in 2011 in the Netherlands. They took reasonable efforts to normalize their (rather noisy) Twitter data and created a reasonably large annotated collection of Dutch political tweets (annotated as negative or non-negative). Sadly, they chose to discard tweets that mentioned multiple parties; those could have been useful for something more similar to aspect-based sentiment analysis. Their work, in the end, merely counted their annotated tweets and used that ratio, compared to polls at the time, as an

adjustment factor to be applied to the number of tweets that subsequently mentioned political parties. O'Connor *et al.* (2010) analyzed political opinion in the United States, but merely counted positive and negative words in order to predict tweets' sentiments. Calais Guerra *et al.* (2011) analyzed the 2010 election in Brazil, and take an interesting approach, trying to estimate the bias expressed by Twitter users, which they argue can be more useful to predict election results than simple sentiment analysis techniques. They analyze many tweets per user and try to model users' opinions, which also alleviates the problems encountered in some other work of having aggregated results skewed by some strongly opinionated users tweeting more frequently than others. Skoric *et al.* (2012) analyzed the 2011 election in Singapore, but their work, like many other efforts to predict elections from tweets, merely counts mentions of parties and candidates, which barely constitutes sentiment analysis. A recent review paper covering Twitter mining to predict elections is offered by Gayo-Avello (2013).

Some work has been pursued in which the goal is to predict the political affiliation of people by analyzing their social media feeds. Pla and Hurtado (2014) aim to classify Twitter users as right-wing, centrist, or left-wing by using a natural language pipeline built around lexicons. Deeper work is presented by Paul *et al.* (2010), who use topic-aspect modelling (introduced by Paul and Girju (2010), and similar to latent Dirichlet allocation, a more common method for topic modelling) to extract phrasal summaries of reasons given for political opinions; for example, being against single-payer health care because of "too much government". This work takes an interesting approach, but many of the extracted phrases are not inherently opinionated (being in favour of single-payer health care because, say, "my kids have no healthcare", an objective statement).

Social media

Sentiment analysis has been performed on blogs, Twitter (for purposes beyond elections, discussed previously), and online discussion forums. This work is not closely related to the task of aspect-based sentiment analysis, but some useful techniques could be gleaned from such work.

Blogs tend to offer relatively well-written text that may be replete with opinions. Ku *et al.* (2006) mine news articles and blog postings about animal cloning then try to extract opinionated sentences to assess, over time, to what degree a particular author is supportive or non-supportive.

Twitter is a favourite source for text for sentiment analysis tasks, despite the small size of the individual messages and the periodically poor spelling and grammar that might infiltrate the data. Pak and Paroubek (2010) approach sentiment mining on Twitter as a problem that

can be solved at data collection time, collecting data that contain happy or sad emoticons (corresponding to positive and negative tweets). They make some linguistic observations by comparing parts-of-speech in objective tweets compared to positive and negative tweets. Finally, they build a sentiment classifier that uses n-gram features. Go *et al.* (2009) pursue a tweet classification task that predicts whether tweets on a given topic (or about a particular product) are positive or negative, trying a few different machine learning classifiers in the process. They also use emoticons as a basis for determining positive versus negative. Agarwal *et al.* (2011) classify tweets as positive, negative, or neutral by representing tweets as trees (that are similar to parse trees in practice) and use these trees as features in a support vector machine classifier. Thelwall *et al.* (2011) try to measure sentiment strength in tweets that concern current news events. They characterize how sentiment orientation and strength tend to change as current events unfold. Mohammad *et al.* (2013) use support vector machine classifiers to classify the sentiment of tweets (and SMS texts) as well as the sentiment of terms within such messages, a more complex task. They also created a large word-sentiment lexicon in the process. Liu *et al.* (2013a) and Liu *et al.* (2013b) mine sentiment on Twitter using co-training and support vector machines. There are also a plethora of papers claiming to mine Twitter for sentiment by mere term counting (e.g., Tumasjan *et al.* (2010)). These are not of particular interest, as there is an argument to be made that these are neither sentiment analyses nor natural language processing. There are criticisms of the predictive power of sentiment mining on Twitter (e.g., Gayo-Avello *et al.* (2011), Chung and Mustafaraj (2011), Metaxas *et al.* (2011)). Chief criticisms are sample bias, the use of Twitter by political parties and advocacy groups (and the associated problem of tweets that are not factual or not trustworthy), and lack of statistical significance testing in existing work.

Online forums are occasionally examined with sentiment analysis systems (although are generally used more frequently by NLP folks interested in discourse analysis). Biyani *et al.* (2013) use co-training to identify sentiment on an online healthcare-related community.

There is even occasionally overlap between sentiment analysis of social media and other NLP tasks; as an example, Kim and Hovy (2004) perform a sentiment-driven search task, finding people who hold opinions about a given topic, and trying to classify those stated opinions as positive or negative. This bridges the divide between information retrieval and sentiment analysis.

3.5 Learning subjective language

Applications that seek to mine the sentiment of text are, to some degree, built upon earlier work that investigated the fundamental nature of subjective language.

There has been notable work in trying to infer and observe the nature of subjective language through corpus analysis. A decade ago, Wiebe *et al.* (2004) sought to learn subjective language from a Wall Street Journal (WSJ) corpus annotated for opinion pieces and non-opinion pieces, plus a small set of WSJ and newsgroup data annotated at the phrase level. Their work focused on learning using collocations and distributional similarity, and they used a bootstrapping algorithm not dissimilar to co-training. Hatzivassiloglou and McKeown (1997) use a log-linear regression model to cluster and classify positive and negative adjectives, with fairly impressive results (using pairs of conjunct adjectives to infer semantic orientation). Wiebe (2000) takes the work a bit further, performing a classification experiment on the data with some extra semantic features thrown in for good measure. Hu and Liu (2004) perform a similar review mining and summarization task. Riloff *et al.* (2003) learn subjective nouns by bootstrapping using extraction patterns and then classifying further nouns using Naïve Bayes. Riloff and Wiebe (2003) learn such extraction patterns using a co-training-like algorithm, while Wiebe and Riloff (2005) go a bit further by trying to classify sentences as subjective or objective using extraction patterns and only non-annotated text. Choi and Cardie (2005) use extraction patterns to find the "source" of opinions (the opinion bearer and/or the third party reporting about the sentiment). The feature subsumption work of Riloff *et al.* (2006) is somewhat similar to the extraction pattern/dependency tree idea that inspired some work in this dissertation. Turney and Littman (2003) learn the semantic orientation and strength of words by measuring occurrences near known positive and negative words. Taboada *et al.* (2011) investigate lexical aspects of opinion, including adjectives/adverbs, negation and intensification. They consider opinions on a -5 to +5 scale. They build a lexicon and test their work on a corpus of opinions.

Some work goes beyond trying to learn the language of sentiment, and seeks to infer the language of aspects or opinion holders. Choi *et al.* (2006) jointly extract expressions of opinions and the sources of those opinions, as well as inferring a relation that links the two; they borrow constraint-based approaches from operations research. Lazaridou *et al.* (2013) use Bayesian modelling to simultaneously model sentiment, aspect, and discourse structure in an unsupervised fashion.

Wilson *et al.* (2004) try to classify the strength of opinions stated in sentences in the Multi-perspective Question Answering (MPQA) corpus, and take an interesting approach to

annotator disagreement, noting that annotators agree much more frequently on the relative ordering of annotations by strength. Taking the work further, Wilson *et al.* (2009) investigate feature selection for neutral-versus-polar classification and use some dependency tree features; they evaluate their work on the MPQA corpus and offer some advances in working with polarity shifters. In a similar vein, Kim and Hovy (2006c) use trees to identify opinions, orientation, and opinion holders; detecting opinion-bearing words using the MPQA corpus.

Wiebe and Cardie (2005) detail a corpus sentiment annotation task. They describe in excellent detail the frames and linguistic structures in which opinion can reside.

A pros-and-cons approach to classifying sentiment is offered by Hu and Liu (2004) and by Liu *et al.* (2005). Later work by Liu (2010) deals in depth with the language and sentence structure used to express opinion. Sokolova and Lapalme (2011) investigate how words that are not normally emotionally charged (e.g., tall, often) can be used to predict opinions contextually in product reviews.

3.6 Standing on the shoulders of giants

Having reviewed literature on sentiment identification, is seems supportable that one might be able to use product review data (as it contains opinionated and non-opinionated text) to do aspect-specific sentiment extraction (rather than just sentence- or document-level sentiment classification); use machine learning to perform said sentiment extraction; and perhaps even choose a support vector machine (SVM) classifier to do so. Such a technique might, at first glance, seem to be entirely statistical and not linguistic; but, in fact, a plethora of work into the nature of subjective language can inform such an approach. These tenets seem to offer a plausible beginning for an interesting experiment.

Chapter 4

The aspect-specific sentiment problem

This thesis tackles the problem of aspect-based sentiment analysis: figuring out what particular aspects of a product a writer mentions and what positive or negative opinions (if any) he or she might be expressing about such aspects.

For example, consider the sentence:

> *I love my new iPhone because of its amazing screen but the battery is barely sufficient to get me through the day.*

There are three sentiments expressed in this sentence:

- a positive sentiment about the iPhone itself;
- a positive sentiment about the screen; and
- a negative sentiment about the battery or battery life.

The screen and the battery life are two aspects of the product iPhone. I seek to automatically annotate these two aspects in such a sentence and correctly infer that the writer has a positive sentiment about the screen and a negative sentiment about the battery life, without being confused by the positive sentiment about the phone itself. (Perhaps a very simple natural language processing system might see that *battery* and *love* appear in the same sentence, and infer that the writer has a positive opinion of the battery life; avoiding such incorrect inferences is a challenge of doing aspect-based sentiment analysis well.)

This work tries to take advantage of unlabelled data to find these aspects, rather than using only human-annotated data; the former is much cheaper and easier to procure, and is more readily available.

The co-training algorithm developed to do this aspect-specific sentiment analysis is one that offers high precision: that is, it is highly likely to get its predictions correct, at the expense of making fewer predictions (or, put more archaically: it makes sins of omission, but few sins of commission). High precision matches a naïve intuition of "correctness" fairly well; and high-precision, lower-recall systems can be combined in ensemble learning to create powerful voting systems like IBM's Watson.

While sentiment classification of text has been attempted computationally for roughly twenty years now (see Section 3.4), aspect-specific sentiment identification is a newer task in natural language processing that is undergoing active research at present (e.g., as part of the SemEval-2014 competition, wherein there was a shared task called Aspect Based Sentiment Analysis[1] that attracted 126 valid submissions from 31 teams).

Co-training, similarly, has been used for various tasks since 1998 (see Section 3.3), but has never, to my knowledge, been applied to the task of aspect-specific sentiment extraction.

Aspect-based sentiment extraction is an inherently useful pursuit. It is a step towards having computers understand the nuance of language; and, more practically, it can be used to compare two or more products based on their aspects. For example, a particular consumer shopping on a large online retailer might be interested in phones that only have oversized screens so as to replace a tablet; while another consumer might be much more interested in a compact phone that fits in a child's hands (one that has a small screen). A phone with a small screen might receive a poor overall review from the former consumer, since it is a poor match for his/her needs, and the latter consumer would be disserved by a set of poor reviews for a phone that fits his or her needs well. Current online retailing sites tend to only have one overall numeric score for a given product; whereas if a largely complete set of product aspects and aggregated aspect-specific ratings were available, a consumer could better select appropriate products. (An alternative would be for such sites to force their users to rate products on all their aspects, like filling out a report card, which would presumably be time-consuming and irritating for those contributing reviews.)

Reviews written by casual consumers are now prevalent and widely available, and yet little such data has been annotated for machine learning research (as to do so is costly and difficult; humans are poor at producing annotations upon which they can agree). Co-training seems to be one of the few techniques that can take advantage of unlabelled data to perform tasks that might otherwise be better suited to supervised machine learning. Aspect-specific sentiment extraction seems to be well-suited to supervised machine learning, so it may follow

[1] A description of the Aspect Based Sentiment Analysis SemEval-2014 task is available at http://alt.qcri.org/semeval2014/task4/

that using co-training to do aspect-based sentiment extraction on product reviews might be a productive pairing.

Chapter 5

Data selection and preprocessing

The software developed for this dissertation identifies aspect-specific opinions in text. Two data sets were used: a set of product reviews from Amazon, and a set of restaurant and laptop reviews used in a SemEval-2014 competition. A brief description of these data is offered along with a rationale for using them.

It was necessary to process the data somewhat before experimentation; the preprocessing steps are also described herein, largely for repeatability.

Finally, a healthy criticism of the data is offered; as with any human-classified data, there are errors and other phenomena therein that could reasonably construe a performance ceiling for machine classification.

5.1 Selecting useful & usable data for experimentation

There does not appear to be a universally-accepted set of metrics for evaluating the quality of a data set for natural language processing applications.[1] It's difficult to select an ideal data set when there is no agreement on what constitutes a good data set.

Accordingly, I chose (and applied) two criteria for evaluating possible data sets for experimentation:

- Is the data set useful?
- Is the data set usable?

[1]Luminaries in the field seem to have informal opinions on which data sets are good or bad, but there seem to be no published editorials on the matter. Passionate discussions on the saliency of data sets do pop up on Corpora-List (http://www.hit.uib.no/corpora/), a vibrant discussion group about corpora for NLP applications.

Useful is taken to mean a data set that, should the experiments be successful, could tell us something interesting about how language works; could have an obvious and plausible application in the real world; and one that approximates normal[2] human writing (as opposed to, say, structured writing like medical records or survey cards).

Usable is taken to mean a data set that is annotated for the task at hand (namely: aspect-specific sentiment), and is tagged at a sufficiently fine-grained level of detail; one that is freely available (making it easy for me to use and also making it trivial for others to try to reproduce my work if they so choose); one on which others have attempted sentiment-extraction tasks (so as to have a standard against which to compare results); one that has a good mix of opinionated and objective sentences; is sufficiently large; does not have such obviously stated sentiments so as to be banal; and, on the other hand, is not so informal and full of misspellings, emoticons, and other challenging tokens as to be unparseable.

It would also be desirable that the annotations in the data set be correct. For all but the most trivial of annotation tasks, this is a high bar; perhaps unreasonably so. At a minimum, a good data set is one that is produced by multiple annotators, where the agreement among the annotators is reported, and where there is high agreement among the annotators.

Riezler (2013) makes an interesting argument about some synthetic data sets created for natural language processing applications (including data sets where the authors of a paper perform the annotation task): that they cause a circular/self-referential problem in the research based on them, as the data themselves are created with the same general properties and biases as, for example, the machine learning features used in a typical machine learning application that might aim to classify the data. In particular, the author cites sentiment extraction tasks on Amazon product reviews and on IMDB movie reviews as being largely immune to this property, noting that "a real-world task that is *extrinsic* and *independent of any scientific theory* avoids any methodological circularity in data annotation and enforces an application-based evaluation" (emphasis added). Both the Amazon reviews and the SemEval-2014 data used in the experiments in this dissertation appear to be both extrinsic and independent in this manner.

Finally, an ideal data set is one that has been used for other similar research tasks that

[2]A good counter-example is that of newspaper articles used for machine summarization experiments, where simply taking the first paragraph – or indeed, the first two or three sentences – of the article is generally an excellent summary. The forced structure of news articles suggests that tools that are very good at summarizing news articles may be largely useless in summarizing other human-written text. Specifically: a "summarization" tool that merely takes the first three sentences of a news article would create compelling summaries of the articles but would have little use on other text. In selecting a data set for the work described in this dissertation, I had a strong desire to avoid this characteristic.

could be used as either a baseline or a meaningful comparison for my experiments.

This final point informed my search for a good data set. I surveyed data sets used in sentiment mining tasks in NLP papers (Section 3.4), and informally evaluated them according to the aforementioned criteria.

I chose one data set prepared by Hu and Liu (2004) that contained five sets of product reviews from Amazon.com. These reviews cover five products:

- an Apex AD2600 DVD player (99 reviews containing 740 sentences)
- a Canon G3 digital camera (45 reviews containing 597 sentences)
- a Creative Labs Nomad Jukebox Zen Xtra 40GB MP3 player (95 reviews containing 1716 sentences)
- a Nikon COOLPIX 4300 digital camera (34 reviews containing 346 sentences)
- a Nokia 6610 cell phone (40 reviews containing 546 sentences)

The reviews are written by the general public and are not edited (nor are they obviously, in many cases, spell checked nor proof-read). A typical review consists of a short title chosen by the reviewer and five to ten sentences of review text. Some reviews contain bulleted lists (i.e., a sentence consisting merely of "pros:" followed by several sentences consisting of mere sentence fragments).

The data contain spelling errors, incorrect choice of homophones, and other such characteristics as one might find in casual writing.

The data in this data set have been human annotated sentence-by-sentence. Each sentence is tagged with zero or more product aspects (like screen size for a digital camera, or audio quality for an MP3 player) and a corresponding sentiment orientation in the range of [-3, 3], where a ranking of negative three suggests a very negative opinion about the aspect, and positive three indicates a very positive sentiment about the product aspect. Aspects that are mentioned in an objective (non-sentiment-bearing) context are not annotated. The words in the sentence that indicate the sentiment are not annotated; and, in fact, are sometimes challenging to identify for a human.

The annotators intended to annotate only aspect-specific sentiments in the data, so there should be no annotations of sentiments about the products themselves. The annotators were only partially successful in this endeavour; many product-specific sentiments appear in the data. This of course presents two interesting real-world challenges of ignoring the noisy

presence of product-level sentiments: those that are tagged but should not be; and those that are not tagged.

There are long-distance anaphora in the reviews, which means that products, product aspects, and sentiments are sometimes not concretely expressed in a given sentence, or may be present only by pronominal reference.

The data have been used for sentiment identification tasks in at least four other peer-reviewed research papers.

In short, this data set appears to meet my criteria for being useful and usable.

I chose a second data set prepared originally by Ganu *et al.* (2009), containing restaurant review text from Citysearch New York, which was modified and enlarged for an aspect-specific sentiment extraction task at SemEval-2014 (Pontiki *et al.*, 2014). This SemEval data set consists of sentences extracted from reviews of restaurants (3041 training sentences and 800 test sentences) and reviews of laptops of different brands (3045 training sentences and 800 test sentences). Aspects that appear in the sentence are tagged and assigned a sentiment polarity of positive, neutral, negative, or "conflict", the latter referring to cases where a both positive and negative sentiments about the aspect appear in the same sentence (along the lines of "the service was friendly but slow"). The data set contains only sentences from reviews, not entire reviews with titles as in the Amazon data set. The data are, once again, human-written by casual writers, but, subjectively, the quality of the writing appears to be somewhat better than in the Amazon reviews; spelling errors and instances of odd formatting (like informally-bulleted lists) seem to be fewer in number.

As in the Amazon reviews, the sentiment-bearing words themselves are not tagged, so it is up to the software to determine in some other manner how and where the sentiment is expressed in the sentence.

This particular data set offers a good basis for comparison for my approach to sentiment extraction. The competition drew 57 submissions for the first phase of evaluation and 69 for the second phase of evaluation.

This data set, too, appeared to meet my criteria for being both useful and usable.

5.2 Partitioning data for experimentation and testing

There is a risk when developing machine learning systems of creating systems that are closely tied to the data used during development; for example, inadvertently selecting machine learning features that model the decision-making process used by the expert annotators of the data (Riezler, 2013). Such features might carefully and tightly model the development data but may

not generalize well .

Accordingly, I developed my system using only a subset of the data. I used the Apex DVD player data during development; it was the median-sized set among the five products' data annotated by Hu and Liu (2004).

The support vector machine classifiers I selected have two parameters that need to be tuned (C and γ, which are explained in detail in Section 6.3.4). Tuning these parameters on all data tends to lead to overfitting, where the learned model accurately represents the training data but does not generalize well to data that have not been seen, such as testing data. Accordingly, for SVM tuning, I partitioned the data. I used the first 20% of each of the five data sets from the data sets prepared by Hu and Liu (2004) and took a logarithmic mean of the best C and γ parameters achieved on a coarse grid search over each such development set.

For training and testing the system on the Hu and Liu data, I used fivefold cross validation or used different products for training, co-training and testing phases.

The SemEval-2014 data were made available only after all my experiments on the Hu and Liu data were complete, and so were not used as development data; they constitute purely training and test data, as they were completely unseen as the experiments were developed. While these data were made available partitioned into development, training, and test sets, I eschewed the development data entirely (as it was rather small; and since I re-used the same C and γ learned from the Hu and Liu data, there was no particular benefit in using it), and used only the training and test data.

In the co-training experiments with the SemEval-2014 data, I divided the training data sets in two: one for the initial seed, and one to be considered unlabelled data for co-training.

5.3 Tokenizing the text

The Amazon data set was tokenized by its authors at time of creation, which is both a blessing and a curse. It makes different research on the same data set more comparable (as authors of conference papers tend to omit basic details about how they tokenized their text, which can impact what counts as a word/token, which in turn influences token-level classification tasks); on the other hand, there are some unfortunate tokenization properties. The SemEval-2014 Task 4 data, on the other hand, were not tokenized; tokenization is undertaken by my system.

There are several notable characteristics of the tokenization of these data sets.

5.3.1 Amazon review data

The Amazon data comes pre-tokenized. For example, contractions in the data have already been split into their constituent words, as in:

> *play[-2], disc[-2]* **wo n't** play a lot of discs .
>
> *player[-3][p]* who knows how many uses **you 'll** get out of it before it craps out , but it probably will .

Note too that punctuation is clearly separated by spaces, which is convenient.

Capitalization has been removed from the data set, which makes the task a bit harder; it's less certain, for example, whether an instance of the word "canon" refers to the camera brand, the military weapon, or religious law; whereas, capitalized when not appearing as the first word in a sentence, one might be able to better conclude that it is the camera brand sense. I made no effort to reconstruct the capitalization.

Hyphenation is inconsistent, which is a shame. While, in general, hyphens and double hyphens are surrounded with spaces, this is not always the case, as in the following two examples:

> *[no aspects]* i purchased this as a christmas gift on **12-4 - 03** .
>
> *picture[-3]* it worked from 12-26 to 1-9 at which time the picture failed completely (see other negative **reviews-my** unit was n't the only one this happened to .

However, for consistency, I left such tokenization errors as-is. In the latter example, the tokenization tools in the Stanford CoreNLP package (as described in section 5.4) were able to separate *reviews - my* into three separate tokens. The Amazon data is in effect tokenized both by its authors and by the Stanford CoreNLP tokenizer; the latter provides some consistency with the tokenization of the SemEval-2014 data and can correct some minor tokenization errors that are present in the data.

5.3.2 SemEval-2014 data

The SemEval data do not come pre-tokenized as did the Amazon review data. For example:

```
<sentence id="337">
   <text>However, the multi-touch gestures and large tracking area make having an
       external mouse unnecessary (unless you're gaming).</text>
   <aspectTerms>
      <aspectTerm term="multi-touch gestures" polarity="positive" from="13" to="33"/>
      <aspectTerm term="tracking area" polarity="positive" from="44" to="57"/>
      <aspectTerm term="external mouse" polarity="neutral" from="73" to="87"/>
      <aspectTerm term="gaming" polarity="neutral" from="115" to="121"/>
   </aspectTerms>
</sentence>
```

Note that the contraction *you're* should be manually tokenized into two tokens; that the two parentheses need to be considered separately from the tokens to which they are adjacent; and that the final period needs to be separated from the closing parenthesis that precedes it. As with the Amazon review data, hyphenation in the SemEval-2014 data is rather arbitrary and is similarly cleaned up during tokenization.

Although not specifically related to tokenization, it may be worth pointing out that the aspects are tagged in the XML representation, along with the initial and final character positions (the *from* and *to* attributes) of the surface form of the attribute. This would have been nice to have in the Hu and Liu data.

For my experiments, the SemEval-2014 data were tokenized using the Stanford CoreNLP tools, which in effect yields sentences that are tokenized much like the pre-tokenized Amazon data. As the SemEval-2014 data and Amazon review data were used in independent experiments, there was no particular effort made to make the tokenization perfectly consistent between the two data sets; though, since they both pass through the Stanford CoreNLP tokenizer, the output is similar in both cases.

5.4 Preprocessing the sentences

Once the sentences in the data set were tokenized, further preprocessing was minimal.

