FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

SENTIMENT ANALYSIS

Feature-based Sentiment Analysis in the Area of Technical Product Reviews

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Wirtschaftsinformatik

eingereicht von

Sebastian Lesslhumer
Matrikelnummer 0625828

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuer: ao.Univ.-Prof. Mag. Dr. Wolfdieter Merkl

Wien, 28.11.2011

(Unterschrift Verfasser) (Unterschrift Betreuung)

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

SENTIMENT ANALYSIS

Feature-based Sentiment Analysis in the Area of Technical Product Reviews

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Business Informatics

by

Sebastian Lesslhumer
Registration Number 0625828

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: ao.Univ.-Prof. Mag. Dr. Wolfdieter Merkl

Vienna, 28.11.2011

(Signature of Author) (Signature of Advisor)

Erklärung zur Verfassung der Arbeit

Sebastian Lesslhumer
Grabenweg 1, 3413 Hintersdorf

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 28. November 2011 Sebastian Lesslhumer

Danksagung

An erster Stelle gebührt mein Dank meinem betreuenden Professor Dr. Wolfdieter Merkl, der mich mit der spannenden und herausfordernden Thematik von Natural Language Processing im Allgemeinen und Opinion Mining im Speziellen vertraut gemacht hat. Unsere interessanten und motivierenden Gespräche, sowie die konstruktiven Anregungen waren für mich von großer Hilfe und Bedeutung.

Weiters möchte ich Dipl. Ing. Veronika Zenz, Dr. Michael Dittenbach und Mag. Andreas Pesenhofer danken. Durch unsere gemeinsamen Diskussionen haben sich mir zusätzliche Aspekte erschlossen, die mir bei der praktischen Implementierung diesr Arbeit geholfen haben, beziehungsweise diese erst ermöglichten.

Besonders bedanken möchte ich mich bei meinen Eltern Johanna und Anton Lesslhumer, die mich auf meinem bisherigen Lebensweg nach all Ihren erdenklichen Möglichkeiten untersützt haben und ohne deren Beistand vieles nicht möglich gewesen wäre, auf das ich heute mit großer Freude und Stolz zurückblicke.

Abstract

The continuous widespread availability of broadband Internet access accompanied by a more excessive usage has lead to an enormous growth of user generated content in the last years. People use blogs, e-commerce websites and social networks to share their experiences on a daily basis. Based on its popularity and the availability the Internet has become one, if not the most important source of information in recent years. This has great impacts on human decision making process. People's decision is strongly influenced by the experiences of other ones. Especially when it comes to buying decisions, the World Wide Web, has become a fundamental decision making instrument. For almost every product several reviews and blog entries exist. Given the amount of information hidden in the depths of the World Wide Web it is a very time consuming and wearisome task to find relevant product reviews and consequently read and compare them.

In order to automate and accelerate this process *sentiment analysis* systems are used. Sentiment analysis, which is also referred to as *opinion mining* is the computational task to automatically find sentiment and subjectivity in written text documents. The goal is to determine whether a whole document, a sentence or just a single phrase expresses a positive or negative opinion. Sentiment analysis has not only gained significant popularity in the research community but has also made its way to the perception of business people who have realised the value of such systems.

In this thesis a novel approach for *feature-based sentiment analysis* of the German language is presented. The objective of the proposed system is to automatically detect sentiment in product reviews and to set them in relation to specific product features. A product feature can be a part or an attribute of a product or even the product itself. The system is realized via a hybrid approach that combines machine learning techniques for resolving the sentence structure and identifying product features together with a comprehensive set of rules for determining the feature orientation as being either positive, negative or neutral. The system is designed for analysing documents in the field of technical product reviews. Because of its learning component the system can also be successfully applied to other domains (e.g. movie reviews).

Kurzfassung

Die kontinuierliche Verbreitung von Breitbandanschlüssen, sowie die damit einhergehende verstärkte Internetnutzung haben in den letzten Jahren zu einem starken Anstieg des sogenannten “user generated content” geführt. Internetnutzer verwenden vielfach die Möglichkeit in Blogs, auf Foren und “E-Commerce”-Seiten, sowie in sozialen Netzwerken ihre Erfahrungen und Eindrücke mit anderen Benutzern zu teilen. Aufgrund der Popularität, sowie der bereits erwähnten Verfügbarkeit, ist das Internet zu einer der wichtigsten Informationsquellen geworden. Dies wirkt sich auch sehr stark auf den menschlichen Entscheidungsprozess aus. Eine Entscheidung wird sehr stark von den Eindrücken und Erfahrungen anderer beeinflusst. Besonders bei Kaufentscheidungen verlässt sich der Mensch auf die Erfahrungen anderer Benutzer. Mittlerweile kann man im Internet zu fast jedem Produkt eine Vielzahl von Kundenrezessionen finden. Jedoch sind die Suche und das Lesen dieser Rezessionen eine oft zeitintensive und anstrengende Aufgabe.

Um diesen Prozess zu vereinfachen und zu beschleunigen verwendet man sogenannte *Sentiment Analysis* Systeme. Sentiment Analysis, das in der Literatur auch als *Opinion Mining* bezeichnet wird, beschreibt den rechnergestützte Prozess um in Textdokumenten subjektive Eindrücke und Meinungen zu analysieren. Das Ziel ist es zu bestimmen, ob das gesamte Dokument, einzelne Sätze oder einzelne Phrasen eine positive beziehungsweise eine negative Stimmung ausdrücken. Aufgrund der Vielzahl an Möglichkeiten, die ein solches System bietet, hat Sentiment Analysis nicht nur bei Forschern, sondern auch in der Wirtschaft großes Interesse erweckt.

In dieser Master’s Thesis wird ein neuartiger Ansatz für *Feature-based Sentiment Analysis* vorgestellt. Das postulierte System wurde für die Analyse von Textdokumenten in der deutschen Sprache konzipiert. Es unterstützt die automatische Identifikation von Meinungen sowie von Produktmerkmalen. Ein Produktmerkmal kann ein Teil eines Produktes, eine Produkteigenschaft oder das Produkt selbst sein. Das System wurde als hybrider Ansatz realisiert. Es verwendet „machine learning” Algorithmen sowie ein umfassendes Regelwerk, um die Orientierung (positiv, neutral, negativ) der Produktmerkmale zu bestimmen. Das System wurde ursprünglich entwickelt um Rezensionen technischer Produkte zu analysieren. Aufgrund der unterstützten Lernfähigkeit kann es auch auf andere Domänen (z.B. Filmrezensionen) erfolgreich angewendet werden.

Contents

List of Figures

List of Tables

CHAPTER 1

Preface

1.1 Motivation

In the 21^{st} century, which is often referred to as the *Information Age*, information is the most important resource for organizations. The business models of companies like Google or Facebook are based on collecting user data and obtaining relevant information, which can be used for personalized advertisement. Vendors strive to gain feedback of buyers and understand the reasons for the decisions buyers made. What are the pros and cons of certain products or services? How can the product be improved? What are the main reasons that encouraged a buyer to purchase the good? These are just a few questions which need to be answered by marketers.

In the last years a large increase in market share of e-commerce transactions and online shopping could be observed. The market research institute Forrester Research (2011*b*) recently postulated that the US online retail sales grew 12.6% in 2010 to reach 176.2 billion US dollars. Furthermore they expect a 10% compound annual growth rate from 2010 to 2015. US e-commerce is expected to reach $278.9 billion in 2015. In a separate report Forrester Research (2011*a*) states similar growth rates for Europe. European online retail sales grew by 18% from 2009 to 2010 and are expected to grow another 13% in 2011. In 2015 the total sales for Europe are forecast to reach 133.6 billion Euros.

Along with this trend the numbers of product reviews and technology blogs available online have recorded highest growth rates. Besides, the popularity of social networks has been growing steadily and inexorably for the last years. In the third quarter of 2011 the network Facebook[1] claims that it has already gained more than 800 million users worldwide.

But why do this impressive numbers matter for business? Surveys showed that peoples' decisions are strongly influenced by other persons' opinions (Pang and Lee, 2008). When faced with a product selection, consumers are suggested to perform an internal search (e.g., relying on their prior knowledge of brands) and an external search (Senecal et al., 2005). The external search comprises activities like gathering information about certain products or asking others

[1] cf. `http://www.facebook.com/press/info.php?statistics` (accessed on October 19^{th}, 2011)

about their recommendations. Before the widespread of the World Wide Web the first thing in the decision making process was to ask family members, friends and relatives about their experiences. But due to the growth of the Internet and the dispersion of broadband Internet access the way how people communicate and how they share their experiences has fundamentally changed. The Internet is not limited to professionals only. Rather, users can easily create their own websites, write online reviews, set up blogs to share their experiences and join social networks to "talk" with friends. This, so called *user-generated content* lead to an enormous increase in the number of documents available online. Thus the Web has become an important - if not the most important - source of information.

In fact, a resent survey[2] showed that Americans already use the Internet as the major information source (Pew Internet & American Life Project, 2010):

- 58% of Americans have done research online about the products and services they buy
- On a typical day, 21% of adults search for product information online
- Among internet users, 78% say that they at least occasionally conduct product research

Beside researching more and more people use the Internet to express their experience with products (Pew Internet & American Life Project, 2010):

- About a quarter (24%) of the Americans have posted comments or reviews online about the things they buy
- 32% of Internet users report that they have posted online product comments

Nowadays, if one wants to purchase a product, he/she is no longer limited to asking his/her friends and families. There are a huge number of reviews available online, which give opinions and expressions of existing users of the product (Liu, 2010). Companies also benefit from this user-generated content. To understand the cause and motivation of people to buy or not to buy a product is an essential question companies strive to understand. Companies no longer need to conduct own surveys or hire external consultants to examine the market. Rather, all the necessary information is already hidden in the World Wide Web. Thus, harvesting the relevant information from blogs, review sites and social networks is of great importance for companies. In order to utilize all this information mature systems are needed that can automatically detect and process this information.

As a matter of fact traditional SQL queries do not succeed in this field. The information is not available as structured data in a relational databases. Instead it is necessary to gather the data from different documents written in natural language and in a next step harvest the desired information. Mature information retrieval (IR) and natural language processing (NLP) systems as well as machine-learning methods present the fundamental basis. In addition to traditional classification approaches an interpretation and analysis of the language is required. Textual

[2]The survey was conducted between August 9^{th} and September 13^{th}, 2010 by the Pew Research Center's & Internet American Life Project. The survey was administered to a sample of 3,001 adults, age 18 and older (Pew Internet & American Life Project, 2010).

classification of documents based on facts has been around in the research community for a long time. A lot of methods have been postulated in this domain. Classical representatives of such text categorization systems are search engines. While text classification categorizes documents based on their topic, other applications require to process subjective information. The essential job of these systems is to detect sentiment and opinions expressed in documents. This task is referred to as *sentiment analysis*. As its name indicates, the goal is to analyse given texts and examine whether sentences, phrases or whole documents state a positive, neutral or a negative sentiment (Gindl et al., 2010).

A typical example for an e-commerce site is Amazon[3]. Amazon provides users with the functionality to rate a product by assigning one to five stars. In addition it is also possible for a user to explicitly write down his/her impressions and experiences in a detailed product review. While it is a rather trivial task to evaluate the five-star rating (traditional classification problem), it is a challenging job to automatically detect the subjective information stated in the review texts (Tang et al., 2009). These texts bear a great deal of information. While the stars just reflect an overall rating, the reviews allow to express sentiment about certain aspects of a product. The detailed information about positive and negative rated aspects is of great importance for the vendors, since it helps them to understand the advantages and consequently the drawbacks of their product.

As mentioned above, sentiment analysis has gained great attention by researchers and also by business people. The application of sentiment analysis is not limited to product reviews only. Rather, sentiment analysis can be used in many different application areas. In fact, companies of economic sectors but also governmental entities have understood the value of information hidden in the depths of the World Wide Web and have realized the needs for systems that allow to gather and make use of these information.

1.2 Problem Definition

Today, frameworks exist that support sentiment analysis. But so far, the most work on sentiment analysis has been done on the document level, for example distinguishing positive from negative reviews (Wilson et al., 2005). However, a lot of applications require sentiment analysis at the sentence-level or even at the phrase-level. Many of the existing approaches lack the ability to analyse documents on a more detailed level. In order to "understand" the polarity of a sentence it is necessary to establish relations between the sentiment bearing words to the underlying meaning of a sentence.

For example, consider the following sentence that expresses a user's experience with a mobile phone. In this case the reviewer states his/her impressions about certain aspects or parts of the phone:

"*The display is really bright but the responsiveness of the touch screen is bad.*"

This sentence contains one positive sentiment word (bright) and one negative sentiment (bad). The challenge is to set the sentiment bearing words or phrases in relation to the con-

[3] `http://www.amazon.at` (accessed on October 17^{th}, 2011)

text. For an analyst it would be interesting to know that the writer was quite happy with the display but there seemed to be some problems with the touch screen. Many existing approaches lack the ability to establish these relationships. For that reason no statement can be made if the sentiment is related to an attribute, a certain part, to the product itself or if it does not stand in context at all. This strongly reduces the expression power of sentiment analysis and limits the practical usage of these systems.

Researchers are eager to create systems that are able to understand the context. This research field is referred to as *feature-based sentiment analysis*. The idea is to use natural language processing (NLP) and stochastic machine learning methods to find feature terms (e.g. display, touch screen) for topic bearing words (e.g mobile phone) and establish relationships to the sentiment bearing words or phrases (e.g. bright, not so good).

1.3 Aim of the Work

The goal of this thesis is to research different sentiment analysis approaches and implement a solution for a feature-based sentiment analysis system. The prototype is developed for analysing text documents written in the German language. Instead of just stating the overall rating of a text the system allows to detect sentiment words at the sentence-level and at the phrase-level. In a next step the objective is to establish relationships to the product features that appear in the text.

The system is based on the utilization and combination of different analysis techniques. In general statistical methods and natural language processing techniques are used for sentiment analysis. Natural language processing techniques are based on the actual syntax of a sentence. They use part of speech taggers and dependency parsers, which allow to obtain the syntactical structure. Statistical methods use machine learning models (e.g. Support Vector Machines, Hidden Markov Models, Conditional Random Fields) to train models based on manually crafted training documents. These models are then used to determine the tag sequence of new documents. A sentence is tagged by assigning the tag sequence with the highest probability.

The main problem for specifying and developing a feature-based sentiment analysis system is the ambiguity of the human language. In addition the World Wide Web is not know for mature writing. In forum entries or product reviewss writers often do not put great emphasis on correct spelling, proper usage of small or capital letters, grammar and punctuation. This makes it hard for automatic language processing models to correctly identify and consequently classify the different words.

1.4 Methodological Approach

The methodology for this Master's Thesis is based on a three step approach.

In a first step a comprehensive literature study in the field of sentiment analysis was conducted. A broad groundwork of different techniques is presented to the reader in this thesis. Consequently the focus lies on state-of-the-art systems in feature-based sentiment analysis.

The next step was centred on developing an approach for feature-based sentiment analysis of text documents written in German. Based on machine learning methods and natural language processing a system was modelled and implemented. This prototype was executed on a data set

of product reviews and product comparisons. In addition the solution was abstracted in a way, that it also fetches valuable results for other domains.

In the third and final step the approach was evaluated in respect to performance and quality. The performance criteria was evaluated by looking at the runtime of the algorithm. The quality, by calculating the accuracy and comparing the results with the manually crafted gold standard, an example data set examined by a human being.

1.5 Structure of the Thesis

The remainder of the Master's thesis is organized in five main chapters:

Chapter 2 - Introduction to Sentiment Analysis: In this chapter a comprehensive introduction to sentiment analysis is given. First a clarification of the terminology used in this research field is given. Additionally the problems and challenges researchers face in this domain are addressed but also the many application domains are presented. Finally, an overview of the resources that are needed for sentiment analysis is provided.

Chapter 3 - Related Word: This chapter concentrates on existing state-of-the-art approaches for sentiment analysis. Different machine learning algorithms and rule based approaches are presented for both sentiment classification and feature-based sentiment analysis.

Chapter 4 - Implementation: In this chapter the novel approach for feature-based sentiment analysis is discussed. At first a general overview of the practical work is given. In the following the used technologies are presented and the challenges for analysing documents written in German are addressed. In the remainder of this chapter a detailed description of the system and its modules can be found.

Chapter 5 - Evaluation: In the evaluation chapter an introduction to evaluation approaches in the area of sentiment analysis is provided. Based on this theoretical background the evaluation results of the feature-based sentiment analysis system are presented to the reader.

Chapter 6 - Discussion: Finally, in the last chapter a summary of the Master's thesis is provided and an overview for possible future research is presented.

CHAPTER 2

Introduction to Sentiment Analysis

2.1 Defining the Terminology

So far the term *sentiment analysis* has been stressed a lot in this work. But what exactly is the meaning of sentiment analysis and how is it different from other buzz words like opinion mining, sentiment detection, sentiment classification or subjectivity?

2.1.1 Facts and Opinions

Statements about the world can be categorized into two main types: *facts* and *opinions* (Liu, 2010). Facts represent true objective statements about an entity in the world. Most current information processing techniques (e.g., search engines) work with facts, which can be expressed with topic keywords (Liu, 2010).

> **Fact:** "A thing that is known and whose truth can be proved" [1]

The second type are opinions, which are subjective impressions of one person about something. Therefore opinions vary from person to person. Someone might perceive something in a different way than someone else does.

> **Opinion:** "A view, personal belief or judgement formed about something, not necessarily founded on proof or certainty" [2]

Regarding this definition, two kinds of opinions can be distinguished (Kim and Hovy, 2006*b*). (1) *beliefs* about the world, with values such as "true", "false", "possible", etc. Statements like "*I believe that it will rain*" and "*The phone will get cheaper soon*" are examples for believes.

[1] cf. `http://oxforddictionaries.com/definition/fact` (accessed on August 21st, 2011)
[2] cf. `http://oxforddictionaries.com/definition/opinion` (accessed on August 21st, 2011)

(2) *judgements* about the world, with values such as "good", "bad", "fast", etc. Examples for judgements are "*The processor is really fast*" and "*The display is really good*". Kim and Hovy (2006*b*) argue that judgements and beliefs are not necessarily mutually exclusive. The statement, "*I believe that the processor of the phone is fast*" carries both a belief and a judgement.

Indeed, the concept of an opinion is a very broad one. In the context of this work, an opinion is understood as a **positive** or **negative** subjective judgement someone has about a feature of an object or about the object itself. Following the work of Liu (2010) there are six basic concepts needed to define the model of an opinion on an object:

- **Opinion holder:** An opinion holder h is the person or organization that holds a specific opinion on a particular object expressed in a document.
- **Document**: A document d is a collection of sentences $S = \{s_1, s_2, \ldots, s_n\}$ that evaluates one or more objects.
- **Object:** An object O is an entity, which can be a product, topic, person, event or organization, on which an opinion is expressed. An object can be represented as a hierarchy of components and sub-components. This taxonomy is realised as a *part-of* relationship.
- **Feature:** A feature f represents the attributes and the components of the object. An object O can have a set of features $F = \{f_1, f_2, \ldots, f_n\}$. Each feature $f_i \in F$ can be expressed by a set of words or phrases $W_i = \{w_{i_1}, w_{i_2}, \ldots, w_{i_m}\}$ which are synonyms of the feature or indicated by one of a set of feature indicator $I_i = \{i_{i_1}, i_{i_2}, \ldots, i_{i_q}\}$.