I parsed the sentences with a Stanford CoreNLP 3.3.1 pipeline (Manning *et al.*, 2014). The pipeline I chose consists of the following steps:

- `tokenize`: separate the sentence (a single string input) into tokens where whitespace or punctuation occurs
- `ssplit`: split the sentence (as it appears in the source data, in which sentences almost always appear one at a time) into sentences; in almost all cases, this is effectively a non-operation, and was performed solely in case there were instances where the parser thought that an input sentence was in fact multiple sentences

- `pos`: tag each token with its predicted part of speech using a tagger built on both a maximum entropy model (Toutanova and Manning, 2000) and on a cyclic dependency network (Toutanova *et al.*, 2003), which is a form of maximum entropy model that considers text sequences both left-to-right and right-to-left
- `lemma`: lemmatize each word (non-destructively)
- `ner`: flag each token that is predicted to be a location, person, or organization, implemented by Finkel *et al.* (2005) using conditional random fields and some extra techniques to incorporate information about long-distance dependencies in the sentence
- `truecase`: predict whether, in perfect writing, a given token would normally have an initial capital letter (though it appears that this particular module may have been trained on news corpora or the like; it seems largely ineffective herein)
- `parse`: parse the sentence into both syntactic and semantic dependency trees using the English lexicalized partial context-free grammar (PCFG) syntactic parser implemented by Klein and Manning (2003) and the dependency parser implemented by De Marneffe *et al.* (2006)
- `dcoref`: try to resolve coreferences; that is, try to link any pronouns in the sentence to the nouns to which they refer, using a hierarchical set of models developed by Raghunathan *et al.* (2010) and Lee *et al.* (2011), combined with a rules-based system contributed by Lee *et al.* (2013)

After all this sentence processing, a fairly rich representation of the sentence is available for further analysis.

The Stanford CoreNLP tools offer very good performance. For example, Socher *et al.* (2013) noted that the current incarnation of the Stanford parser achieves an impressive F_1 score of 0.904; they compare the Stanford parser to several other commonly-used parsers, and conclude that its performance exceeds that of the previous Stanford partial context-free grammar parser; the Collins parser; the Berkeley parser; and scoring only very slightly lower than the Charniak parser (which achieved an F_1 score 0.006 higher). Furthermore, the ability of the Stanford tools to do named entity recognition, correct capital letters, and resolve coreferences made it useful for more than mere parsing; being able to use a single tool to manage the majority of the pre-processing work was advantageous.

While the Stanford parser is rather well-regarded and offers fairly impressive performance, it is not perfect. One of its key proponents argues that, for example, while its part-of-

speech tagging is very impressive at 97.3% token accuracy (as of 2011), it offers a mere 56% sentence accuracy, and further improvements using supervised or semi-supervised learning may be quite limited (Manning, 2011). Because I have chosen to use linguistic features, I am subject to the constraint that all these preprocessing steps are a "best guess" by the Stanford toolkit, and are not in fact ground truth. This could limit or alter the success of my techniques.

I noticed some obvious annotation errors in the data. In particular, despite reasonable instructions to the contrary, some annotators in both the Amazon data and the SemEval decided to annotate product-level sentiments. I removed non-aspectual (that is, product-level) opinions in a very crude and simple manner; for each set of reviews, I assigned a single term that represented the generic product category so as to filter out incorrectly tagged features of the product ("dvd player", "camera", "player", and "phone", as appropriate in the Amazon review data, and "restaurant" and "laptop" in the SemEval-2014 data). For example, if a particular sentence in the Amazon review data were tagged as "dvd player[+2]", I chose to omit that annotation from both the training and the testing data. This penalizes the performance of the system I developed to some extent; but it is a more fair and realistic way of evaluating the system's ability to differentiate between aspects and products.

Similarly, I decided to omit the brand name of each product in the Amazon review data, as it was known ahead of time. (This data was not easily inferred at the sentence level in the SemEval-2014 data, though perhaps using a master list of known restaurant names and laptop brands would have been useful.) My goal, in omitting brand names tagged as aspects in the Amazon review data, was to discard brand and brand reputation as a product aspect: just as it would be undesirable to consider the token "Apple" to be a positive sentiment word just because people tend to like their iPhones, so too was it undesirable for such a brand name to become an aspectual feature word.

I encountered no difficulties parsing the tagged aspect-sentiment pairs in the XML in SemEval-2014 data, as the files were well-formed. On the other hand, I made extra efforts to parse the simple syntax of the sentiment annotations in the Amazon dataset. For example, there were cases where the square brackets used to denote the orientation of the opinion did not conform to the specification (for example, *sound quality[+2)* instead of *sound quality[+2]*). I developed a small set of rules to deal with all cases where the aspect parsing failed. I was ultimately able to parse all the annotations as they were presumably intended, but this might help or hinder comparison to others' work on the same dataset.

I made no effort to correct spelling nor punctuation in any of the data (though such a task, if feasible, might improve parsing quality, which would presumably in turn improve my results).

The Amazon review data contained review titles, which I chose to omit from my analyses, as they were never tagged with sentiments about product aspects. There were no such review titles in the SemEval-2014 data.

Finally, in the Amazon review data, I separated sentences for analysis even when they appeared in the same review; while this breaks long-distance anaphoric resolution, it provided two benefits: it made it trivial to sort sentences into roughly equally-sized bins for cross validation, and it was a natural fit for evaluation, as the annotations were at the sentence level. The SemEval-2014 data were already separated thusly; there was no concept of a review as an atomic unit in the SemEval-2014 data, so no further effort was necessary.

5.5 Tagging aspectual and sentiment words

The SemEval-2014 data had aspects tagged explicitly, including character positions. For example:

```
<text>But the staff was so horrible to us.</text>
<aspectTerms>
   <aspectTerm term="staff" polarity="negative" from="8" to="13"/>
</aspectTerms>
```

It was thus trivial to reconcile the aspects to their surface expressions for further processing, even when multi-word expressions were present. However, sentiment expressions were not explicitly tagged (e.g., *horrible* in the sentence above), so it was necessary to develop a heuristic to tag them.

It was more difficult to reconcile the product aspects with their lexical/surface forms in the Amazon product review data. They are not tagged at the word level, and there is no information about where in the sentence the surface form appears; rather, they are simply listed for a given sentence, and the aspects listed for the sentence are not always identical to the surface form appearing in the sentence itself. Accordingly, it was useful (and perhaps even necessary) to try to reconcile the annotations for each sentence and the words/tokens within the sentence.

Sentences in the Amazon data set are labelled with their product aspects and corresponding sentiment orientations (indicating that the writer has expressed an opinion about that particular product feature). For example:

> *color[+2]* silverish color really adds a special touch .

Notably, while the *colo[u]r* aspect is directly mentioned in the sentence, the token itself is not tagged in the data; in no place does the data set explicitly point out that the second token

in the sentence is a product aspect. Furthermore, there is no annotation that indicates which word(s) indicate the positive sentiment that has been annotated; *adds a special touch* is one option, though *really adds a special touch* and simply *special* could be argued as well.

Accordingly, I wrote a heuristic to tag aspect tokens (given the sentence-level aspect-opinion tags) in the Amazon review data and tag sentiment-bearing tokens in both the Amazon review data and the SemEval-2014 data.

5.5.1 Identifying product aspects in Amazon review data

I identify direct mentions of product aspects in the Amazon training data using a heuristic. This is not ideal, but is a limitation of the data set chosen. The heuristic seems to work well in practice.

Some aspects do not appear in the sentence exactly as extracted. Some words are slightly different in morphology (plurals, for example), so I compare both full tokens and their lemmas (such that if the data set has annotated the surface token *qualities* in the sentence as the aspect of *quality*, I can reconcile the two easily). There were particular challenges in tagging multi-word aspects like *picture quality*, which appeared in several cases as *quality of the pictures* and in other cases as *photo quality*. In the former case, I tag the tokens *quality* and *pictures*; in the latter case I only tag the token *quality*, unfortunately. My simple heuristic tries to tag all tokens in the labelled aspect, if they exist, but does not search for synonyms.

Some aspects are not mentioned explicitly at all in the sentence. I handle some indirect mentions of aspects, wherein an adjective implicitly suggests a feature. For example, "My car is fast" suggests that the product *car* has an implicit feature *speed*. To handle such cases, for all tokens tagged by the Stanford parser as an adjective (JJ), comparative adjective (JJR), or superlative adjective (JJS), I look up the first sense of the adjective in Princeton WordNet 3.1 (Fellbaum, 1998). I then examine its *attribute* relation(s), if any exist; these are nouns for which the adjective expresses values (Miller, 1995). Continuing the previous example, the first sense of *fast* in WordNet has an *attribute* relationship to the synset *speed, swiftness, fastness*. I tag the adjective with the first term in the attribute synset (in this example, *speed*).

In the data set, some of the annotated aspects are explicitly labelled as not being present in the sentence (particularly in cases where the only lexical realization of the aspect is a pronominal reference). I did not attempt to reconcile these cases, particularly as the annotation of these pronominal references was rather inconsistent in the annotation. This would necessarily reduce recall in the results.

From a machine classification perspective, the product aspect data is inherently skewed;

most tokens in the sentence are not product aspects. This makes the classification task more difficult.

There is a final uncertainty in the data that, mercifully, rarely manifests itself. In the SemEval-2014 data, because the character positions delineating the product aspects about which a sentiment is expressed are made clear, it is easy to reconcile ambiguous opinions about aspects in a given sentence; so, if an aspect is mentioned twice in a sentence, the SemEval data make it explicitly clear (excepting annotation errors) which mention of the aspect is the subject of the sentiment. Not so in the Amazon data, where if an annotated aspect appears in the sentence multiple times, it must be guessed which instance in the sentence corresponds to the annotation. In such cases, I tag all instances of the aspect in the sentence; this is not ideal, but is perfectly serviceable.

I thus created a usable (though somewhat imperfect) heuristic to use the sentence-level annotations in the Amazon review data to tag individual tokens in the sentence that indicate product aspects; and used to annotations in the SemEval-2014 data to produce similar output.

5.5.2 Identifying sentiment terms

Sentiment-bearing or opinion-conveying tokens are not explicitly annotated in any of the data used. This is rather a shame, as it would be useful if there were some annotation to indicate what lexical terms are used to indicate opinions. There is perhaps interesting corpus-based linguistic research to be pursued on that matter (beyond existing work into correlating tokens with sentiments en masse).

Given a set of annotated product aspects in a sentence and their corresponding opinion polarities, there appeared to be two obvious ways to try to infer where the opinions were being expressed in the sentence:

- Use a bag of dependency relations in the sentence as features and use a classifier to decide if the product aspect is positive or negative in its context. (For example, for a given product aspect, its features could be the shortest dependency relation to every other token in the sentence plus a guess as to whether each related token were positive, negative, neutral, or an indicator of negation).

- Classify words in the sentence as being likely to bear sentiment, then use close dependency relations to match the sentiment words to their corresponding product aspect words, if any.

I chose the latter method, largely for its similarity to the product aspect identification task. Thus, I created a module that tags tokens in a sentence as being either inside or outside opinion-bearing clauses. For example, in the following sentence, taken from the Hu and Liu data, my system would ideally tag the token *special* as being a positive opinion-bearing word:

color[+2] silverish color really adds a special touch .

This sentence demonstrates the complexity of tagging sentiment words. An easy algorithm would simply tag all direct modifiers of *colo[u]r* (e.g., "nice colour"). In this sentence, *special* modifies the noun *touch* instead of *colo[u]r*, and yet it is still the key bearer of sentiment here; if *special* were substituted with *horrifying*, for example, the polarity of the sentiment would change. (By similar logic, if *adds* were replaced with *takes away*, the sentiment polarity would change; but in this case, *takes away* could be better considered a negation instead of a sentiment word.)

The opinions annotated in the Amazon review data were given a polarity between -3 (very negative) and +3 (very positive), inclusive. No data were explicitly annotated as neutral; rather, when a product aspect was mentioned in an objective manner (e.g., "the camera has a *viewfinder*"), it was simply not annotated. I chose to classify opinions as merely positive or negative (or -1 to +1); the annotations seemed pleasantly finely grained, but I did not see an obvious use for that level of granularity. As a consequence, I did not consider tagging intensifiers like *very* or *really*.

The aspect opinions annotated in the SemEval-2014 data were given polarities of negative, positive, neutral, or "conflict". I treated the conflict case as instances of both positive and negative; that is, I would convert one annotation of *conflict* into one positive annotation and one negative annotation. An added bonus of this approach is that, in cases where my system was only able to identify either the positive or the negative sense of the conflict, it would at least achieve half marks, which seems fair. There were relatively few cases where the *conflict* annotation was used. In this manner, I was able to use the same three sentiment classes (positive, objective, negative) as in the Amazon data.

Since the sentiment-bearing words were not themselves tagged in the data, I decided to bootstrap them from the sentiment word lexicon developed by Hu and Liu (2004) and Liu *et al.* (2005); the compiled lists[3] of positive and negative terms are available at Bing Liu's website.[4] I did not take any particular measures to tag only sentiment-bearing words that related to

[3]http://www.cs.uic.edu/~liub/FBS/opinion-lexicon-English.rar
[4]http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html

product aspects; I simply tagged them all and relied on a heuristic later in the pipeline to reconcile them with product aspects.

Bootstrapping from such emotion lexicons seems to be supported in literature on the matter. For example, in discussing extracting sentiment and emotion from text, Mohammad (2012) posits that, while "n[-]gram features tend to be accurate, they are often unsuitable for use in new domains. On the other hand, affect lexicon features tend to generalize and produce better results than n-grams when applied to a new domain". That is, since n-grams by their nature are a mixture of domain-specific and language-general terms, separating the two is a more useful approach to determining sentiment, since the sentiment lexicon is reusable in new contexts.

The appearance of an opinion word in a sentence does not necessarily indicate that a sentence is opinionated; and if the sentence happens to be opinionated, the presence of a particular opinion word gives no particular hint as to the polarity of the opinion (Liu, 2010), due to negation[5], for example. Notwithstanding, my system classifies sentiment words independently from product aspects and other words in a given sentence, classifying only whether, at a given token, the system believes that it is inside or outside a sentiment-bearing expression.

The opinion-bearing word data is inherently skewed; most tokens in the sentence are not indicating opinions. This makes the classification task more difficult.

The SemEval-2014 task 4 restaurant data seems to contain more colourful, varied, and complex negation language (e.g., "The two waitress's *(sic)* looked like sucking lemons" is tagged as a negative opinion of *waitress*) than either the SemEval-2014 task 4 laptop data or the Amazon review data. Modal verbs also seemed to be used more frequently than in the laptop data to indicate pseudo-negation (e.g., "The staff should be a bit more friendly", where the modal verb *should* implies that the staff are insufficiently friendly; this particular example is annotated as a negative opinion of the aspect *staff*). This restaurant data also contains more cases of *conflict*, where both positive and negative sentiments are offered about a single aspect in the same sentence (e.g., "The *guac[amole]* is *fresh*, yet *lacking flavo[u]r*...").

5.6 Considering some performance limitations

The limitations of the data sets chosen and the choices made in pre-processing them impose a necessary and not entirely unreasonable performance ceiling. This performance ceiling is independent of the experiments themselves.

[5]The features for classifying sentiment words include a feature noting whether it is a part of a local negation clause (as determined by the sentence parser).

Text written by the general public is imperfect, and common phenomena in casual writing can affect the performance of NLP systems. Consider, for example, that simple misspellings can affect recall. In data such as the Amazon data used herein, where annotations are added by typing rather than by marking them in a graphical user interface, the spellings of the annotations do not always match the form that appears in the annotated sentence, and since many misspellings will be unique in a given data set, it is difficult for a system to learn and account for all such errors. Eisenstein (2013) describes such issues admirably and considers several approaches to adapting imperfect writing for NLP applications (e.g., domain adaptation and normalization), and concludes implicitly that it seems most defensible to process the given text as-is and accept the consequences of poor performance compared to, for example, experimenting with newswire text.

It is inherently challenging to annotate a data set for product aspects and their associated sentiments. Carletta (1996) proposes the kappa statistic as a measure of agreement among multiple data annotators as one that can accommodate chance agreement, and criticizes four other common metrics for analyzing annotator agreement on a boundary-marking task.

Sadly, no agreement measure is mentioned for the data annotation for the Amazon review data. Hu and Liu (2004) were themselves the annotators for the Amazon data. While they have done a commendable job, in general, they concede that the annotation task is "somewhat subjective", and expressed particular difficulty in some cases deciding whether an opinion was present at all. They decided to use consensus to annotate these difficult aspects of data, rather than measuring the agreement with any of the common metrics mentioned by Carletta (1996). Thus, for the computer classification task, it will be difficult to define the performance ceiling and floor to be expected by inter-annotator agreement and chance agreement, respectively. Using a simple metric, if the annotators agreed on their annotations, say, 90% of the time, and the computer agreed with each of the annotators roughly 90% of the time, one could perhaps form a reasonable argument that the computer has achieved human performance. In practice, if we knew the agreement rate of the two annotators, we could calculate the kappa statistic for the computer, and if it were greater than, for example 0.6., indicating substantial agreement (Viera *et al.*, 2005), we could again make a reasonable argument that the computer can perform the task about as well as humans.

The issue surfaces when one looks at some of the aspects that have been annotated. For example, the aspect "DVD player" is annotated on multiple occasions in the Apex DVD player data set; "DVD player" is most decidedly not a product aspect of a DVD player like "sound quality" or "picture quality" or "ease of use". Although I have incorporated a correction for this particular example in the aspect tagging task in the Amazon data, no such simplification

was reasonably possible in the SemEval-2014 data. Further, synonyms of "DVD player" like "unit" appear as well; I did not try to fix anything beyond the most obvious generic term for the product. Due to the sheer difficulty of the tagging task, it seems reasonable that some aspect annotations are missing and that some are incorrect or unsuitable.

In one instance, "recognize" is annotated as a product aspect of the DVD player. It is clearly not a product aspect in and by itself, although there may be a related implicit aspect of "disc recognition" or "fault tolerance" or "scratch tolerance". In the DVD reviews, the word "recognize" only ever appears in negative reviews, but is not inherently negative. A naïve approach might infer that "recognize" is both a product aspect and a negative sentiment word; both inferences are supported by the data, but both are incorrect. This offers a hint that the task is a difficult one; and that the annotations are such that some degree of disagreement with the annotations is necessary if the system is to offer defensible results. This need for divergence from the annotations dictates a performance ceiling, and one that is again difficult to measure. Vexingly, the system may occasionally do a better job of annotating the data than humans did (that is, where the gold standard annotations are poor); but the system gets no credit for doing so.

Finally, the data pre-processing tasks themselves are prone to introduce errors that could limit the effectiveness of the experiments.

Lack of correct punctuation can be particularly vexing for sentence parsing. Consider two sentences (Figure 5.1) that differ only by a comma, shown with their respective Stanford CoreNLP dependency parses.[6]

The first sentence in Figure 5.1 is a simulated sentence showing a parse that captures the writer's intended statement, where *voice* is the nominal subject of *clear* (linked by the copular verb *is*). The second is a sentence that appears in the Nokia phone reviews in the Amazon data. Note that in the latter, the closest semantic link between the sentiment-bearing word *clear* and the product attribute *voice [quality]* is two "hops" away, and two strange hops at that. That is to say: the implied pattern in the top sentence of "a sentiment-bearing word can be linked to a product aspect by a single nsubj (nominal subject) relationship" should hold true in other sentences; whereas the implied pattern in the lower sentence of "a sentiment-bearing word can be linked to a product aspect that is the direct object of the verb phrase it is modifying as part of an adverbial clause" would probably not hold for other [well-formed] examples. Accordingly, any heuristic using dependency trees to link sentiment-bearing words to product aspect words may have a performance ceiling imposed by parser performance.

[6]Diagrams were generated by the Stanford CoreNLP online demonstrator at http://nlp.stanford.edu:8080/corenlp/process, which uses the Brat rapid annotation tool (http://brat.nlplab.org) to create such diagrams.

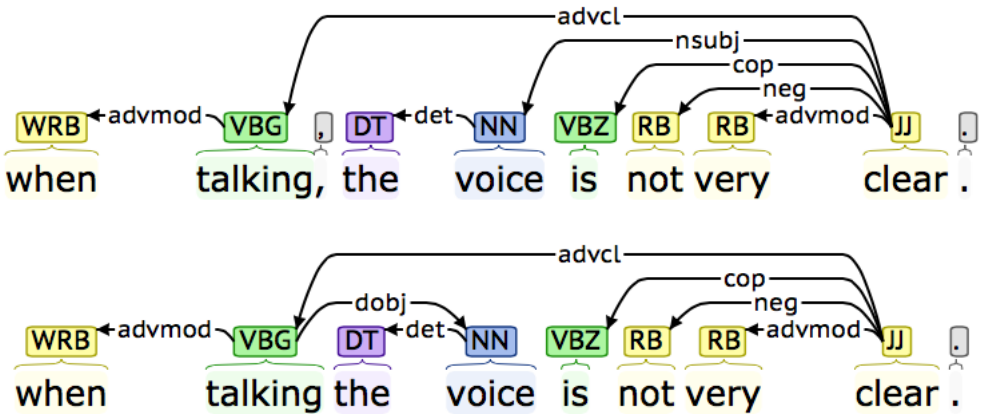

Figure 5.1: Punctuation affects parsing: with comma after second word (top), the parser correctly infers a direct semantic relationship between *clear* and the *voice [quality]* it describes; without comma (bottom), it does not.

Similar arguments can be made for part-of-speech tagging, lemmatization, and other steps used to pre-process the data for the experiments.

Of course, if there are consistent parsing or tagging errors that repeat regularly, they may in fact cancel each other out; if the parsing errors in the training data mirror those in the testing data, perhaps the correct results may be distilled nonetheless.

Despite the likelihood of such limitations on system performance, the data set is both usable and useful; the limitations mentioned are necessary byproducts of using text written causally by regular folks and using off-the-shelf tools that maximize the repeatability of the experiments.

Chapter 6

Method and experiments

6.1 Finding aspect-specific sentiments

I work with text from product reviews collected online by Hu and Liu (2004) and by Pontiki *et al.* (2014). This text is written by the general public, and contains some opinions about specific aspects of products as well as some non-opinionated sentences. For example, a particular sentence might indicate that a person likes the screen size of a phone but dislikes its battery life; there are two aspects in such a sentence (screen size, battery life) and an opinion about each aspect.

The goal of my work is to detect sentiments about aspects of products. This can be separated into two sub-goals:

- classify each word in a sentence as being inside or outside either a clause that indicates a product aspect or a sentiment (assuming lexical mutual exclusion between these two classes)
- match aspects with any corresponding sentiments in a sentence and determine the polarities (positive/negative) of these aspect-sentiment pairs

Thus, the software I developed takes a sentence as input and returns a list of zero or more tagged stated opinions about aspects of a product.

The system is based on machine learning plus a handful of heuristics.

6.2 Developing a co-training algorithm

Co-training is a semi-supervised classification algorithm that augments a small set of labelled data with a large set of unlabelled data to reduce the error rate in a classification task (Blum and Mitchell, 1998). A main motivation of such an approach is that labelled data is "expensive" (as it is usually hand-labelled by humans, which incurs time and/or monetary costs), and so any improvement in results that can be gleaned from unlabelled data is essentially "free" (Blum and Mitchell, 1998).

Co-training uses two conditionally independent "views" of the data being classified. Each such view must be (at least theoretically) sufficient to classify the data. Co-training iteratively builds up each classifier's knowledge by adding high-confidence classified cases to the training set; the expertise of one classifier is used to train the other, iteratively.

I have chosen to use [surface-level] lexemes (including predicted part-of-speech) and syntactic features as the two views. In English, as in many languages, there is not a one-to-one relation between lexemes and their part of syntax; they may be independent for all but the most basic of functional words (conjunctions, particles, and the simplest adverbs and personal pronouns, for example).

The lexical view is inspired by collocations. For example, a word following the fragment "I like my phone's ..." is fairly likely to be a product aspect, and is unlikely to be a sentiment-bearing word (unless, perhaps, it is a superlative adjective followed by the aspect).

The syntactic view is inspired by the observation that both product aspects and sentiment-bearing words appear in a limited number of grammatical structures. For example, a noun that is the direct object of a verb may be more likely to be a product aspect than the verb itself; in contrast, a verb is more likely to express sentiment than it is to be a product aspect.

The co-training algorithm posited by Blum and Mitchell (1998) assumed that the two views were conditionally independent; that is, given three events, and given that the first event occurs, knowing that the second event occurs (or does not occur) does not give any particular hint that the third event will or will not occur, and likewise, knowing that the third event occurs (or does not occur) does not give any particular hint that the second event will occur or not occur. In a system that processes text token-by-token, for example, we might say that the first event is the set of words that have occurred up to a certain point under current analysis; so, knowing the sequence of tokens that has occurred prior to the token being considered, two views are conditionally independent if neither view can predict the value (feature set) of the other view for the token being considered. Blum and Mitchell use the example of web page text and web page URLs as two views for using co-training to classify web pages according to

topic. They justify this example by assuming that different authors write these two types of text. One could imagine, however, that one might develop an algorithm that can reasonably often predict the name of the HTML file of a page given its contents, though not the reverse. My selection of lexical and syntactic views follows a similar pattern: they seem reasonably conditionally independent, as one can be reasonably assured of not being able to predict the lexical view given the syntactic view, although one might be able to predict (albeit poorly) the syntactic view given the lexical view on occasion.

The two views I have selected are assumed to be sufficiently and usefully close to being conditionally independent to borrow some of the principles of Blum and Mitchell (1998); though, admittedly, the two views may not be strictly conditionally independent given a set of preceding tokens. Given some set of preceding tokens (including the null set), there is a probability distribution of the [surface] token to follow (which one might be able to estimate from a very large corpus, for example). There is also a probability distribution of the part-of-speech to follow (which one might also be able to estimate from a very large corpus). Knowing that the token under consideration reads "buffalo", as an example, only slightly changes the probability distribution of the part-of-speech, since "buffalo" could (infamously[1]) be a noun, verb, or proper noun doing the work of an adjective. Similarly, knowing that the part-of-speech of the next token is (for example) a noun limits surface token choice, but due to the great number of tokens that have a word sense that is a noun, the probability distribution is perhaps not greatly changed.

The conditional independence assumption was relaxed by Goldman and Zhou (2000). The only requirement of their co-training algorithm is that each classifier be able to divide the data into "equivalence classes"; that is, that it be able to classify the data into n bins. My algorithm certainly fits that description. Several other papers contribute evidence that the fairly strict assumptions of the original Blum and Mitchell co-training algorithm can be relaxed (see Section 3.3); so, while it appears that my system may meet the strict criteria, exhaustive proof thereof may not be strictly necessary.

Blum and Mitchell (1998) took a small random sample of unlabelled data at each iteration to classify and add to the labelled set. Subsequent improvements to co-training, such as those introduced by Dasgupta *et al.* (2002), used confidence-based classification, where the classifier estimates its own competence and confidence in its own predictions. Similarly, Huang *et al.* (2012) developed an algorithm very similar to the Blum and Mitchell algorithm, but instead of taking a small random sample at each iteration, sampled the data where their two views' classifiers agreed the most, based on the classifiers' confidence.

[1] http://www.cse.buffalo.edu/~rapaport/buffalobuffalo.html

Given:

- a set L of labelled training examples with features x
- a set U of unlabelled examples with features x

Create a pool U' of examples by choosing u examples at random from U

Result: An enlarged pool L'

initialization;

for $i \leftarrow 1$ **to** k **do**

Use L to train a classifier h_1 that considers only the x_1 portion of x;
Use L to train a classifier h_2 that considers only the x_2 portion of x;
Allow h_1 to label p positive and n negative examples from U';
Allow h_2 to label p positive and n negative examples from U';
Add these self-labelled examples to L;
Randomly choose $2p + 2n$ examples from U to replenish U';

end

with typical $p = 1$, $n = 3$, $k = 30$, $u = 75$;

Algorithm 1: Co-training algorithm from Blum and Mitchell (1998) (largely verbatim)

Given:

- a set L of labelled training examples with features x
- a set U of unlabelled examples with features x

Create a pool U' of all examples from U

Result: An enlarged pool L'

initialization;
$i = 1$;

while $i = 1$ **or** $n_{i-1} > 0$ **do**

Use L to train a classifier h_1 that considers only the x_1 portion of x;
Use L to train a classifier h_2 that considers only the x_2 portion of x;
Allow h_1 to label all examples from U';
Allow h_2 to label all examples from U';
Sort these self-labelled examples in descending order of *max(confidence of h_1, confidence of h_2)*;
Add the top n most confidently labelled examples to L where $n \leq m$ and the confidence of the prediction of every such example is greater than c;
$i \leftarrow i + 1$;

end

with typical $m = 2500$, $c = 0.55$, $i_{\max} \approx 8$;

Algorithm 2: Co-training algorithm using confidence-based classification

I started with the algorithm introduced by Blum and Mitchell (1998) (Algorithm 1, left column, page 52), and modified it to use confidence-based classification (Algorithm 2, right column, page 52).

Both algorithms begin with set (L) of labelled training examples and a set (U) of unlabelled examples. The Blum and Mitchell algorithm then selects a random pool of unlabelled examples (U') that is much smaller than the full unlabelled set, and enlarges this pool every iteration; whereas my algorithm considers all remaining unlabelled examples (U') at each iteration. The Blum and Mitchell algorithm iterates a fixed number of times (k); whereas my algorithm keeps running as long as the number of unlabelled examples that could be classified in the previous iteration (n_{i-1}) is greater than zero. Each iteration, both methods have a classifier train itself on a single view of all labelled data (including data that have been labelled successfully in previous iterations), then classify the data in (U'). At this point, Blum and Mitchell randomly pick one positive and three negative examples to add to the set of labelled data; whereas my algorithm adds to the set of labelled data those data about which it was most confident. This confidence metric is defined as the confidence of the most confident classifier; a more complex scoring function could be used, such as the amount of agreement or the amount of disagreement between the two classifiers, as suggested by Huang *et al.* (2012).

If, in both algorithms, the classifier is very confident at all times, and the maximum number of data labelled per iteration by my algorithm (m) is set to four (roughly equivalent to how the Blum and Mitchell algorithm accepts one positive and three negative examples per iteration), the two algorithms end up performing in a very similar manner. If, however, the classifier has moments of uncertainly, as one might tend to in complex tasks like analyzing language, the algorithm I present should offer an advantage, as only the most certain examples are learned at each iteration. If a co-training algorithm is used in a scenario where it is desirable to know the confidence of the final output (as in a system where many machine learning modules vote), the algorithm I present offers the benefit of being able to estimate that confidence with no further work.

As an added bonus, this confidence-based classification approach avoids the need to tune two "magic numbers" in the Blum and Mitchell method: the number of positive predictions p and the number of negative predictions n to make at each iteration, which Blum and Mitchell determined experimentally on their data, but which may not hold for other data sets; they note that "empirically the performance of the algorithm was sensitive to this issue" of weighting p and n both in absolute terms and relative to each other (Blum and Mitchell, 1998, Section 5). The co-training algorithm they implemented accepted $p + n = 4$ new examples per iteration; for any classifier that takes a non-trivial amount of time to train its models, this

limits the scalability of the approach.

The most notable divergence from the Blum and Mitchell algorithm is the decision to add a large number of examples at each iteration, so long as the classifiers are confident in their classification of such unlabelled examples. I have chosen to implement an upper limit in the algorithm, so that, rather than accepting all new unlabelled examples that can be classified with a confidence of, say, 55%, it will only accept the top m-most cases. The intuition is that it may be desirable, especially in the early iterations, to add only the most confident examples and retrain so as to be able to more confidently label the next set; the expertise added by only accepting the highly confidently-labelled examples may be sufficient to more confidently classify the merely marginal unlabelled examples that may have been classified with confidence at or just above the threshold. In practice, the algorithm tends to use this upper limit in only the first several iterations; after roughly the fifth iteration, the confidence threshold determines the number of unlabelled examples added at each iteration, as the most obvious examples have already been added to the labelled set.

While Blum and Mitchell's algorithm takes as an input the maximum number of iterations k (which would also presumably have to scale proportionally to the size of the data set), my algorithm requires the maximum number of new examples to label in each iteration, which roughly determines the number of iterations for a given confidence threshold.

Except where otherwise specified, I have run my experiments with no more than 2500 unlabelled examples being classified in each iteration. The effect of varying this parameter is examined in Table 7.7 (page 95).

The confidence threshold c in my algorithm is tuneable. This parameter serves as classification confidence floor; the algorithm will not include any labelled examples when the confidence in that example's classification is less than this floor. The support vector machine classifier I selected offers fairly good classification performance, so I set this threshold to a relatively low 0.55 for all experiments described herein. (A grid search classifying the development data with confidence thresholds $c \in \{0.00, 0.45, 0.50, 0.55, 0.65, 0.75, 0.85, 0.95\}$ revealed that 0.55 was close to optimal.)

The classifiers h_1 and h_2 mentioned in Algorithm 2 refer to two support vector machine classifiers, one for the lexical view and one for the syntactic view. These are described in greater detail in the following section.

The bootstrapping algorithm presented by Yarowsky (1995) re-classifies all examples that have been labelled to date so as to reject examples that fall below the classification threshold (termed "escaping from initial misclassifications"). This could be a natural extension of my algorithm; though I thought that the algorithmic complexity of such an approach – and

particularly the possibility of infinite loops as marginal cases are included and excluded from the training set in successive iterations – might not be justified.

This co-training algorithm was used to identify and learn sentiment-bearing terms and product aspects from both the Amazon review data and the SemEval-2014 task 4 data.

6.3 Implementing classification

The co-training approach used relies on two machine learning classifiers. The classifier was selected with care, and it was necessary to determine suitable features to use and reasonably optimal classifier tuning parameters.

6.3.1 Selecting a classifier

In a sentence, it is necessary to identify which words indicate product aspects (general features of product categories) and which words indicate sentiments. Aspects and sentiments might be explicitly stated (e.g., "good battery life") or implicit in the language used (e.g., "it's impressively fast", suggesting a positive sentiment for the aspect *speed*).

I construe this as a classification task, where each word in a sentence can be classified as one of three mutually exclusive cases:

- The word is inside (is part of) a product aspect
- The word is inside (is part of) an expression of a sentiment
- The word is outside of a product aspect or expression of a sentiment

This approach to classification contrasts with, say, considering a sentence as a bag of words and considering all the possible product aspects to be classes. Sentence-level (predicting the positive/negative orientation of an entire sentence) and document-level (predicting a product rating on a 0-to-5 scale) classification, while easier to implement, are inadequate in this task.

Given this framing of the problem as a token-level classification task, it was necessary to select a useful and usable classifier.

There were two key properties I considered in selecting a classifier. First, it had to be able to generate class probabilities when classifying a datum, so that I would be able to experiment with a co-training style algorithm (wherein one starts with a small set of training cases and then iteratively builds larger and larger classification models by successively adding a

small number cases from the test set to the training set when the confidence of successful classification is very high). Second, since I wanted to be able to consider words and lemmas as dimensions of machine learning features, it had to be able to handle a very large number of features while being computationally tractable.

I chose LibSVM (Chang and Lin, 2011), a support vector machine classifier, as it met these two criteria, and has been used in other natural language processing applications.

It is perhaps worth noting for a moment that my algorithm could (in theory) be used with any classifier that can estimate percentage confidence. There is nothing inherent in my method that is classifier-specific; I simply use a binary product aspect word classifier and a three-class opinion word classifier with probability estimates and a large number of features.

There is some academic support for SVMs being used in natural language processing tasks. Basili and Moschitti (2002) suggest that SVM is a better choice for text classification than K-Nearest-Neighbours (K-NN) (though perhaps not statistically significantly), a neural network implementation, decision trees, and Naïve Bayes, among others. There is some evidence that linear regression classifiers can not practically handle a large feature set for similar tasks (as in (Ghazi *et al.*, 2014)). Manning *et al.* (2008) compare text classification performance on a news corpus with results from (Li and Yang, 2003), (Joachims, 1998), and (Dumais *et al.*, 1998); SVM appears to consistently outperform Naïve Bayes, Rocchio, k-nearest-neighbour, and tree classifiers in this meta-analysis. They note three caveats: that the observed classifier performance in the three papers varies more between tasks than between classifiers; that the conclusion that SVMs perform better may only be true when trained and tested on "independent and identically distributed data"; and that, in the real world, it is a regular occurrence that skilled practitioners are unable to build complex classifiers that outperform Naïve Bayes. Caruana and Niculescu-Mizil (2006) perform a large evaluation of ten supervised learning methods, including support vector machines, and mention that even the best classifiers perform poorly on some tasks; they found that calibrated boosted trees, calibrated random forests, bagged trees, calibrated SVMs and uncalibrated neural nets performed best, in order of decreasing performance. They also observed particularly poor performance with or without calibration in Naïve Bayes, logistic regression, and decision tree classifiers. They conclude that SVMs (along with other relatively modern learning methods) offer excellent performance.

At a minimum, then, it appears that support vector machines are not a poor choice for the text classification task, and there is some evidence to suggest that they may be better than other approaches.

I chose a radial basis function (RBF) kernel for the support vector machine classifier. This seemed a better choice than a linear kernel or a low-order polynomial kernel, as an RBF

kernel can theoretically create decision boundaries that better conform to groups of features that interact (i.e., cases where, considering features independently, the classes are not linearly separable), while still being able to use single simple features to classify when possible. The creators of LibSVM propose the RBF kernel as a good default choice (Hsu *et al.*, 2003) for its ability to deal with nonlinear class separation; for the fact that RBF has fewer parameters to tune than a polynomial kernel; and for some numeric/computational feasibility benefits over polynomial kernels. They note that the linear kernel is merely a special case of the RBF kernel. They also note that, for cases where the number of features is very large, like in my experiments, the linear kernel might be a better choice.

Support vector machines seem to work best on linearly separable problems. It is quite possible (and, in fact, probable) that, due to the creative nature of language, the product aspect words are not entirely separable from the rest of the corpus; that is, there may be many objective/factual statements made about aspects of products. Similarly, the words that bear opinion about aspects are almost certainly not separable from the rest of the corpus, since there may certainly be other opinions in the sentence that have no bearing on product aspects. However, it seems quite plausible that opinion-bearing words may be separable from product aspect words (and vice versa). We might imagine that product aspects could be mostly nouns (e.g., screen, brightness, speed, service) and adjectives that are closely associated with attributes (e.g., fast, bright, flavourful, loud) while sentiment-bearing words might be more skewed towards verbs (e.g., like, hate, love, appreciate) and gradable general-purpose adjectives (e.g., great, worse, awesome). This mutual exclusion between the two sought-after classes in the data may be useful, and may provide further (if weak) support for using SVMs; it seems unlikely that an SVM would, given reasonable features, confuse aspects for opinions, because the two classes seem quite separable. Even if both the aspect word and sentiment word classifiers were to both try to classify a particular token positively, only the classifier with the higher confidence would be allowed to do so; it would become a positive example in the more confident classifier's training data in the next iteration, and a negative example in the other classifier's training data in the next iteration. In this manner, the level of expertise of both classifiers should improve.

This is a critical advance on some other contemporaries' work in the field. One could imagine that a simple bag-of-words review classifier would probably infer that "iPhone" connotes a positive opinion and perhaps that "Blackberry" connotes a negative opinion (both inferences being false).

In choosing a classifier, I specifically rejected Naïve Bayes classifiers on the fundamental basis that, while they offer reasonable performance on some tasks, I am uncomfortable with

the independence assumption (that features are independent given the class label) that underlies the method, particularly as it applies when doing word-level classification. As a simple example, one might assume that the surface form of a token and its part of speech might both be good features to consider, but it would be difficult to argue that they are independent. Put succinctly, in the real world, there are many instances where we can predict a word given its antecedents; this clashes with the Bayes assumption of conditional independence of feature values given class membership (Lewis, 1998), particularly when one uses neighbouring tokens as features (as I have chosen to).

With LibSVM selected as a classifier, it was necessary to select features to feed the support vector machine, as described in the following section.

6.3.2 Selecting features

I use two broad categories of features, each corresponding to a co-training view: lexical features (tokens and stemmed versions thereof, plus part-of-speech, which is somewhat tightly tied to the lexeme), and syntactic features, including dependency tree features and some simple role labels determined solely from syntax.

For each category of features, I consider a window around the word to be classified. In the lexical view, for example, I consider a three-token window on either side; this is somewhat inspired by work on extraction patterns, where a pattern like "I like my *some_product_name* despite its rather poor *some_product_attribute*" can be used to extract product names and product attributes with fairly high confidence. For syntactic tree features, this window extends upwards: I consider it important that a word is part of a noun phrase within a verb phrase, for example. I also pay special attention to dependency roles (e.g., direct and indirect objects of verbs), and whether the dependency roles of a token include negation. Such dependency roles can be manifested any number of tokens away in a sentence, so even if, for example, a negation word is ten words away in the sentence, it is incorporated so long as it is one dependency relation away. My intent is to work in a manner similar to extraction patterns but to do so in a way that reflects the complexity of language, particularly long-distance dependencies that might not be accounted for in an n-gram model.

The list of machine learning features for each of the views follows immediately; an explanation and justification of how the features were chosen follows these lists.

Lexical view features

- the token itself
- the token's lemma (its root form; e.g., the lemma of *been* is *be*, while the lemma of *prawns* is *prawn*)
- the token's (predicted) part-of-speech (e.g., noun, verb, adjective, superlative adjective, adverb; using Penn Treebank part-of-speech labels[2])
- the first WordNet *attribute*[3] indicated by the first WordNet sense of the token (only if the token is an adjective or adverb); e.g., the adjective *fast* describes the WordNet attribute *speed, swiftness, fastness*, so for a given token *fast* in our text, the recorded attribute feature is *speed*, which itself might appear nowhere in the sentence
- for each of the three tokens preceding the token:
  - its token
  - its lemma
  - its part-of-speech
- for each of the three tokens following the token:
  - its token
  - its lemma
  - its part-of-speech

[2]A reasonably comprehensive list of Penn Treebank part-of-speech labels is available at http://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

[3]Defined as "a noun for which adjectives express values" in the WordNet Glossary at http://wordnet.princeton.edu/man/wngloss.7WN.html

Syntactic view features

- the "local" chain of nodes above this token in the parse tree; that is, the chain of nodes from this token up to the first sentential head (a node marked *S* in the parse tree; e.g., [S, VP, VP, PP][4])
- the entire chain of nodes above this token in the parse tree (which may be identical to or longer than the local chain; e.g., [VP, SBAR, S, VP, VP, NP, S, VP, VP, PP][5])
- the node immediately above this node in the parse tree (e.g., PP)
- whether the token is referenced by a pronoun elsewhere in the sentence
- the anaphoric expression back to which a pronoun refers, if the reference appears in the same sentence (e.g., in "The waiter was quick and he was efficient", the word *he* would have an anaphoric feature *the waiter*)
- a list of immediate (local) dependency relations[6] in the sentence, as determined by the Stanford CoreNLP semantic parser (e.g., in "The waiter was good", *good* would have two semantic relations, one noun subject "waiter" and one copula "was".)
- the specific category of semantic role in the sentence (e.g., passive nominal subject, labelled *nsubjpass* by the parser)
- the broad category of semantic role in the sentence (e.g., subject, labelled *subj* by the parser)
- the incoming dependency graph edge, if any (e.g., the noun an article introduces)
- whether any of the immediate outgoing dependency graph edges contains negation (*not* or *n't*) or a modal verb that indicates negation (*could*, *would*, or *should*)
- whether the token has been tagged as a named entity (a proper noun indicating a person, organization, or location)

[4]This chain indicates that, from right to left, the token at hand is part of a prepositional phrase (PP) that is part of a verb phrase (VP) that is in turn part of a larger verb phrase (VP) that is part of a sentence (S)

[5]The chain shown here is for the word *with* in the sentence *Our waiter is friendly and it is a shame that he didn't have a supportive staff to work* ***with***.

[6]A full list of dependency relations is available at http://nlp.stanford.edu/software/dependencies_manual.pdf

- for both the preceding and following token:
  - whether the token is referred to by a pronoun elsewhere in the sentence
  - if the token is a pronoun, the anaphoric expression it refers to
  - whether the token is part of a named entity
  - its specific semantic role
  - its general semantic role
  - whether any of its dependency graph outgoing edges contains negation
  - the part-of-speech of its immediate parent in the parse tree

The number of nonzero features recorded for each token varies. A token will have 24, 29, 35, or 41 lexical features, depending on how close it is to the beginning or end of a sentence; a token at the beginning or end of a sentence will have 24 lexical features, while a token in the middle of a long sentence will have 41 lexical features. Typically, a token will have a minimum of 14 syntactic features, and an average of approximately 32 syntactic features. Because any number of immediate syntactic relations can be encoded for a given token, the number of syntactic features is highly variable. It would be unusual for more than 40 syntactic features to be recorded for any given token.

The feature set (and particularly the lexical view thereof) is inspired by an n-gram model. It has been observed since the 1950s that "it is possible to define a linguistic structure solely in terms of the distributions (patterns of co-occurrences) of its elements" (Harris, 1954). Concordances are modelled: each token in the preceding trigram and each token in the trailing trigram is encoded (separately) as a feature. Encoding the preceding trigram is inspired by the Markov assumption: that, in the words of Manning and Schütze (1999, page 193), "only the prior local context – the last few words – affects the next word". If we make the simplifying assumption that both sentiments and aspects are expressed lexically, then we can take advantage of the Markov assumption for classifying sentiment-bearing and aspect words. Encoding the trailing trigram as three word features is inspired by the same principle; one can imagine that aspect words might be followed shortly thereafter by a verb (e.g., "…the screen is…") or by a comparative or superlative adjective, for example. While there is no generally accepted theory that a word can be predicted by the sequence of words behind it (in English), including them seems intuitively to be of benefit. It is also worth considering that the order of words is not arbitrary (although may be particularly subject to creativity in English); and so if we allow that words can be predicted by their predecessors, perhaps we may allow that the

word followed by a particular trigram cannot be completely arbitrary. It follows that there is at least weak associativity in both directions.

Trigrams are a reasonable starting point for language modelling.[7] Manning and Schütze (1999, page 195) note that "the lexical co-occurrence, semantic, and basic syntactic relationships that appear in this very local [trigram] context are a good predictor of the next word, and such systems work surprisingly well". Trigram models are able to capture some instances of phenomena such as collocations[8] and idiomatic expressions[9].

These preceding trigrams and trailing trigrams are encoded not atomically, but as individual tokens; otherwise, we effectively learn a 7-gram (preceding trigram + token of interest + trailing trigram) model where every such 7-gram is very likely to be unique in any given data set, and therefore not particularly useful for learning. Using atomically-encoded bigrams instead of unigrams, given data of sparseness similar to the experiments herein, was found to offer worse performance in sentiment classification tasks by both Pang *et al.* (2002) and Ng *et al.* (2006); whereas encoding the adjoining trigrams as individual token features should provide something akin to back-off[10], where, if the $unigram_{@position\text{-}3}$, $unigram_{@position\text{-}2}$, and $unigram_{@position\text{-}1}$ have all been seen in that combination previously, it is a very strong hint about how to classify the token; if only $unigram_{@position\text{-}2}$ and $unigram_{@position\text{-}1}$ have been seen, the model performs as if it were a bigram model, offering a less strong hint; if only $unigram_{@position\text{-}1}$ has been seen previously, it's a weaker still hint about how to classify the token under consideration, but effectively performs as a unigram model.

To some extent, this use of n-gram neighbours takes advantage of the *one sense per collocation* property noted by Yarowsky (1995): that is, a token in a given collocation is likely to be of the same class (sentiment-bearing word, aspect word, or neither) when such a collocation is found elsewhere. To a lesser extent, Yarowsky's *one sense per discourse* observation also

[7] "When the trigram model works, it can work brilliantly … The four-gram model is entirely useless in general." (Manning and Schütze, 1999, page 201) Four- and five-grams could perhaps be useful in limited circumstances provided that a very large corpus is available; the relative rarity of non-singleton four- and five-grams in a given corpus is a limiting factor.

[8] Collocations are sequences of words that frequently appear together to provide a slightly stronger meaning; contrast *fast food* (a collocation) with *quick food* and *food that is fast*, for example.

[9] Idiomatic expressions are language-dependent figurative expressions like *kick the bucket* and *feeling blue*.

[10] This idea of n-gram back-off is suggested by Manning and Schütze (1999, page 201): "Examining the table [of unigram, bigram, and trigram prior probabilities in a given text] suggests an obvious strategy: use higher n-gram models when one has seen enough data for them to be of use, but back off to lower order n-gram models when there isn't enough data." They claim that this is a widely used strategy, though, as seen in the plethora of bag-of-unigrams solutions identified in the related work (Chapter 3), that may not be entirely true in sentiment analysis.

applies: a token that appears as an aspect is probably an aspect throughout the same review, and perhaps so too in other similar product reviews in the same domain.

Part-of-speech (POS) features are also recorded in a similar fashion: the predicted[11] part-of-speech of each of the preceding three words and of each of the next three words is recorded as a feature. This should work in concert with the aforementioned back-off strategy, increasing the likelihood that, even if $unigram_{@position\text{-}3}$, $unigram_{@position\text{-}2}$, and $unigram_{@position\text{-}1}$ have not been seen previously, there may be a good chance that $POS_{@position\text{-}3}$ followed by $unigram_{@position\text{-}2}$ and $unigram_{@position\text{-}1}$ might have been seen before, which may be a stronger hint to the classifier than simply the preceding bigram alone.

In instances where lexemes are used as features, it seemed reasonable to include both the surface forms of the words in the sentence as well as their lemmas. The intent was to improve performance in cases where a word has not been seen by a model, but where a morphological variant has been seen previously (e.g., for the purposes of classifying aspects for a particular high tech device, *battery* and *batteries* could be interchangeable, and lemmatization helps reconcile such variations). I chose to lemmatize instead of stem so as to try to capture similarities; a good lemmatizer should be able to lemmatize *better* as *good*, whereas a simple stemmer might stem it as *bet* or *bett*. Being able to collapse down, for example, *good*, *better*, and *best* to the same lemma feature should assist the sentiment classifier.

A secondary reason for including lemmas is that negation can be marked morphologically (Bender, 2013). In English, the prefixes dis-, in-, non-, de-, un-, and the suffix -less can all mark negation (Cartoni and Lefer, 2011); and from the point of view of a classifier that seeks to mark sentiment words in particular, being able to recognize that, if *unsuitable* is a known negative sentiment word, than perhaps *suitable* might be a sentiment word as well. Encoding both the surface form and the lemma as features allows us to encode both the positive lemma and the negative token usage in such a case.