  If a feature f or one of its synonyms appears in a sentence s it is called *explicit feature*. If neither the feature nor its synonyms appear, but f is implied, it is referred to as *implicit feature*.
- **Opinion:** The opinion itself is a judgemental view, attitude, or appraisal on a feature f or the object O itself from an opinion holder (e.g., "good", "bad", "(I) like", etc.)
- **Polarity:** The polarity of an opinion p on a feature f indicates whether the opinion is positive p^+, negative p^- or neutral p^o.

In other works the notions *source* (opinion holder), *topic* (object), *aspect* (feature), *valence, evaluation* (opinion) and *semantic orientation* (polarity) have been introduced (Kim and Hovy, 2004; Choi et al., 2005; Kim and Hovy, 2006*b*; Kobayashi et al., 2007). The understanding is the same as for the terms introduced above and can be used interchangeably.

By putting together all these concepts the model of a direct opinion on an object can be defined as follows (Liu, 2010):

> In an opinionated document $d \in D$ an opinion holder h comments on a subset of features $E \subseteq F$ of the object O. The opinion holder expresses a positive, negative or neutral opinion on each feature $f_j \in E$, where each feature f_j can be expressed by a number of synonyms $W_j = \{w_{j_1}, w_{j_2}, \ldots, w_{j_m}\}$ or indicated by one of the feature indicators $I_j = \{i_{j_1}, i_{j_2}, \ldots, i_{j_q}\}$.

For better understanding of these concepts consider Figure 2.1, illustrating a product review obtained from Amazon.com:


By **Bonnie S.** - See all my reviews

This review is from: Samsung i9100 Galaxy S II Unlocked GSM Smartphone with 8 MP Camera, Android OS, 16 GB Internal Memory, Touchscreen, Wi-Fi, GPS - No Warranty - Noble Black (Wireless Phone Accessory)

I am in love with my SII.....the amoled screen is unreal for a cellphone. The only thing I would change is the battery.....it seems non-existent at times.....guess I will just have to deal with it!!!! Still 5 stars!!

Figure 2.1 – Example of a product review obtained from Amazon.com[3]: slightly modified screenshot of a user review of the Samsung Galaxy SII

In this case the Samsung Galaxy SII can be identified as the object the review is about. Bonnie S. is the author of this review and therefore the opinion holder. This text contains two features (1) *amoled screen* and (2) *battery life* as well as three opinions that are related to the product or its features. *"I am in love with my SII"*. Here Boonie S. expresses an opinion on the actual object (Samsung Galaxy SII). The opinion has clearly a positive polarity. The second opinion is expressed on the amoled display *"the amoled screen is unreal for a cellphone"*, which also states a positive polarity. Finally an opinion about the battery life *"it seems non-existing at times"*. This sentence contains an implicit feature, since battery life is not explicitly mentioned. "Non-existing" in context of the battery life has a negative polarity.

2.1.2 Sentiment Analysis

In general *sentiment analysis* is the task to analyse textual documents and find subjectivity within the text. A lot of different concepts have been introduced by researches, which more or less address the same tasks. Other concepts like *opinion mining*, *subjectivity detection* and *sentiment detection* have been applied in different works. While sentiment analysis is a rather new research field it has already gained great attention in the last years by research communities. The increasing importance of sentiment analysis can also be recognized by the fact that since 2010 an annual workshop[4] completely dedicated to sentiment analysis takes place.

Sentiment analysis uses textural information processing and therefore is based on Natural Language Processing (NLP). In general, NLP is a discipline of computational linguistics and is concerned with developing machines capable of understanding human language. Liddy (2001) provides a definition of Natural Language Processing as "a theoretically motivated range of computational techniques for analysing and representing naturally occurring texts at one or more levels of linguistic analysis for the purpose of achieving human-like language processing for a range of tasks or applications."

[3] `http://www.amazon.com` (accessed on August 20^{th}, 2011)

[4] Workshop on Computational Approaches to Subjectivity and Sentiment Analysis (WASSA); cf. `http://gplsi.dlsi.ua.es/congresos/wassa2011/` (accessed on August 21^{st}, 2011)

In contrast to other information processing techniques, which have been focused on mining and retrieval of factual information (e.g. information retrieval, Web search, text classification) sentiment analysis is focused on extracting subjective information. The detection of subjectivity in textual documents was first introduced by Wiebe (1994). He referred to subjectivity as aspects of language used to express opinions and evaluations. A definition for sentiment analysis based on the concept of subjectivity was provided by Tang et al. (2009), who defined *sentiment detection* as "the job to detect subjective information contained in texts, include viewpoint, fancy, attitude and sensibility".

The phrase *opinion mining* first appeared in a paper by Dave et al. (2003). They describe opinion-mining as a process, to "search results for a given item, generating a list of product attributes (quality, features, etc.) and aggregating opinions about each of them (poor, mixed, good)". Esuli and Sebastiani (2006) defined opinion mining, as "a sub-discipline at the crossroads of information retrieval and computational linguistics, which is concerned not with the topic a text is about, but with the opinion it expresses". Following a description stated by Liu (2008) opinion mining "is a method that aims to extract attributes and components of the object that has been commented on in each document of a set of evaluative text documents and to determine whether the comments are positive, negative or neutral".

Each of the definitions stated above is more or less centred around one certain aspect of sentiment analysis. But what all these definitions have in common is the explicit mention of subjectivity or sentiment and the focus on analysing written text documents. Thus, a very general definition for sentiment analysis could be:

> *Sentiment analysis or opinion mining is the computational task to automatically detect sentiment and subjectivity in text documents and to determine whether they express a positive or negative polarity.*

Besides the terms sentiment analysis and opinion mining other concepts have been introduced in this young research field, causing a lot of confusion. To overcome this ambiguity Pang and Lee (2008) tried to clarify the terminology. They argued that sentiment analysis and opinion mining deals with the same field of study and can be used interchangeably. Opinion mining can be seen as a sub discipline of text analysis in general. The research field of sentiment analysis can itself be roughly divided into *sentiment classification* and *feature-based opinion mining*. The former is focused on determining the overall sentiment of a text on the document level or the sentence level. The latter analyses a text on the phrase-level. One might argue, that feature-based sentiment analysis is just another refinement of sentence classification. But feature-based sentiment analysis is a more complex approach. Instead of just considering the topic of a review, feature-based opinion mining strives to extract certain object features. These features are set in relation to opinion bearing words. In this case statements about the orientation of each single feature of an object can be made. Figure 2.2 illustrates the categorization of opinion mining research.

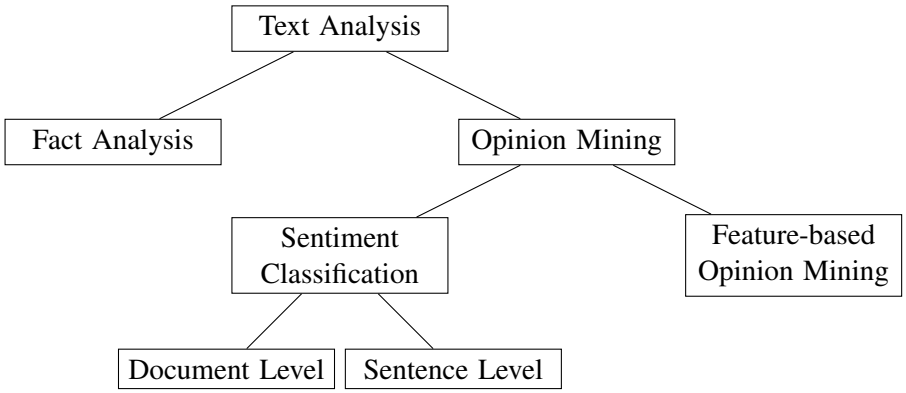


Figure 2.2 – Classification of opinion mining research (Bhuiyan et al., 2009)

2.2 Problems and Challenges of Sentiment Analysis

On a first though the detection of subjectivity and classification of polarity might seem similar to traditional topical classification tasks. A very basic approach is based on using a dictionary containing several sentiment words of each polarity (Tang et al., 2009). The document under consideration is searched for appearances of sentiment words by comparing each word with the dictionary entries. Then the occurrences of the words of each polarity are summed up. Finally, by calculating the ratio of the two sums the overall polarity of the document is determined. Even this simple procedure implies several challenges some of which are presented in the following.

2.2.1 Topic Classification and Sentiment Classification

Classical topical categorization attempts to sort documents according to their subject matter (Pang et al., 2002; Pang and Lee, 2008). Documents are assigned to classes based on the underlying topic (e.g. sport vs. politics). The goal of sentiment classification on the other hand is to identify the sentiment and polarity of documents. While in topical classification a lot of different categories can turn up, sentiment classification usually only supports few classes (good vs. bad, five star, etc.) that generalize across many domains and users (Pang and Lee, 2008). In topical classification the classes could be completely unrelated or items can be assigned to several classes. For example, consider an article about a politician doing sports. This article can be assigned to the categories "sport" and/or "politics". In sentiment classification the classes typically represent opposing categories or ordinal/numerical categories. Therefore the categories are mutually exclusive. A feature of a product is perceived either good, bad or neutral.

2.2.2 Detection of Keywords

A very basic problem that researchers face in sentiment analysis is to actually detect the right sentiment bearing words. In general, these words are selected by human-beings by hand. In a standard approach these keywords are enlarged by synonyms and antonyms from dictionaries.

In an early study Pang et al. (2002) showed that it is not that easy to figure out the right keywords. They focused on classifying reviews from the movie domain, using the Internet Movie Database[5] (IMDB) as their data source. Two students were asked to independently choose good indicator words for positive and negative sentiments in movie reviews. Pang et al. (2002) hypothesized that there are certain words people tend to use to express strong sentiments, so that it might suffice to simply produce a list of such words by introspection and rely on them alone to classify the texts. Table 2.1 shows the seed words the students picked to express strong sentiments. To prove their hypothesis Pang et al. (2002) applied the seed words to a set of movie reviews. As shown in Table 2.1, the accuracy for the human based classifiers were 58% and 64%, representing rather poor performance results. Also the tie rates - percentage of documents where the two sentiments were rated equally likely - were quite high.

Table 2.1 – Baseline results for human word lists (Pang et al., 2002)

	Proposed word lists	**Accuracy**	**Tie**
Human 1	positive: *dazzling, brilliant, phenomenal, excellent, fantastic* negative: *suck, terrible, awful, unwatchable, hideous*	58%	75%
Human 2	positive: *gripping, mesmerizing, riveting, spectacular, cool, awesome, thrilling, badass, excellent, moving, exciting* negative: *bad, cliched, sucks, boring, stupid, slow*	64%	39%

In a next step Pang et al. (2002) chose some word lists of the same size but based on the examination of the text corpus' statistics. As Table 2.2 shows this yielded much better results (accuracy of almost 70% and tie rate of only 16%) even though someone would not initially expect "still" to be a good sentiment indicator.

Table 2.2 – Results for baseline using introspection and simple statistics of the data (Pang et al., 2002)

	Proposed word lists	**Accuracy**	**Tie**
Stats	positive: *love, wonderful, best, great, superb, still, beautiful* negative: *bad, worst, stupid, waste, boring, ?, !*	69%	16%

[5] `http://www.imdb.com/` (accessed on August 21st, 2011)

2.2.3 Subjectivity Detection

Another problem is linked to the differentiation between facts and opinions in a document. A typical example for this problem is a newspaper article. What are the actual facts and what is the author's personal attitude?

In general, it is a difficult task to define what an opinion is. Often, an expression that is considered as an opinion in one domain might not be an opinion in another. As a result it is hard or not reasonable to apply opinion bearing words collected from one domain to another domain (Kim and Hovy, 2006*b*). For instance consider the following statement:

> "*The screen is very big.*"

This sentence can reflect a positive judgement for a screen review, but it could also be a mere fact. Therefore it is necessary to collect specific opinions within individual domains. Generally it is not possible and if, just to a limited extent, to apply training data from existing sources to other domains. An additional difficulty for detecting opinions and facts is that expressions like "*in fact*" and "*the fact that*" do not necessarily state the objective truth and bigrams[6] like "no sentiment" do not guarantee the absence of opinions either (Pang and Lee, 2008).

Especially in the field of reviews and product comparisons it is necessary to retrieve subjectivity and facts. Since facts usually have greater relevance they have to be handled differently.

> "*Phone A is more expensive than Phone B but I think its worth the money because the former has a better display.*"

This statement expresses the fact, that in order to buy Phone A more money is needed than to buy Phone B but also the subjective information that the opinion holder thinks that Phone A has the better display.

2.2.4 Ambiguity of the Human Language

The main challenge in sentiment analysis is the ambiguity of the human language. In comparison to face based analysis things can be expressed in a more subtle manner, making it hard to be identified (Pang and Lee, 2008). Many different ways exist to express the same thing. The meaning of an expression can completely change in different domains. It is possible to change the word order, to use negation, to refer to nouns by using pronouns and so on.

Different meanings of words: One and the same word can have completely different meanings when the context changes. It is a trivial task for a human being to read a text and set a word in relation to the underlying context. But this task being perceived easy for humans presents a big challenge for machines.

[6]Bigrams are a special case of n-grams. N-grams are groups of n written words. An n-gram with a size of one is referred to as an "unigram", groups having two or three items are called "bigrams" respectively "trigrams". N-grams are commonly used in natural language processing for statistical analysis of texts; cf. `http://en.wikipedia.org/wiki/N-gram` (accessed on October 18^{th}, 2011).

Consider the following sentence:

"*The mobile phone has a great display.*"

On one hand "*great*" can refer to the size of the display. The writer can express his/her sentiment that the display is big compared to others. On the other hand, "*great*" can also be used in the way to express that the display is really good.

Negation: The usage of a negation term completely changes the polarity of a sentence. Consider the sentence:

"*I do not like the mobile phone.*"

Even though a positive sentiment word appears the sentence has a negative polarity. The negation word "*not*" changes the meaning from positive to negative. So far this does not represent a big problem. It is possible to define rules that a specific prefixed term negates the meaning of a sentence. But negation can be expressed in a more subtle manner. The negation can appear at the beginning or ending of a sentence, or one can use double negation.

"*It's not that I don't like the film but it has not fully convinced me".*

This example represents the challenges of negation. In order to consider all cases a huge set of rules would be needed and still they would not cover every eventuality.

Indirect speech: Indirect speech is a sophisticated form of speech act in which speakers convey their message in an implicit way (Tsur et al., 2010). An important manifestation of indirect speech is irony or sarcasm. Ironic statements typically imply a meaning in opposition to their literal meaning. One is stating the contrary of what he/she actually means. Sarcasm can be defined as "the use of remarks which clearly mean the opposite of what they say, and which are made in order to hurt someone's feelings or to criticize something in a humorous way"[7].

Irony or sarcasm are mainly transported via speech because the inflection or tone of voice are often needed to express it (Tsur et al., 2010). Nonetheless, sarcastic writing is common in online communities and is popular in blog posts and product reviews. The ambiguous nature of sarcasm sometimes makes it hard even for humans to decide whether a statement is sarcastic or not. The automatic recognition of sarcasm is a novel task in natural language processing and only few works address the issue. So far sarcasm is seen as "a hard nut that is yet to be cracked" (Tsur et al., 2010). Following examples present the difficulties of sarcasm detection. Even tough both sentences express a clear positive statement the writer can also want to express the exact opposite. Humans might identify the sarcasm as such when considering the context, but machines lack this ability.

"*This is probably the best mobile phone I have ever had.*"

"*This is everything I was looking for.*"

[7]cf. `http://dictionary.cambridge.org/dictionary/british/sarcasm?q=sarcasm/` (accessed on August 24^{th}, 2011)

Domain dependence: Subjectivity is rather domain dependent. The exact same expression can have total different meanings in different domains (Pang and Lee, 2008).

> "*Go read the book!*"

Generally one would say that this sentence bears a positive polarity. But in the domain of movie reviews one would conclude that the movie is bad and it would be better to read the book instead (Pang and Lee, 2008).

Order effects: The order in which different opinions occur, has a great impact on the overall sentiment. In fact, the order has more importance than the frequency (Pang and Lee, 2008). Consider following text from a movie review[8]:

> "*This film should be brilliant. It sounds like a great plot, the actors are first grade, and the supporting cast is good as well, and Stallone is attempting to deliver a good performance. However, it can't hold up.*"

Even though the number of terms with a positive sentiment is by far higher than the number of negative items the overall polarity is negative. The final sentence completely changes the polarity of this excerpt.

Another implication of order dependencies is that comparisons are not symmetric. "A is better than B" is the complete opposite of "B is better than A".

Mature writing: A major problem for natural language processing is the fact that people do not put so much emphasis on correct spelling when writing texts online. Especially product reviews available online often lack correct spelling and punctuation. In addition, writing in minuscule and ignoring capitalization makes it hard for machines to correctly process the text. Another complexity is based on the common use of abbreviations (e.g. *sync* instead of *synchronization*) requiring to consider all possible variants. Especially in tech blogs the so called leet speak[9] is used, which cannot be analysed using traditional lexicons.

So far, some of these issues have been solved by researchers but still there are a lot of problems that have not been fully addressed in the literature and which are still subject for further research.

[8] This example was adopted from Pang and Lee (2008)

[9] Leet speak (1337 Sp34k): an alternative alphabet for the English language that is used primarily on the Internet. It uses various combinations of ASCII characters to replace Latinate letters. Leet speak has its origins in the online gaming community but by now it is also used in other online domains. Typical examples for leetspeak are "c00l" (cool) or "suxx" (sucks); cf. `http://www.urbandictionary.com/define.php?term=leet+speak` (accessed on August 24^{th}, 2011).

2.3 Application Domains of Sentiment Analysis

Sentiment analysis can be used in a lot of different application domains and represents a big economic factor. The ability to understand reasons for humans' decision making is an essential advantage when it comes to competitiveness and profit maximization. Since customers want to know which product to buy they read reviews, collect information about products or use recommender systems to support their decision making process. For that reason opinions presented online are getting more and more attention. Online opinions represent wealth of information which can be beneficial for the industry as well as for consumers. But sentiment analysis or opinion mining is not just limited to the domain of electronic business. It can also be used in many other domains. Entertainment, research and development or electronic learning indicate other important application domains (Archak et al., 2007). Even governmental entities can benefit from opinion mining to find out public opinions and views towards politicians, political parties or towards political manifestos.

2.3.1 Review Analysis

The very basic application of sentiment analysis is inherent in the field of analysing online reviews. It is common for e-commerce companies to provide customers the possibility to write reviews about products. Basic opinion mining methods determine the overall polarity of such a review text. Sentiment classification automatically processes several documents about a product and assigns the product (object) to a class (Tang et al., 2009). Possible classes are "recommended" and "not-recommended".

While this method just expresses an overall impression reviews allow to discuss the utility of a product in more detail. A product usually has several features. Some features may be perceived positive, while others might have shortcomings in the eyes of the reviewer. An automated analysis of one product allows to compare opinions from several reviewers and so the advantages and disadvantages of the product can be clearly seen. Customers can compare different products by feature-to-feature comparisons based on consumer opinions and easily find the right product.

Review analysis is not limited to products only. For instance, many approaches have been published that strive to extract sentiment information from movie reviews (Pang et al., 2002; Chaovalit and Zhou, 2005; Zhuang et al., 2006; Zhao and Li, 2009). This domain is especially used in experimental works, because there are large online collections of movie reviews available (e.g. The Internet Movie Database[10], Rotten Tomatos[11]).

2.3.2 Business Intelligence and Marketing

Another application area of sentiment analysis can be found in business intelligence. As mentioned above customers can benefit from review analysis and product comparison systems helping them to decide which product to buy. But also suppliers and sellers can use sentiment analysis for their advantage. They gain insight into the consumers' decision processes and see which product features posses a positive perception and which do not.

[10] `http://www.imdb.com` (accessed on August 24^{th}, 2011)

[11] `http://www.rottentomatoes.com` (accessed on August 24^{th}, 2011)

Consider a mobile phone manufacturer selling a new smart phone, but its sales figures being below the expectations[12]. Disappointed by the selling numbers the manufacturer seeks to answer the question: "Why are people not buying my product?" Concrete data like the price of the phone or the price of the competitors' models are obviously relevant. But a satisfactory answer to that question requires to focus more on the personal views of the customers. Statements like "*The display is not bright*" or "*The respond time of the phone is really slow*" bear a lot more useful information than bare facts, since they help manufacturers to understand the drawbacks of their products.

Sentiment analysis also allows to examine other sources of information like newsgroups, forums and webblogs. Especially tech blogs contain valuable information. Some works even aim to perform trend prediction in sales. A study by Gruhl et al. (2005) showed that peaks in references to books in weblogs are likely to be followed by peaks in their sales. In their work Mishne and Glance (2006) examined if sentiment correlates with the product's financial performance. They proposed a framework to predict sales figures of movies by analysing sentiment to the movie in webblogs.

In addition opinion mining can also be well used for reputation management. Understanding what people think of a specific company helps to launch public relation initiatives and to position the company on the market in a positive way.

2.3.3 Sentiment Analysis as an Enabling Technology

Sentiment analysis has also gained great importance as an enabling technology for other applications. Opinion mining applications find use in *recommender systems*. Such a system proposes potentially interesting products for a certain customer based on personal interests, interests of comparable customers and on product features. Of course, these recommender systems should avoid suggesting products that received negative feedback. Therefore an analysis of product reviews is a crucial part of mature recommender systems.

Pang and Lee (2008) also proposed sentiment analysis applications to detect opinion spam[13] and flame[14]. The detection of opinion spam has become increasingly important, since opinion spam may tamper the trustworthiness of online opinions and as a result, opinions on the web can even become useless to mining applications. Jindal and Liu (2008) proposed an approach to detect opinion spam in reviews. They distinguish between three types of spam reviews. (1) Untruthful opinions, which can be classified in *hyper spam*, giving undeserving positive reviews to some target objects in order to promote the objects and *defaming spam*, giving unjust or malicious negative reviews to some other objects in order to damage their reputation. (2) Reviews on brands only: commenting only the brands, the manufacturers or the sellers of the products and (3) non-reviews: advertisements and other irrelevant reviews containing no opinions at all. In general, spam detection can be regarded as a classification problem with two classes, *spam* and *non-spam*.

[12] This example is adopted from Pang and Lee (2008, p. 13)

[13] Spam: unsolicited and mostly irrelevant electronic messages sent out in mass quantities;
cf. `http://www.urbandictionary.com/define.php?term=Spam` (accessed on August 24^{th}, 2011)

[14] Flame: to insult someone electronically, in emails or other types of communication;
cf.`http://www.urbandictionary.com/define.php?term=Flame` (accessed on August 24^{th}, 2011).

Sentiment analysis is also used in politics and in rule making of governmental entities. Opinions and believes matter a great deal in politics. *Electronic rulemaking* (eRulemaking) uses information technology and especially sentiment analysis to make the process of rule-making more transparent and accessible to the public. Several papers have been published in this area (Yang and Callan, 2005; Cardie et al., 2006).

Opinion mining in legal Weblogs (referred to as "blawgs") has also gained great attention by researchers. The blawgosphere has been growing at a rapid pace and many potential applications for opinion detection and monitoring are arising as a result (Conrad and Schilder, 2007). The mining of blawgs can be valuable to a legal researcher who is looking for perspectives on legal issues.

2.3.4 Social Media Analysis

The popularity of social networks makes the mining of these platforms increasingly important. Recently, a lot of studies have been published focusing on mining opinions from twitter[15] messages (Barbosa and Feng, 2010; Go et al., 2009; Kim et al., 2009; Pak and Paroubek, 2010). Some works cover the prediction of the near future based on tweets. In a recent paper Asur and Huberman (2010) analysed twitter messages to forecast the revenue of movies. Tumasjan et al. (2010) proposed an approach to forecast the outcome of elections.

Sites like tweetfeel [16] or twittersentiment[17] gather tweets from twitter in real-time and analyse the feelings towards a certain object. Even special social networks exist (e.g. mattermeter[18]) based on the idea of members building relationships strictly based on opinions.

Ahkter and Soria (2010) focused on mining Facebook[19] status messages. They argue that Facebook status messages are more succinct than reviews, and are easier being classified than tweets because their ability to contain more characters allows for better writing and a more accurate conveying of emotions.

2.4 Resources for Opinion Mining

In order to establish a sentiment analysis system that operate properly several resources are needed. The very basic resource is the opinion lexicon, which lists a number of positive and negative seed words that are used for identifying opinions in the text. However, besides the opinion lexicon also the actual documents need to be collected and prepared for opinion mining. Opinion mining systems based on machine learning approaches also require datasets to train the classifiers. The resources and the necessary processing steps are presented in the following.

[15] `http://www.twitter.com` (accessed on August 24^{th}, 2011)
[16] `http://www.tweetfeel.com` (accessed on August 24^{th}, 2011)
[17] `http://twittersentiment.appspot.com` (accessed on August 24^{th}, 2011)
[18] `http://www.mattermeter.com` (accessed on August 24^{th}, 2011)
[19] `http://www.facebook.com` (accessed on August 24^{th}, 2011)

2.4.1 Opinion Lexicon

An opinion word, also referred to as opinionated word or sentiment word is used to express someone's feeling towards a person, object, situation or state. Positive opinion words express desired states while negative opinion words are used to express undesired states (Liu, 2010). Examples of positive opinion words are *good*, *beautiful*, *wonderful*, *awesome*, etc. Examples of negative opinion words are *bad*, *poor*, *corrupt*, etc.

Opinion words can be discriminated into two types, the *base type* and the *comparative type* (Liu, 2010). The base type is used to express an opinion towards an object "*My new mobile phone is awesome*". The examples stated above belong to the base type. Opinion words of the comparative type express relationships between two or more objects "*I like my new phone better than my old one*". Typical examples for the comparative type are words like *better*, *worse*, *best*, *worst*. Opinion words of the comparative type are usually generated as the comparative and superlative of the basic adjectives.

Commonly, finding an entry for an opinion lexicon can be seen as word sentiment classification. A word, which is found to express a sentiment is assigned to the corresponding class depending on its polarity. Different approaches exist to generate opinion lexicons. The standard approach is to manually create a complete list of relevant opinion words. But this is a very laborious and time consuming task. In addition, it does not necessarily lead to the best results as discussed in section 2.2.2 - *Detection of Keywords*. Instead, it is common to combine the manual task with automated methods. Two different types of automatic approaches are in use, *dictionary based approaches* and *corpus based approaches*.

2.4.1.1 Dictionary Based Approaches

As the name already states, dictionary based approaches fall back on the utilization of dictionaries. A simple technique is based on bootstrapping, starting with a small set of manually labelled seed words with known polarity . A dictionary is automatically searched for synonyms and antonyms of these words Liu (2010). The set of seed words is iteratively extended. The process stops when a certain number of seed words is reached or when no more seed words can be found in the dictionary.

A really important dictionary is WordNet[20], which is hosted from the Princeton University. WordNet is a comprehensive database of the English language. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms, each expressing a distinct concept. These sets of synonyms are called *synsets*. Synsets are interlinked by means of conceptual-semantic relations (e.g. hyponym/hypernymy[21]) and lexical relations (e.g. synonymy/antonymy). Figure 2.3 shows the general structure of WordNet by means of the adjective "fast".

Usually opinions are expressed by using adjectives. Adjectives share the same orientation as their synonyms and opposite orientations as their antonyms. Based on this circumstance Hu

[20] `http://www.wordnet.princeton.edu` (accessed on August 30^{th}, 2011)

[21] Hypernymy is a semantic relation between two word meanings. A hyponym is included within the semantic field of another word, called the hypernym. In computer science hypernymy is also called the *IS-A relation*. It generates a hierarchical semantic structure. E.g. iPhone is a hyponym of smart phone and mobile phone is a hyernym of smart phone; cf. `http://en.wikipedia.org/wiki/Hyponymy` (accessed on August 31^{st}, 2011).

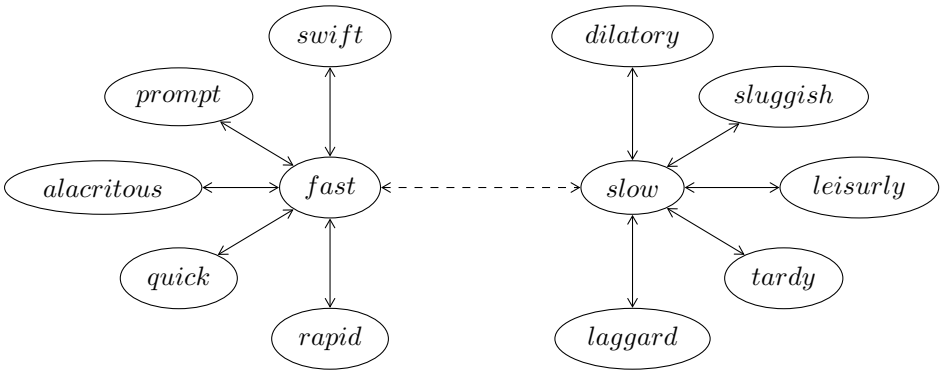


Figure 2.3 – General structure of WordNet represented by means of the adjective "fast". Continuous lines indicate the synonymy relationship; dotted lines indicate the antonymy relationship (Hu and Liu, 2004*a*)

and Liu (2004*a*) applied a bootstrapping method using WordNet. They defined a set of seed adjectives with known polarity. Searching WordNet other adjectives are obtained. Once an adjective's orientation is predicted it is added to the seed list.

Another bootstrapping method to determine the orientation of a word was presented by Turney and Littman (2003). Starting with two minimal sets of seed terms S_p, S_n the semantic association between the target term t and every seed term $t_i \in S_p \cup S_n$ is calculated. The seed terms serve as the indicators of the categories POSITIVE and NEGATIVE and are marked as

- $S_p = \{$good, nice, excellent, positive, fortunate, correct, superior$\}$,
- $S_n = \{$bad, nasty, poor, negative, unfortunate, wrong, inferior$\}$.

Kamps et al. (2004) proposed a graph-theoretic model for representing the synonymy relationship in WordNet. Given a graph $\mathcal{G}(W, syn)$ with W a set of words in WordNet and syn a set of edges connecting each pair of synonymous words. The distance $d(w_i, w_j)$ between two words w_i and w_j is the length of a shortest path between w_i and w_j. Kamps et al. (2004) defined a function EVA that measures the relative distance of a word w to the two reference words "good" and "bad".

Applying the former methods, only adjectives can be evaluated. But opinions can also be expressed by means of nouns and verbs. Esuli and Sebastiani (2005) presented a method that also considers verbs and nouns as opinion words. Their method exploits the glosses (textual definitions) a word has in dictionaries. They used WordNet as the source of lexical relations. Their postulated process composes four steps. First a seed set (S_p, S_n) is created, where S_p contains positive words and S_n negative words. Second, lexical relations (e.g. synonymy) are used in order to find new terms that are added to S_p or S_n. For each entry $t_i \in (S_p \cup S_n)$ a textual representation is created by obtaining all glosses of t_i from WordNet. Finally an opinion word classifier is trained on the terms in $(S_p \cup S_n)$ and applied to the terms in the test set. The

approach of Esuli and Sebastiani (2005) is based on the idea that terms with similar orientation tend to have "similar" glosses.

Semantic Tag Extraction Program (STEP) is an extension to the basic approaches and was first presented by Andreevskaia and Bergler (2006). It combines the original bootstrapping method with glosses. STEP starts with a small list of seed words (S_p, S_n). In a next step WordNet is searched for synonyms and antonyms which are added to the initial seed list. Then STEP goes through all WordNet glosses and identifies the entries that contain the sentiment-bearing words in their definitions. The identified head words are also added to the seed list. In the last step the opinion lexicon is disambiguated by eliminating double entries and ambiguous words.

An important lexical resource based on WordNet is SENTIWORDNET. Each synset syn listed in WordNet is associated to three numerical scores $Obj(syn)$, $Pos(syn)$ and $Neg(syn)$, describing how Objective, Positive, and Negative the terms contained in the synset are (Esuli and Sebastiani, 2006). Figure 2.4 shows the graphical representation of a term's subjectivity and polarity in SENTIWORDNET.

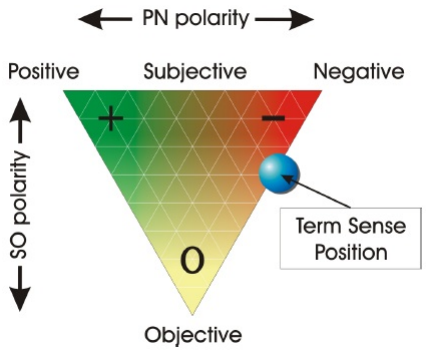


Figure 2.4 – Graphical representation of a term's subjectivity and polarity in SENTIWORDNET (Esuli and Sebastiani, 2006)

German Opinion Lexicons: So far, all presented approaches are focused on the English language. But also some opinion lexicons for other languages exist[22]. An important example is GERMANET[23], the German version of WordNet. Like in WordNet the semantic concept is modelled by the means of synsets (Henrich and Hinrichs, 2010). Another example is Sentiment Wortschatz or SentiWS[24]. SentiWS lists positive and negative sentiment bearing words (Remus et al., 2010). The strength of the semantic association between two words w_1, and w_2 is calculated and scaled to the interval of $[-1; +1]$ with $+1.0$ being absolutely positive and -1.0 being absolutely negative.

[22] In the following lexical resources for the German language are presented, since they are important for the practical part of this thesis.

[23] GERMANET is developed and hosted by the Division of Computational Linguistics of the Linguistics Department, Eberhard Karls Universität Tübingen, Germany; cf. `http://www.sfs.uni-tuebingen.de/lsd/index.shtml` (accessed on August 30^{th}, 2011).

[24] Sentiment Wortschatz is developed by the University of Leipzig; cf. `http://wortschatz.informatik.uni-leipzig.de/` (accessed on August 30^{th}, 2011).

2.4.1.2 Corpus Based Approaches

Dictionary based approaches can be easily and fast generated and produce comprehensive lists of opinion words. But Liu (2010) argues that dictionary based approaches have a major shortcoming since their entries are domain independent. It is not possible to find opinion words with domain specific orientation. For example consider the following statement about the durability of a mobile phone, "*This phone lasts very long*". In this case the word "long" bears a positive sentiment. But long can also have a negative meaning when used in a different context: "*The phone takes really long to open an application*". In this case long refers to the time an application needs to get started, which the author considers to take too much time.

Corpus based approaches strive to overcome this shortcoming by learning opinion words from the actual text corpus. The corpus based methods rely on syntactic or co-occurrence patterns (Liu, 2010). Like in dictionary based methods the starting point is a manually labelled set of seed words categorized in having positive or negative polarity. These initial seed words are extended by means of analysing the text corpus and finding relationships.

In an early paper Hatzivassiloglou and McKeown (1997) postulated a method that extracts domain-dependent information from the text corpus and automatically adapts to a new domain when the corpus is changed. Their method is based on the hypothesis that adjectives, which are connected by certain conjunctions share the same semantic orientation. *"The mobile phone is fast **and** intuitive"*. Assuming that "fast" is known to be positive, it can be inferred that "intuitive" is also positive. The situation is reversed for adversative conjunctions like "but", which usually connects two adjectives of different orientations "*The mobile phone is fast **but** heavy*". Hatzivassiloglou and McKeown's (1997) algorithm comprises four steps. In the first step all conjunctions of adjectives are extracted from the text corpus. Then, in the next step a regression model is used to determine if two conjoined adjectives are of the same or different orientation. In the third step a clustering algorithm separates the adjectives into two subsets of different orientation. Finally, the average frequencies in each group are compared and the group with the higher frequency is labelled as positive, since usually more positive than negative words are used in texts (Hatzivassiloglou and McKeown, 1997). Liu (2010) refers to this approach as *sentiment consistency* even though this method is not always consistent in practice. Because of its simplicity and the ability the score good results this method is still commonly used.

Context coherency is an unsupervised learning method proposed by Kanayama and Nasukawa (2006) to acquire domain dependent lexical knowledge. Instead of adjectives they used so called *polar atoms*. A polar atom is a minimum syntactic structure specifying polarity in an expression. Kanayama and Nasukawa (2006) assumed that polar clauses with the same polarity appear successively unless the context is changed by adversative conjunctions. The polarity of a set of domain-depended polar clauses is determined by considering clauses adjacent to the polar atoms listed in the initial lexicon. Two kinds of polar clauses can be distinguished, clauses appearing within a sentence (intra-sentential) and clauses between neighbouring sentences (inter-sentential).

The following sentence provides examples for inter-sentential and intra-sentential clauses:

> *"My new mobile phone is great. It is fast and very intuitive. The display is very bright but not very big. Nonetheless I am still satisfied."*

Figure 2.5 represents the intra- and inter-sentential contexts of this text. The words "but" and "nonetheless" are adversative conjunctions changing the polarity of the polar clauses appearing afterwards. The existence of an adversative conjunction is denoted by the symbol $\otimes$.

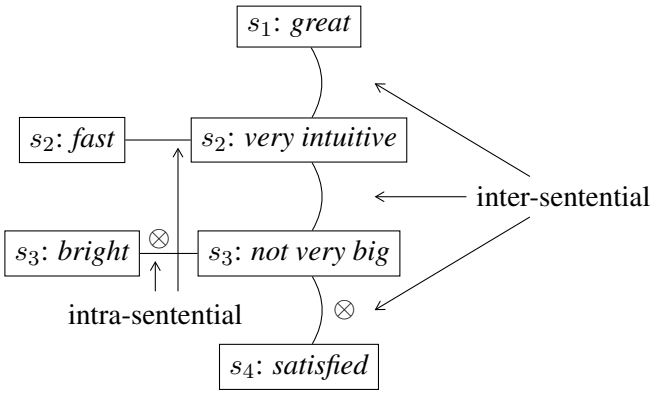


Figure 2.5 – Concept of intra- and inter-sentential contexts (Kanayama and Nasukawa, 2006)

Other approaches exploit the relationships between sentiment words and product features. Qiu et al. (2009) argued that opinion words almost always appear together with a product feature. Therefore, the opinion words can be recognized by identified features and vice versa features can be recognized by known opinion words. An early system exploiting this relationship was postulated by Hu and Liu (2004*b*). An advancement was presented by Qiu et al. (2009). They distinguished three types of relationships: (1) relationships between sentiment words, (2) relationships between features and sentiment words and (3) relationships between features and features. Beginning with a set of seed words, other opinion words as well as object features can be found. The obtained features and opinion words are then used to find more concepts. Since the lexicon propagates due to opinion words and object features, this approach is called double propagation.