One final lexical feature included is the WordNet *attribute* of a given word. This, even more so than with lemmatization, is an attempt to give the classifier hints about new language that may be synonymous with – but entirely different than – previously seen language. In particular, I hypothesize that sentiment-bearing words are more likely than other words to have a usable attribute relationship recorded in WordNet, so the mere presence of an attribute relationship may be a reasonable hint that a sentiment-bearing word is present.

Using dependency relations in the syntactic view is both inspired by and supported some-

[11] Part-of-speech labels are generated during sentence parsing. For the purposes of building the classifiers, the labels are considered to be correct, with the hope that, even when POS labels are incorrect, they are probably at least consistently incorrect in similar contexts.

what by Ng *et al.* (2006). They concerned themselves only with adjective-noun, subject-verb and verb-object relations, but make the argument that verb-object relations, for example, "may allow the learner to learn that the author likes the actors and not necessarily the movie" (Ng *et al.*, 2006, page 615). They conclude that dependency relations are useful for their sentiment classification task when bigrams and trigrams are not used, rejecting their initial hypothesis that bigrams/trigrams and dependency relations encode redundant information. This weak conclusion is somewhat counter-intuitive when analyzing non-trivial opinionated sentences (as in the SemEval-2014 datasets, for example): long-distance dependencies can be reconciled using dependency relations but might not be captured with trigrams. This is particularly important in the present task; whereas those authors were performing document-level (review-level) sentiment analysis, the task of disambiguating product sentiments from aspect-specific sentiment seems likely to benefit from dependency relations. Basili and Moschitti (2002, page 405) conclude that such linguistically-motivated features are superior, concluding that building features from parsing and using major grammatical relations (subject/object pairs, for example) is tenable.

I chose to encode all local dependency relations for each token, letting the classifier sort out what is salient and what is not, rather than pre-selecting certain relations (like verb-object).

I record three levels of parse tree above each given token: its immediate parent; a chain of local relations up to the most local sentential unit; and the entire chain of relations to the head of the sentence. This, as with the n-gram modelling approach, is an attempt to incorporate cases where knowledge is sparse but fairly certain (an aspect buried deep in a complex sentence may well have a unique parse tree lineage, but should that entire exact lineage appear again, that may be a very strong indication that such a token is also an aspect) as well as common but less certain knowledge (knowing that a token is in a noun phrase may increase the likelihood of it being an aspect, for example, even though many words that are not aspects also appear in noun phrases).

An attempt is made to reconcile local pronominal references. If an aspect is explicitly stated in the sentence and a pronominal reference is made in the same sentence, the pronoun is marked as if it were the noun. This is an attempt to be able to classify pronouns as aspects even when the noun itself is not present.

Negation is also explicitly encoded in the classifier features. Negation is noted by the sentence parser in the dependency relation *neg*; dependencies of this type, when lexicalized as *not* or *n't*, were given a machine learning feature noting negation. Three modal verbs were

also considered as negation: *could*, *would*, and *should*.[12] These appeared often in reviews in the form of *the salads could be better*; in such a construction, the modal *could* effectively negates the positive sentiment word *better*.

All of the textual features (token, lemma, POS, dependency relations) are encoded as sparse numbered binary features. Considering only tokens to simplify the explanation, there is a feature for every token that has been seen; one such feature will be 1, and all others 0. For example, if the word *alpha* has been assigned feature number 0, the word *buttery* has been assigned the feature number 1, *crass* has been assigned feature number 2, and *definite* has been assigned feature number 3, the feature representation for an instance of *buttery* would be the matrix [0, 1, 0, 0], while the matrix feature representation of *definite* would be [0, 0, 0, 1]. There is an arbitrary assignment of feature numbers to tokens. The other textual features follow a similar pattern.

These machine learning features were selected in order to classify sentiment-bearing and aspect words in sentences, given data from product and restaurant reviews.

6.3.3 Scaling data

There was no need to scale the data for the classifiers. All machine learning features were binary and sparse, indicating the presence or absence of a particular property of interest.

6.3.4 Tuning classifiers

Support vector machines have several parameters that can be tuned to minimize the rate of classification errors. Tuning these parameters can dramatically improve classification accuracy (Hsu *et al.*, 2003).

In the LibSVM implementation that I chose, there are two key parameters that need to be tuned with care: C and γ (gamma). The former is associated with SVMs in general; the latter is a particular parameter for radial basis function (RBF) kernels. Together, these control the process for deciding how to mathematically optimize the plane(s) separating the classes of data.

The C parameter is a cost (penalty) parameter that serves as a trade-off between errors in the model (points that end up on the wrong side of the plane) versus how flat/uniform the separating plane is (Manning *et al.*, 2008, page 328). It is the cost of ignoring outlier data to build a model. A larger value of C will produce a model that will omit fewer outliers

[12] Other tenses (e.g., can, shall, will) were not considered to cause negation.

in building the model, and will thus have fewer errors based on the training data, but may produce a very complex plane that might be undergeneralized/overfitted at testing time (as an extreme, imagine a plane that curves tightly around every single data point; such a plane would be error-prone when faced with any other nearby data points at classification time). A lower C will ignore some outliers; particularly in data sets that are not perfectly separable. Consequently, C can be thought of as a noise filter.

The γ parameter, by contrast, controls the width of the RBF margin, the orthogonal distance from the plane to any given point; within a certain margin, values are ignored. This can be thought of as a figurative demilitarized zone: a zone that belongs to neither side. Setting this margin to be small (larger values of γ) tells the support vector machine that, since the margin is small, it must work harder to make sure that more data points are sufficiently far away from the border; large values of γ tend to cause overfitting. (It is worth noting that SVM RBF implementations other than LibSVM tend to use a parameter σ instead of γ; where $\gamma = 1/\sigma$. The effect is the same.)

The parameters have similar (if numerically opposite) effects; tuning C controls how many points can be ignored in favour of a simpler model, whereas γ controls the degree to which a solution must wrap around points that must be included in the model.

I followed the suggested tuning heuristic in (Hsu *et al.*, 2003) and incorporated advice from (Joachims, 2002) and (Cherkassky and Ma, 2004). I chose an RBF kernel. I took the first 20% of sentences from each of the five Amazon review data sets, and did a coarse grid search using each such sample to estimate reasonable values for C and γ for each view for each classifier. I then took logarithmic averages of the parameters of the five trials. Taking small samples of the data and then averaging the parameter values over these five samples was an attempt to avoid over-fitting the parameters to the data.

I then performed an iterative hill-climbing optimization by varying the four variables for each classifier (two views per classifier, each with two tuneable parameters). At each iteration, I would try doubling and halving each parameter in one classifier while leaving the values for the other classifier static, then try the same with the other classifier, finally continuing with the best four parameters from that iteration.

For example, if starting with [C_{view1}, γ_{view1}, C_{view2}, γ_{view2}] = [1.0, 0.1, 1.0, 0.1], I would run eight complete supervised classification task using the 20% sample of the Amazon review data with parameters, trying the following variations:

No change in view 1; trying to find an improving direction in view 2:

- $[1.0, 0.10, 2.0, 0.20]$
- $[1.0, 0.10, 2.0, 0.05]$
- $[1.0, 0.10, 0.50, 0.20]$
- $[1.0, 0.10, 0.50, 0.05]$

Trying to find an improving direction in view 1; no change in view 2:

- $[2.0, 0.20, 1.0, 0.10]$
- $[2.0, 0.05, 1.0, 0.10]$
- $[0.50, 0.20, 1.0, 0.10]$
- $[0.50, 0.05, 1.0, 0.10]$

In many instances it was not necessary to calculate all permutations, e.g., if we have doubled all values of interest from the previous iteration to get to the current optimum, it is not necessary to recalculate the permutation in this iteration where all current values are halved.

Also, C and γ are not strictly independent; in general, as C increases, γ should decrease; and conversely, as γ increases, C should decrease (Ben-Hur and Weston, 2010). Accordingly, at most iterations, it was sufficient to simply calculate the permutation that is in the same direction as the previous direction of greatest improvement, and if the optimum improves, take a short cut to the next iteration without calculating all permutations. For example, if the general trend is that an increasing C_{view1} and a decreasing C_{view2} have given a better solution at the previous iteration, we can check whether again increasing C_{view1} (and, by extension, decreasing γ_{view1}) and a decreasing C_{view2} (and, by extension, increasing γ_{view2}) gives us a better solution, and move immediately in that direction (climbing the slope further) without checking all directions for a better improvement.

I iterated thusly until none of the attempted possibilities offered a better solution than the best solution from the previous iteration. Using relatively large macro changes was an attempt to avoid fitting the parameters too closely to the training data; being in a reasonable ballpark of the optimal solution for the sample should be sufficient.

I did some brief experiments to see if weighting C by class[13] would have any positive impact (e.g., hoping it might effectively overweight positive cases to partially make up for the fact that, for each classifier, there are many more negative examples than positive). Put simply, increasing C for data in the minority class tells the classifier to try harder to get those data points correct; doing so would tend to move the hyperplane closer to to majority class.

[13]This concept is illustrated well at http://scikit-learn.org/stable/auto_examples/svm/plot_separating_hyperplane_unbalanced.html

	C	γ	ε
Sentiment-bearing word classifier			
Lexical view	147.998	0.00336540	1.0×10^{-4}
Syntactic view	2.87744	0.102704	1.0×10^{-4}
Aspect word classifier			
Lexical view	1.26562	0.0343323	1.0×10^{-3}
Syntactic view	39.0625	0.0234375	1.0×10^{-3}

Table 6.1: Tuning parameters for the support vector machine classifiers

This effort was ultimately unfruitful, however. Similarly, I tried oversampling the minority class (duplicating or triplicating examples in the minority class) and, separately, undersampling the majority class (omitting half the examples); these efforts were not successful in improving results either.

Finally, there is a third parameter available in LibSVM, ε, that specifies the stopping condition for the training. Considered as a hill climbing optimization problem, the support vector machine is trying to place an ideal splitting plane as far away as possible from each of the given training data while minimizing the error (training data that necessarily fall on the wrong side of the plane; this can happen if the data for two classes overlap in some dimensions); ε governs how close to an optimum the classifier will try to get before completing the model.[14] A smaller value instructs LibSVM to perform more iterations, in effect. A typical value (recommended by the authors of LibSVM) is ε = 0.001. A smaller value may marginally improve the classification performance of a model while increasing the time required to train it. Early on, I varied this parameter for each classifier for a given C and γ, trying $\varepsilon \in [0.1, 0.01, 0.001, 0.0001, 0.00001]$ and found that ε = 0.001 was suitable for the product aspect classifier, but that ε = 0.0001 offered about 3%-6% better classification performance in the sentiment word classifier without too great a training performance penalty.

The values chosen for these three SVM tuning parameters are listed in Table 6.1.

[14]Pseudo-code of a support vector machine offered in Appendix B of Fan *et al.* (2005) exemplifies how ε is used in practice.

6.4 Inferring opinions from product aspects and sentiment words

A task implicit in the works to date on the Amazon review data but ignored in the SemEval-2014 Task 4 challenge is the reconciliation of tagged product aspects and predicted sentiments into a combined task of tagging product aspects and sentiments simultaneously.

There are many sentences in each dataset that:

- contain aspect words but no sentiments
- contain sentiment-bearing words but no aspects
- contain sentiment-bearing words used in an objective context (e.g., describing a processor throttling down temporarily to save battery life as *slow*, independent of a processor that is inherently too slow at all tasks); it is conceivable that there might also exist polysemous words that exhibit a similar property, where the lexeme might appear in a lexicon of opinion words but the particular sense used is not opinion-bearing
- contain both aspect words and sentiment-bearing words that are unrelated (e.g., perhaps sentiment-bearing words are describing the brand or the product itself, not aspects thereof)
- contain sentiment-bearing words that indicate sentiments about aspect words elsewhere in the sentence

Of course, compound and complex sentences can meet several of these criteria simultaneously.

Identifying aspect words in a sentence and separately identifying the sentiments of tagged aspects, as in the SemEval-2014 task 4 challenge, is important, but is only a part of the natural language "understanding" required to disambiguate how the aspects and sentiments are related. Being able to classify a mention of an aspect in a sentence as positive or negative is of only partial value.

I developed a heuristic that, given a sentence with zero or more tokens tagged as product aspects and zero or more tokens tagged as sentiment-bearing words, tries to use dependency relations to reconcile the two. This heuristic is applied after classification.

A useful starting point is offered by Nigam and Hurst (2004): that if a sentence contains topical information and polar sentiment language, that the two can be assumed to be related.

Words that describe product aspects were more rare than sentiment-bearing words. The heuristic identifies each n-gram of consecutive product aspect words. For each such mentioned aspect, the heuristic examines dependency relations at distance d = 1, then d = 2, then d = 3 to see if any tagged sentiments are present, first considering incoming edges, then outgoing edges if no such sentiment-bearing terms have been identified in a three-hop distance.

The maximum distance of three hops was chosen as a compromise between larger distances that would classify more aspect-sentiment relations by chance and smaller distances that would offer more certainty (e.g., sentiment and aspect words that are one dependency hop away are almost certainly related). The maximum distance of three is sufficient, for example, to reconcile the aspect word *screen* and the salient sentiment-bearing word *good* in *the design is sleek and the color screen offers, in my mind, good resolution* (where the chain of dependency relations is *screen* → *offers* → *resolution* → *good*).

When a related sentiment-bearing word is found, its negation feature is checked; the orientation of the sentiment-bearing word is determined by considering both the polarity of the word (determined by the classifier) and whether it is tagged with a negation feature.

Note that this heuristic handles long-distance dependencies well; using word distance to model sentiment words and the concepts they are describing has been tried by Kim and Hovy (2004) and Ding and Liu (2007), but such a naïve approach cannot account for anything but the simplest sentence structure.

This heuristic is not perfect; it misses sentiment words that are still salient but that are more distantly related in the dependency tree, and it can incorrectly infer links between aspect words and sentiments that are unrelated. Perhaps an uncertainty-based measure, such as that posited by Rohrdantz *et al.* (2012), might be more effective at the expense of added complexity; or perhaps the work of Qiu *et al.* (2009) might be applicable (they used sentiment lexicons to learn product aspects and vice-versa, iteratively). Notwithstanding such options, however, my simple heuristic seems to work reasonably well in many (and perhaps most) circumstances.

It could be reasonable to improve this heuristic directly (by considering only certain dependency relations, for example) or implement it using machine learning, training the system to determine what sets of hops are productive in linking sentiment-bearing words and related aspects in the training data.

Sentence-level simultaneous classification of both product aspects and sentiments expressed thereof was performed on both the Amazon review data and on the SemEval-2014 task 4 data.

6.5 Validating the classifiers on unseen language

One of my goals in developing the software for this dissertation was to be able to handle new language. A system that can handle previously unseen language is one that can better handle sparsity of data (which often rears its head in NLP problems). Further, a system that can handle new terms that it has never seen should be able to better handle data from different subject domains. A philosophical argument is that naïve systems like bag-of-words perform many tasks well, but disregard linguistic knowledge that intuitively should have value; whereas a system that can, without any retraining, correctly handle ("understand") new words and new phrasings is one that can be argued to use such information implicitly, putting the *learning* back in machine learning.

I performed two classes of synthetic tests to determine if my system was able to handle new and unseen data. These tests were neither conclusive nor extensive, but demonstrated that, unlike many supervised machine learning models, my system can adapt to new terminology and new sentence structures.

I took the approach of training impoverished minimal models and evaluating them on test sentences.

First, I wrote a minimal test to see if my system could handle new terms. I trained (in a supervised manner) a system with only the following four synthetic sentences:

- *feature[+2]* the car has a wonderful set of features .
- *feature[+2]* the camera has a wonderful set of features .
- *lens[+2]* the camera has a great lens .
- *grip[+1]* the camera has a fine grip .

I then tested the model on the following sentence (and variations thereof):

- *shmork[+2]* the camera has a cromulent shmork .

The system was able to correctly infer that *cromulent* was a sentiment-bearing adjective; it classified it as a positive[15] sentiment-bearing word. It also correctly classified *shmork* as a product aspect.

Some credit is due to the Stanford parser, which correctly inferred that *cromulent* might be an adjective and *shmork* a noun.

[15]It is debatable whether *cromulent* is positive or negative; as the training data were all positive, the system reasonably classified it as positive.

Such a test can be criticized. The training data is unusually clean (containing no contradictory examples where, for example, in a review of a digital camera, *camera* is a product, not an aspect; whereas in a cell phone review, *camera* is an aspect, not a product). The sentences all follow a similar sentence structure, so the pattern is easily learned; whereas in real life, people might use many different sentence structures to express the same idea. Nonetheless, the system is able to handle, to at least some degree, new language that it has never seen before.

The system is similarly able to handle new sentence structures and verb tenses that it has not previously observed. I created an impoverished synthetic model using four sentences:

- *feature[+2]* the car 's features are wonderful .
- *feature[+2]* the camera has a wonderful set of features .
- *lens[+2]* the camera has a great lens .
- *grip[+1]* the camera has a fine grip .

The system was able to successfully handle the following two sentences, correctly classifying *cromulent* as the positive sentiment of the aspect *shmork*:

- *shmork[+2]* the camera 's shmork is cromulent .
- *shmork[+2]* the camera 's shmork was cromulent .

Interestingly, because the system does not have *cromulent* in its lexicon, and because, in this formulation, the word has not been specifically annotated, the sentiment word classifier marks it as a false positive; a misclassification that reveals an error in the original data annotation rather than poor system performance. There were many notable such false positives in the data sets used in the experiments described in this dissertation, where arguably my system outperformed human annotators on occasion.

This latter test has its limits. In testing *the camera 's shmork could be cromulent*, the system only tags *cromulent* properly; similarly, in *i really liked the camera 's shmork* the system is only able to tag *shmork* successfully. This is a byproduct of the extreme sparsity of the training data.

The system demonstrably handles new terminology and new sentence structure.

6.6 Applying the system to the chosen data sets

Two main experiments were performed.

First, the system developed was applied to the Amazon review data sets to determine whether it could offer some reasonable ability to extract aspect-specific sentiments from the customer reviews. Results were compared against three other papers that undertook some parts of this task on the same data.

Second, the system was applied to the SemEval-2014 task 4 data. It was used to extract product aspects; separately, to classify the sentiment expressed for tagged aspects; and, finally, extending beyond the SemEval-2014 task itself, classifying the aspects and sentiments thereof in sentences simultaneously.

6.7 Adapting to (slightly) different domains

The Amazon review data were not split into training and test sets, so a little creative experimentation was required. I performed two sets of experiments, one where I simply took each set of data and did cross-validation; and, as a second experiment to stretch and stress the system somewhat, I decided to use four out of the five products' data for training and then test on a fifth product, round-robin. This demonstrates some aptitude for domain adaptation, as the language learned at training time for, say, DVD players, is not directly transferable to test sentences pulled from cell phone reviews. Results are in Table 7.3 (page 85).

The SemEval-2014 task 4 data, on the other hand, were released in three sets: a very small development set; a large training set; and a reasonably large test test. This lent itself to easy experimentation: I trained my system on the training data and tested on the testing data, (having developed the algorithm and machine classification parameters using the Hu and Liu data, rather than the development set offered for the SemEval-2014 task). The language in the restaurant reviews seemed fairly different from the language in the laptop reviews; no domain adaptation experiment was attempted, although could be interesting future work.

6.8 Considering inherent limitations

There are a few limitations that may impose a ceiling on the performance of the system I developed.

Ambiguous language, whether manifesting as polysemy or as errors in sentence parsing, may cause problems.

The heuristic for reconciling product aspects and associated sentiments is surely imperfect, though it seems to be a reasonable compromise.

Scores on the sentence-level task may not intuitively reflect good performance of the individual aspect- and sentiment-extraction tasks. A similar situation is lamentably present in even the most capable part-of-speech tagging systems: the Stanford parser, for example, offers 97% part-of-speech accuracy on a word-by-word basis, but this translates into only about 54% accuracy at the sentence level (Manning, 2011).

Sentences are considered only in isolation. Particularly in the Amazon review data, resolving long-distance anaphora might improve performance, but the conception of the problem as a sentence processor that considers sentences only in isolation makes this problematic.

There are, of course, going to be annotation errors in the data; there will likely be some occasional instances where the computer does a better job annotating sentences than humans would, but will not get credit for doing so.

Despite such challenges, the system was developed to do a reasonable job identifying aspect-specific sentiments in opinionated text.

Chapter 7

Evaluation of experimental results

Aspect-specific sentiment structure was extracted from two data sets: text compiled from Amazon reviews of five products, and a set of restaurant and laptop reviews that were used in the SemEval-2014 task 4 competition.

Performance of the semi-supervised co-training algorithm is presented against that of a similar fully-supervised model. In addition, these results are compared against, for the Amazon reviews, three academic papers that did roughly comparable sentiment recognition tasks on the same data set, and for the SemEval-2014 data, against the results of the 31 teams that entered the competition.

Results of the individual aspect word and sentiment word classifiers are offered (indicating how well the system could pick the correct terms in the sentence), as well as results achieved at the sentence-level (indicating how well the system could distinguish different aspect-sentiment pairs in a sentence).

7.1 Metrics

Four key metrics (Witten and Frank, 2005, pages 163, 171, and 172) are used to evaluate my aspect-based sentiment extraction performance against others who have done similar (or identical) tasks on the same data:

- precision (P): the percentage of all positive predictions that were correct
  $P = \frac{truepositives}{truepositives + falsepositives}$
- recall (R): the percentage of all positive samples that were correctly predicted
  $R = \frac{truepositives}{truepositives + falsenegatives}$

- F_1 score (F_1): the harmonic mean of precision and recall, and a measure of accuracy
  $F_1 = 2 \times \frac{precision \times recall}{precision + recall}$
- accuracy (A): the percentage of all predictions that were correct
  $A = \frac{truepositives + truenegatives}{truepositives + truenegatives + falsepositives + falsenegatives}$

Using the product aspect classifier as an example, true positives are instances where a token that appears in the gold standard is a product aspect and the classifier classifies it as a product aspect. False positives occur when a token that is not a product aspect is classified (incorrectly) as a product aspect. A false negative comes about when a token is indeed a product aspect but the classifier classifies it as not being an aspect term, missing it. A true negative is a case where a token is not a product aspect and the classifier correctly predicts that it is not a product aspect.

Not all four metrics (P, R, F_1, A) are reported in all comparable tasks for all data sets. All available results are reported in the tables in this chapter; where other authors' results are not indicated, none were reported.

It could be argued that there are better metrics for analyzing classifiers (e.g., ROC curves); but the four aforementioned metrics are the only metrics reported in work on the two chosen data sets, and are, in various combinations, widely used in natural language processing tasks in general and in sentiment analysis specifically.

7.2 Performance on Amazon reviews

The experimental results that follow presently are those achieved working with the five sets of Amazon product reviews of Hu and Liu (2004). The data consisted of sets of reviews for an Apex AD2600 DVD player; a Canon G3 digital camera; a Creative Labs Nomad Jukebox Zen Xtra 40GB MP3 player; a Nikon COOLPIX 4300 digital camera; and a Nokia 6610 cell phone.

The data sets were each relatively small (ranging from 346 sentences to 1716 sentences). Each contained a mix of sentences with no opinions, single opinions, and multiple opinions. Sentences were pulled verbatim from user reviews, so some sentences are parts of informally bulleted lists. Many contain spelling errors. Some contain long-distance anaphora. The data set is rather challenging, but since three other strong papers have been published that have attempted sentiment classification tasks on this data, it provides a basis for meaningful comparison.

Two analyses are offered: one of the product aspect classifier, and one of the sentence-by-sentence performance of the system, including an experiment to analyze whether the system

can adapt itself to somewhat different product domains.

7.2.1 Classifier performance

Classifying product aspects

The supervised results of the aspect classifier alone (not working in concert with the sentiment classifier) are offered in Table 7.1. Because the data sets were small, it was not reasonable to attempt co-training at this stage; the system would be far too impoverished initially to be able to have sufficient confidence to label the new tokens to be learned at the first iteration.

Two versions of my results are offered: one where only aspect-specific sentiment expressions are considered (that is, I omit some obvious cases where products have been annotated as if they were aspects), denoted as *aspects only* in the accompanying results; and one where the gold standard is considered as-is, denoted *aspects + some products*. The former is the more interesting task, since one of the challenges in learning aspect-specific sentiments is to have the algorithm learn to ignore the products themselves. The latter is included only for ease of comparison to other work in the field. Examples abound in the data set where, for example, *DVD player* is labelled incorrectly as a product aspect, when it is instead a product. Such product-level annotations are particularly inconsistent in the data; so, while my system's recall improves when they are considered, accuracy drops due to a higher number of false positives when the products themselves are not annotated in the data. Nonetheless, the first set of my results omits such labels, while the second set includes them.

Two matching strategies are considered: one where only an exact match of aspect is allowed; and one where a partial match is given credit. The latter is useful for counting the effects of near misses, like tagging only *screen* instead of *LCD screen*, where either annotation could be considered reasonably correct, and where the difference could be attributable to the style of the annotator rather than to any inherent ideal and agreed-upon label.

My results were achieved by performing fivefold cross validation within each data set.