Niazi et al. (2011) presented a straight forward approach employing term frequency. The polarity of domain dependent opinion words is determined by counting their appearances (term frequency) in two training corpora, one corpus being a set of positive reviews, the other corpus containing only negative reviews. A word is assigned to the class corresponding to the corpus in which the frequency is higher.

One might think that generating the opinion lexicon and the actual process of opinion mining are more or less the same. Both methods use machine learning or rule-based approaches and strive to find words or phrases with positive or negative polarity. Liu (2010) argues that this tasks are indeed different. Just because an opinion word is listed in a lexicon does not mean that it is actually expressing an opinion. However, corpus-based approaches are strongly related to feature-based opinion mining and are well integrated in opinion mining systems.

2.4.2 Document Corpora

A document corpus is a collection of documents usually obtained from online sources (e.g. online shopping sites, movie review portals, etc.) for the purpose of linguistic description and analysis. Before a system is able to identify sentiment in unlabelled documents it is necessary to prepare the documents for the subsequent analysis. The document preparation comprises several steps to eliminate ambiguity and to generate processable textual documents.

2.4.2.1 Sentence Detection and Tokenization

The basic task before being able to work with a document is to split it into a number of sentences and in a next step identify the single tokens of each sentence. A token can be an actual word or a special character like parentheses, punctuation marks, and others. Consider following sentence:

I really like the new phone.

The tokens for this sentence are:

```
I, really, like, the, new, phone, .
```

2.4.2.2 Part-Of-Speech Tagging

Part-of-speech (POS) tagging is probably the most important preparation task. Part-of-speech tags build the foundation for many opinion mining systems. Especially in feature-based opinion mining POS tags are widely applied.

In general, part-of-speech tagging is the process of assigning part-of-speech tags to the words in a sentence (Toutanova et al., 2003). These tags are then used to find opinion holders, objects, object features and opinion words. The tagging process requires a standard set of tags. For the English language several different tag sets exist that have been applied in research. A very simple set consists of nine basic part-of-speech tags: noun (N), verb (V), adjective (Adj), adverb (Adv), preposition (Pre), article (Art), pronoun (Pro), conjunction (Con) and interjection (Int). In practice this level of granularity is too coarse. There are many different cases that have to be considered for each tag. For example plural, possessive, and singular forms can be distinguished in case of a noun. Therefore finer-grained sets are usually used for opinion mining systems. In the English language the Penn Treebank[25] tag set is widely applied. It contains 36 POS tags and 12 other tags for punctuation and currency symbols. An introduction to the Penn Treebank tag set and a detailed listing of all the tags can be found in Marcus et al. (1993). Considering the sentence from above POS-tagging leads to the following result:

```
I[PRP] really[RB] like[VBP] the[DT] new[JJ] phone[NN] .[.]
```

Different methods for POS tagging exist. Rule-based taggers use large collections of rules and constraints to identify part-of-speech (Brill, 1992, 1994). Stochastic tagging on the other hand uses probabilistic models and machine learning algorithms to train taggers based on manually labelled training sets (Brants, 2000; Toutanova and Manning, 2000; Toutanova et al., 2003).

[25] `http://www.cis.upenn.edu/~treebank/` (accessed on August 30^{th}, 2011)

2.4.2.3 Stemming

Once the single tokens are identified, it is reasonable to consider the many morphological variants a word can have. Usually the different morphological variants of words have similar semantic interpretations. Thus, these words can be considered as equivalent. The process of reducing inflected words to their stems is called *stemming*[26]. A stem is the part of the word that never changes even when morphologically inflected. The stem of the word "declared" is "declar". This is because of words like "declaration".

Stemming uses a set of rules to handle different words and their forms. Usually suffix stripping algorithms are used that remove suffixes like "-ing", "-ed", etc. Typical examples of stemming rules are shown in Table 2.3.

Table 2.3 – Possible rules for stemming

Part-of-speech	**Rule**		**Example**
NN	plural ⇒ singular	remove "-s"	phones ⇒ phone
		remove "-ing"	writing ⇒ writ
V	conjugation ⇒ infinitive	remove "-d", "-ed"	heard ⇒ hear
ADJ	comparison ⇒ positive	remove "-er", "-est"	fastest & faster ⇒ fast

The advantage of stemming is that terms of a document are represented by stems rather than by the original words. Therefore, the frequency count is higher compared to considering each word form on its own. Stemming also leads to a reduction in processing time and storage space since only the stems have to be saved. The problem of stemming is broadly addressed in information retrieval research and a lot of algorithms exist for this task.

2.4.2.4 Lemmatization

Lemmatization[27] is closely related to stemming. But while stemming is focused on the stem of a word, lemmatization uses the lemma. The lemma is the base form of a word. For example the lemma of the word "declared" is "declare", while the stem is "declar". The combination of the lemma with the part of speech is often called the lexeme of the word.

Lemmatization is a more complex task then stemming but it also produces more accurate results. A stemmer operates on a single word without knowledge of the context. Lemmatization on the other also considers the context of a word depending on the part of speech. A typical example where stemming does not lead to a result are the two adjectives "best" and "better". These adjectives can not be transformed to the base form by using stemming since the lemma of "best" and "better" is "good". In order to determine the base form a dictionary lookup is required.

[26] cf. `http://en.wikipedia.org/wiki/Stemming` (accessed on October 14^{th}, 2011)
[27] cf. `https://en.wikipedia.org/wiki/Lemmatization` (accessed on October 14^{th}, 2011)

2.4.2.5 Shallow parsing

Besides tokenization and part-of-speech tagging mature systems also require to identify the phrases the words belong to. This process is called *shallow parsing* or *chunking* since chunks of a sentence are produced. In this task a text is divided into a set of syntactically correlated parts of words. Theses parts are non-overlapping meaning that a word can only be member of one phrase (Tjong Kim Sang and Buchholz, 2000). Chunking is considered an intermediate step towards full parsing since only the constituents (noun groups, verb groups, adverb groups, etc.) are identified but neither the internal structure, nor the role in the main sentence are specified.

The following excerpt shows the result of shallow parsing on the basis of the sample sentence mentioned before:

```
(ROOT
  (S
    (NP (PRP I))
    (ADVP (RB really))
    (VP (VBP like)
      (NP (DT the) (JJ new) (NN phone)))
    (. .)
  )
)
```

The parsed text annotated with additional syntactic information is called a treebank. A treebank is usually displayed in a tree structure. Figure 2.6 shows the tree structure of the example sentence.

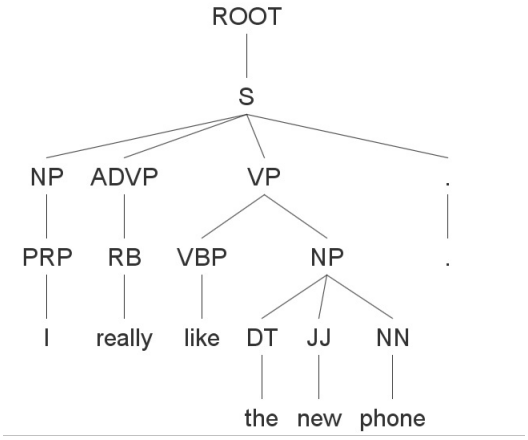


Figure 2.6 – Graphical representation of the treebank of the sentence "*I really like the new phone*".

2.4.2.6 Dependency Parsing

In contrast to chunking dependency parsing also allows to determine the internal structure of a sentence. Thus, a sentence is enriched with additional information allowing to represent the grammatical relationships (De Marneffe and Manning, 2008). A relationship is given as a triple with the two words and the type of relationship between them. For example $\langle subject, like, I \rangle$, with the meaning "the subject of *like* is *I*". The dependencies can be represented as a directed graph starting from a root node, leading to a tree structure similar to a treebank (De Marneffe and Manning, 2008). Figure 2.7 represents the typed dependencies and the resulting dependency tree of the sample sentence as produced by the Stanford parser[28]:

```
nsubj(like-2, I-1)
root(ROOT-0, like-2)
det(phone-5, the-3)
amod(phone-5, new-4)
dobj(like-2, phone-5)
```

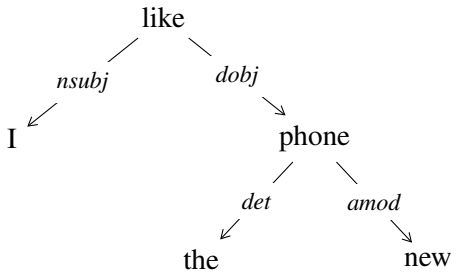


Figure 2.7 – Dependencies and resulting dependency tree of the sentence *"I like the new phone"*.

For a more complex sentence the following dependencies are created. The resulting dependency tree is illustrated in Figure 2.8.

"The new phone that was manufactured by Apple is really fast but it costs a lot."

```
det(phone-3, The-1)
amod(phone-3, new-2)
nsubj(fast-11, phone-3)
nsubjpass(manufactured-6, that-4)
auxpass(manufactured-6, was-5)
rcmod(phone-3, manufactured-6)
prep(manufactured-6, by-7)
pobj(by-7, Apple-8)
cop(fast-11, is-9)
advmod(fast-11, really-10)
cc(fast-11, but-12)
nsubj(costs-14, it-13)
conj(fast-11, costs-14)
det(lot-16, a-15)
dobj(costs-14, lot-16)
```

[28] cf. `http://nlp.stanford.edu:8080/parser/index.jsp` (accessed on October 23rd, 2011)

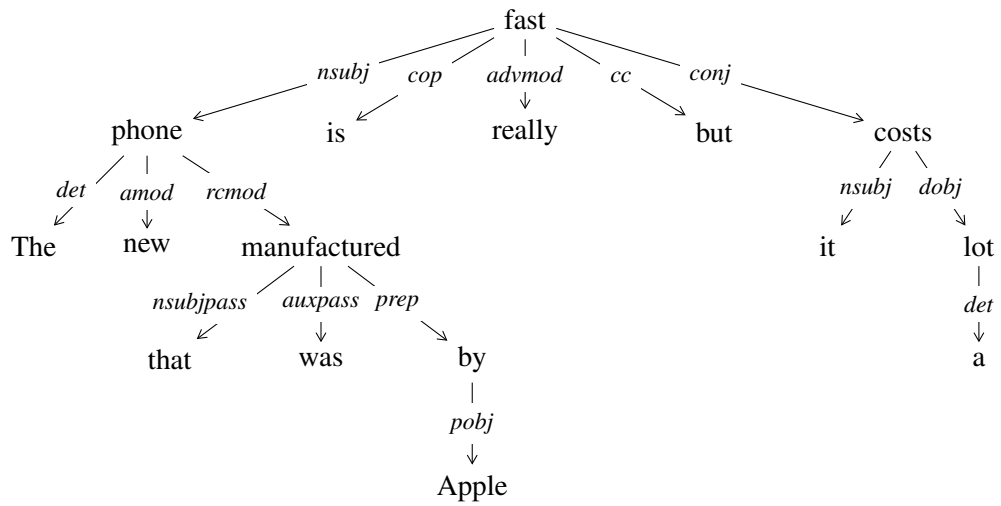


Figure 2.8 – Dependencies and resulting dependency tree of a more complex sentence: "*The new phone that was manufactured by Apple is really fast but it costs a lot.*

Note that the grammatical representation of sentences is a complex task itself. A detailed description of the different methods and their mathematical foundations is beyond the scope of this thesis. Nonetheless, some mathematical methods - especially probabilistic networks - used in POS-tagging and dependency parsing are also important for the actual opinion mining task. Some of these mathematical methods are presented in the next chapter of this thesis. However, for a comprehensive overview the interested reader may have a look at (Manning and Schütze, 1999; Toutanova and Manning, 2000; Toutanova et al., 2003) for POS-tagging and (De Marneffe and Manning, 2008) for dependency parsing. Other important information sources are the Penn Treebank project page[29] or the Stanford Parser project page[30].

2.4.3 Datasets

In order to create an opinion mining system it is often necessary to train a model. For training researchers use special training datasets. These are usually sets of manually labelled data, which are then applied to the learning algorithms. In addition, datasets to measure the quality of a system are needed. These manually labelled test sets are referred to as *gold standard*.

[29] `https://www.cis.upenn.edu/~treebank/` (accessed on September 23rd, 2011)

[30] `http://nlp.stanford.edu/software/lex-parser.shtml` (accessed on September 23rd, 2011)

2.4.3.1 Generating the datasets

For generating the training and test sets a text corpus is manually annotated with necessary information. If large data sets are available a part of the data is used to train the model, and the rest is used for testing it (Gupta, 2006). For this purpose a huge dataset is separated in several subsets used alternately for training and testing. The subsets should be obtained by random sampling to avoid having biases. Several methods exist for generating the datasets, some of which are presented in the following (Gupta, 2006):

1. **Holdout method:** The holdout method represents a straight forward approach for estimating accuracy. A dataset is divided into a set for training and a set for testing. Once the classifier is trained the test set is used to measure the accuracy. This method is subject to a trade-off. A bigger training set leads to a better classifier, while a bigger test set results in a better estimate of accuracy.

2. **k-fold cross validation:** This is a very common method and widely applied by researchers. The idea is to randomly separate a big data set in k subsets (k-fold) of the same size. One subset is used to test the model and the remaining k-1 subsets are used to train the classifier. This procedure is repeated k times, so that every subset is once used for testing. Finally, the accuracy of the model is determined by calculating the mean of the k runs. Values of five or ten folds are usually applied and turned out to be accurate.

3. **Leave-one-out cross validation:** This method is a special case of k-fold cross validation with k equal to N. Exactly one item is used to test the classifier while all other items are used for training. This is repeated N times, so that every item in the sample is used once for testing.

2.4.3.2 Examples of Important Datasets

To compare the results of different opinion mining systems they have to be executed on exactly the same data set. This requires the usage of the same documents and the same categories as well as the same split between training and test sets (Tang et al., 2009).

Following data corpora have been widely studied by researchers in the area of opinion mining and used to estimate accuracy of the different approaches and to compare them. The data collections were obtained from several different domains.

1. **Movie reviews:** So far a lot of research has be done on the movie domain. This domain is widely used because there are large online collections of movie reviews available (Internet Movie Database, Rotten Tomatoes). These review sites also support functionality to rate a movie based on a predefined scale (e.g. ten-star rating, in the case of the IMDD). This allows reviewers to summarize their overall sentiment with a machine-extractable rating indicator (Pang and Lee, 2008). Pang et al. (2002) created a dataset of movie reviews. The corpus consists of 700 negative and 700 positive reviews obtained from the IMDB. Pang and Lee (2004) proposed an extension to the original dataset. The new dataset (v2.0) contains 1000 positive and 1000 negative reviews, with a cap of 20 reviews per author

(312 authors total) per category. Another movie dataset focused on the sentence level was proposed by Pang and Lee (2005). This sentence polarity dataset covers 5,331 positive and 5,331 negative processed sentences.

All three datasets v1.0, v2.0 and the sentence polarity dataset are available at `http://www.cs.cornell.edu/people/pabo/movie-review-data/`.

The movie datasets from Pang and Lee have been used in a lot of studies. A list of works using the datasets can be found at `http://www.cs.cornell.edu/people/pabo/movie-review-data/otherexperiments.html`.

2. **Customer reviews:** Several datasets exist in the domain of product reviews. An important dataset was introduced by Hu and Liu (2004*a*,*b*). The dataset contains annotated customer reviews of five products. The reviews were obtained from amazon.com and ClNet. The dataset is available at `http://www.cs.uic.edu/~liub/FBS/CustomerReviewData.zip`. An additional review dataset containing reviews of nine products was used in work from Ding et al. (2008) and is available at `http://www.cs.uic.edu/~liub/FBS/Reviews-9-products.rar`.

   More information on these datasets and works, where they have been applied can be found at `http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html`.

   The Amazon Product Review Data presented by Jindal and Liu (2008) consists of five million product reviews from Amazon.com. The data contain information about reviewers, review text, ratings and product info. The data collection is available at `http://131.193.40.52/data/productinfo.rar`.

3. **MPQA-corpus:** Another important corpus of text documents is the Multi-Perspective Question Answering (MPQA) corpus that was originally collected and annotated as part of the summer 2002 ARDA-NRRC[31] Workshop. This corpus contains 530 news articles manually annotated using an annotation scheme for opinions and other forms of subjectivity.

   The original documents and their annotations are available at `http://nrrc.mitre.org/NRRC/publications.htm`

All these collections consist of reviews written in English. As of today, there are no labelled datasets publicly available containing German texts.

[31] Advanced Research and Development Activity - Northeast Regional Research Center; cf. `http://nrrc.mitre.org/index.htm` (accessed on October 1st, 2011)

CHAPTER 3

Related Work

As stated in section 2.1.2 the research field of sentiment analysis can be roughly divided into two research directions *sentiment classification* and *feature-based opinion mining*. The former is focused on determining the sentiment on document or sentence level, while the latter is concerned with sentiments on the phrase level. Sentiment classification uses rule based approaches or machine learning algorithms to classify documents or single sentences in respect to their polarity. In feature-based opinion mining a sentence is seen as a bag of words or phrases, where each phrase can express a different sentiment on different objects. This allows to set the opinionated words in context to the objects commented on in the document and to identify opinions on specific features of an object. Systems for feature-based opinion mining usually combine machine learning approaches with language patterns and rules.

3.1 Sentiment and Subjectivity Classification

Sentiment classification is probably the most widely studied topic in the research field of sentiment analysis (Liu, 2010). The idea of sentiment classification is to assign an opinionated document or sentence to a class depending on the overall sentiment.

In general there are two main types of approaches for sentiment classification problems. Symbolic techniques comprise a huge set of manually crafted lexicons and rules. These resources are used to categorize documents in respect to their polarity. The goal is to find relevant words, phrases and patterns in the text that match the rules. The second type are machine learning methods. Machine learning describes computer systems that automatically improve with experience based on algorithms and statistics. To some extent machine learning methods also use lexicons and rules. But these initial resources are automatically extended by the learning component of the system. Research showed that machine learning outperforms pure symbolic methods (Turney, 2002).

Two learning techniques can be distinguished:

- **Supervised learning**: The idea of supervised learning is to build a model based on a training set with already known classes. When a new object (in the case of sentiment classification a document or a sentence) is presented to the model, the classifier is used to assign the object to one of the existing classes in respect to the predictor features (Kotsiantis, 2007).
- **Unsupervised learning**: On the other hand, in unsupervised learning there are no predefined classes. Instead, the goal is to find patterns or hidden structures in the unlabelled input data, by grouping the items based on their similarity or dissimilarity. A typical example for unsupervised learning is clustering (Ghahramani, 2004).