Precision is generally slightly worse than the rules-based methods of other papers that have done work on the same data set. Recall is very poor, which also contributes to poor F_1 scores. No other papers reported accuracy results for this task; my accuracy results are offered for future comparison, and to offer some reassurance that, despite poor recall, the system is able to perform the task with some competence.

[1]In Table 7.1, the recall of the simpler system in Hu and Liu (2004) reported for the Nikon camera could be a typo; but I present it as it was originally published. The F_1 score was calculated by me and may also be affected.

Data set	Apex DVD player (740 sentences)				Canon digital camera (597 sentences)				Creative MP3 player (1716 sentences)			
	P	R	F_1	A	P	R	F_1	A	P	R	F_1	A
Hu and Liu (2004)												
- rules-based pruning + heuristic for infreq. aspects	0.743	0.797	0.769	-	0.747	0.822	0.783	-	0.692	0.818	0.750	-
- with only machine-learned aspect lexicon, no pruning	0.531	0.754	0.623	-	0.552	0.671	0.606	-	0.573	0.652	0.610	-
- using FASTR instead	0.030	0.163	0.051	-	0.031	0.190	0.054	-	0.021	0.140	0.037	-
Popescu and Etzioni (2007)												
- only explicitly listed aspects	0.950	0.780	0.857	-	0.940	0.800	0.864	-	0.940	0.790	0.858	-
- more similar task	-	-	-	-	-	-	-	-	-	-	-	-
Ly *et al.* (2011)	0.867	0.797	0.831	-	0.842	0.822	0.832	-	-	-	-	-
Carter (aspects only)												
- exact match only	0.606	0.076	0.129	0.569	0.562	0.074	0.120	0.682	0.761	0.207	0.323	0.676
- partial match allowed	0.618	0.088	0.152	0.576	0.697	0.121	0.201	0.704	0.774	0.223	0.343	0.682
Carter (aspects + some product)												
- exact match only	0.497	0.073	0.125	0.536	0.630	0.088	0.144	0.602	0.768	0.195	0.308	0.632
- partial match allowed	0.571	0.095	0.162	0.546	0.680	0.135	0.214	0.616	0.777	0.211	0.328	0.639

Data set	Nikon digital camera (346 sentences)				Nokia cellular phone (546 sentences)				Mean			
	P	R	F_1	A	P	R	F_1	A	P	R	F_1	A
Hu and Liu (2004)												
- rules-based pruning + heuristic for infreq. aspects	0.710	0.792	0.749	-	0.718	0.761	0.739	-	0.722	0.798	0.758	-
- with only machine-learned aspect lexicon, no pruning	0.594	0.594[1]	0.594	-	0.563	0.731	0.636	-	0.563	0.680	0.614	-
- using FASTR instead	0.044	0.188	0.072	-	0.028	0.149	0.046	-	0.031	0.166	0.052	-
Popescu and Etzioni (2007)												
- only explicitly listed aspects	0.930	0.730	0.818	-	0.950	0.730	0.826	-	0.942	0.766	0.845	-
- more similar task	-	-	-	-	-	-	-	-	0.790	0.760	0.775	-
Ly *et al.* (2011)	-	-	-	-	0.769	0.761	0.765	-	0.826	0.793	0.809	-
Carter (aspects only)												
- exact match only	0.712	0.179	0.282	0.674	0.597	0.191	0.274	0.618	0.648	0.145	0.226	0.644
- partial match allowed	0.737	0.203	0.312	0.682	0.662	0.221	0.318	0.632	0.698	0.171	0.265	0.655
Carter (aspects + some product)												
- exact match only	0.786	0.261	0.387	0.615	0.655	0.221	0.319	0.557	0.667	0.168	0.257	0.588
- partial match allowed	0.804	0.286	0.419	0.629	0.689	0.255	0.360	0.573	0.704	0.196	0.297	0.600

Table 7.1: Performance of product aspect classifiers/extractors working on the same data

My system did particularly poorly on the DVD player data. Notably, it was the data set where the annotations contained the most non-aspect sentiment annotations (e.g., *DVD player* was a frequently rated aspect, even though it is a product, not an aspect of the product). When I removed these annotations from my gold standard data, I penalized myself. In the case where I considered such product-level annotations anyway, my system did no better, as the DVD player was not consistently annotated as an aspect, so while my system learned that annotation, it scored many false positives that would have counted as true positives, had the annotation been consistent.

Despite apparent weakness, all is not lost: while the tasks undertaken in the other papers are similar to the task I undertook, they are not identical.

I sought to extract the aspect terms in the sentence that matched the annotations, fair and square, using only the machine learning approach described in the previous chapter.

The Hu and Liu (2004) paper, which broadly aims to classify aspect-specific sentiments at the sentence level using the Amazon data, takes a relatively similar approach, with one major exception: their algorithm does a pre-pass over the entire data set (i.e., they do not make any effort to separate their data into training and test sets) to build a lexicon of aspects for that product. It undertakes four major steps: it automatically builds a lexicon of aspects that are mentioned frequently; then prunes that list (trying to narrow it down to only collocations); then tries to build a lexicon of aspects that are mentioned infrequently; and then combines these lexicons. At testing time, they use this lexicon to predict the aspects in the sentences. Their algorithm and results are impressive in their own right; but, whereas their system can use 100% of the data at training time to build knowledge about aspect labels, my system can use only 80% of the data in any given cross-validation iteration to try to build knowledge of aspects. (It may be reasonable to assume that a good number of the infrequently-mentioned aspects only appear once in a given data set, and would not get learned if separating training and test data, as I did by using cross-validation for evaluation.)

One part of their pruning algorithm involves heuristics to make sure that the word *life* does not enter the lexicon if it only appears in the aspect *battery life*; the rules for the heuristic they used to do so are a rather complex, whereas in my system I rely on a relative naïve classifier to determine if a token is inside or outside of a multi-word aspect expression. Perhaps such hand-crafted rules can offer advantages in a particular domain but the effort of hand-crafting and tuning them is high. For fairness' sake, I report their results both with and without this pruning and the heuristic to add rarely-mentioned aspects to their lexicon.

Hu and Liu (2004) also offered aspect classification results from a production system called

FASTR[2] as a baseline; these are also included. FASTR seems to be more of a noun phrase indexing tool than an aspect extraction tool, so its poor performance is perhaps not surprising.

The results reported in (Popescu and Etzioni, 2007), while impressive, are only for explicitly labelled noun phrase aspects, not all aspects. They do not consider implied features (e.g., *It is too bulky to fit in my pocket* would be annotated in the data with a negative opinion of the aspect *size*, which does not appear anywhere in the sentence). Effectively, the authors chose to exclude an entire set of difficult cases which would likely have reduced their recall, whereas I did not. The authors did not run any experiments with such implicit features on this dataset. The "more similar task" aggregate results listed for the (Popescu and Etzioni, 2007) paper reflects their efforts at classifying whether token-opinion-sentence tuples are aspects or not. This approach is rather unusual, but is more similar to the task I worked on, so the results might offer a more realistic comparison.

The work by Ly *et al.* (2011) is very similar to the work by Hu and Liu (2004), with the addition of some extra filtering of noun phrases when building the lexicon of possible product aspects; they parse sentences first and then only consider nouns or noun phrases that serve as either a subject or an object in the sentence. The same comments and caveats made of the Hu and Liu (2004) methodology therefore apply to the Ly *et al.* (2011) as well. These authors only applied their method to three of the five data sets; no reason for excluding the remaining two was given in their paper. It is unusual that they achieved exactly the same recall as Hu and Liu in all three data sets, while achieving better precision in each case; one might expect to see recall affected in at least some slight way (whether positively or negatively) if the method is indeed doing something novel.

Of all the comparisons of extracting aspects, the work by Hu and Liu (2004) without the hand-tuned heuristics for pruning aspects from the lexicon and finding infrequent aspects is the most similar task, and the most fair comparison; this is one where my system offers consistently (if slightly) better precision and poorer recall, although the difference in recall would probably be less if those authors had separated their data into training and test sets.

My system's ability to classify aspects in sentences in this difficult data set has precision that is in the ballpark of more task-specific rules-based approaches that undertake similar or simpler tasks. My system, however, suffers from relatively poor recall even taking into account the differences among systems.

[2]FASTR was created by Christian Jacquemin, and is available at http://perso.limsi.fr/jacquemi/FASTR/.

Classifying the sentiments of aspects

Sentiment-bearing terms were not tagged in the Amazon data, so there is no gold standard against which to compare.

7.2.2 Performance finding all aspect-sentiment pairs in a sentence

It is a more challenging and more interesting task to classify both product aspects and the associated sentiments in a sentence than is classifying aspects alone. To be clear: this is not the same task as classifying a whole sentence as positive or negative; rather, it is the task of finding sentiments expressed about stated aspects while ignoring sentiments about the products and services themselves.

A brief word is offered about how I counted results. As this section of the evaluation considers sentences, I considered a sentence a full true positive if and only if it correctly extracted all aspects and correctly predicted the sentiment of each aspect. In cases where I was only partially successful, I apportioned a percentage of the result to each class. For illustration, consider a sentence containing mentions of two aspects, each with a corresponding sentiment (and, for simplicity, it is assumed that there are no objective mentions of aspects in such a sentence). Let us pretend that my classifier correctly identifies one aspect and its sentiment (one true positive); misses the other mentioned aspect entirely (one false negative); and classifies another token in the sentence as an aspect though it is not one (one false positive). For the purposes of calculating precision, recall, and accuracy, I would consider this to be $^1/_3$ of a true positive, $^1/_3$ of a false negative, and $^1/_3$ of a false positive, and apportion no contribution to the count of true negatives. Correctly reconciling a product aspect and a sentiment-bearing token but incorrectly classifying the sentiment orientation (positive/negative) was considered a false negative, just as if the aspect had been missed or if the sentiment-bearing word had been missed.

In this manner, I gave partial credit to my system if it was able to reconcile parts of very long compound sentences that mentioned multiple aspects.

As in all human-annotated data, some annotation errors were noticed. One particularly egregious one is the following sentence in the cell phone reviews:

> *t-mobile[-2]* their network coverage is very sporadic , and the network always seems overloaded , resulting in very unpleasant calling experience .

which my system tagged as follows:

> their **network coverage**$_{\text{aspect}}$ is very **sporadic**$_{\text{sentiment-neg}}$, and the **network**$_{\text{aspect}}$ always seems **overloaded**$_{\text{sentiment-neg}}$, resulting in very **unpleasant**$_{\text{sentiment-neg}}$ calling experience .

When aspect words and sentiment-bearing words were reconciled, the system tagged two aspect-sentiment pairs: a negative sentiment about the network coverage (reconciling *network coverage* with *sporadic*), and a negative sentiment about the network (reconciling the latter with *overloaded*). The latter could be improved by recognizing that *network* in this context means *network capacity*, specifically. The system seems to have done a good job of ignoring *calling experience*, which looks like an aspect at first glance, but is arguably not so; in the pre-smartphone era, a bad calling experience is not entirely removed from saying that it's merely an inherently bad phone (just as, arguably, a car that *drives badly* does not have an aspect of *driving*). At best, *calling experience* seems like it might fall in a grey area, so its exclusion is perhaps forgivable. And nonetheless, the annotations that my system provided seemed much better, in my opinion, than rolling it all up and blaming the cellular carrier (in this case, T-Mobile); it seems reasonable to state that *T-Mobile* is not an inherent aspect of the cell phone model (as the phone may be perfectly serviceable in a different state, on a different carrier, or in a different country), but that poor ability to pick up a network signal and maintain a connection to the network could well be aspects of that particular phone (and so should not be discarded entirely). In the evaluation of my system's performance on this sentence, I scored a 0.67 false positive and 0.33 false negative, which counts as a total failure; such a score is perhaps not entirely reflective of the adequate (or better) system performance in this example. Finally, from a sentence-processing perspective, there is no local indication that the phone is on the T-Mobile network, making that aspect annotation particularly challenging.

Two sets of sentence-level experiments were performed. In the first, each product's data set was considered only against itself, and the system was run through fivefold cross validation (as before). In the second sentence-level experiment, I used four of the products' data to train the system, and tested on the fifth product's data, in order to demonstrate whether the system had the ability to test on out-of-domain data. The inclusion of two digital cameras in the data made for an interesting aspect; the two trials that tested on digital cameras also had some digital camera data in their training set, whereas in the other three such trials, the test data was in an entirely different product domain than the training data.

Data set	Apex DVD player				Canon digital camera				Creative MP3 player			
	P	R	F_1	A	P	R	F_1	A	P	R	F_1	A
Hu and Liu (2004)	0.607	0.653	0.629	0.927	0.643	0.719	0.679	0.927	0.589	0.784	0.673	0.842
Popescu and Etzioni (2007)	-	-	-	-	-	-	-	-	-	-	-	-
Ly *et al.* (2011)	-	-	0.791	-	-	-	0.753	-	-	-	-	-
Carter (fivefold cross-validation)												
- aspects only	0.767	0.037	0.069	0.578	0.804	0.119	0.204	0.713	0.880	0.153	0.260	0.674
- aspects + some products	0.969	0.043	0.081	0.553	0.813	0.135	0.224	0.635	0.854	0.143	0.244	0.630

Data set	Nikon digital camera				Nokia cellular phone				Mean			
	P	R	F_1	A	P	R	F_1	A	P	R	F_1	A
Hu and Liu (2004)	0.554	0.634	0.591	0.946	0.815	0.675	0.738	0.764	0.641	0.693	0.662	0.881
Popescu and Etzioni (2007)	-	-	-	-	-	-	-	-	0.679	0.676	0.678	-
Ly *et al.* (2011)	-	-	-	-	-	-	0.722	-	-	-	0.755	-
Carter (fivefold cross-validation)												
- aspects only	0.870	0.182	0.294	0.691	0.819	0.191	0.299	0.652	0.828	0.136	0.225	0.662
- aspects + some products	0.936	0.255	0.396	0.645	0.822	0.218	0.335	0.594	0.879	0.159	0.256	0.611

Table 7.2: Comparing results of fivefold cross-validation within each data set to similar sentence-level tasks

Comparative results - supervised case

Performance of my system running in a supervised fashion (with fivefold cross-validation) is listed in Table 7.2, along with results of several other papers that have tried to do somewhat comparable aspect-specific sentiment extraction on the same data (or a subset thereof).

Once again, my results are compared to tasks that are similar but not identical. The second set of results, labelled *aspects + some products*, makes no effort to discard obvious cases where products were annotated as if they were product aspects, and is perhaps the more fair comparison against others' work. The first set of results, labelled *aspects only*, is the more interesting set of results, as product annotations (incorrectly annotated as product aspects) were removed from the data.

To calculate accuracy, Hu and Liu (2004) used a voting system to determine opinion polarity only at the sentence level (not at the product aspect level). For example, if a sentence has two positive opinions of two aspects and a negative opinion of an aspect, the authors consider the sentence positive and deem their algorithm successful if the sentence is classified as positive. This is a simpler task than I have undertaken; this approach glosses over individual

errors tagging the sentiments of aspects. The authors list precision and recall at the sentence level, as well: they calculate whether they have correctly classified a sentence as containing opinions or not, which is also a simpler task.

The same caveat that was noted in the aspect classifier results applies here too: Hu and Liu (2004) did not separate data into training and test sets, and built a product aspect lexicon using the entire data set, which would presumably boost their recall, particularly when they can get credit for product aspects that only appear once in a given data set.

A note on reconciling the results: the authors listed their digital camera results as "digital camera 1" and "digital camera 2" in (Hu and Liu, 2004), which is ambiguous. Thankfully, in (Ding *et al.*, 2008), which is co-authored by Bing Liu, the number of reviews for each of the digital cameras is listed, so I have been able to determine that that "digital camera 1" refers to the Canon G3, and that "digital camera 2" refers to the Nikon Coolpix 4300. Popescu and Etzioni (2007) simply enumerate all five data sets, but since they include comparative results from (Hu and Liu, 2004), I manually reconciled their results.

Popescu and Etzioni (2007) use a different sentence-level experimental setup: they select 800 (token, aspect, sentence) tuples from the five product review sets, manually annotate each tuple as positive/negative/neutral (rather than inferring it from the sentence-level annotations in the source data, as I chose to), and then separately report the results of extracting the opinions and extracting the opinion polarity (whereas I and the other papers chose to report the results of the combined task). I have provided an estimate for the combined task by multiplying the precision results for their two separate tasks, and multiplying the recall results for their two separate tasks. As the raw numeric results are not strictly percentages, this is perhaps not a perfect proxy, but gives an optimistic indication how a system with their reported precision and recall might perform on a combined task.

Ly *et al.* (2011) offer several variations on their experimental method. Their best results were using non-hierarchical clustering of product aspects, reported in Table 3 in their paper; these are the results I compare against. They report only F_1 score. The experiment they performed, however, was somewhat limited. The authors selected only three out of five of the data sets (the Canon camera, the Apex DVD player, and the Nokia phone). In each data set, they extracted only opinionated sentences (thereby omitting all objective sentences). Furthermore, they manually partitioned the sentences into subtopics, one partition per opinion, and manually mapped similar and synonymous terms for product features into clusters. These decisions simplify their task, reducing the likelihood of false positives, so perhaps their results should be taken with a grain of salt.

The precision of my system is notably better on this sentence-level task than the results

Data set	Apex DVD player				Canon digital camera				Creative MP3 player			
	P	R	F_1	A	P	R	F_1	A	P	R	F_1	A
Hu and Liu (2004)	0.607	0.653	0.629	0.927	0.643	0.719	0.679	0.927	0.589	0.784	0.673	0.842
Popescu and Etzioni (2007)	-	-	-	-	-	-	-	-	-	-	-	-
Ly *et al.* (2011)	-	-	0.791	-	-	-	0.753	-	-	-	-	-
Carter (domain adaptation)												
- aspects only	0.900	0.028	0.054	0.576	0.763	0.138	0.234	0.718	0.763	0.105	0.185	0.653
- aspects + some products	0.833	0.029	0.056	0.546	0.760	0.159	0.263	0.643	0.768	0.110	0.193	0.612

Data set	Nikon digital camera				Nokia cellular phone				Mean			
	P	R	F_1	A	P	R	F_1	A	P	R	F_1	A
Hu and Liu (2004)	0.554	0.634	0.591	0.946	0.815	0.675	0.738	0.764	0.641	0.693	0.662	0.881
Popescu and Etzioni (2007)	-	-	-	-	-	-	-	-	0.679	0.676	0.678	-
Ly *et al.* (2011)	-	-	-	-	-	-	0.722	-	-	-	0.755	-
Carter (domain adaptation)												
- aspects only	0.838	0.207	0.332	0.699	0.887	0.143	0.247	0.647	0.8302	0.124	0.210	0.659
- aspects + some products	0.882	0.233	0.369	0.631	0.900	0.135	0.235	0.571	0.829	0.133	0.223	0.601

Table 7.3: Comparing results of domain adaption experiment (training on four data sets and testing on the fifth) to similar sentence-level tasks

in (Hu and Liu, 2004) and in (Popescu and Etzioni, 2007), though my recall is rather weak, albeit achieved on a more difficult formulation of the task.

Domain adaptability

A final experiment on the Amazon review data was conceived to test whether my system can adapt to domains that are at least somewhat different than the data upon which it was trained (a scenario similar to the task of being able to handle unforeseen language); and, simultaneously, to see if the system would perform better on a larger training set. (While the size of the training set is not of particular concern to rules-based systems, a large training set is rather important for machine learning systems.)

In each trial, I trained my system on the reviews of four products and tested on the fifth product. I compare my results to others' somewhat similar sentence-level tasks in Table 7.3.

Results were not terribly different than the experiment where I merely performed fivefold cross-validation within each set of product reviews. Aggregate precision went up trivially, and aggregate recall went down trivially. This performance could be considered unimpres-

sive or commendable. One might expect that, because the system was trained on a much larger data set in this experiment, it should perform better than in the cross-validation experiment. On the other hand, in this particular experiment, at training time, the system was deprived of the domain-specific terms that it would learn in the cross-validation experiment (though to a lesser extent in the two trials where a set of digital camera reviews was used as the test set, since at least some digital camera data would appear care of the other digital camera's reviews). Perhaps the increase in the amount of training data offsets the decrease in the domain applicability of the training data, effectively cancelling each other out. Overall, results were similar whether the system was performing cross-validation within a single product's reviews or whether it was training on four products' reviews and training on a fifth, characterized by good precision and poor recall.

7.3 Performance on SemEval-2014 task 4 data

The experimental results that follow presently are those achieved analyzing the data used in the SemEval-2014 task 4 competition (Pontiki *et al.*, 2014). There are two sets of data: one consisting of sentences extracted from reviews of a large number of restaurants in the New York City area, and one consisting of sentences extracted from reviews of a large number of laptops of various makes and models.

There were two versions of the SemEval-2014 data prepared; almost all competitors used the version one data; and those competitors who submitted results using version two data generally also submitted results of the same experiments on version one data. Accordingly, I chose to use the version one data. There were few differences between the two versions.

Competitors had the ability to state whether they would be creating a *constrained* or *unconstrained* system. The distinction was introduced so that simple systems that had relatively little world knowledge (hence, constrained) could be compared against each other, and systems trained on data not provided as a part of the competition could be compared against each other (hence, unconstrained). Put more explicitly, the intent was to give advantage to simple systems over systems that were trained on, for example, all of Wikipedia. The organizers deemed, however, that any number of lexicons could be used in the constrained task, and offered no particular constraints for how such lexicons could be created; so, in practice, there was little discernible difference in the constrained and unconstrained solutions.[3] Off-

[3]One team (Kiritchenko *et al.*, 2014) that did very well in constrained versions of the tasks attributed their success in general to the work compiling many lexicons solely for the competition, and specifically credits their excellent results to one particular domain-specific word-aspect lexicon they developed for the task. Crafting a

the-shelf statistical parsers (e.g., that in Stanford CoreNLP) had been considered acceptable by the organizers in the constrained task, even though such parsers are trained on many corpora from outside the competition domain. Accordingly, I have chosen to ignore such considerations and compare my results against both the constrained and unconstrained results; though, had I compiled a lexicon of common (or even of all) WordNet adjective-attribute pairs rather than looking them up on-the-fly, my system would presumably have been considered to be constrained by the organizers' definition.

Two analyses are offered: one of the individual sentiment word and product aspect word classifiers (corresponding to the subtasks 1 and 2, respectively, in the SemEval-2014 task 4 competition), and one of the sentence-by-sentence performance of the system. My results are compared against the 31 other teams that competed in the SemEval-2014 task 4. It is probably unreasonable for this sole M.A.Sc. student to outperform the best of the best in this competition; the most successful entrants appeared to be teams of four or more people where members have doctorate degrees in natural language processing and impressively illustrious careers, including (in some cases) previous successes in SemEval and other such competitions. Notwithstanding, it seems reasonable to try to achieve results at or near the mid-field.

The two SemEval-2014 task 4 data sets are much larger than those in the Amazon review data set. The laptops data set contains 3045 training sentences and 800 test sentences, while the restaurants data set contains 3041 training sentences and 800 test sentences; by comparison, the five Amazon review sets contained 346, 546, 597, 740, and 1716 sentences, a further 20% of which were excluded from training when running cross-validation. Also, whereas many sentences in the Amazon reviews contained no aspect-specific sentiments, it appears that all of the sentences in the SemEval-2014 task 4 data included at least one aspect-sentiment expression. A machine learning system such as the one I developed can probably be expected to perform better when fed with more data in this manner.

7.3.1 Classifier performance

The performance measurements of the aspect word and sentiment-bearing word classifiers are offered in Tables 7.4 and 7.5, respectively. These results examine solely whether the systems can tag aspect words in a sentence, and, given an aspect word, whether the system can predict its polarity. This is a simpler challenge than evaluating performance on entire sentences (which is offered in the following subsection), where multiple aspects might be present

specific and nominally single-use lexicon using resources from, e.g., Wikipedia or WordNet, starts to blur the distinction between the constrained and unconstrained tasks.

in one sentence, and where a system has to both identify the word and classify its sentiment orientation.

My system is compared against the aggregated results of the 31 teams that participated in the SemEval-2014 task 4. Rather than listing all 31 teams' results[4], four results are listed (per subtask): the mean performance of all competitors; the performance of the least successful team; the performance of the median team; and the performance of the most successful team. Note that the organizers of the SemEval-2014 task 4 released only precision, recall, and F_1 scores for the first subtask (tagging aspects in sentences), and only accuracy scores for the second subtask (predicting polarity given tagged aspects).

In general, all systems (including mine) appeared to have an easier time with the restaurant reviews compared to the laptop reviews; lowest, highest, mean and median precision and recall were higher in the restaurant reviews.

Classifying product aspects

I compared my system's ability to label product aspects in sentences (independent of any effort to glean stated sentiments) to the results of those who participated in the SemEval-2014 task 4 subtask 1 challenge.

In the fairest comparison (Table 7.4, *exact match only* rows), my system, operating in a supervised manner and allowing only exact matches in cases where aspects were composed of multiple words, offered higher precision than all other systems on the laptop reviews, though perhaps not significantly so (Figure 7.1, left side). My system – whether running in a supervised manner or using co-training with only half of the training data being labelled – offered precision on the restaurant reviews that was roughly tied with the top competitor, and much higher than the mean and median (Figure 7.1, right side).