Most existing techniques for sentiment classification use supervised learning but there are also some approaches, which use unsupervised learning techniques or a combination of both, so called semi-supervised learning techniques. In the following important supervised and unsupervised machine learning algorithms for both document and sentence-level sentiment classification are presented.

3.1.1 Sentiment Classification on Document Level

Document-level sentiment classification uses the document as the basic information unit. The goal is to classify each document in respect to its overall polarity. In document classification usually the two classes "positive" and "negative" are distinguished. Figure 3.1 depicts the basic functionality of a document-level sentiment classification system. Formally the task can be described as (Liu, 2010):

> **Task:** Given a set of documents D about a certain Object O the objective is to determine the polarity p of each opinionated document $d_j \in D$ and assign it to the correct class $c_i \in C$.

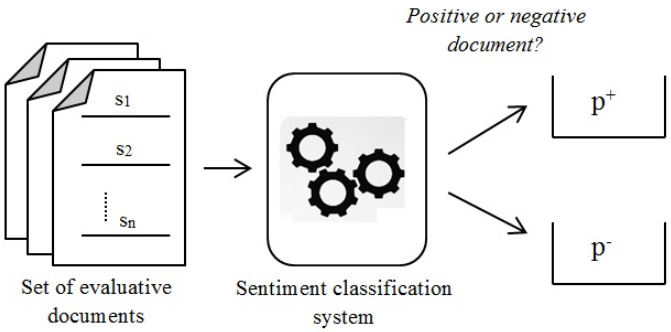


Figure 3.1 – Document-level sentiment classification

3.1.1.1 Supervised Machine Learning Methods

Naturally, sentiment classification can be formulated as a supervised learning problem with two class levels ("positive" and "negative") (Liu, 2010). The classifier is trained by using a set of already classified documents. Review sites often allow users to rate a product based on a predefined scale (e.g., "five star rating", "Thumb up vs. Thumb down", "Like" , etc.). These values can be used for training the classifier and consequently as a test set for determining the accuracy of the classifier.

Sentiment classification is similar to traditional fact-based or topic-based classification. Still, some minor differences exist. Fact-based algorithms classify documents into predefined topic classes, e.g., "politics", "sports", "science", etc. In contrast, sentiment classification is focused on determining the overall sentiment of a document. Topic related words are uninteresting, words that rather indicate positive or negative sentiment are important, e.g., "great", "excellent", "beautiful", "good", "awful", etc. (Liu, 2010). Nonetheless, the similarity between those two approaches allows to apply traditional supervised machine learning techniques like *Naïve Bayes*, *Support Vector Machines* (SVM) or *Maximum Entropy* to sentiment classification problems. Those three methods are presented on the following pages. Especially Maximum Entropy and Support Vector Machines represent the foundation of most classification systems and are implemented in many opinion mining applications. Improvements and extensions of these methods are still researched and widely used in modern approaches.

Naïve Bayes: Naïve Bayes is a supervised learning method that uses conditional probability to compute the probability of a hypothesis that an object belongs to a particular class (Gupta, 2006). The Naïve Bayes algorithm is based on the work by Thomas Bayes (1702 - 1761).

The conditional probability is the probability of an event A, given the occurrence of some other event B. Therefore $P(A|B)$ stands for the probability that event A will happen, given that event B has already happened. This is read as "the (conditional) probability of A, given B" or "the probability of A under the condition B". Conversely $P(B|A)$ is "the probability of B, given A". The conditional probability is calculated as:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}, \tag{3.1}$$

$$P(B|A) = \frac{P(B \cap A)}{P(A)}. \tag{3.2}$$

$P(A \cap B)$ is the probability that the two events A and B occur together. Since $P(A \cap B) = P(B \cap A)$ is valid, the Bayes' theorem can be deduced by dividing the first equation by the second:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}. \tag{3.3}$$

In opinion mining a set of documents D is given. Based on a fixed set of classes $C = \{c_1, c_2, \ldots, c_n\}$ a document $d \in D$ is assigned to the class that maximises the conditional probability. The classes are also called categories or labels (Manning et al., 2008) and are usually human defined (e.g., $C = 2$ and the classes are "recommended" and "not-recommended").

Using a training set T of labelled documents with their corresponding class $\langle d_t, c_t \rangle \in D \times C$ a classifier is trained. This Naïve Bayes classifier is used to assign a new document d to the class c_i that maximises the conditional probability $P(c_i|d)$. From Bayes' theorem the following equation can be derived:

$$P(c_i|d) = \frac{P(c_i)P(d|c_i)}{P(d)}, \tag{3.4}$$

where

- $P(c_i|d)$ is the probability of the document d belonging to class c_i
- $P(d|c_i)$ is the probability of obtaining d if known that it belongs to c_i
- $P(c_i)$ is the probability that a randomly picked document belongs to class c_i.
- $P(d)$ is the probability of obtaining a document whatever class it belongs to

In order to be able to calculate $P(c_i|d)$ the parameters $P(d|c_i)$, $P(c_i)$ and $P(d)$ need to be estimated. Since the denominator $P(d)$ is independent of c_i it is not required to be known. Instead it is sufficient to compute $P(c_i)$ and $P(d|c_i)$. To estimate $P(c_i)$ the maximum likelihood estimate (MLE) (Manning et al., 2008) is used, which is simply the relative frequency and corresponds to the probability of a document belonging to class c_i:

$$P(c_i) = \frac{N_{c_i}}{N}. \tag{3.5}$$

N_{c_i} being the number of documents in class c_i and N the total number of training documents. In the next step $P(d|c_i)$ has to be estimated. $(f_1, f_2, \ldots, f_m)$, is a set of predefined features[1] that appear in a document d and are part of the vocabulary V used to identify the class. m is the number of such features in d. To estimate $P(d|c_i)$ a naive approach is applied, assuming that all features $\{f_1, f_2, \ldots, f_m\}$ are conditionally independent:

$$P(d|c_i) = P(f_1, f_2, ..., f_m|c_i) = \prod_{j=1}^{m} P(f_j|c_i). \tag{3.6}$$

$P(f_j|c_i)$ is a measure of how much evidence f_j contributes that c_i is the correct class. The parameter $P(f_j|c_i)$ can be estimated as the relative frequency of f_j in documents belonging to class c_i:

$$P(f_j|c_i) = \frac{n_j(c_i) + 1}{\sum_{f \in V}(n(c_i) + 1)}. \tag{3.7}$$

[1]The features of machine learning are different from the object feature introduced earlier. In machine learning a feature is understood as a data attribute. Usually in the context of opinion mining a feature is a term.

$n_j(c_i)$ is the number of occurrences of feature f_j in training documents of class c_i. To avoid having a zero in case a feature-class combination does not occur in the training set $+1$ is added to each count. Consequently $P(f_j|c_i)$ can be written as:

$$P(f_j|c_i) = \frac{n_j(c_i) + 1}{\left(\sum_{f \in V} n(c_i)\right) + |V|}. \tag{3.8}$$

From 3.4 and 3.6 the following final equation can be derived,

$$P(c_i|d) = P(c_i) \left(\prod_{j=1}^{m} P(f_j|c_i) \right). \tag{3.9}$$

Note that $P(d)$ can be removed since it is independent from c_i and thus has no effect. A new document d is assigned to the best class, that is the most likely or maximum a posteriori (map) class c_{map}:

$$c_{map} = \underset{C}{\operatorname{argmax}}\, P(c_i) \left(\prod_{j=1}^{m} P(f_j|c_i) \right). \tag{3.10}$$

Example: For better understanding consider the following simplified example[2]. Given a training set of four classified documents $\{\langle d_1, rec\rangle, \langle d_2, rec\rangle, \langle d_3, rec\rangle, \langle d_4, \overline{rec}\rangle\}$ the goal is to classify a new document about a mobile phone into either one of the two classes $C = \{rec, \overline{rec}\}$. Table 3.1 shows the features and the values identified in the documents as well as the classes they were assigned to. The vocabulary is {good, bright, excellent, fast, slow, bad} and the presented document d_5 is given with {good, good, good, slow, bad}.

Table 3.1 – Table showing the features and classes of the training documents

DocID	Features in the documents	Class
1	good, bright, good	rec
2	good, good, excellent	rec
3	good, fast	rec
4	bad, slow, good	$\overline{rec}$

Three of the documents are assigned to class rec and one is classified as $\overline{rec}$. Hence the probability of a document belonging to class "recommended" is $P(rec) = 3/4$ while the probability of not belonging to this class is $P(\overline{rec}) = 1 - P(rec) = 1/4$. Table 3.2 shows the feature occurrences for both classes (rec and $\overline{rec}$).

[2]This example is adopted from Manning et al. (2008).

Table 3.2 – Number of occurrences of feature f_j in the classes rec and $\overline{rec}$

Feature	$n_j(rec)$	$n_j(\overline{rec})$
good	5	1
bright	1	0
excellent	1	0
fast	1	0
slow	0	1
bad	0	1
$\sum_{j=1} n_j(c_i)$	8	3

Using the data in the table the conditional probability $P(f_j|c_i)$ can be calculated. Note that it is only necessary to calculate the probability of those features that actually appear in d_5 (good, slow, bad).

$$\begin{aligned} P(good|rec) &= (5+1)/(8+6) = 3/7 \\ P(slow|rec) = P(bad|rec) &= (0+1)/(8+6) = 1/14 \\ P(good|\overline{rec}) &= (1+1)/(3+6) = 2/9 \\ P(slow|\overline{rec}) = P(bad|\overline{rec}) &= (1+1)/(3+6) = 2/9 \end{aligned}$$

Finally, $P(c_i|d) = P(c_i)\left(\prod_{j=1}^{m} P(f_j|c_i)\right)$ can be estimated for d_5 = {good, good, good, slow, bad}:

$$P(d_5|rec) = \frac{3}{4} \times \left(\frac{3}{7}\right)^3 \times \frac{1}{14} \times \frac{1}{14} \simeq 0.0003.$$

$$P(d_5|\overline{rec}) = \frac{1}{4} \times \left(\frac{2}{9}\right)^3 \times \frac{2}{9} \times \frac{2}{9} \simeq 0.0001.$$

Therefore, the classifier assigns the new document d_5 to the maximum a posteriori class $c_{map} = rec$. The reason for this classification decision is that the three occurrences of the positive indicator “good” in d_5 outweigh the occurrences of the two negative indicators “slow” and “bad”.

The Naïve Bayes method assumes that the terms in a document are independent and that a good training sample exists to train the model. But in practical use the independence assumption is always valid. There are lots of features that are correlated and words which occur together. Nonetheless, the Naïve Bayes method performs reasonably well and has important advantages that makes it a good algorithm for sentiment analysis. NB-based techniques are usually very simple and are known to be robust to noise and to be high dimensional (Zhang, 2004).

Naïve Bayes was originally used within text categorization problems, but has also been successfully applied in the sentiment classification domain. Pang et al. (2002) implemented different machine learning methods and tested them on a set of movie reviews they had obtained from the Internet Movie Database. Their dataset comprised 700 positive reviews and 700 negative reviews. A comparison of the results showed that the Naïve Bayes method scores good results for certain problem classes. Pang and Lee (2004) conducted an experiment on a bigger dataset of movie reviews (1000 positive, 1000 negative). Again they compared different machine learning approaches. In this setting the Naïve Bayes classifier achieved an accuracy of 86.4%. Because of its simplicity and good performance the Naïve Bayes algorithm is still a very important classifier and researchers use it in combination with other approaches.

Maximum Entropy: Another important supervised machine learning method is called Maximum Entropy (MaxEnt) classification. Like Naïve Bayes, MaxEnt uses probabilities to construct a stochastic model (Berger et al., 1996). This technique has been proven to be effective in a number of natural language processing applications and can outperform the Naive Bay in sentiment classification (Pang et al., 2002).

MaxEnt is a general technique for estimating probability distributions from data. Like in Naïve Bayes the objective is to estimates the conditional distribution $P(c|d)$ of the class label c given a new document d and to find the maximum a posteriori class c_{map}. In comparison to Naïve Bayes, MaxEnt does not assume statistical independence of the features that serve as predictors for the classes.

The first step of MaxEnt classification is to identify a set of features that will be useful for classification. In case of machine learning a feature is understood as an indicator function of properties of the input. Generally, a document is represented by a set of word count features (Nigam et al., 1999). F_j states if a specific feature appears in the document. Depending on the application a feature can be a single word, an n-gram or a phrase. For each feature its expected value over the training data is estimated (Nigam et al., 1999). The equation to calculate $P(c|d)$ is given with:

$$P(c|d) = \frac{1}{Z(d)} \exp\left(\sum_i \lambda_i F_i(d,c)\right), \tag{3.11}$$

where each $F_i(d,c)$ represents a *feature/class function* for the feature f_i and class c. λ_i is a weight parameter for the feature function. A higher value for the parameter indicates an important feature. $Z(d)$ is a normalized function to ensure a proper probability and is defined as:

$$Z(d) = \sum_c \exp\left(\sum_i \lambda_i F_i(d,c)\right). \tag{3.12}$$

For their Maximum Entropy classification system Pang et al. (2002) used word counts as features. Hence, $n_i(d)$ is the number of how often the feature f_i appears in document d. For

each word-class combination the binary feature function F_i is defined as:

$$F_i(d,c) = \begin{cases} 1 & \text{if } n_i(d) > 0 \text{ and } c' = c \\ 0 & \text{otherwise} \end{cases} \tag{3.13}$$

Thus, a feature function only has value one if feature f_i appears at least once in the document. For instance, a feature function only fires if and only if the bigram "very bad" appears in d.

The advantage of MaxEnt models is that they do not suffer from any independence assumptions. By using the feature-weight parameters λ_i the relevance of a feature f_i can be adjusted. Large values of λ_i indicate that f_i is considered a strong indicator for a class c and vice versa. Therefore, the effect of two words appearing together can be reduced. MaxEnt discounts the λ_i for each of these features in such a way that their weight towards classification is appropriately reduced by half (Nigam et al., 1999). A scaling algorithm to determine λ_i was postulated by Della Pietra et al. (1997) and applied by Pang et al. (2002) in their sentiment classification system.

Ahkter and Soria (2010) postulated an approach on mining Facebook[3] status messages by applying Maximum Entropy classifiers on both binary and multi-class sentiment labelling. Experiments showed that a MaxEnt classifier augmented with part of speech data performs the best, achieving an average binary classification score of approximately 85% and an average multi-class score of approximately 67%.

Support Vector Machines: Support Vector Machines (SVM) are supervised learning methods used to analyse data and recognize patterns. The SVM algorithm was first invented by Vapnik (1979) and published by Cortes and Vapnik (1995). An improvement was postulated by Vapnik in 1998.

A SVM is a non-probabilistic linear classification method that uses a kernel function, which can be seen as a similarity measure (Vapnik, 1998). Given a set of already classified instances kernel methods determine the class of a novel instance by comparing it to the classified training instances using this kernel function. The kernel function of SVMs is based on the margin with which they separate the data and not on the number of features (Joachims, 1998). The fundamental idea is to find a hyperplane[4] that separates the documents in one class from the other. A good separation is achieved by the hyperplane that has the largest total margin possible. Figure 3.2 depicts the basic concept of the Support Vector Machines in the two-dimensional space.

The hyperplane H_3 does not separate the two classes at all. Hyperplane H_2 separates the classes but with just a small margin. H_1 represents the maximum-margin hyperplane having the largest possible margin.

A data point (document) in Support Vector Machines is viewed as an n-dimensional real vector $\vec{d} \in \mathbb{R}^n$. Each document $\mathbf{d}$[5] belongs to a class $c \in \{1, -1\}$ (corresponding to positive consequently negative). Given a training set with m documents together with their classes, $T = \{(\mathbf{d_1}, c_1), (\mathbf{d_2}, c_2), \ldots, (\mathbf{d_m}, c_m)\}$, the goal is to find the *maximum-margin hyperplane* H that divides the documents having $c_i = 1$ from those having $c_i = -1$ (Abe, 2010).

[3] `http://www.facebook.com` (accessed on September 5^{th}, 2011)

[4] A hyperplane is a concept in geometry. It is a generalization of the plane into a different number of dimensions.

[5] Instead of $\vec{x}$ an n-dimensional vector of the form $\vec{x} = (x_1, x_2, \ldots, x_n)$ can also be written as $\mathbf{x}$.

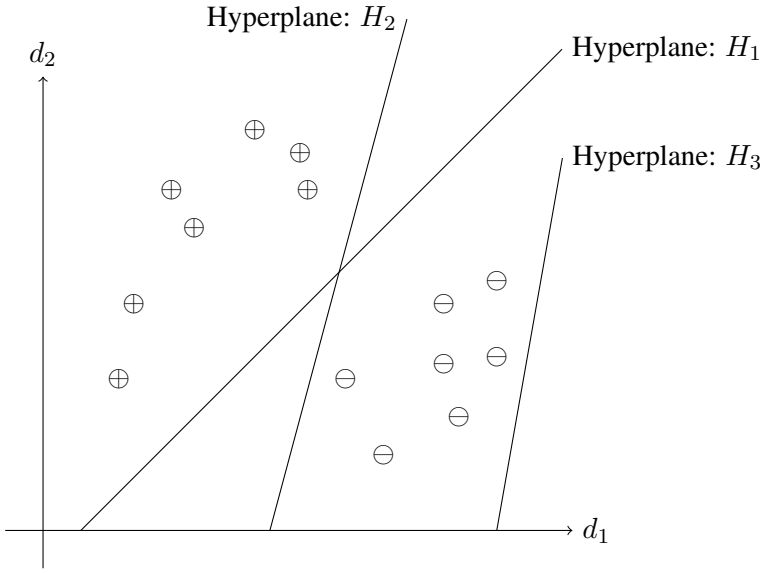


Figure 3.2 – Concept of Support Vector Machines (Abe, 2010)

The hyperplane H can be written as the set of points $\mathbf{w}$ satisfying the condition:

$$\mathbf{w} \cdot \mathbf{d_i} + b = 0, \tag{3.14}$$

where b represents the offset of the origin and $\cdot$ is the dot product of the vectors $\mathbf{w}$ and $\mathbf{d_i}$. The dot product is defined as:

$$\mathbf{w} \cdot \mathbf{d} = \sum_{i=1}^{n} w_i d_i. \tag{3.15}$$

The objective is to choose $\mathbf{w}$ and b to maximize the margin, or distance between the parallel hyperplanes that are as far apart as possible while still separating the data. These hyperplanes can be described by the equations:

$$H_1 = \mathbf{w} \cdot \mathbf{d_i} + b = +1, \tag{3.16}$$

$$H_2 = \mathbf{w} \cdot \mathbf{d_i} + b = -1. \tag{3.17}$$

To prevent any data point to fall into the margin, the following constraints need to be defined:

$$\begin{cases} \mathbf{w} \cdot \mathbf{d_i} + b \geq 1 & \text{if } c_i = +1 \\ \mathbf{w} \cdot \mathbf{d_i} + b \leq -1 & \text{if } c_i = -1 \end{cases} \tag{3.18}$$

Figure 3.3 illustrates the separating hyperplanes and the classification constraints.