My system offered weaker performance in recall; somewhat below the mean and median when processing the laptop reviews in a supervised manner, and roughly tied with the mean and well below the median when examining restaurant reviews. Co-training offered much worse recall, though still better than the worst of the SemEval-2014 task 4 competitors; though, of the two, the reduction in recall when co-training was much less pronounced in the restaurant review data.

With high precision and relatively weak recall, my system achieved F_1 scores that placed mid-pack among SemEval competitors (Figure 7.2) when considering all test data in a supervised manner. When co-training with the laptop data, my F_1 was below average; whereas

[4]Full SemEval-2014 task 4 results are available at http://alt.qcri.org/semeval2014/task4/

Data set	Laptop reviews				Restaurant reviews			
	P	R	F_1	A	P	R	F_1	A
SemEval task 4 subtask 1 (aspect term extraction)								
mean performance	0.690	0.504	0.562	-	0.767	0.672	0.708	-
lowest performance	0.231	0.148	0.239	-	0.371	0.340	0.383	-
median performance	0.756	0.551	0.605	-	0.818	0.720	0.727	-
highest performance	0.848	0.671	0.746	-	0.909	0.827	0.840	-
Carter (fully supervised, using all training data, exact match only)	0.863	0.401	0.547	0.632	0.915	0.681	0.781	0.647
Carter (training with first half, co-training with second half, exact match)	0.822	0.292	0.430	0.581	0.909	0.587	0.713	0.589
Carter (training with second half, co-training with first half, exact match)	0.829	0.224	0.353	0.559	0.910	0.616	0.734	0.606
Carter (fully supervised, using all training data, allowing partial match)	0.899	0.523	0.661	0.710	0.934	0.783	0.852	0.764
Carter (training with first half, co-training with second half, partial match)	0.884	0.410	0.560	0.655	0.928	0.697	0.796	0.704
Carter (training with second half, co-training with first half, partial match)	0.886	0.338	0.489	0.625	0.928	0.723	0.813	0.722

Table 7.4: Comparing aspect word extraction results on SemEval task 4 (subtask 1) data

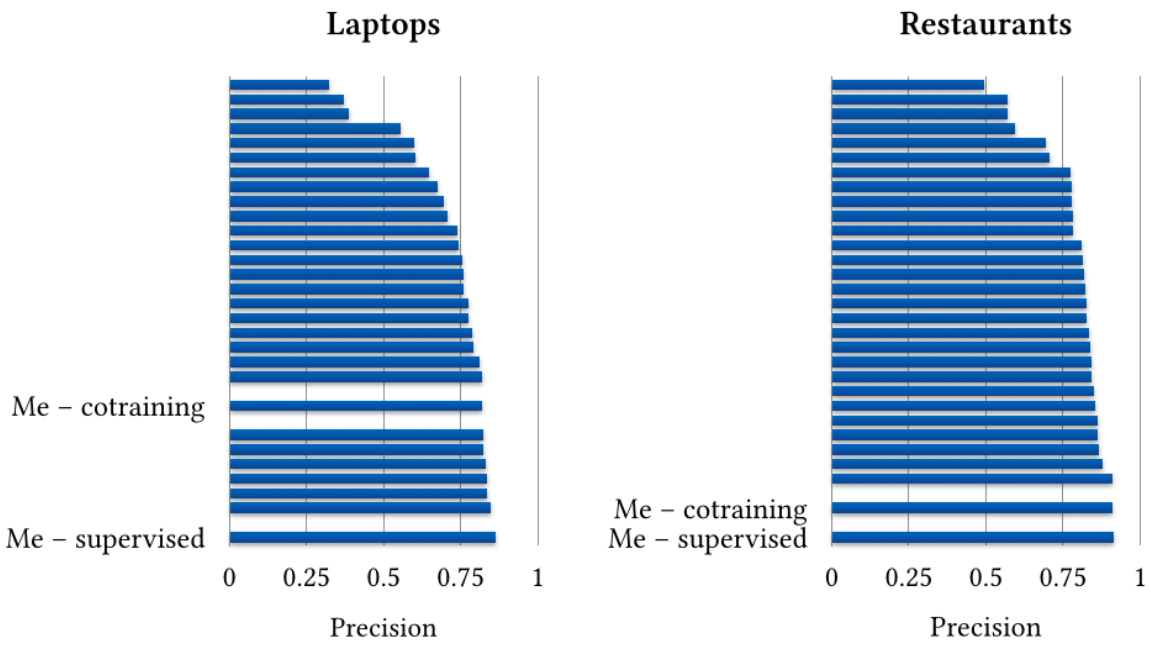

Figure 7.1: Comparing aspect word identification precision on SemEval task 4 (subtask 1) data

when co-training with the restaurant reviews, my F_1 scores were slightly above the mean and tied with the median.

Allowing a partial match (Table 7.4, *partial match* rows) when considering aspects (e.g., if an aspect in the gold standard data is labelled *LCD screen* and I give myself credit for labelling only *screen*), my results improve across the board; in particular, the poor recall previously observed in the laptop review data seems to increase to competitive levels (above mean but below median when running in a supervised manner). This is not an entirely fair comparison, as no other competitors were evaluated in this manner; it does provide an indication, however, that the classifier is building a reasonable base of knowledge.

The product aspect classification performance of my system on the SemEval-2014 data can be described as being roughly average among the 31 teams, and is characterized by very high precision and rather low recall.

Accuracy was not reported in the competition results, but I offer my system's accuracy performance for future comparison and to illustrate that, even in cases where recall is low, accuracy remains at reasonable levels.

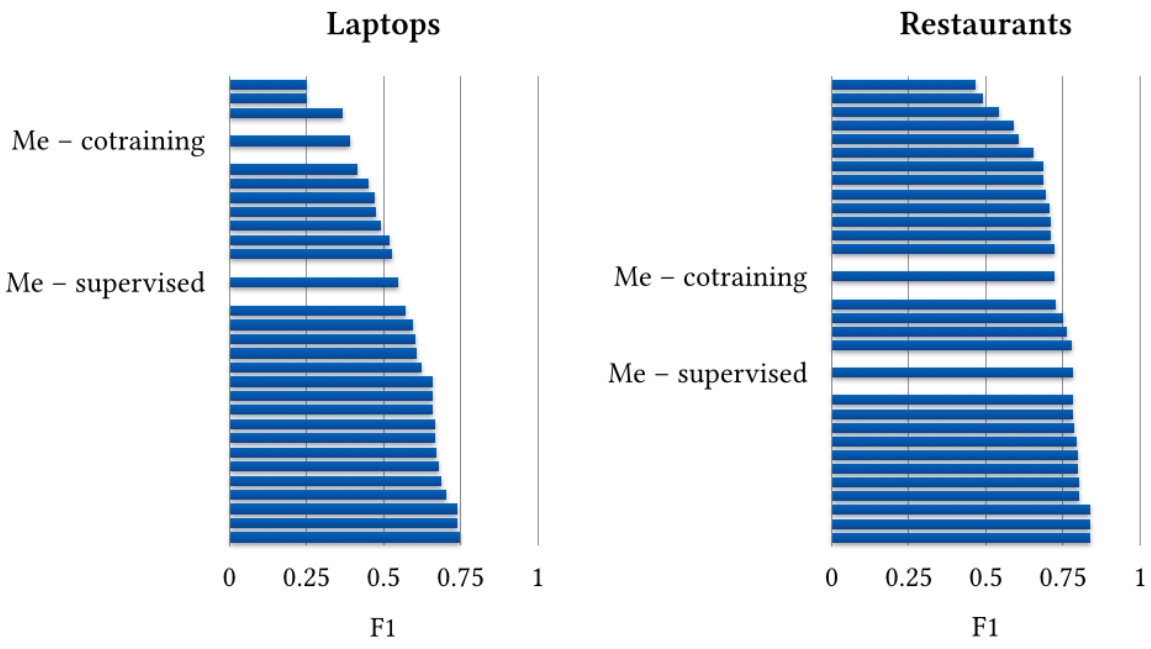

Figure 7.2: Comparing aspect word identification F_1 scores on SemEval task 4 (subtask 1) data

Classifying the sentiments of aspects

The second subtask in the SemEval-2014 task 4 challenge was to predict the stated aspect-specific sentiment of sentences where the product aspect(s) was/were already labelled. I compared my system's performance on this task to those who entered the challenge (Table 7.5 and Figure 7.3).

Accuracy was the only metric reported by the challenge organizers. This makes some sense: since the task is one of choosing among four possible sentiment values (positive, negative, objective, and "conflict", indicating both positive and negative sentiments about the aspect in the same sentence), there can be no concept of a false positive. By extension, a system that gets at least one correct result in the entire data set will score a precision of 1.0, making precision a largely useless metric in this task.

When my system was used in an entirely supervised manner, it (just barely, and probably not significantly) beat all competitors in the SemEval-2014 task 4 challenge. Even the co-training results are well above both mean and median on the laptop reviews. On the other hand, when trying to classify sentiments in the restaurant reviews, performance of the supervised system was tied with the mean and very slightly lower than the median competitor; and the accuracy when co-training was almost 10% worse than the mean and median competitor,

Data set	Laptop reviews				Restaurant reviews			
	P	R	F_1	A	P	R	F_1	A
SemEval task 4 subtask 2 (determine polarity of given aspects)								
mean performance	-	-	-	0.590	-	-	-	0.691
lowest performance	-	-	-	0.365	-	-	-	0.417
median performance	-	-	-	0.586	-	-	-	0.708
highest performance	-	-	-	0.705	-	-	-	0.810
Carter (fully supervised, using all training data)	1.00	0.467	0.637	0.719	1.00	0.591	0.743	0.690
Carter (training with first half, co-training with second half)	1.00	0.370	0.541	0.668	1.00	0.529	0.692	0.643
Carter (training with second half, co-training with first half)	1.00	0.360	0.529	0.662	1.00	0.513	0.678	0.631

Table 7.5: Comparing sentiment orientation classification results (given tagged aspects) on SemEval-2014 task 4 (subtask 2) data

though still much better than the least successful teams that participated in the challenge.

My system thus appears to offer fairly compelling performance in classifying the sentiments expressed about known product aspects in these data sets, even when co-training with only half of the training data being labelled.

7.3.2 Performance finding all aspect-sentiment pairs in a sentence

It is a more challenging and more interesting task to classify both product aspects and the associated sentiments in a sentence than is classifying aspects in isolation. Sadly, this was not a part of the SemEval-2014 task 4 challenge, although it would be a natural extension thereof. My system's sentence-level results are listed in Table 7.6.

Compared to the performance of my system analyzing full sentences in the Amazon review data (Table 7.2, page 83), my system performs much better on the SemEval-2014 task 4 data, with somewhat better precision and much better recall.

The performance of co-training in a real-world and suitably difficult task can be analyzed here. The co-trained models were trained using only half as much labelled data as the supervised model. Precision remained sufficiently high to conclude that it was tied with the

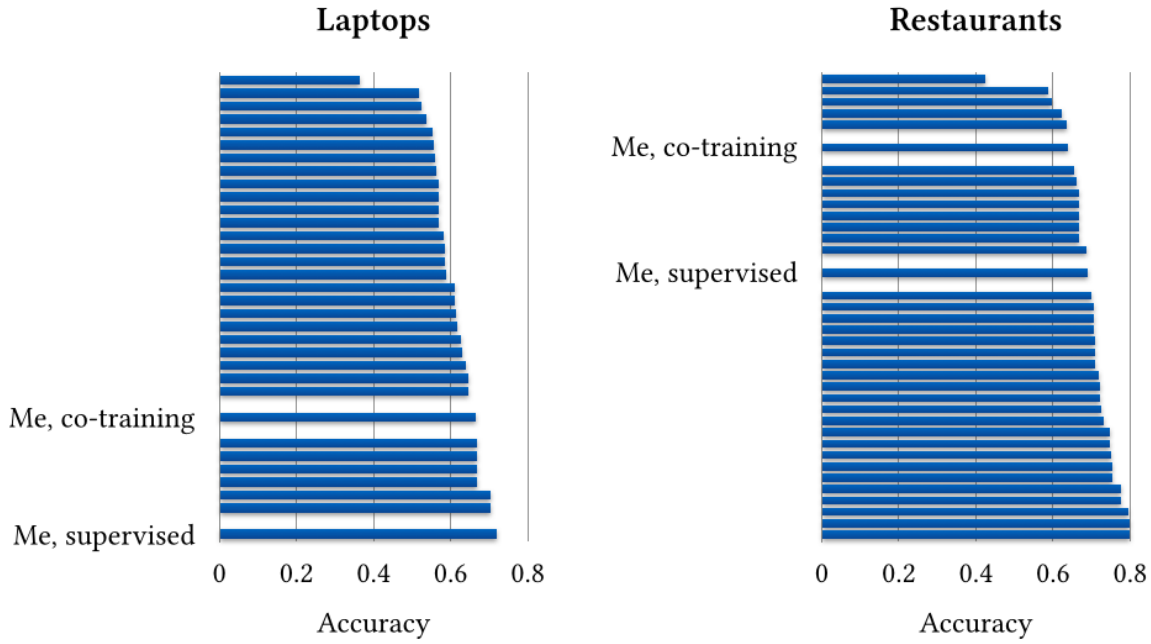

Figure 7.3: Comparing sentiment orientation classification results (given tagged aspects) in SemEval-2014 task 4 (subtask 2) data

Data set	Laptop reviews				Restaurant reviews			
	P	R	F_1	A	P	R	F_1	A
Carter - test sets A & B combined								
- fully supervised, using all training data, all test data	0.890	0.268	0.412	0.596	0.936	0.507	0.658	0.600
- training with first half, co-training with second half	0.880	0.121	0.213	0.528	0.933	0.321	0.477	0.468
- training with second half, co-training with first half	0.935	0.103	0.185	0.523	0.923	0.354	0.512	0.488
- mean performance loss when using co-training with only half of the labelled training data	-2%	58%	52%	12%	1%	34%	25%	20%

Table 7.6: Aspect-specific opinion extraction results on SemEval-2014 task 4 competition data (classifying aspects and sentiments simultaneously, given unlabelled sentences)

supervised model. Recall dropped quite a bit. In the laptop reviews, the F_1 score roughly halved, whereas in the restaurant reviews it dropped an average of 25%. Accuracy suffered 22% in the worst of the trials. These results are somewhat comforting: using only half as much training data seems to reduce the F_1 by half, at worst, while maintaining high precision. This could be an acceptable trade-off in a particular application domain, since labelled data is both difficult and expensive to produce.

By comparison, Nigam and Hurst (2004) offers insight into humans' classification performance. Humans seem to be able to classify polarity at the sentence level with roughly 88% precision and 70% recall. My system, performing a more nuanced task of classifying aspect-specific sentiments at the sentence level, certainly meets this level of precision, if not exceeding it; though does nowhere near as well at recall. (Human brains, viewed as a natural language processing machine, are trained on much larger language models than my system, so one could intuitively expect that humans might have better recall than mere natural language processing software trained on about 3000 sentences of data.)

7.4 Finer points of the co-training algorithm

The co-training algorithm I developed (Algorithm 2, right column, page 52) has several parameters that can be tuned.

At each iteration, the algorithm will add to the training set a maximum given number of new examples. Reducing this number increases the computational burden and tends to lead to higher precision and lower recall (Table 7.7).[5] In effect, the algorithm can be directly tuned for higher precision or higher (albeit still low) recall, which is may be useful for some applications. (By comparison, support vector machines have been criticized in some circles for not being directly tuneable for precision versus recall.)

The other key tuneable parameter in my algorithm is the prediction confidence floor to be used for data added at each iteration; the algorithm requires that the SVM classifier have at least a certain percentage confidence of its prediction for the exemplar to be considered for inclusion in the next round of training data. It appears that the support vector machine's raw performance makes this value largely moot; increasing this threshold above 55% results in usable data being omitted from the trained model. This may not be true of all data sets, but at least for the natural language processing tasks explored in this dissertation, using this

[5]The results of in this table indicate sentence-level performance, but reflect a mildly different set of features than was used in other experiments in this dissertation; as such, the numbers in this table are not directly comparable to other results, but the trend is illustrated nonetheless.

Maximum number of unlabelled tokens added to each trained model per co-training iteration	Laptop reviews				Restaurant reviews			
	P	R	F_1	A	P	R	F_1	A
250	0.943	0.117	0.209	0.531	0.936	0.241	0.384	0.413
2500	0.885	0.127	0.223	0.531	0.920	0.321	0.475	0.464
10 000	0.879	0.155	0.264	0.543	0.926	0.372	0.530	0.502
no co-training	0.897	0.228	0.364	0.579	0.946	0.432	0.593	0.551

Table 7.7: Effect of changing maximum number of tokens classified per co-training iteration, using SemEval-2014 task 4 data and subject to confidence floor of 55%

confidence floor threshold offered relatively little benefit beyond pushing marginal examples to later iterations of co-training; with more data at later iterations, the support vector machine has more estimated confidence in its classification, so marginal cases get added much later in the co-training process, or get rejected with greater certainty.

7.5 Are these good results?

I strove to make software that was both useful and usable on a novel task.

Performance on three related tasks was analyzed:

- the ability to identify product aspects in opinionated text taken from reviews
- the ability to predict the sentiment (positive or negative) of given aspects
- the ability, given an unlabelled sentence, to identify any product aspects mentioned and any particular sentiment expressed about such aspects

My system's performance when operating in a supervised manner (using all available labelled data) was illustrated; and, when appropriate, so too was its performance when using only half of the labelled data as the seed for co-training, and considering the other half of the training data as if it were not labelled.

A subjective summary of performance on the aspect identification and sentiment classification tasks is offered in Table 7.8.[6] It is difficult to make a strong claim about the results

[6]The table summarizes the F_1 scores of the aspect labelling and the accuracy scores of the sentiment prediction. *Excellent* indicates performance better than any other comparable effort. *Very good* indicates performance above the mean and median but below the best comparable work. *Good* indicates performance around the

Task	Supervised training	Co-training
Identify product aspects in a sentence		
- Amazon reviews of five products	Poor	N/A (not enough text)
- SemEval-2014 task 4 laptops	Good	Poor
- SemEval-2014 task 4 restaurants	Very good	Good
Classify sentiment expressed about aspect		
- Amazon reviews of five products	N/A (dissimilar gold standard)	N/A (dissimilar gold standard)
- SemEval-2014 task 4 laptops	Excellent	Very good
- SemEval-2014 task 4 restaurants	Good	Mediocre

Table 7.8: Subjective evaluation of performance on the evaluation tasks

gleaned from the Amazon data, as direct comparison to others' work was difficult. When processing the SemEval-2014 data, my system showed better results on the restaurant data in both subtasks than it did when considering the data about laptops; but, compared to other teams who participated in the SemEval-2014 competition, it performed *relatively* better when working with the restaurant data. There were notable differences in language between the restaurant and laptop data sets; anecdotally, sentiment-bearing words in the restaurant data seemed to be more varied and creative. My system was tuned on the Amazon reviews of five electronic products, and the language in those reviews bears more similarity to the laptop data used in SemEval-2014 than to the restaurant data; this could account for the better performance in the experiments with the laptop reviews.

The more interesting task of starting with an unlabelled sentence and finding the opinions expressed about specific aspects therein is harder to evaluate comparatively. My performance on this task was numerically poor if considering others' work on the Amazon reviews; however, there were assumptions or methodological differences in others' work that made the comparison difficult, and there were notable annotation errors where product sentiments were included when they should have been omitted. Performance analyzing unlabelled sentences in the SemEval-2014 task 4 data was noticeably better; but since sentence-level per-

mean and median. *Mediocre* indicates performance no more than 10% below the mean and median. *Poor* means performance much worse than the mean and median. *N/A* indicates that no meaningful comparison could be performed.

formance was not part of the challenge, there are no results against which to compare mine. Notwithstanding, at this challenging task of extracting nuanced opinions from raw sentences, my system offered excellent precision and fairly weak recall. Co-training using only half of the labelled data did not affect precision, but caused quite a drop in recall.

These are good results. The task is quite challenging, and my system's performance appears to be competitive.

My system generally offers very good precision and relatively poor recall. While the F_1 score is perhaps the customary metric on which to focus, an argument can be made that precision is the more appropriate metric in this task. For example, Sokolova and Lapalme (2009) analyze and contrast various common performance measures in NLP. They note that, in contrast to recall, accuracy, and F_1 score, "Precision…may be more reliable when manual labelling follows rigorous rules for a negative class" (Sokolova and Lapalme, 2009, Section 5, Invariance I4), which is the case in the annotation schemes used for the data sets used in these experiments. They also note that recall[7] (and, by extension, F_1 score) is less "reliable if the representative power of positive and negative classes is uncertain" (Sokolova and Lapalme, 2009, Section 5, Invariance I8), which is the case here, since, in any new sentence, it is uncertain whether the proportion of sentiment-bearing words, aspect words, and other words is remotely proportional to the training data (or, more simply: since sentences have different authors, a new sentence may be completely surprising in its language compared to the learned models). These two properties suggest that precision may in fact be a more useful measure for the task at hand given the data sets used.

The system can perform aspect-based sentiment extraction and is available (Appendix A). This makes it *usable*.

The high precision of my system would help it excel in an ensemble learning scenario, which makes it both relevant and interesting: when my system attempts to extract an aspect-sentiment pair, it tends to do so correctly; but it misses many such pairs. Simply: when it guesses, it tends to guess right; it knows its own expertise.

The effectiveness of artificial intelligence techniques that share this particular property has been demonstrated in IBM's demonstrations of its Watson question answering system, which is an ensemble of hundreds of specialized language analysis (and information retrieval) components, each with high precision and good confidence estimation; recall is not terribly important.[8] In such a system, where many components might vote on a result, it is important

[7] along with sensitivity and AUC, two measures that are less common in NLP applications

[8] In a paper describing Watson's performance competing on the *Jeopardy!* television show, it was noted that "confidence, precision, and answering speed are of critical importance." (Ferrucci *et al.*, 2010)

that each component vote only if it is confident in its answers, so high precision is critical. Low recall can be worked around by adding new components with different expertise; effectively building coverage by having components which can each handle a small subset of data in a complementary fashion. Watson can be considered to be among the most advanced and effective systems in the natural language processing domain at present. Accordingly, my system having low recall, while not ideal, does not disqualify it from excellence; rather, it demonstrates a property that makes it useful for a particular set of NLP applications.

Furthermore, my system could estimate its own competence on a given sentence, since its underlying classifiers offer estimates of classification confidence; it is relatively trivial to heuristically or using AI combine these into a sentence-level confidence estimate. Such an ability makes this technique applicable in systems like Watson, whose creators noted that:

> "Confidence estimation was very critical to shaping our overall approach in DeepQA. There is no expectation that any component in the system does a perfect job — all components post features of the computation and associated confidences, and we use a hierarchical machine-learning method to combine all these features and decide whether or not there is enough confidence in the final answer to attempt to buzz in and risk getting the question wrong." (Ferrucci *et al.*, 2010, page 60)

> "What is far more important than any particular technique we use is how we combine them in DeepQA such that over-lapping approaches can bring their strengths to bear and contribute to improvements in accuracy, confidence, or speed." (Ferrucci *et al.*, 2010, page 68)

Accordingly, there is at least one well-known and successful NLP system for which high precision and the ability to estimate one's own competence are key properties of a component that can operate in an ensemble learning scenario. My software has these two properties. I believe this makes my software *useful*.

The system thus appears to be both usable and useful; and thus, I postulate that these may be good results indeed.