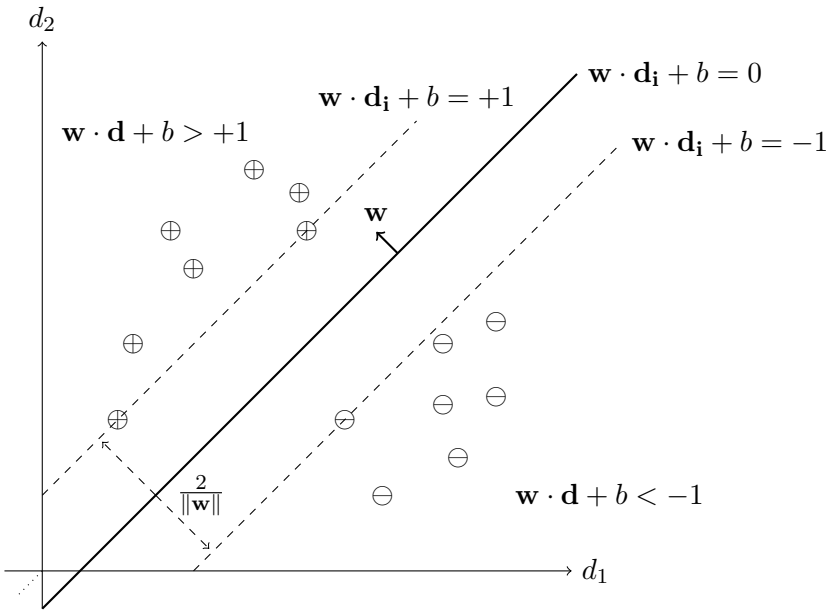


Figure 3.3 – The separating hyperplanes (Abe, 2010)

By transformation the constraints given in Equation 3.18 can be expressed in compact form:

$$c_i(\mathbf{w} \cdot \mathbf{d_i} + b) \geq 1. \tag{3.19}$$

By applying these constraints the distance between the two hyperplanes H_1 and H_2 can now be calculated. Therefore, consider d_0 to be a point on H and d_1 to be a point on H_1. This assumption leads to following equations:

$$\begin{cases} \mathbf{w} \cdot \mathbf{d_0} + b = 0 \\ \mathbf{w} \cdot \mathbf{d_1} + b = 1 \end{cases} \tag{3.20}$$

Subtracting the first equality from the second produces:

$$\mathbf{w} \cdot (\mathbf{d_1} - \mathbf{d_0}) = 1. \tag{3.21}$$

Since vector $\mathbf{w}$ and $(\mathbf{d_1} - \mathbf{d_0})$ are orthogonal to the hyperplanes H_1 and H the equation 3.21 can be written as:

$$|\mathbf{w} \cdot (\mathbf{d_1} - \mathbf{d_0})| = \|\mathbf{w}\| \times \|\mathbf{d_1} - \mathbf{d_0}\| = 1. \tag{3.22}$$

Thus, the distance between H_1 and H is defined as:

$$\|\mathbf{d_1} - \mathbf{d_0}\| = \frac{1}{\|\mathbf{w}\|}. \tag{3.23}$$

Since this is also valid for the assumption that d_2 is a point on H_2 and d a point on H the total margin between H_1 and H_2 can be written as:

$$2 \times \frac{1}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|}. \tag{3.24}$$

The objective of Support Vector Machines is to maximise the distance between H_1 and H_2, which is $\frac{2}{\|\mathbf{w}\|}$. This is equivalent to minimising $\frac{1}{2}\|\mathbf{w}\|^2$ (Vapnik, 1998). The problem can be solved by quadratic programming techniques.

The vector $\mathbf{w}$ that defines the maximum-margin hyperplane can be expressed by terms of linear combination of the training vectors as:

$$\mathbf{w} = \sum_{i}^{m} \alpha_i c_i \mathbf{d_i}, \tag{3.25}$$

where $\alpha_i \geq 0$ is a parameter, indicating the relevance of the points. Only the data points d_i that lie on the margin and satisfy the condition $c_i(\mathbf{w} \cdot \mathbf{d_i} + b) = 1$ are important.

Once the maximum-margin hyperplane is trained it is possible to predict the membership of new documents. The class of a new document only depends on the sign of the expression $\mathbf{w} \cdot \mathbf{d_k} + b$. The unknown document $\mathbf{d_k}$ is classified into:

$$\begin{cases} c_k = +1 & \text{if} \quad \mathbf{w} \cdot \mathbf{d_k} + b > 0 \\ c_k = -1 & \text{if} \quad \mathbf{w} \cdot \mathbf{d_k} + b < 0 \end{cases} \tag{3.26}$$

Support Vector Machines have great importance in sentiment analysis. Pang et al. (2002) used SVM to classify movie reviews they obtained from the IMDB. Comparisons with Naïve Bayes and MaxEnt showed that in general SVM scores the best results. Using word unigrams as features lead to an accuracy of 82.7%. The employment of other features lead to similar good results. Mullen and Collier (2004) presented an approach using word bigrams as features. Experiments by Pang and Lee (2004) on a larger dataset of movie reviews lead to even better results using SVM. Matsumoto et al. (2005) conducted a study on both movie datasets. Their results showed an accuracy of 87.3% on the first dataset and even 92.9% on the second dataset. A novel approach postulated by Kennedy and Inkpen (2006) combines contextual valence shifters together with SVM. Furthermore researchers also applied SVM to product reviews and customer feedback data (Gamon, 2004).

So far, the discussed methods only support binary classification (the documents are assigned to either one of two classes: positive or negative). In their paper Pang and Lee (2005) addressed the *rating-inference problem*. In this case a document is classified with respect to a multi-point scale (e.g., one to five stars). A set of training documents belonging to N different classes is given and the goal is to train a classifier that correctly predicts the class to which a new document belongs. On a dataset of Internet movie reviews Pang and Lee (2005) examined different approaches, all based on SVM. (1) *One-vs-All* (OVA) is an extension to the SVM for the n-ary case. The idea is to train N different binary classifiers. Each one is trained in a way to distinguish the samples in one class from the samples in all the other classes. In order to classify a new document all N classifiers are run and the classifier that produces the largest value is

chosen (Rifkin and Klautau, 2004). (2) *Support Vector Regression* (SVR), treats neutrals as intermediate between positives and negatives with a threshold ε. Thus, the new documents are classified as positive if $\mathbf{w} \cdot \mathbf{d_k} + b > +\varepsilon$ and consequently as negative if $\mathbf{w} \cdot \mathbf{d_k} + b < -\varepsilon$ (Rifkin and Klautau, 2004). All those documents that lie in between the interval $[+\varepsilon, -\varepsilon]$ are neutrals. In an additional paper Koppel and Schler (2006) introduced the neutral class. They used two different types of labelled corpora to train and test their model. The TV corpus contains posts of chat groups devoted to popular U.S. television shows. The second corpus consists of product evaluations obtained from shopping.com[6]. Their research showed that linear regression models yield better accuracy for a 3-point rating scale than OVA SVMs.

3.1.1.2 Unsupervised Machine Learning

Contrary to supervised machine learning, unsupervised approaches do not require any human labelled training data. All the required data is directly found in the document. Thus all the data is free from noise and once the unsupervised system is set up it can be immediately used without labelling training data. Unsupervised methods are often combined with supervised learning methods. Using an unsupervised system an initial data set is generated (word lists, opinion lexicon), which is then applied to a supervised learning system. The combination of supervised and unsupervised methods is referred to as semi-supervised learning.

Pointwise Mutual Information: Liu (2010) and Tang et al. (2009) present an unsupervised machine learning method that was originally postulated by Turney (2002). This method is based on pointwise mutual information (PMI) to classify reviews as either *recommended* or *not recommended*. The method comprises three steps.

In a first step the algorithm extracts phrases that contain adjectives or adverbs. By using a Part-of-Speech (POS) tagger two consecutive words are extracted if their tags conform to the rules depicted in Table 3.3. The JJ tags indicate adjectives, the NN tags are nouns, the RB tags represent adverbs, and the VB tags stand for verbs.

Table 3.3 – Patterns of tags for extracting two-word phrases from reviews (Turney, 2002)

	First word	**Second word**	**Third word** (Not extracted)
1.	JJ	NN or NNS	anything
2.	RB, RBR or RBS	JJ	not NN nor NNS
3.	JJ	JJ	not NN nor NNS
4.	NN or NNS	JJ or NNS	not NN nor NNS
5.	RB, RBR or RBS	VB, VBD, VBN or VBG	anything

[6] `www.shopping.com` (accessed on September 7^{th}, 2011)

The first pattern defines that two consecutive words are extracted if the first word is an adjective and the second word is a noun. Consequently, the second rule states that the first word needs to be an adverb and the second word an adjective, but the third word cannot be a noun. Consider the following sentence, "*This mobile phone produces beautiful pictures*". In this case "beautiful pictures" will be extracted since it fulfils the first rule.

Once the phrases are extracted the semantic orientation (polarity) has to be estimated. This is achieved by using the *Pointwise Mutual Information* (PMI) measure, which is defined as follows:

$$\text{PMI}(w_1, w_2) = \log_2 \left(\frac{P(w_1, w_2)}{P(w_1)P(w_2)} \right), \tag{3.27}$$

where w_1 and w_2 are words or phrases that can appear in a text. $P(w_1, w_2)$ is the probability that w_1 and w_2 occur together. $P(w_1)P(w_2)$ gives the probability that the two words co-occur if they are statistically independent. Thus, the ratio between $P(w_1, w_2)$ and $P(w_1)P(w_2)$ is a measure of the degree of the statistical dependence between the words (Turney, 2002).

The polarity of a new phrase $p(phrase)$ is calculated by using the PMI measure in respect to the positive indicator word "excellent" and the negative indicator word "poor":

$$p(phrase) = \text{PMI}(phrase, \text{“excellent”}) - \text{PMI}(phrase, \text{“poor”}). \tag{3.28}$$

The reference words "excellent" and "poor" were chosen because in the five star review rating system it is common to define one star as "poor" and five stars as "excellent" (Turney, 2002).

The Pointwise Mutual Information is usually calculated by issuing queries to search engines (Liu, 2010). The search engine Altavista[7] provides the NEAR operator that only finds documents where those words appear in a maximum distance of ten words, in either order. For example, the search expression "phone NEAR display" only retrieves documents where the words "phone" and "display" co-occur within ten words.

Let "hits(*query*)" be the number of hits returned given the query *query*. From 3.27 and 3.28 the following equation for the polarity $p(phrase)$ of a phrase can be derived:

$$p(phrase) = \log_2 \left(\frac{\text{hits}(phrase \text{ NEAR “excellent”}) \times \text{hits}(\text{“poor”})}{\text{hits}(phrase \text{ NEAR “poor”})\text{hits} \times (\text{“excellent”})} \right). \tag{3.29}$$

Finally, in the last step the algorithm computes the average polarity $\bar{p}$ over all the phrases in the document and classifies the document in respect to a positive or negative average:

$$\bar{p} = \frac{1}{N} \sum_{i}^{N} p(phrase_i). \tag{3.30}$$

On a test corpus of 410 documents the unsupervised classification algorithm of Turney (2002) scored an average accuracy of 74%. Experiments in the domain of movie reviews yielded an accuracy of only 66%.

[7] `http://www.altavista.com` (accessed on September 8^{th}, 2011)

Turney (2002) explained this result on account of the structure of movie reviews. The sum of all the single review parts does not necessarily represent the overall polarity of the review. In other domains (bank and automotive) the algorithm gained significantly better results of more than 80% accuracy.

An extension to this algorithm was postulated by Gamon and Aue (2005). In addition to the assumption of Turney (2002) that sentiment terms of similar orientation tend to co-occur in documents, they added the second assumption that sentiment terms of opposite orientation do not tend to co-occur at the sentence level. In their approach, instead of regarding the document as a whole, each sentence is contemplated as one information item and analysed on its own. This method yielded significant better results than the analysis on document level did.

3.1.2 Sentiment Classification on Sentence Level

The disadvantage of sentiment classification on document level is that it only leads to coarse results. Sentence-level sentiment classification overcomes this drawback by considering the sentence as the basic information unit. Instead of analysing the whole document as one, each sentence is analysed on its own. As well as in document-level classification the sentence-level sentiment classification does not support the identification of object features that have been commented on (Bhuiyan et al., 2009).

The basic functionality of sentence-level sentiment classification is depicted in Figure 3.4. Starting with a set of evaluative documents each document is split into its single sentences. The sentences are then analysed if they express subjectivity. This step is referred to as *subjectivity detection*. Finally, in the last step the orientation of the subjective sentences is determined.

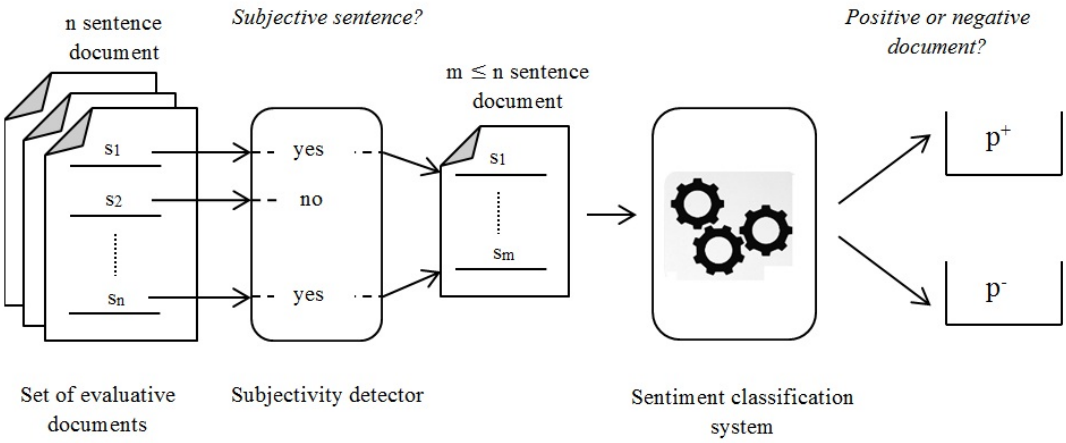


Figure 3.4 – Sentence-level sentiment classification (Pang and Lee, 2004)

Task: Sentence-level sentiment detection comprises two subtasks:

- **Subjectivity detection:** First of all, it is necessary to determine if a sentence actually contains sentiment. A document d consists of a total number of sentences $S = \{s_1, s_2, \ldots, s_n\}$. Some sentences express subjective information S_s and other sentences express factual or objective information S_o (Tang et al., 2009). In the first subtask it is necessary to analyse whether sentence s_i represents subjective information.
- **Sentence-level sentiment classification**: Once the subjective sentences is identified, the next task analyses if those sentences S_s express a positive or negative sentiment.

Both tasks can be seen as classification problems. Thus, traditional machine learning approaches, some of which were already discussed in *Section 3.1.1* can be applied, e.g., Naïve Bayes (Wiebe and Riloff, 2005; Gamon et al., 2005) or Maximum Entropy (Kim and Hovy, 2006*a*). New approaches combine different machine learning algorithms (Wang and Liu, 2011) or use graph representation to model the sentence granularity (Pang and Lee, 2004).

Cut-based classification Pang and Lee (2004) proposed a framework integrating sentence-level subjectivity detection with document-level sentiment polarity. They used a graph-based formulation for finding *minimum cuts*.

Given a set of n items $\{x_1, x_2, \ldots, x_n\}$ the objective is to assign each item x_i to either class c_1 or c_2 depending on the item's total score. Each item x_i has an (1) *individual score:* $ind_j(x_i)$ representing an estimate for the preference of x_i being in c_j and an (2) *association score:* $assoc(x_i, x_k)$, an estimate of how important it is that x_i and x_k belong to the same class. In order to find the correct class the score of each item x_i needs to be calculated by subtracting its individual score of being in one class from the individual score of being in the other class. In addition, the association score of the items classified into different classes also needs to be subtracted. An item x_i is assigned to the class having the maximum score. Consequently, this leads to the optimization problem to assign x_i to c_1 and c_2 so as to minimize the partition costs:

$$\sum_{x \in c_1} ind_2(x) + \sum_{x \in c_2} ind_1(x) + \sum_{\substack{x_i \in c_1 \\ x_k \in c_2}} assoc(x_i, x_k). \tag{3.31}$$

In order to solve this optimization problem Pang and Lee (2004) proposed to portray it as an undirected graph $\mathcal{G}$ with vertices $(v_1, v_2, \ldots, v_n, s, t)$, where s and t respectively are the *source* and the *sink*. In addition n edges of the form (s, v_i), each having a weight $ind_1(x_i)$ and n edges (v_i, t), each with weight $ind_2(x_i)$ are added. Finally, $\binom{n}{2}$ edges (v_i, v_k), each with weight $assoc(x_i, x_k)$ are added. The graphical representation of this graph formulation is illustrated in Figure 3.5.

As the name of the algorithm states, the goal is to find the *minimum cut*. A cut (S, T) of $\mathcal{G}$ is a partition of its nodes into sets $S = \{s\} \cup S'$ and $T = \{t\} \cup T'$, where $s \notin S'$ and $t \notin T'$. Its cost $cost(S, T)$ is the sum of the weights of all edges crossing from S to T. A *minimum cut* of $\mathcal{G}$ is one of minimum cost (Pang and Lee, 2004).

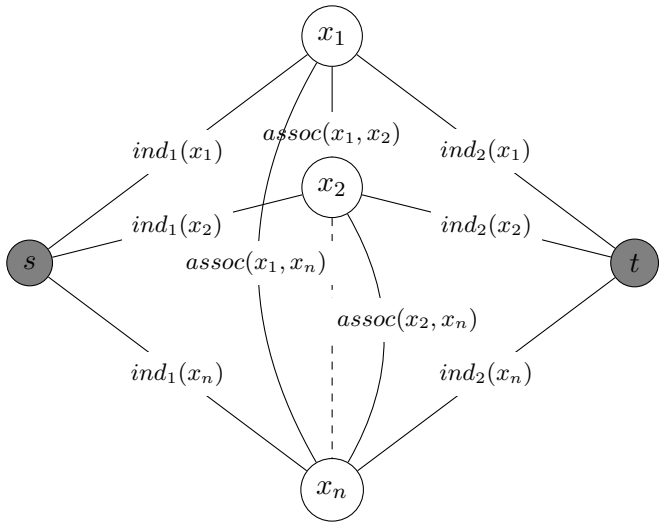


Figure 3.5 – Graph representation of the cut-based optimization problem (Pang and Lee, 2004)

Example: For clarification consider the following graph $\mathcal{G}$ containing the three nodes (X, Y, Z) as shown in Figure 3.6.

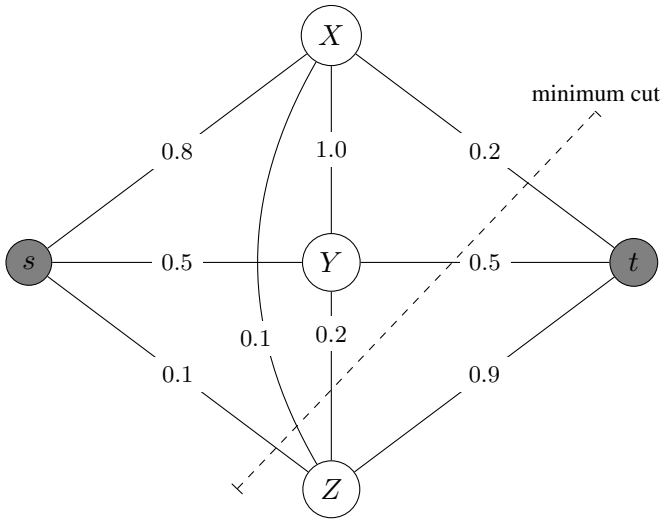


Figure 3.6 – Graph for classifying three items (Pang and Lee, 2004)

Table 3.4 lists the total costs (individual costs + association costs) of each possible separation. For X,Y belonging to class c_1 and Z being in class c_2 the total costs are the lowest. Thus, the minimum-cut, indicated by the dashed line, puts X,Y in c_1 and Z in c_2.

Table 3.4 – Values for determining the minimum-cut (Pang and Lee, 2004)

c_1	c_2	**Individual penalty**	**Association penalty**	**Cost**
X,Y	Z	$0.2 + 0.5 + 0.1$	$0.2 + 0.1$	1.1
-	X,Y,Z	$0.8 + 0.5 + 0.1$	0	1.4
X,Y,Z	-	$0.2 + 0.5 + 0.9$	0	1.6
X	Y,Z	$0.2 + 0.5 + 0.1$	$1.0 + 0.1$	1.9
Z	X,Y	$0.8 + 0.5 + 0.9$	$0.1 + 0.2$	2.5
Y	X,Z	$0.8 + 0.5 + 0.1$	$1.0 + 0.2$	2.6
X,Z	Y	$0.2 + 0.5 + 0.9$	$1.0 + 0.2$	2.8
Y,Z	X	$0.8 + 0.5 + 0.9$	$1.0 + 0.1$	3.2

Calibrated Expectation Maximization: Calibrated Expectation Maximization is a mature semi-supervised classification method combining Naïve Bayes and Expectation Maximization (EM). The idea is to train a Naïve Bayes classifier via the EM algorithm. This approach was originally postulated by Nigam et al. (2000) for applications on the document level.

Given a set of labelled data the Naïve Bayes classifier is trained by estimating the parameters from the labelled data. Then, the Expectation Maximization algorithm is used to improve the Naïve Bayes classifier. EM is an iterative algorithm for maximum likelihood estimations. It comprises two steps (E-step and M-step). In the E-step the current classifier is used to classify the unlabelled examples. In the M-step the parameters of the original classifier are updated based on the calculations of the E-step. These steps are repeated until convergence is reached (Nigam et al., 2000; Tsuruoka and Tsujii, 2003).

A new study on sentiment detection on sentence level was presented on this year's *Workshop on Computational Approaches to Subjectivity and Sentiment Analysis* by Wang and Liu (2011). Their proposed system utilizes Calibrated Expectation Maximization to classify single sentences. In a first step a subjectivity lexicon is used to create an initial training set. Each entry in the lexicon is categorized into strong and weak subjective clues. For each word w_j in s the subjectivity score $sub(w_k)$ is determined. 1 indicating a very strong subjectivity clue, 0.5 is used for very weak clues and 0 for every other word. Consequently the subjectivity score of the sentences is calculated as the sum of the subjectivity scores of all words of the sentence, normalized by the sentence length:

$$sub(s) = \sum_{w \in s} \frac{sub(w)}{N_s}, \tag{3.32}$$

where N_s is the number of words in the sentence s, thus the length of the sentence.

The top m sentences with the highest subjectivity scores are labelled as SUBJECTIVE and the top m sentences with the highest objectivity scores are labelled OBJECTIVE. These $2m$ sentences are then used to train the iterative two step classification model.

In the E-step the Naïve Bayes classifier is used to calculate the posteriori conditional probability that a sentence s_i belongs to class $c_j \in \{\text{SUBJECTIVE } (sub), \text{OBJECTIVE } (obj)\}$:

$$P(c_j|s_i) = \frac{P(c_j)\prod_{k=1}^{N_i} P(w_k|c_j)}{\sum_{c_l \in C} P(c_l)\prod_{k=1}^{N_i} P(w_k|c_l)}, \tag{3.33}$$

The M-step updates the parameters of the model. Therefore it uses the probabilistic results from the E-step to recalculate the parameters of the Naïve Bayes classifier, the probability of word w_t being in class c_j and the prior probability of class c_j:

$$P(w_t|c_j) = \frac{0.1 + \sum_{s_i \in S} N(w_t, s_i)P(c_j|s_i)}{0.1 \times |V| + \sum_{k=1}^{|V|}\sum_{s_i \in S} N(w_k, s_i)P(c_j|s_i)} \tag{3.34}$$

and

$$P(c_j) = \frac{0.1 + \sum_{s_i \in S} P(c_j|s_i)}{0.1 \times |C| + |S|}, \tag{3.35}$$

where $N(w_t, s_i)$ is the count of word w_t in the sentence s. S is the set of sentences, $|C|$ the number of classes and $|V|$ is the total size of the vocabulary from the entire data set. For smoothing $\alpha = 0.1$ is added to the numerator and denominator of the equations.

In the first iteration $P(c_j|s_i)$ is estimated by using the training data. If a sentence is labelled SUBJECTIVE, then $P(sub|s_i)$ is 1 and $P(obj|s_i)$ is 0. Vice versa if a sentence is labelled OBJECTIVE, then $P(sub|s_i)$ is 0 and $P(obj|s_i)$ is 1.

Wang and Liu (2011) applied their method on three different data sets: (1) Movie reviews from the IMDB, (2) articles form the MPQA Corpus and (3) meeting protocols from the AMI corpus. Their method scored good results on movie reviews. Using a seed of 2,000 self-labelled samples an accuracy of 90.15% could be reached. For the AMI and MPQA the system did not score such good results. The Calibrated Expectation Maximization method needed more iterations and only scored around 58% and 70% for these data sets. Wang and Liu (2011) explain these results due to the structure of the documents in those two datasets. The most highly ranked sentences according to the subjective lexicon are not the most subjective sentences.

Pulse Based on the idea of using a Naïve Bayes classifier trained by Expectation Maximization Gamon et al. (2005) developed a system (*Pulse*) to analyse a set of 406,818 customer car reviews on the sentence level. Pulse combines a clustering technique with a supervised machine learned sentiment classifier. In a first step Pulse creates a taxonomy of major categories (cars) and minor categories (models of cars). The sentences are then extracted from the reviews for each car and model. In the next step a clustering algorithm is applied for finding salient patterns in the sentences. Based on a small set of labelled data D_L the Naïve Bayes classifier is trained and by Expectation Maximization extended to the large set of unlabelled data D_U. In total, the car review dataset contains almost 900,000 sentences. The sentences are classified as either being "positive", "negative" or "other". The latter category includes sentences with no sentiment.

3.2 Feature-based Sentiment Analysis

Analysing texts on the document or the sentence-level does not reveal what the author likes or dislikes about an object. It only allows to identify the overall sentiment and its polarity. A positive document does not necessarily mean that the opinion holder is happy with all aspects or features of the product. Likewise, a negative document does not mean that the author dislikes everything (Bhuiyan et al., 2009; Liu, 2010). In a typical review the author writes about the positive and negative aspects of the object. In order to analyse this detailed information it is necessary to move to the phrase level. Based on the model of an opinion, as presented in Section 2.1.1 the task of feature-based sentiment analysis can be defined as:

> **Task:** Given a set of evaluative text documents D, each document $d \in D$ containing opinions about an object o, feature-based opinion mining aims to extract a number of features $E \subseteq F$ of the object that has been commented on in d by an opinion holder h and to determine whether the comments on a feature $f_k \in E$ have a positive p^+, negative p^- or neutral p^o polarity.

Given the task description from above six steps of feature-based sentiment analysis can be determined (Liu, 2010):

1. **Identify the opinion holder:** The opinion holder is someone who expresses a certain opinion on an object. In the field of reviews the opinion holder usually is the author of the text. It is also possible that different opinion holders appear in a text. This is very common in news paper articles in which the author cites someone else.

2. **Determine the object:** In a normal review the object is defined in the title. But it is also possible that several objects are commented on in a text. This is especially the case when considering comparative product reviews, comparing the features of two or more products.

3. **Identify the object features:** The goal of this subtask is to find the features of an object that have been commented on in the text.

4. **Identify opinions regarding the object features:** Once the object features are identified the next step is to find opinions expressed on the features.

5. **Determine the polarity:** Determine whether the opinions on the features are positive, negative or neutral.

6. **Summarize the discovered information:** In the last subtask the information is prepared and presented to the user in a legible way.

Based on these steps Figure 3.7 depicts the possible architecture of a feature-based sentiment analysis system as presented by Hu and Liu (2004*a*). A set of documents about a certain topic are collected by crawling certain review sites. A POS-tagger is used to create the tags for all words of each document. The tagged documents are then used to find the named entities as well

as the frequent features. By comparing adjectives that are close to the frequent features to the entries in an opinion lexicon the relevant opinion words are identified. In a next step the opinion words are used to identify those infrequent features that are are not frequently commented on. Finally, the three resources (named entities, features and opinion words) created in the previous stages are used for determining the polarity and strength of each feature.

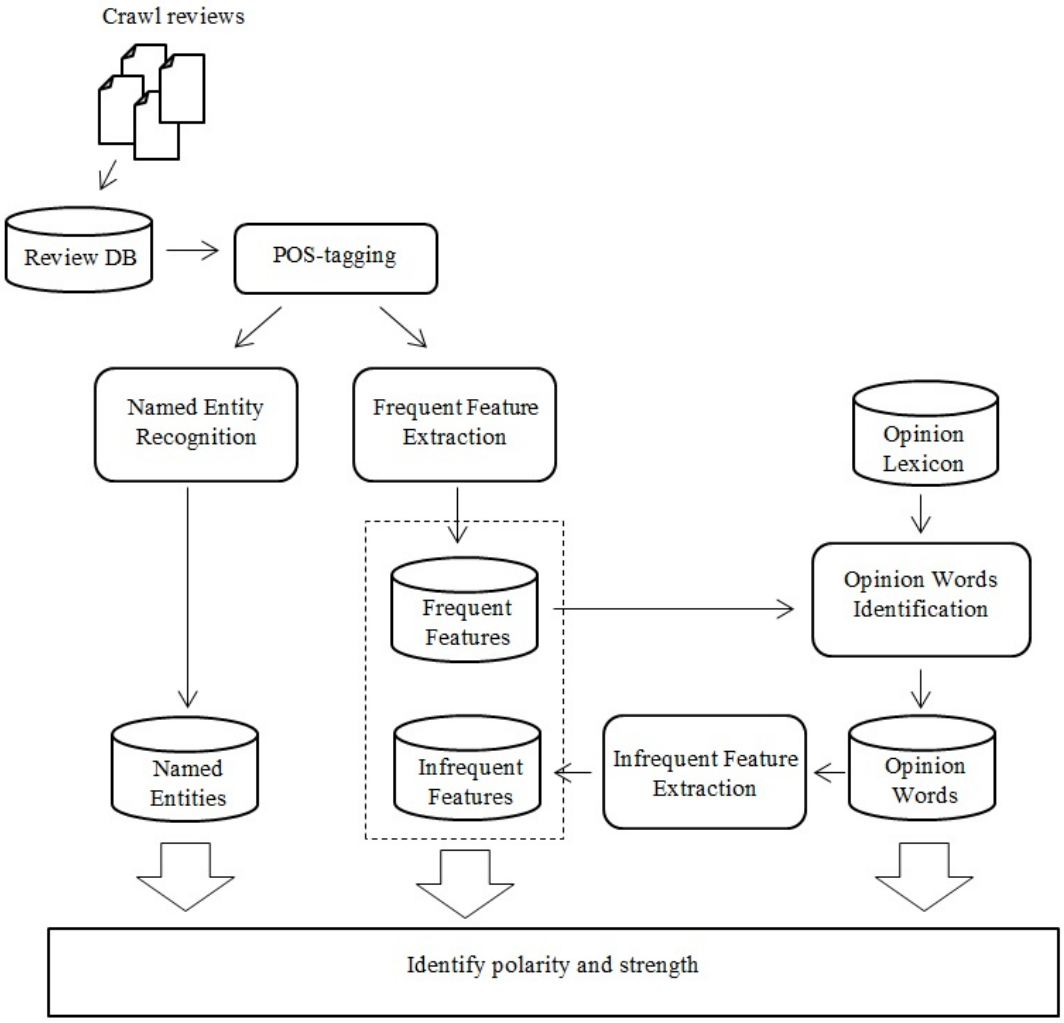


Figure 3.7 – Architecture of a feature-based sentiment analysis system (Hu and Liu, 2004*a*)

3.2.1 Named Entity Recognition

Generally, the goal of Named Entity Recognition (NER) is to find atomic concepts in a text, like names of persons, organizations, locations, quantities or products. If NER is applied in the domain of opinion mining it is mostly dedicated to identifying sources of opinions. In general it is necessary to distinguish between direct and indirect sources. The following two sentences show an example for a direct and an indirect source.

s_1: "*According to the report, the iPhone has a bright display.*"
s_2: "*Tom said that the report about the new iPhone was positive.*"

In the sentence s_1 the "report" is a direct source focusing on the quality of the phone's display. The similar sentence s_2 denotes an indirect (second hand, third hand, etc.) source "Tom" on an opinion, whose direct source is "report".

Several approaches have been postulated to identify and extract named entities in texts. They use grammar-based techniques (set of rules) as well as statistical models (Conditional Random Fields, Maximum Entropy, etc.). These NER approaches can also be applied in sentiment analysis to find sources of opinions; cf. Choi et al. (2005); Kim and Hovy (2006*b*); Choi et al. (2006). However, most research in the area of feature-based sentiment analysis is focused on object feature extraction (Liu, 2010). Especially for analysing product reviews the task of extracting opinion holders from the actual text is not so important. Usually, review sites only allow registered users to write comments. Hence, the opinion holder as the author of the review can be easily identified by the user's ID or the login name. The product the review is about is also known, since it is stated in the review's title.

3.2.2 Feature Extraction

The most crucial task in feature-based sentiment analysis is to extract those product features from the text that the author commented on in the review. There are different formats on how a review can be written. In their work Liu et al. (2005) proposed a differentiation of review formats in three main types.

1. **Format 1** - Pros and Cons: The reviewer is asked to describe Pros and Cons separately.

2. **Format 2** - Free format: The reviewer can write freely. There is no explicit separation of Pros and Cons.

3. **Format 3** - Pros, Cons and detailed review: The reviewer is asked to describe Pros and Cons separately and also write a detailed review.

Each format needs different techniques to perform feature extraction and to estimate polarity. However, format 3 more or less is a mixture of the other two formats. Thus, approaches used for reviews written in format 1 and format 2 are combined for mining reviews of format 3.

3.2.2.1 Feature Extraction from Reviews of Format 1

In this case the opinion polarity (positive or negative) is known, since the classes are discussed separately. Thus, there is no need to identify them. Only the product features that have been commented on need to be analysed. A review obtained from CNet.com[8] is shown in Figure 3.8.

Figure 3.8 – Review with separate description of the pros and cons

In this example the author expresses opinions on several features of a smart phone. Table 3.5 shows an overview of the pros and cons of this review.

Table 3.5 – Pros and cons of a product review with separate descriptions

Opinion	Product feature
Pros	
awesome	screen
very light	*light* ⇒ weight
fairly thin	*thin* ⇒ size
buttery smooth	UI
Cons	
heats up	use
short	battery life

The opinions are expressed in short sentence segments, consisting of a few words. Each segment is focused on one feature. For processing, it is necessary to distinguish between explicit

[8] `http://reviews.cnet.com/smartphones` (accessed on August 29^{th}, 2011)

and implicit features. An explicit feature is a concrete feature (e.g. user interface, screen) mentioned in a segment. However, also implicit features may appear in a review (Liu et al., 2005). Notice that "light" is an implicit feature referring to the actual feature "weight". The same applies to "thin", which refers to the "size" of the phone. Liu et al. (2005) argue that int addition to nouns also verbs can be features, e.g. "use".

Label sequential rules: Probably the most important work in this field was postulated by Liu et al. (2005). They prosed a supervised rule discovery approach for finding features. The rules are called *label sequential rules*, short LSR (Liu, 2010). Before the rules can be used to extract certain language patterns in novel reviews, they need to be generated from a training set.

In a first step the training set has to be prepared for analysis. For each word in the segments a POS tag[9] is generated. These tags are used to generate general language patterns. The two examples below illustrate the results of this step. <N> represents a noun, <Adj> and adjective and <V> a verb.

"<N> UI <V> is <Adj> buttery <Adj> smooth"
"<Adj> awesome <N> screen"

In a next step the actual feature words in a sentence are manually labelled and replaced by the label "[feature]". Different products have different features. Thus, this replacement ensures the identification of general language patterns. For implicit features, the feature indicator is also replaced with "[feature]". Then larger segments are split to shorter segments. Liu et al. (2005) use 3-grams (three words with their corresponding POS-tags). Therefore the example from above turns to:

"<Adj> awesome <N> [feature]"
"<N> [feature] <V> is <Adj> buttery"
"<N> [feature] <V> is <Adj> smooth"

Once the preprocessing is done the segments (3-grams) are stored in a transaction file. This file is used for finding relevant rules via association rule mining. Association rule mining is an important data mining model (Gupta, 2006). Given a set of n items $I = \{i_1, i_2, \ldots, i_n\}$ and N transactions $T = \{t_1, t_2, \ldots, t_N\}$. Each transaction consisting of m items with $m \leq n$. An association rule is written as $X \rightarrow Y$, where $X \subset I$, $Y \subset I$ and $X \cap Y = \emptyset$, with the meaning that, whenever X appears Y also tends to appear. This is called *confidence* and is the probability of Y given X, $P(Y|X)$ (conditional probability). The probability of X and Y appearing together is called *support*:

$$\text{Support}(X, Y) = P(X \cap Y) = \frac{X \cap Y}{N}, \tag{3.36}$$

$$\text{Confidence}(X, Y) = P(Y|X) = \frac{P(X \cap Y)}{P(X)}. \tag{3.37}$$

[9]More information about part of speech tagging is presented in Section 2.4.2 - Document Corpora (page 24).

The text is searched in order to find rules that fulfil the conditions of having a minimum support and a minimum confidence. Below, two sample rules are given:

(a) <Adj>, <N> → [feature]
(b) <N>, [feature] → <Adj>

In a rule the POS-tags as well as words may appear. An issue that has to be considered is that people use different words to describe one and the same feature, e.g. "display" and "screen" refer to the same feature of a mobile phone. In order to find relevant rules it is important to group features with similar meaning together. Liu et al. (2005) applied WordNet to find synonyms of features.

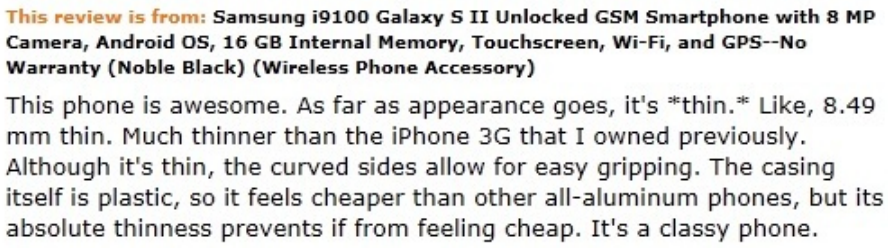
Not all generated rules are also relevant. Only those are important that have "[feature]" on the right side of →, since the objective is to predict the label "[feature]" and extract the feature. Rule (b) does not fulfil the condition and is therefore removed. The remaining rules are used to create the language patterns, according to the ordering of the items in the rules and the feature location.