Chapter 8

Conclusions and future work

8.1 Conclusions

Several conclusions can be drawn from this work:

1. Software was developed that can label sentiments expressed about specific aspects of a product. This software is both usable and useful. (Section 7.5)

2. The software is better at classifying the sentiments expressed about known attributes in laptop reviews than any of the 31 teams who performed the same task in a recent international NLP challenge (SemEval-2014 task 4). (Table 7.5)

3. The software developed is characterized by its very high precision and somewhat weak (or, in some cases, very weak) recall. (Chapter 7)

4. The software can be trained with only labelled data (supervised learning), or can be trained with fewer labelled data and a set of unlabelled data (co-training). Unlabelled data are more readily available and much cheaper to procure or produce. (Chapter 7)

5. When using co-training to perform this aspect-specific sentiment analysis, precision remains high or improves very slightly, at the expense of some recall. (Chapter 7)

6. This appears to be the first application of co-training to aspect-based sentiment analysis. (Chapter 3)

7. The algorithm presented is directly tuneable, albeit to a small extent, for greater precision or greater recall. (Section 7.4)

8. The system developed is able to handle new language that it has never seen before (a criticism of some simpler NLP systems). (Section 6.5)

9. The algorithm implemented differs from the commonly accepted co-training algorithm of Blum and Mitchell (1998), offering better scalability and taking advantage of the ability of newer machine learning classifiers to estimate the confidence in their own predictions. The tuneable parameters of the algorithm herein are more intuitive than those in (Blum and Mitchell, 1998). (Section 6.2)

10. My system has some demonstrated ability to adapt to slightly different product domains where the language of the product aspects and the words used to express sentiment can be quite different. (Section 7.2.2, Domain Adaptibility)

11. An NLP system with very high precision and somewhat weak recall (as mine can be characterized) can be used on its own or perhaps to greater advantage in ensemble learning, where when it votes, it will tend to vote correctly most of the time, staying quiet otherwise. (Section 7.5)

12. The system handles casually-written text that includes misspellings, incorrect word usage, and other quality phenomena that are prevalent in informal text written by the public at large. (Chapter 5)

13. The task of identifying aspect-specific sentiments is both challenging and current; it is an area of current interest and research, not a solved problem. (Chapter 7)

8.2 Summary of contributions

The contributions of this dissertation and the work it describes are:

- The first use of co-training for aspect-specific sentiment analysis (simultaneously training or classifying both the aspects of the product/service and the sentiment expressions)
- The natural language processing application of co-training using lexical information as one co-training view and syntactic information as another co-training view
- A machine learning approach that is demonstrated to be able to correctly handle words that have never been seen before; many tools in the sentiment analysis domain cannot be easily ported to new domains, whereas mine has a demonstrated ability to do so because of its ability to handle new language

- Particularly high precision product aspect tagging, achieving higher precision on both data sets in the SemEval-2014 task 4 subtask 1 than all 31 teams who participated therein
- High accuracy predicting the semantic orientation of opinions expressed about known aspects in the SemEval-2014 task 4 subtask 2 data; besting all teams who participated in the challenge on one of the two data sets
- A first usable result in a sentence-level task on the SemEval-2014 task 4 data, where entire unlabelled sentences are analyzed for both product aspects and the sentiments expressed about them simultaneously, a real-world task extending the artificially constrained tasks in the SemEval-2014 competition

8.3 Future work

There are several ways that the work herein could be improved or extended.

Extensions

The co-training algorithm developed in the course of this dissertation could be applied to other tasks, both within the natural processing domain and outside of it. (By comparison, Blum and Mitchell's co-training algorithm has found diverse applications as seen in Section 3.3).

It could be interesting to incorporate work on opinion strength. At present, I lump together all positive and all negative opinions, whereas in natural language, opinions are more nuanced. If a consumer is using comparative ratings of an aspect-specific sentiment classification system to make informed choices, it is probably advantageous that the strength of the opinions be known and aggregated (e.g., a cell phone with many weakly negative opinions about the battery life might be preferable to one with a similar number of very strong negative opinions about its battery life). There is some existing academic work on strength-based sentiment classification, e.g., (Wilson *et al.*, 2004) and (Turney and Littman, 2003), so that would seem a natural pairing.

One necessary compromise in trying to learn only aspect-specific sentiments herein was a willful and purposeful ignorance of sentiments expressed about the products (atomically) or the products' brands. A step forward might be incorporating classifiers designed to label such expressions at the same time as labelling aspect-specific sentiment terms; the sentiment-bearing word classifier could likely be used as-is. The architecture of the system developed

herein can be extended to any *n* lexically mutually exclusive classes; this could include named entities, competing brands, or retailers. With additional learning models for products (and synonyms thereof) and brands (perhaps by using a named entity tagger), a better picture of both the broader and more specific opinions expressed in text might be gleaned, for a better understanding of the messages conveyed by the creators of the text.

If this aspect-specific opinion classifier were used in a production environment, perhaps to generate "report cards" on competing products, it would then be useful to incorporate opinion spam detection, as in (Jindal and Liu, 2008), for example.

Improvements

There are a few areas where the work herein could be improved.

The heuristic reconciling product aspects and the sentiments expressed about them is relatively rudimentary (though effective). It makes rather strong simplifying assumptions about the dependency structure of opinions expressed in natural language; a more appropriate approach might be to use machine learning to learn from data the dependency structure between sentiment words and the aspects they modify. (Simple learners based on token distance have been tried and, according to the literature, do not work very well.)

It seems reasonable to expect that the feature selection could be improved. Perhaps the addition of more precise semantic role labels could help; or perhaps some graph-based or label propagation methods might provide a boost. Specifically, the weak recall of the system could stem from the small data sets used to train the system; or, perhaps it could be improved by further such features.

Incorporating advances in more accurate and precise sentence parsing could help. The Stanford CoreNLP folks issue compelling updates at an impressive rate, and as dependency parsing improves, so too should my system. Alternatively, other parsers could be tried.

While the support vector machine classifier selected was a compelling choice, other classifiers could be used. LibLinear could possibly offer comparable classification performance to LibSVM but should be able to train models in much less time; with faster training, it might be possible to perform co-training iterations wherein a much smaller number of new data are added at a time, which is not reasonably feasible with LibSVM on any non-trivial data set. Similarly, Bayesian classifiers have a different set of strengths than support vector machines; they are generally better at classifying negative examples, whereas SVM classifiers shine when classifying positive examples (Sokolova *et al.*, 2006). Perhaps using a different core classifier could provide an improvement.

Perhaps a better sentiment-bearing word lexicon could be used. At least one top contender in the SemEval-2014 task 4 challenge cited their sentiment lexicon as being key to their strong performance, whereas I simply took an off-the-shelf hand-made lexicon (Section 5.5.2) whose contents seemed reasonably explainable. Perhaps using more complex and/or larger sentiment lexicons could improve recall.

Finally, some semi-supervised algorithms (e.g., that in (Yarowsky, 1995)) run a prediction on all *training* data at each iteration to see if, for example, a borderline example that was added in a previous iteration should now be rejected from the training data because it now falls below a particular threshold due to the new knowledge gained by the classifier in the mean time (termed "escaping from initial misclassifications" in the Yarowsky paper). That could be a compelling addition to my approach.

Appendix A

The software

The software developed in the course of this dissertation is made available for experimentation and for repeatability.

The software is written in Java[1], and is made available at:

https://github.com/davecart/cotraining

An executable .jar file (carter-thesis-executable-jar.zip) can be run from the command line. Arguments can be listed by running the command:

```
java -jar processreviews-sept2014.jar help
```

The original data sets[2], which need to be downloaded in order to run the software, are available at:

- Amazon reviews of five products (Hu and Liu, 2004)
  http://www.cs.uic.edu/~liub/FBS/CustomerReviewData.zip
- Reviews of restaurants and laptops used in SemEval-2014 task 4 (Pontiki *et al.*, 2014)
  http://metashare.ilsp.gr:8080/repository/browse/semeval-2014-absa-test-data-gold-annotations/b98d11cec18211e38229842b2b6a04d77591d40acd7542b7af823a54fb03a155/

[1] Java 6 is required. Java 7 or higher is recommended.

[2] These data sets are the intellectual property of their respective creators and are not a contribution of this dissertation.

References

Ahmed Abbasi, Hsinchun Chen, and Arab Salem (2008), Sentiment Analysis in Multiple Languages: Feature Selection for Opinion Classification in Web Forums, *ACM Trans. Inf. Syst.*, 26(3):12:1–12:34, ISSN 1046-8188, doi:10.1145/1361684.1361685, URL http://doi.acm.org/10.1145/1361684.1361685.

Steven Abney (2002), Bootstrapping, in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pp. 360–367, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1073083.1073143, URL http://dx.doi.org/10.3115/1073083.1073143.

Douglas Adams (2002), *The Salmon of Doubt: Hitchhiking the Universe One Last Time*, Random House LLC.

Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau (2011), Sentiment analysis of Twitter data, in *Proceedings of the Workshop on Languages in Social Media*, pp. 30–38, Association for Computational Linguistics.

Massih-Reza Amini and Cyril Goutte (2010), A co-classification approach to learning from multilingual corpora, *Machine Learning*, 79(1-2):105–121, ISSN 0885-6125, doi:10.1007/s10994-009-5151-5, URL http://dx.doi.org/10.1007/s10994-009-5151-5.

Nikolay Archak, Anindya Ghose, and Panagiotis G. Ipeirotis (2007), Show Me the Money!: Deriving the Pricing Power of Product Features by Mining Consumer Reviews, in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pp. 56–65, ACM, New York, NY, USA, ISBN 978-1-59593-609-7, doi:10.1145/1281192.1281202, URL http://doi.acm.org/10.1145/1281192.1281202.

Maria-Florina Balcan, Avrim Blum, and Ke Yang (2004), Co-training and expansion: Towards bridging theory and practice, in *Advances in neural information processing systems*, pp. 89–96.

Roberto Basili and Alessandro Moschitti (2002), Intellitent NLP-Driven Text Classification, *International Journal on Artificial Intelligence Tools*, 11(03):389–423, doi: 10.1142/S0218213002000952, URL http://www.worldscientific.com/doi/abs/10.1142/S0218213002000952.

Asa Ben-Hur and Jason Weston (2010), A UserÕs Guide to Support Vector Machines, in *Data mining techniques for the life sciences*, pp. 223–239, Springer.

Emily M Bender (2013), Linguistic Fundamentals for Natural Language Processing: 100 Essentials from Morphology and Syntax, *Synthesis Lectures on Human Language Technologies*, 6(3):1–184.

Prakhar Biyani, Cornelia Caragea, Prasenjit Mitra, Chong Zhou, John Yen, Greta E. Greer, and Kenneth Portier (2013), Co-training over Domain-independent and Domain-dependent Features for Sentiment Analysis of an Online Cancer Support Community, in *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '13, pp. 413–417, ACM, New York, NY, USA, ISBN 978-1-4503-2240-9, doi:10.1145/2492517.2492606, URL http://doi.acm.org/10.1145/2492517.2492606.

John Blitzer, Mark Dredze, and Fernando Pereira (2007), Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification, in *ACL*, volume 7, pp. 440–447.

Avrim Blum and Tom Mitchell (1998), Combining Labeled and Unlabeled Data with Co-training, in *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, COLT' 98, pp. 92–100, ACM, New York, NY, USA, ISBN 1-58113-057-0, doi:10.1145/279943.279962, URL http://doi.acm.org/10.1145/279943.279962.

Sergey Brin (1999), Extracting patterns and relations from the world wide web, in *The World Wide Web and Databases*, pp. 172–183, Springer.

Samuel Brody and Noemie Elhadad (2010), An Unsupervised Aspect-sentiment Model for Online Reviews, in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pp. 804–812, Association for Computational Linguistics, Stroudsburg, PA, USA, ISBN 1-932432-65-5, URL http://dl.acm.org/citation.cfm?id=1857999.1858121.

Pedro Henrique Calais Guerra, Adriano Veloso, Wagner Meira Jr, and Virgílio Almeida (2011), From bias to opinion: a transfer-learning approach to real-time sentiment analysis, in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 150–158, ACM.

Jean Carletta (1996), Assessing Agreement on Classification Tasks: The Kappa Statistic, *Computational Linguistics*, 22(2):249–254, ISSN 0891-2017, URL http://dl.acm.org/citation.cfm?id=230386.230390.

Bruno Cartoni and Marie-Aude Lefer (2011), Negation and lexical morphology across languages: insights from a trilingual translation corpus, *Poznań Studies in Contemporary Linguistics PSiCL*, 47:795.

Rich Caruana and Alexandru Niculescu-Mizil (2006), An Empirical Comparison of Supervised Learning Algorithms, in *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pp. 161–168, ACM, New York, NY, USA, ISBN 1-59593-383-2, doi:10.1145/1143844.1143865, URL http://doi.acm.org/10.1145/1143844.1143865.

Chih-Chung Chang and Chih-Jen Lin (2011), LIBSVM: A library for support vector machines, *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Vladimir Cherkassky and Yunqian Ma (2004), Practical selection of SVM parameters and noise estimation for SVM regression, *Neural networks*, 17(1):113–126.

Yejin Choi, Eric Breck, and Claire Cardie (2006), 2006. Joint extraction of entities and relations for opinion recognition, in *In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 431–439.

Yejin Choi and Claire Cardie (2005), Identifying Sources of Opinions with Conditional Random Fields and Extraction Patterns, in *In HLT/EMNLP 2005*.

Jessica Elan Chung and Eni Mustafaraj (2011), Can collective sentiment expressed on Twitter predict political elections?, in *AAAI*.

Stephen Clark, James R. Curran, and Miles Osborne (2003), Bootstrapping POS Taggers Using Unlabelled Data, in *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pp. 49–55, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1119176.1119183, URL http://dx.doi.org/10.3115/1119176.1119183.

Michael Collins and Yoram Singer (1999), Unsupervised models for named entity classification, in *Proceedings of the joint SIGDAT conference on empirical methods in natural language processing and very large corpora*, pp. 100–110.

Corinna Cortes and Vladimir Vapnik (1995), Support-Vector Networks, *Mach. Learn.*, 20(3):273–297, ISSN 0885-6125, doi:10.1023/A:1022627411411, URL http://dx.doi.org/10.1023/A:1022627411411.

Mark Craven, Dan DiPasquo, Dayne Freitag, Andrew McCallum, Tom Mitchell, Kamal Nigam, and Seán Slattery (1998), Learning to Extract Symbolic Knowledge from the World Wide Web, in *Proceedings of the Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence*, AAAI '98/IAAI '98, pp. 509–516, American Association for Artificial Intelligence, Menlo Park, CA, USA, ISBN 0-262-51098-7, URL http://dl.acm.org/citation.cfm?id=295240.295725.

Hang Cui, Vibhu Mittal, and Mayur Datar (2006), Comparative Experiments on Sentiment Classification for Online Product Reviews, in *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2*, AAAI'06, pp. 1265–1270, AAAI Press, ISBN 978-1-57735-281-5, URL http://dl.acm.org/citation.cfm?id=1597348.1597389.

Sanjiv R. Das, Mike Y. Chen, To Vikas Agarwal, Chris Brooks, Yuk shee Chan, David Gibson, David Leinweber, Asis Martinez-jerez, Priya Raghubir, Sridhar Rajagopalan, Ajit Ranade, Mark Rubinstein, and Peter Tufano (2001), Yahoo! for amazon: Sentiment extraction from small talk on the web, in *8th Asia Pacific Finance Association Annual Conference*.

Sanjoy Dasgupta, Michael L Littman, and David McAllester (2002), PAC generalization bounds for co-training, *Advances in neural information processing systems*, 1:375–382.

Kushal Dave, Steve Lawrence, and David M. Pennock (2003), Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews, in *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, pp. 519–528, ACM, New York, NY, USA, ISBN 1-58113-680-3, doi:10.1145/775152.775226, URL http://doi.acm.org/10.1145/775152.775226.

Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, *et al.* (2006), Generating typed dependency parses from phrase structure parses, in *Proceedings of LREC*, volume 6, pp. 449–454.

Xiaowen Ding and Bing Liu (2007), The Utility of Linguistic Rules in Opinion Mining, in *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, pp. 811–812, ACM, New York, NY, USA, ISBN 978-1-59593-597-7, doi:10.1145/1277741.1277921, URL http://doi.acm.org/10.1145/1277741.1277921.

Xiaowen Ding, Bing Liu, and Philip S. Yu (2008), A Holistic Lexicon-based Approach to Opinion Mining, in *Proceedings of the 2008 International Conference on Web Search and Data Mining*, WSDM '08, pp. 231–240, ACM, New York, NY, USA, ISBN 978-1-59593-927-2, doi:10.1145/1341531.1341561, URL http://doi.acm.org/10.1145/1341531.1341561.

Jun Du, Charles X. Ling, and Zhi-Hua Zhou (2011), When Does Cotraining Work in Real Data?, *IEEE Trans. on Knowl. and Data Eng.*, 23(5):788–799, ISSN 1041-4347, doi:10.1109/TKDE.2010.158, URL http://dx.doi.org/10.1109/TKDE.2010.158.

Susan Dumais, John Platt, David Heckerman, and Mehran Sahami (1998), Inductive learning algorithms and representations for text categorization, in *Proc. CIKM*, pp. 148–155, ACM Press, New York, NY, USA, ISBN 1-58113-061-9, doi:doi.acm.org/10.1145/288627.288651.

Jacob Eisenstein (2013), What to do about bad language on the internet, in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 359–369, Association for Computational Linguistics, URL http://aclweb.org/anthology/N13-1037.

Rong-En Fan, Pai-Hsuen Chen, and Chih-Jen Lin (2005), Working Set Selection Using Second Order Information for Training Support Vector Machines, *J. Mach. Learn. Res.*, 6:1889–1918, ISSN 1532-4435, URL http://www.csie.ntu.edu.tw/~cjlin/papers/quadworkset.pdf.

Christiane Fellbaum, editor (1998), *WordNet: an electronic lexical database*, MIT Press.

David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, *et al.* (2010), Building Watson: An overview of the DeepQA project, *AI magazine*, 31(3):59–79.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning (2005), Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling, in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL

'05, pp. 363–370, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1219840.1219885, URL http://dx.doi.org/10.3115/1219840.1219885.

Michael Gamon (2004), Sentiment Classification on Customer Feedback Data: Noisy Data, Large Feature Vectors, and the Role of Linguistic Analysis, in *Proceedings of the 20th International Conference on Computational Linguistics*, COLING '04, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1220355.1220476, URL http://dx.doi.org/10.3115/1220355.1220476.

Michael Gamon, Anthony Aue, Simon Corston-oliver, and Eric Ringger (2005), Pulse: Mining Customer Opinions from Free Text, in *In Proc. of the 6th International Symposium on Intelligent Data Analysis*, pp. 121–132.

Gayatree Ganu, Noemie Elhadad, and Amélie Marian (2009), Beyond the Stars: Improving Rating Predictions using Review Text Content, in *Proceedings of the 12th International Workshop on the Web and Databases (WebDB 2009)*, URL http://people.dbmi.columbia.edu/noemie/papers/webdb09.pdf.

Daniel Gayo-Avello (2013), A Meta-Analysis of State-of-the-Art Electoral Prediction From Twitter Data, *Soc. Sci. Comput. Rev.*, 31(6):649–679, ISSN 0894-4393, doi:10.1177/0894439313493979, URL http://dx.doi.org/10.1177/0894439313493979.

Daniel Gayo-Avello, Panagiotis Takis Metaxas, and Eni Mustafaraj (2011), Limits of electoral predictions using Twitter, in *ICWSM*.

Diman Ghazi, Diana Inkpen, and Stan Szpakowicz (2014), Prior and Contextual Emotion of Words in Sentential Context, *Comput. Speech Lang.*, 28(1):76–92, ISSN 0885-2308, doi:10.1016/j.csl.2013.04.009, URL http://dx.doi.org/10.1016/j.csl.2013.04.009.

M. Ghiassi, J. Skinner, and D. Zimbra (2013), Twitter Brand Sentiment Analysis: A Hybrid System Using N-gram Analysis and Dynamic Artificial Neural Network, *Expert Syst. Appl.*, 40(16):6266–6282, ISSN 0957-4174, doi:10.1016/j.eswa.2013.05.057, URL http://dx.doi.org/10.1016/j.eswa.2013.05.057.

Anindya Ghose and Panagiotis G. Ipeirotis (2007), Designing Novel Review Ranking Systems: Predicting the Usefulness and Impact of Reviews, in *Proceedings of the Ninth International Conference on Electronic Commerce*, ICEC '07, pp. 303–310, ACM, New York, NY, USA, ISBN 978-1-59593-700-1, doi:10.1145/1282100.1282158, URL http://doi.acm.org/10.1145/1282100.1282158.

Anindya Ghose and Panagiotis G Ipeirotis (2011), Estimating the helpfulness and economic impact of product reviews: Mining text and reviewer characteristics, *Knowledge and Data Engineering, IEEE Transactions on*, 23(10):1498–1512.

Natalie Glance, Matthew Hurst, Kamal Nigam, Matthew Siegler, Robert Stockton, and Takashi Tomokiyo (2005), Deriving Marketing Intelligence from Online Discussion, in *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, KDD '05, pp. 419–428, ACM, New York, NY, USA, ISBN 1-59593-135-X, doi: 10.1145/1081870.1081919, URL http://doi.acm.org/10.1145/1081870.1081919.

Alec Go, Richa Bhayani, and Lei Huang (2009), Twitter sentiment classification using distant supervision, *CS224N Project Report, Stanford*, pp. 1–12.

Andrew B. Goldberg and Xiaojin Zhu (2006), 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*, TextGraphs-1, pp. 45–52, Association for Computational Linguistics, Stroudsburg, PA, USA, URL http://dl.acm.org/citation.cfm?id=1654758.1654769.

Sally Goldman and Yan Zhou (2000), Enhancing supervised learning with unlabeled data, in *Proceedings of the 17Th International Conference on Machine Learning*, pp. 327–334, Morgan Kaufmann.

Zellig S Harris (1954), Distributional structure, *Word*, 10:146–162.

Vasileios Hatzivassiloglou and Kathleen R. McKeown (1997), Predicting the Semantic Orientation of Adjectives, in *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, ACL '98, pp. 174–181, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/976909.979640, URL http://dx.doi.org/10.3115/976909.979640.

Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, *et al.* (2003), A practical guide to support vector classification, URL http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf.

Minqing Hu and Bing Liu (2004), Mining and Summarizing Customer Reviews, in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pp. 168–177, ACM, New York, NY, USA, ISBN 1-58113-888-1, doi: 10.1145/1014052.1014073, URL http://doi.acm.org/10.1145/1014052.1014073.

Jin Huang, Jelber Sayyad-Shirabad, Stan Matwin, and Jiang Su (2012), Improving multi-view semi-supervised learning with agreement-based sampling, *Intell. Data Anal.*, pp. 745–761.

Wei Jin, Hung Hay Ho, and Rohini K. Srihari (2009), 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*, KDD '09, pp. 1195–1204, ACM, New York, NY, USA, ISBN 978-1-60558-495-9, doi:10.1145/1557019.1557148, URL http://doi.acm.org/10.1145/1557019.1557148.

Nitin Jindal and Bing Liu (2006), 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, pp. 244–251, ACM, New York, NY, USA, ISBN 1-59593-369-7, doi:10.1145/1148170.1148215, URL http://doi.acm.org/10.1145/1148170.1148215.

Nitin Jindal and Bing Liu (2007), Analyzing and detecting review spam, in *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pp. 547–552, IEEE.

Nitin Jindal and Bing Liu (2008), Opinion Spam and Analysis, in *Proceedings of the 2008 International Conference on Web Search and Data Mining*, WSDM '08, pp. 219–230, ACM, New York, NY, USA, ISBN 978-1-59593-927-2, doi:10.1145/1341531.1341560, URL http://doi.acm.org/10.1145/1341531.1341560.

Yohan Jo and Alice H. Oh (2011), Aspect and Sentiment Unification Model for Online Review Analysis, in *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, WSDM '11, pp. 815–824, ACM, New York, NY, USA, ISBN 978-1-4503-0493-1, doi:10.1145/1935826.1935932, URL http://doi.acm.org/10.1145/1935826.1935932.

Thorsten Joachims (1998), Text categorization with support vector machines: Learning with many relevant features, in *Proc. ECML*, pp. 137–142, Springer, Heidelberg.

Thorsten Joachims (2002), *Learning to Classify Text Using Support Vector Machines: Methods, Theory and Algorithms*, Kluwer Academic Publishers, Norwell, MA, USA, ISBN 079237679X.

Mahesh Joshi and Carolyn Penstein-Rosé (2009), Generalizing Dependency Features for Opinion Mining, in *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, ACLShort '09, pp. 313–316, Association for Computational Linguistics, Stroudsburg, PA, USA, URL http://dl.acm.org/citation.cfm?id=1667583.1667680.

Daniel Jurafsky and James H. Martin (2000), *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition, ISBN 0130950696.

Alistair Kennedy and Diana Inkpen (2006), Sentiment Classification of Movie Reviews Using Contextual Valence Shifters, *Computational Intelligence*, 22:2006.

Soo-min Kim and Eduard Hovy (2004), Determining the sentiment of opinions, in *In Proceedings of the 20th International Conference on Computational Linguistics (COLING?04*, pp. 1367–1373.

Soo-Min Kim and Eduard Hovy (2006a), Automatic Identification of Pro and Con Reasons in Online Reviews, in *Proceedings of the COLING/ACL on Main Conference Poster Sessions*, COLING-ACL '06, pp. 483–490, Association for Computational Linguistics, Stroudsburg, PA, USA, URL http://dl.acm.org/citation.cfm?id=1273073.1273136.

Soo-Min Kim and Eduard Hovy (2006b), 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, pp. 1–8, Association for Computational Linguistics, Stroudsburg, PA, USA, ISBN 1-932432-75-2, URL http://dl.acm.org/citation.cfm?id=1654641.1654642.

Soo-Min Kim and Eduard Hovy (2006c), Identifying and Analyzing Judgment Opinions, in *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pp. 200–207, Association for Computational Linguistics, Stroudsburg, PA, USA, doi: 10.3115/1220835.1220861, URL http://dx.doi.org/10.3115/1220835.1220861.

Soo-min Kim, Patrick Pantel, Tim Chklovski, and Marco Pennacchiotti (2006), Automatically assessing review helpfulness, in *In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP*, pp. 423–430.

Svetlana Kiritchenko and Stan Matwin (2001), Email Classification with Co-training, in *Proceedings of the 2001 Conference of the Centre for Advanced Studies on Collaborative Research*, CASCON '01, pp. 8–, IBM Press, URL http://dl.acm.org/citation.cfm?id=782096.782104.

Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif M Mohammad (2014), NRC-Canada-2014: Detecting Aspects and Sentiment in Customer Reviews, in *Proceedings of the International Workshop on Semantic Evaluation, SemEval*, volume 14.

Dan Klein and Christopher D. Manning (2003), Accurate Unlexicalized Parsing, in *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pp. 423–430, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1075096.1075150, URL http://dx.doi.org/10.3115/1075096.1075150.

Lun-Wei Ku, Yu-Ting Liang, and Hsin-Hsi Chen (2006), Opinion Extraction, Summarization and Tracking in News and Blog Corpora, in *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, volume 100107.

Angeliki Lazaridou, Ivan Titov, and Caroline Sporleder (2013), A Bayesian Model for Joint Unsupervised Induction of Sentiment, Aspect and Discourse Representations, in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pp. 1630–1639.

Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky (2013), Deterministic Coreference Resolution Based on Entity-centric, Precision-ranked Rules, *Computational Linguistics*, 39(4):885–916, ISSN 0891-2017, doi:10.1162/COLI_a_00152, URL http://dx.doi.org/10.1162/COLI_a_00152.

Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky (2011), Stanford's Multi-pass Sieve Coreference Resolution System at the CoNLL-2011 Shared Task, in *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, CONLL Shared Task '11, pp. 28–34, Association for Computational Linguistics, Stroudsburg, PA, USA, ISBN 9781937284084, URL http://dl.acm.org/citation.cfm?id=2132936.2132938.

David D. Lewis (1998), Naive (Bayes) at forty: The independence assumption in information retrieval, in Claire Nédellec and Céline Rouveirol, editors, *Machine Learning: ECML-98*, volume 1398 of *Lecture Notes in Computer Science*, pp. 4–15, Springer Berlin Heidelberg, ISBN 978-3-540-64417-0, doi:10.1007/BFb0026666, URL http://dx.doi.org/10.1007/BFb0026666.

Fan Li and Yiming Yang (2003), A Loss Function Analysis for Classification Methods in Text Categorization, in *Proc. ICML*, pp. 472–479.

Fangtao Li, Chao Han, Minlie Huang, Xiaoyan Zhu, Ying-Ju Xia, Shu Zhang, and Hao Yu (2010a), Structure-aware Review Mining and Summarization, in *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pp. 653–661, Association

for Computational Linguistics, Stroudsburg, PA, USA, URL http://dl.acm.org/citation.cfm?id=1873781.1873855.

Shoushan Li, Chu-Ren Huang, Guodong Zhou, and Sophia Yat Mei Lee (2010b), Employing Personal/Impersonal Views in Supervised and Semi-supervised Sentiment Classification, in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pp. 414–423, Association for Computational Linguistics, Stroudsburg, PA, USA, URL http://dl.acm.org/citation.cfm?id=1858681.1858724.

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

Bing Liu (2012), Sentiment analysis and opinion mining, *Synthesis Lectures on Human Language Technologies*, 5(1):1–167.

Bing Liu, Minqing Hu, and Junsheng Cheng (2005), Opinion Observer: Analyzing and Comparing Opinions on the Web, in *Proceedings of the 14th International Conference on World Wide Web*, WWW '05, pp. 342–351, ACM, New York, NY, USA, ISBN 1-59593-046-9, doi:10.1145/1060745.1060797, URL http://doi.acm.org/10.1145/1060745.1060797.

Bing Liu and Lei Zhang (2012), A survey of opinion mining and sentiment analysis, in *Mining Text Data*, pp. 415–463, Springer.

Jian Liu, Gengfeng Wu, and Jianxin Yao (2006), Opinion Searching in Multi-Product Reviews, in *Computer and Information Technology, 2006. CIT '06. The Sixth IEEE International Conference on*, pp. 25–25, doi:10.1109/CIT.2006.132.

Shenghua Liu, Fuxin Li, Fangtao Li, Xueqi Cheng, and Huawei Shen (2013a), Adaptive Co-training SVM for Sentiment Classification on Tweets, in *Proceedings of the 22Nd ACM International Conference on Conference on Information & Knowledge Management*, CIKM '13, pp. 2079–2088, ACM, New York, NY, USA, ISBN 978-1-4503-2263-8, doi:10.1145/2505515.2505569, URL http://doi.acm.org/10.1145/2505515.2505569.

Shenghua Liu, Wenjun Zhu, Ning Xu, Fangtao Li, Xue-qi Cheng, Yue Liu, and Yuanzhuo Wang (2013b), Co-training and Visualizing Sentiment Evolvement for Tweet Events, in *Proceedings of the 22Nd International Conference on World Wide Web Companion*, WWW '13 Companion, pp. 105–106, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, ISBN 978-1-4503-2038-2, URL http://dl.acm.org/citation.cfm?id=2487788.2487836.

Duy Khang Ly, Kazunari Sugiyama, Ziheng Lin, and Min-Yen Kan (2011), Product Review Summarization from a Deeper Perspective, in *Proceedings of the 11th Annual International ACM/IEEE Joint Conference on Digital Libraries*, JCDL '11, pp. 311–314, ACM, New York, NY, USA, ISBN 978-1-4503-0744-4, doi:10.1145/1998076.1998134, URL http://doi.acm.org/10.1145/1998076.1998134.

Christopher D. Manning (2011), Part-of-speech Tagging from 97% to 100%: Is It Time for Some Linguistics?, in *Proceedings of the 12th International Conference on Computational Linguistics and Intelligent Text Processing - Volume Part I*, CICLing'11, pp. 171–189, Springer-Verlag, Berlin, Heidelberg, ISBN 978-3-642-19399-6, URL http://dl.acm.org/citation.cfm?id=1964799.1964816.

Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze (2008), *Introduction to information retrieval*, volume 1, Cambridge university press Cambridge.

Christopher D. Manning and Hinrich Schütze (1999), *Foundations of Statistical Natural Language Processing*, MIT Press, Cambridge, MA, USA, ISBN 0-262-13360-1.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky (2014), The Stanford CoreNLP Natural Language Processing Toolkit, in *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 55–60, Association for Computational Linguistics, Baltimore, Maryland, URL http://www.aclweb.org/anthology/P/P14/P14-5010.

Justin Martineau, Tim Finin, Anupam Joshi, and Shamit Patel (2009), Improving Binary Classification on Text Problems Using Differential Word Features, in *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, pp. 2019–2024, ACM, New York, NY, USA, ISBN 978-1-60558-512-3, doi:10.1145/1645953.1646291, URL http://doi.acm.org/10.1145/1645953.1646291.

Shotaro Matsumoto, Hiroya Takamura, and Manabu Okumura (2005), Sentiment Classification Using Word Sub-sequences and Dependency Sub-trees, in *Proceedings of the 9th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, PAKDD'05, pp. 301–311, Springer-Verlag, Berlin, Heidelberg, ISBN 3-540-26076-5, 978-3-540-26076-9, doi:10.1007/11430919_37, URL http://dx.doi.org/10.1007/11430919_37.

Julian McAuley and Jure Leskovec (2013), Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text, in *Proceedings of the 7th ACM Conference on Rec-*

ommender Systems, RecSys '13, pp. 165–172, ACM, New York, NY, USA, ISBN 978-1-4503-2409-0, doi:10.1145/2507157.2507163, URL http://doi.acm.org/10.1145/2507157.2507163.

Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and ChengXiang Zhai (2007), Topic sentiment mixture: modeling facets and opinions in weblogs, in *Proceedings of the 16th international conference on World Wide Web*, pp. 171–180, ACM.

Panagiotis Takis Metaxas, Eni Mustafaraj, and Daniel Gayo-Avello (2011), How (not) to predict elections, in *Privacy, security, risk and trust (PASSAT), 2011 IEEE third international conference on and 2011 IEEE third international conference on social computing (SocialCom)*, pp. 165–171, IEEE.

Rada Mihalcea (2004), Co-training and self-training for word sense disambiguation, in *Proceedings of the Conference on Computational Natural Language Learning (CoNLL-2004)*.

George A. Miller (1995), WordNet: A Lexical Database for English, *Commun. ACM*, 38(11):39–41, ISSN 0001-0782, doi:10.1145/219717.219748, URL http://doi.acm.org/10.1145/219717.219748.

Saif Mohammad (2012), Portable Features for Classifying Emotional Text, in *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL HLT '12, pp. 587–591, Association for Computational Linguistics, Stroudsburg, PA, USA, ISBN 978-1-937284-20-6, URL http://dl.acm.org/citation.cfm?id=2382029.2382123.

Saif M Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu (2013), NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets, *arXiv preprint arXiv:1308.6242*.

Satoshi Morinaga, Kenji Yamanishi, Kenji Tateishi, and Toshikazu Fukushima (2002), Mining Product Reputations on the Web, in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pp. 341–349, ACM, New York, NY, USA, ISBN 1-58113-567-X, doi:10.1145/775047.775098, URL http://doi.acm.org/10.1145/775047.775098.

Mohamed M. Mostafa (2013), More Than Words: Social Networks' Text Mining for Consumer Brand Sentiments, *Expert Syst. Appl.*, 40(10):4241–4251, ISSN 0957-4174, doi:10.1016/j.eswa.2013.01.019, URL http://dx.doi.org/10.1016/j.eswa.2013.01.019.

Tetsuya Nasukawa and Jeonghee Yi (2003), Sentiment Analysis: Capturing Favorability Using Natural Language Processing, in *Proceedings of the 2Nd International Conference on Knowledge Capture*, K-CAP '03, pp. 70–77, ACM, New York, NY, USA, ISBN 1-58113-583-1, doi:10.1145/945645.945658, URL http://doi.acm.org/10.1145/945645.945658.

Vincent Ng and Claire Cardie (2003a), Bootstrapping Coreference Classifiers with Multiple Machine Learning Algorithms, in *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, EMNLP '03, pp. 113–120, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1119355.1119370, URL http://dx.doi.org/10.3115/1119355.1119370.

Vincent Ng and Claire Cardie (2003b), Weakly Supervised Natural Language Learning Without Redundant Views, in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pp. 94–101, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1073445.1073468, URL http://dx.doi.org/10.3115/1073445.1073468.

Vincent Ng, Sajib Dasgupta, and S. M. Niaz Arifin (2006), Examining the Role of Linguistic Knowledge Sources in the Automatic Identification and Classification of Reviews, in *Proceedings of the COLING/ACL on Main Conference Poster Sessions*, COLING-ACL '06, pp. 611–618, Association for Computational Linguistics, Stroudsburg, PA, USA, URL http://dl.acm.org/citation.cfm?id=1273073.1273152.

Kamal Nigam and Rayid Ghani (2000), Analyzing the Effectiveness and Applicability of Co-training, in *Proceedings of the Ninth International Conference on Information and Knowledge Management*, CIKM '00, pp. 86–93, ACM, New York, NY, USA, ISBN 1-58113-320-0, doi:10.1145/354756.354805, URL http://doi.acm.org/10.1145/354756.354805.

Kamal Nigam and Matthew Hurst (2004), Towards a robust metric of opinion, in *AAAI spring symposium on exploring attitude and affect in text*, pp. 598–603.

Brendan O'Connor, Ramnath Balasubramanyan, Bryan R Routledge, and Noah A Smith (2010), From tweets to polls: Linking text sentiment to public opinion time series, *ICWSM*, 11:122–129.

Alexander Pak and Patrick Paroubek (2010), Twitter as a Corpus for Sentiment Analysis and Opinion Mining, in *LREC*.

Bo Pang and Lillian Lee (2004), A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts, in *Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1218955.1218990, URL http://dx.doi.org/10.3115/1218955.1218990.

Bo Pang and Lillian Lee (2008), Opinion mining and sentiment analysis, *Foundations and trends in information retrieval*, 2(1-2):1–135.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan (2002), Thumbs up? Sentiment Classification using Machine Learning Techniques, in *IN PROCEEDINGS OF EMNLP*, pp. 79–86.

Michael Paul and Roxana Girju (2010), A two-dimensional topic-aspect model for discovering multi-faceted topics, *Urbana*, 51:61801.

Michael J. Paul, ChengXiang Zhai, and Roxana Girju (2010), Summarizing Contrastive Viewpoints in Opinionated Text, in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pp. 66–76, Association for Computational Linguistics, Stroudsburg, PA, USA, URL http://dl.acm.org/citation.cfm?id=1870658.1870665.

David Pierce and Claire Cardie (2001), Limitations of co-training for natural language learning from large datasets, in *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, pp. 1–9.

Ferran Pla and Lluis-F Hurtado (2014), Political Tendency Identification in Twitter using Sentiment Analysis Techniques, in *Proceedings of the 25th International Conference on Computational Linguistics COLING*, pp. 183–192, URL http://www.aclweb.org/anthology/C/C14/C14-1019.pdf.

Maria Pontiki, Dimitrios Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar (2014), SemEval-2014 Task 4: Aspect Based Sentiment Analysis, in *Proceedings of the International Workshop on Semantic Evaluation (SemEval)*.

Ana-Maria Popescu and Orena Etzioni (2007), Extracting product features and opinions from reviews, in *Natural language processing and text mining*, pp. 9–28, Springer.

Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen (2009), Expanding Domain Sentiment Lexicon Through Double Propagation, in *Proceedings of the 21st International Jont Conference*

on Artifical Intelligence, IJCAI'09, pp. 1199–1204, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, URL http://dl.acm.org/citation.cfm?id=1661445.1661637.

Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nathanael Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning (2010), A Multi-pass Sieve for Coreference Resolution, in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pp. 492–501, Association for Computational Linguistics, Stroudsburg, PA, USA, URL http://dl.acm.org/citation.cfm?id=1870658.1870706.

Stefan Riezler (2013), On the Problem of Theoretical Terms in Empirical Computational Linguistics, *Computational Linguistics*, pp. 235–245, ISSN 0891-2017, doi:10.1162/COLI_a_00182, URL http://dx.doi.org/10.1162/COLI_a_00182.

Ellen Riloff and Rosie Jones (1999), Learning Dictionaries for Information Extraction by Multi-level Bootstrapping, in *Proceedings of the Sixteenth National Conference on Artificial Intelligence and the Eleventh Innovative Applications of Artificial Intelligence Conference Innovative Applications of Artificial Intelligence*, AAAI '99/IAAI '99, pp. 474–479, American Association for Artificial Intelligence, Menlo Park, CA, USA, ISBN 0-262-51106-1, URL http://dl.acm.org/citation.cfm?id=315149.315364.

Ellen Riloff, Siddharth Patwardhan, and Janyce Wiebe (2006), Feature Subsumption for Opinion Analysis, in *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pp. 440–448, Association for Computational Linguistics, Stroudsburg, PA, USA, ISBN 1-932432-73-6, URL http://dl.acm.org/citation.cfm?id=1610075.1610137.

Ellen Riloff and Janyce Wiebe (2003), Learning extraction patterns for subjective expressions, in *Conference on Empirical Methods in Natural Language Processing (EMNLP-03*, pp. 105–112.

Ellen Riloff, Janyce Wiebe, and Theresa Wilson (2003), Learning Subjective Nouns Using Extraction Pattern Bootstrapping, in *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pp. 25–32, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1119176.1119180, URL http://dx.doi.org/10.3115/1119176.1119180.

Christian Rohrdantz, Ming C. Hao, Umeshwar Dayal, Lars-Erik Haug, and Daniel A. Keim (2012), Feature-Based Visual Sentiment Analysis of Text Document Streams, *ACM Trans.*

Intell. Syst. Technol., 3(2):26:1–26:25, ISSN 2157-6904, doi:10.1145/2089094.2089102, URL http://doi.acm.org/10.1145/2089094.2089102.

Stuart J. Russell and Peter Norvig (2003), *Artificial Intelligence: A Modern Approach*, Pearson Education, 2 edition, ISBN 0137903952.

Erik Tjong Kim Sang and Johan Bos (2012), Predicting the 2011 Dutch Senate Election Results with Twitter, in *Proceedings of the Workshop on Semantic Analysis in Social Media*, pp. 53–60, Association for Computational Linguistics, Stroudsburg, PA, USA, URL http://dl.acm.org/citation.cfm?id=2389969.2389976.

Anoop Sarkar (2001), Applying Co-training Methods to Statistical Parsing, in *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics on Language Technologies*, NAACL '01, pp. 1–8, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1073336.1073359, URL http://dx.doi.org/10.3115/1073336.1073359.

Christopher Scaffidi, Kevin Bierhoff, Eric Chang, Mikhael Felker, Herman Ng, and Chun Jin (2007), Red Opal: Product-feature Scoring from Reviews, in *Proceedings of the 8th ACM Conference on Electronic Commerce*, EC '07, pp. 182–191, ACM, New York, NY, USA, ISBN 978-1-59593-653-0, doi:10.1145/1250910.1250938, URL http://doi.acm.org/10.1145/1250910.1250938.

Marko Skoric, Nathaniel Poor, Palakorn Achananuparp, Ee-Peng Lim, and Jing Jiang (2012), Tweets and votes: A study of the 2011 singapore general election, in *System Science (HICSS), 2012 45th Hawaii International Conference on*, pp. 2583–2591, IEEE.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng (2013), Parsing With Compositional Vector Grammars, in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 455–465, Association for Computational Linguistics, URL http://aclweb.org/anthology/P13-1045.

M. Sokolova and G. Lapalme (2011), Learning Opinions in User-generated Web Content, *Nat. Lang. Eng.*, 17(4):541–567, ISSN 1351-3249, doi:10.1017/S135132491100012X, URL http://dx.doi.org/10.1017/S135132491100012X.

Marina Sokolova, Nathalie Japkowicz, and Stan Szpakowicz (2006), Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation, in *AI 2006: Advances in Artificial Intelligence*, pp. 1015–1021, Springer.

Marina Sokolova and Guy Lapalme (2009), A Systematic Analysis of Performance Measures for Classification Tasks, *Inf. Process. Manage.*, 45(4):427–437, ISSN 0306-4573, doi:10.1016/j.ipm.2009.03.002, URL http://dx.doi.org/10.1016/j.ipm.2009.03.002.

Qi Su, Kun Xiang, Houfeng Wang, Bin Sun, and Shiwen Yu (2006), Using pointwise mutual information to identify implicit features in customer reviews, in *Computer Processing of Oriental Languages. Beyond the Orient: The Research Challenges Ahead*, pp. 22–30, Springer.

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

Huifeng Tang, Songbo Tan, and Xueqi Cheng (2009), A survey on sentiment detection of reviews, *Expert Systems with Applications*, 36(7):10760–10773.

Mike Thelwall, Kevan Buckley, and Georgios Paltoglou (2011), Sentiment in Twitter events, *Journal of the American Society for Information Science and Technology*, 62(2):406–418.

Ivan Titov and Ryan McDonald (2008a), A Joint Model of Text and Aspect Ratings for Sentiment Summarization, in *PROC. ACL-08: HLT*, pp. 308–316.

Ivan Titov and Ryan McDonald (2008b), Modeling Online Reviews with Multi-grain Topic Models, in *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, pp. 111–120, ACM, New York, NY, USA, ISBN 978-1-60558-085-2, doi:10.1145/1367497.1367513, URL http://doi.acm.org/10.1145/1367497.1367513.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer (2003), Feature-rich Part-of-speech Tagging with a Cyclic Dependency Network, in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pp. 173–180, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1073445.1073478, URL http://dx.doi.org/10.3115/1073445.1073478.

Kristina Toutanova and Christopher D. Manning (2000), Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-speech Tagger, in *Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora: Held in Conjunction with the 38th Annual Meeting of the Association for Computational*

Linguistics - Volume 13, EMNLP '00, pp. 63–70, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1117794.1117802, URL http://dx.doi.org/10.3115/1117794.1117802.

Andranik Tumasjan, Timm Oliver Sprenger, Philipp G Sandner, and Isabell M Welpe (2010), Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment, *ICWSM*, 10:178–185.

Peter D. Turney (2002), 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*, ACL '02, pp. 417–424, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1073083.1073153, URL http://dx.doi.org/10.3115/1073083.1073153.

Peter D. Turney and Michael L. Littman (2003), Measuring praise and criticism: Inference of semantic orientation from association, *ACM Transactions on Information Systems*, 21:315–346.

Anthony J Viera, Joanne M Garrett, *et al.* (2005), Understanding interobserver agreement: the kappa statistic, *Fam Med*, 37(5):360–363.

Henning Wachsmuth, Martin Trenkmann, Benno Stein, and Gregor Engels (2014), Modeling Review Argumentation for Robust Sentiment Analysis, in *Proceedings of the 25th International Conference on Computational Linguistics COLING*, pp. 553–564.

Xiaojun Wan (2009), Co-training for Cross-lingual 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 1 - Volume 1*, ACL '09, pp. 235–243, Association for Computational Linguistics, Stroudsburg, PA, USA, ISBN 978-1-932432-45-9, URL http://dl.acm.org/citation.cfm?id=1687878.1687913.

Xiaojun Wan (2011), Bilingual Co-training for Sentiment Classification of Chinese Product Reviews, *Computational Linguistics*, 37(3):587–616, ISSN 0891-2017, doi:10.1162/COLI_a_00061, URL http://dx.doi.org/10.1162/COLI_a_00061.

Sida Wang and Christopher D. Manning (2012), Baselines and Bigrams: Simple, Good Sentiment and Topic Classification, in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL '12, pp. 90–94, Association for

Computational Linguistics, Stroudsburg, PA, USA, URL http://dl.acm.org/citation.cfm?id=2390665.2390688.

Wei Wang and Zhi-Hua Zhou (2007), Analyzing Co-training Style Algorithms, in *Proceedings of the 18th European Conference on Machine Learning*, ECML '07, pp. 454–465, Springer-Verlag, Berlin, Heidelberg, ISBN 978-3-540-74957-8, doi:10.1007/978-3-540-74958-5_42, URL http://dx.doi.org/10.1007/978-3-540-74958-5_42.

Wei Wang and Zhi-Hua Zhou (2013), Co-Training with Insufficient Views, in *Asian Conference on Machine Learning*, pp. 467–482.

Wei Wei and Jon Atle Gulla (2010), Sentiment Learning on Product Reviews via Sentiment Ontology Tree, in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pp. 404–413, Association for Computational Linguistics, Stroudsburg, PA, USA, URL http://dl.acm.org/citation.cfm?id=1858681.1858723.

Janyce Wiebe and Claire Cardie (2005), Annotating expressions of opinions and emotions in language. Language Resources and Evaluation, in *Language Resources and Evaluation (formerly Computers and the Humanities*, p. 2005.

Janyce Wiebe and Ellen Riloff (2005), Creating Subjective and Objective Sentence Classifiers from Unannotated Texts, in *Proceedings of the 6th International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing'05, pp. 486–497, Springer-Verlag, Berlin, Heidelberg, ISBN 3-540-24523-5, 978-3-540-24523-0, doi:10.1007/978-3-540-30586-6_53, URL http://dx.doi.org/10.1007/978-3-540-30586-6_53.

Janyce Wiebe, Theresa Wilson, Rebecca Bruce, Matthew Bell, and Melanie Martin (2004), Learning Subjective Language, *Computational Linguistics*, 30(3):277–308, ISSN 0891-2017, doi:10.1162/0891201041850885, URL http://dx.doi.org/10.1162/0891201041850885.

Janyce M. Wiebe (2000), Learning Subjective Adjectives from Corpora, in *In AAAI*, pp. 735–740.

Theresa Wilson (2005), Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis, in *In Proceedings of HLT-EMNLP*, pp. 347–354.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann (2009), Recognizing contextual polarity: An exploration of features for phrase-level sentiment analysis, *Computational linguistics*, 35(3):399–433.

Theresa Wilson, Janyce Wiebe, and Rebecca Hwa (2004), Just how mad are you? Finding strong and weak opinion clauses, in *In Proceedings of AAAI*, pp. 761–769.

Ian H. Witten and Eibe Frank (2005), *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, ISBN 0120884070.

David Yarowsky (1995), Unsupervised Word Sense Disambiguation Rivaling Supervised Methods, in *Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics*, ACL '95, pp. 189–196, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/981658.981684, URL http://dx.doi.org/10.3115/981658.981684.

Hong Yu and Vasileios Hatzivassiloglou (2003), Towards Answering Opinion Questions: Separating Facts from Opinions and Identifying the Polarity of Opinion Sentences, in *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, EMNLP '03, pp. 129–136, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:10.3115/1119355.1119372, URL http://dx.doi.org/10.3115/1119355.1119372.

Jiaming Zhan, Han Tong Loh, and Ying Liu (2009), Gather customer concerns from online product reviews–A text summarization approach, *Expert Systems with Applications*, 36(2):2107–2115.

Lei Zhang, Bing Liu, Suk Hwan Lim, and Eamonn O'Brien-Strain (2010), Extracting and Ranking Product Features in Opinion Documents, in *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pp. 1462–1470, Association for Computational Linguistics, Stroudsburg, PA, USA, URL http://dl.acm.org/citation.cfm?id=1944566.1944733.

Li Zhuang, Feng Jing, and Xiao yan Zhu (2006), Movie review mining and summarization, in *In Proceedings of the International Conference on Information and Knowledge Management (CIKM.*