<Adj> <N> [feature]

Finally, the language patterns are used to identify product features in new reviews. Thereby the system searches for segments in the review that match a pattern. In the case that more patterns apply the one with the highest confidence is chosen.

3.2.2.2 Feature Extraction from Reviews of Format 2

The extraction of features in reviews of format 2 represents a more challenging task. Usually complete and sometimes complex sentences are used for such reviews. In addition, not all sentences do necessarily express an opinion, instead they may contain large amounts of noise. Figure 3.9 shows an excerpt of a typical example of a review in free format obtained from Amazon.com[10].


This review is from: **Samsung i9100 Galaxy S II Unlocked GSM Smartphone with 8 MP Camera, Android OS, 16 GB Internal Memory, Touchscreen, Wi-Fi, and GPS--No Warranty (Noble Black) (Wireless Phone Accessory)**

This phone is awesome. As far as appearance goes, it's *thin.* Like, 8.49 mm thin. Much thinner than the iPhone 3G that I owned previously. Although it's thin, the curved sides allow for easy gripping. The casing itself is plastic, so it feels cheaper than other all-aluminum phones, but its absolute thinness prevents if from feeling cheap. It's a classy phone.

Figure 3.9 – Review in free format

Table 3.6 depicts the opinion words and product features mentioned in this product review. This rather long text just lists a few relevant words, while the rest of the text does not contain any interesting information and can be considered as noise.

[10] `http://www.amazon.com` (accessed on August 08^{th}, 2011)

Table 3.6 – Pros and cons of a product review in free format

Opinion	Product feature
Pros	
awesome	phone
thin	*thin* ⇒ size
easy gripping	*curved sides* ⇒ phone
classy	phone
Cons	
feels cheaper	phone

Frequency based approach: Hu and Liu (2004*a*) postulated a feature-based opinion aggregation system for reviews of format 2. Their approach comprises two subtasks. First, the frequent features are identified. Frequent features are defined as those features that have been talked about by many reviewers. Hu and Liu (2004*a*) used a part of speech tagger to split each review into a set of sentences and produce the POS-tag for each word in a sentence. The identified nouns and noun phrases of a sentence are saved in a transaction file. Other words that are not likely to contain sentiment are abolished. Some pre-processing of words is also performed, which includes removal of stop words, stemming and lemmatization[11].

To find all the frequent features association mining as introduced in Section 3.2.2.1 is applied. When product features are discussed, the words used to express a feature (usually nouns) converge. Thus, the frequent phrases can be found. Irrelevant contents in a review are likely to be infrequent and therefore are not recorded. Since not all obtained features are also genuine features pruning is used to remove those. The remaining features are saved in a database.

However, also infrequent features need to be obtained, because they can be of interest to potential customers or the manufacturer. Those infrequent features are found by making use of opinion words. The same adjectives can be used to express an opinion on different features. Thus, opinion words used to describe frequent features are also used to extract infrequent features. For all sentences that do not contain frequent features but one or more opinion words the nouns or noun phrases closest to the opinion words are stored as infrequent features.

Part-of relationship: Popescu and Etzioni (2005) postulated an improvement to the method mention above to find explicit product features. Their system, called OPINE[12] is built on top of KNOWITALL[13].

[11] A description of these techniques can be found in Section 2.4.2 - Document Corpora (page 25).

[12] `http://www.cs.washington.edu/research/knowitall/opine/` (accessed on August 9^{th}, 2011)

[13] KNOWITALL is a data-driven, Web-based information extraction system, proposed by Etzioni et al. (2005).

OPINE creates a representation of a product class by using the part-of relationship. It identifies both the parts and the properties of the given product class and the parts and properties of its parts. First OPINE extracts the noun phrases and then calculates the Pointwise Mutual Information scores (similar to Turney's PMI-IR algorithm[14]) between the noun phrase and a discriminator phrase. A discriminator phrase consists of the form "is a", "has a", etc.

The Partial Mutual Information is computed as

$$\text{PMI}(f, d) = \frac{\text{hits}(f, d)}{\text{hits}(f) \times \text{hits}(d)}, \tag{3.38}$$

where f is a candidate feature and d a discriminator (Liu, 2010). The PMI score of f and d is the number of hits for a query in case the discriminator and feature appear together, divided by the product of hits for both alone. If the PMI score of a feature is too low (feature and discriminator do not appear together in sufficient count), the feature may not be a part of the product.

This method scored a 22% higher precision than Hu and Liu's system, while the recall dropped by only 3%.

Chunked tokens: An novel approach that utilizes the information of dependency trees was presented by Niazi et al. (2011). Similar to the approaches presented above they applied a POS-tagger. In addition they also installed a Chunker to find out how words interrelated with each other. Niazi et al. (2011) formulated two rules based on which the relevant features are identified in the document:

> **First rule:** if the chunked-token is a B-NP (Begin Noun Phrase) and its POS-tag is either NN or NNP then the token is a feature candidate.

A feature candidate may also have other interrelated tokens. Thus, the second rules is formulated as follows:

> **Second rule:** if the chunked-token is I-NP (Interrelated Noun Phrase) and its POS-tag is either NN or NNP the tokens are extracted as one feature. Otherwise only the token obtained from the first rule is selected as a feature.

For each document the words and the corresponding POS-tags are identified. In a next step the annotated sentences are searched for chunks that fulfil both decision rules. In case of a match the tokens are added to the candidates list. Finally, the candidates are aggregated and the feature list is generated.

3.2.3 Opinion Word Extraction

Once the object features are extracted the next step is to find the opinion words and phrases in the review text. Usually an opinion lexicon is used to find these opinion words. An opinion lexicon contains a list of positive and negative opinionated words.

[14] cf. Section 3.1.1.2 - Unsupervised Machine Learning (page 42)

A straight forward solution for this step is described below:

1. Start with a set of manually labelled seed words.
2. This initial set is expanded by searching for synonyms and antonyms in WordNet or other lexical databases.
3. Ambiguity and double entries are removed from the lexicon.
4. Finally, the opinion lexicon can be applied to the actual review text. Therefore, for all the sentences that contain one or more product features the opinion words are extracted by comparing the words to the entries in the opinion lexicon. The words that match are labelled as opinion words.

A detailed overview of opinion lexicons and methods for training opinion lexicons is provided in Section 2.4.1.

3.2.4 Determining the Feature Orientation

Finding the opinion words is not enough. In addition the semantic orientation of a phrase has to be identified. The following sentence shows an example[15] on how to determine the orientation.

"*I do not like the shape of the new iPhone but the display is great.*"

In this case "shape" and "display" are determined as the product features of the iPhone. According to that "like" and "great" are the relevant opinion words. So far, nothing about the strength and polarity is know. Liu (2010) postulated a four step approach on how to estimate the polarity of an opinion:

1. **Identifying strength:** Opinion lexicons like SENTIWORDNET have numerical scores for each entry that indicate how positive or negative they are. The entries are weighted within an interval, e.g., $[-1, +1]$, with $+1.0$ being absolutely positive and -1.0 being absolutely negative. The adjectives of the opinion sentences are compared with the entries in the opinion lexicon to define their polarity. For this example each positive word has the value $+1$, each negative word the value -1 and domain depended words have the value 0. Therefore, the sample sentence from above becomes "*I do not **like**[+1] the shape of the new iPhone but the display is **great**[0]*". The expression"great" has many different meanings. It can refer to the size of an object but can also be used to express a feeling. Thus, "great" has the value 0.
2. **Handling negations:** So far, this score does not really describe the true polarity of the sentence as a human being would understand it. Indeed, the negation word "not" inverts the polarity of the phrase. Hence, the sentence is turned into "*I do **not like**[-1] the shape of the new iPhone but the display is **great**[0]*". A negation word does not always change

[15] This example is adapted from Liu (2010).

the polarity, e.g. "not yet", "not only", etc. These phrases have a different meaning and have to be addressed separately. How to handle negation correctly is defined by a set of negation rules.

3. **Conjunctions:** Conjunctions help to determine the orientation of neutral opinion words. Coordinating conjunctions like "and" keep the polarity of what follows after the conjunction. On the other hand adversative conjunctions (e.g "but", "nonetheless", etc.) change the polarity of the following words. Therefore, the orientation of the opinion words positioned before the adversative conjunction and mentioned thereafter are opposite to each other. In this example "but" turns the polarity of "great" to $+1$. As a consequence the sentence turns to "*I do **not like***[-1] *the shape of the new iPhone **but** the display is **great***[+1]".

4. **Summarization of opinions:** In this last step the final orientations of the product features in a sentence are estimated. The score of one feature can be calculated as the sum of the orientation scores of each single opinion word related to this feature:

$$\text{score}(f, s) = \sum_{i=1}^{M} p(w_i), \tag{3.39}$$

where f is a feature appearing in s, w_i being an opinion word or an opinion phrase related to the feature and $p(w_i)$ represents the polarity score of the opinion. If the total score is positive, then the opinion expressed on f is positive and vice versa.

The example from above is extended as follows:

> "*I do **not like***[-1] *the shape of the new iPhone **but** the display is **great***[+1]. *It has a **high***[+1] *resolution and has a **very sharp***[+1] *contrast*".

Then the following total scores for the features "display" and "shape" are determined as:

$$\text{score}(display, s_1) = +1,\ \text{score}(shape, s_1) = -1,\ \text{score}(display, s_2) = +2$$

Liu (2010) proposed a method to normalize the score by applying weights to opinion words or phrases, depending on how close they are to the feature. The closer the opinion word the higher its weight:

$$\text{score}(f, s) = \sum_{i=1}^{M} \frac{p(w_i)}{d(w_i, f)}, \tag{3.40}$$

$d(op_i, f)$ being the distance between feature f and the opinion word ow_i.

Keep in mind that the identification of opinion holders, product features and opinion words are not strictly executed in succession. Instead, these tasks are executed iteratively and are often based on each other. For instance, Hu and Liu (2004*a*) first extracted features from the text and in a next step used them to find opinion words. Those opinion words were then used to find infrequent features.

3.2.5 Probabilistic Models for Feature-based Sentiment Analysis

Till now, the presented methods have been based on POS-tagging, association rules mining and predefined rule sets. However, some novel approaches use probabilistic models to identify features, opinions and also sources of opinions in texts. In comparison to association rules mining these sequence models use graphical structures to represent a set of random variables and their conditional dependencies. The conditional independence structure between the random variables is depicted via a graph.

In the following Hidden Markov Models (HMM) and Conditional Random Fields (CRF) are discussed, since they have great importance for feature-based sentiment analysis.

3.2.5.1 Hidden Markov Models:

Hidden Markov models (HMM) are a form of dynamic Bayesian networks and represent a probabilistic graphical model. A set of random variables and their conditional dependencies are modelled via a directed acyclic graph.

A HMM uses a set of random variables Y that represent labels (Sutton and Mccallum, 2006). Each label has a probability distribution along the possible output tokens. The output tokens are random variables X that represent the observations (e.g. a sentence with n words). In a hidden Markov model the label is not directly visible, but the output, dependent on the label is visible. Thus, for a given sequence of output tokens $\mathbf{x} = x_1, x_2, \ldots x_n$ the goal is to find the label sequence $\mathbf{y}$ that maximises $P(\mathbf{y}|\mathbf{x})$. Figure 3.10 shows the graphical structure of a HMM. The conditional probability distribution of the hidden variable Y_i depends only on the value of the hidden variable Y_{i-1}. This is called the Markov property[16]. Similarly, the value of the observed variable X_i only depends on the value of the hidden variable Y_i.

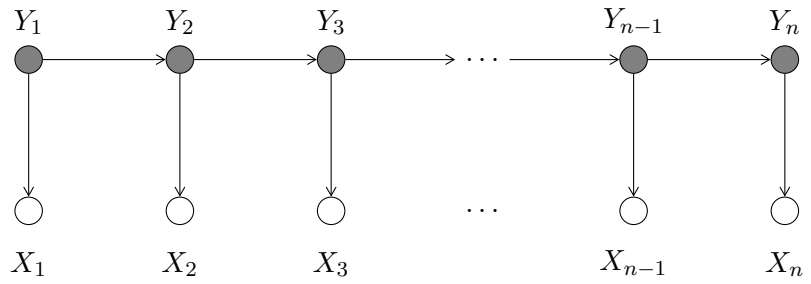


Figure 3.10 – Hidden Markov Model (Sutton and Mccallum, 2006)

Based on these basic concepts Jin et al. (2009) proposed a novel machine learning approach built under the framework of lexical Hidden Markov Models. Their approach integrates additional lexical information like part of speech into the HMM.

[16]The Markov property states, that the next state depends only on the current state and not on other past states (Andersson, 2000).

Given the observations, a sequence of words $\mathbf{w} = w_1, w_2, \ldots, w_n$ with their corresponding POS-tags $\mathbf{s} = s_1, s_2, \ldots, s_n$, the objective is to find an appropriate sequence of tags (hidden labels) $\hat{\mathbf{t}}$, which maximize the conditional likelihood $P(\mathbf{t}|\mathbf{w}, \mathbf{s})$. By taking Bayes' theorem this objective can be expressed as:

$$P(\mathbf{t}|\mathbf{w}, \mathbf{s}) = \frac{P(\mathbf{w}, \mathbf{s}|\mathbf{t})P(\mathbf{t})}{P(\mathbf{w}, \mathbf{s})}. \tag{3.41}$$

Since the probability $P(\mathbf{w}, \mathbf{s})$ remains unchanged for all candidate tag sequences, it can be removed. Thus, the model is given as follows:

$$\hat{\mathbf{t}} = \underset{\mathbf{t}}{\operatorname{argmax}} \left[P(\mathbf{w}, \mathbf{s}|\mathbf{t})P(\mathbf{t})\right] = \underset{\mathbf{t}}{\operatorname{argmax}} \left[P(\mathbf{s}|\mathbf{t})P(\mathbf{w}|\mathbf{t}, \mathbf{s})P(\mathbf{t})\right]. \tag{3.42}$$

In practice this model cannot be calculated since it involves too many parameters. Thus, Jin et al. (2009) made three assumptions for simplifying the problem: (1) the assignment of the current tag t_i is supposed to depend not only on its previous tag but also on the previous J words w_{i-J}, w_{i-1}. (2) The appearance of the current word w_i is assumed to depend not only on the current tag and the current POS, but also on the previous K words w_{i-K}, w_{i-1}. (3) The appearance of the current POS s_i is supposed to depend on both, the current tag and the previous L words w_{i-L}, w_{i-1}. Then the objective is to maximize:

$$\hat{\mathbf{t}} = \underset{\mathbf{t}}{\operatorname{argmax}} \prod_{i=1}^{N} \begin{pmatrix} P(s_i|w_{i-1}, t_i)\times \\ P(w_i|w_{i-1}, s_i, t_i)\times \\ P(t_i|w_{i-1}, t_{i-1}) \end{pmatrix}. \tag{3.43}$$

Jin et al. (2009) used the Maximum Likelihood Estimation (MLE) to estimate the parameters of the statistical model depicted in equation 3.43. For example the parameter $P(s_i|w_{i-1}, t_i)$ can be calculated as:

$$P(s_i|w_{i-1}, t_i) = \frac{C(w_{i-1}, t_i, s_i)}{\sum_s C(w_{i-1}, t_i, s)}, \tag{3.44}$$

where $C(w_{i-1}, t_i, s_i)$ is the count on how often the constellation appears. The MLE values for estimating the other parameters can be computed analogous.

For testing Jin et al. (2009) collected 1728 review documents about cameras from Amazon.com. A subset of 293 documents was manually tagged, while the remaining documents were tagged in a bootstrapping process implementing the HMM classifier. Additionally, the publicly available corpus of Hu and Liu (2004*a*) was used for evaluation. Applying the Lexical-HMM together with the POS-tags the system scored an average recall of 83.9% for entity extraction on the Amazon corpus. For the corpus of Hu and Liu (2004*a*) the system achieved a recall of 86.0%, and the precision almost reached 80.0%.

3.2.5.2 Conditional Random Fields:

An important learning method for feature-based opinion mining are Conditional Random Fields (CRFs). CRFs are conditional probability distributions on an undirected graph model. They relax the strong independence assumptions made in HMMs. CRFs were first introduced by Lafferty et al. (2001).

(X, Y) is a conditional random field, where X is a set of input variables (observations) that need to be labelled (e.g., a sequence of textual words that form a sentence) and Y a set of random variables over the corresponding sequence. Y represents a hidden state variable that needs to be inferred from the given observations. Let $\mathcal{G} = (V, E)$ be an undirected graph, with V a set of vertexes representing the random variables $\{Y_1, Y_2, \ldots, Y_n\}$, and a set of edges E of the form $\{Y_{i-1}, Y_i | 1 < i \leq n\}$, which create a linear chain or line. The graphical structure is depicted in Figure 3.11.

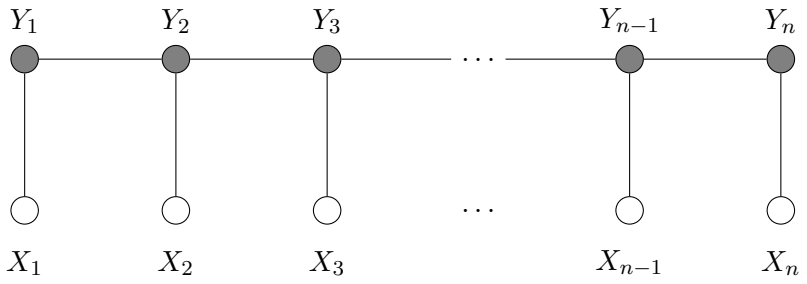


Figure 3.11 – Linear Conditional Random Field (Lafferty et al., 2001; Qi and Chen, 2010)

Given a set of observations $\mathbf{x} = x_1, x_2, \ldots, x_n$ for X the conditional probability $P(\mathbf{y}|\mathbf{x})$ for the linear-chain is estimated as:

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_x} \exp\left(\sum_{k=1}^{K} \lambda_k F_k(\mathbf{y}, \mathbf{x})\right), \tag{3.45}$$

with

$$\lambda_k F_k = \sum_{i=1}^{n} \lambda_k f_k(y_{i-1}, y_i, \mathbf{x}, i) + \sum_{i=1}^{n} \lambda'_k f'_k(y_i, \mathbf{x}, i). \tag{3.46}$$

f_k being a binary feature function for edge (y_{i-1}, y_i) with $k \in K^e$ and f'_k a binary feature function for a vertex y_i with $k \in K^v$, where K^v and K^e respectively represent the sets of indices of vertex features and edge features. λ_k and λ'_k are parameters to weight the impact of a feature function. Z_x portrays a normalization factor for $\mathbf{x}$:

$$Z_x = \sum_{y} \exp\left(\sum_{k=1}^{K} \lambda_k F_k(\mathbf{y}, \mathbf{x})\right). \tag{3.47}$$

The goal in CRF is to find the most probable sequence of labels $\hat{\mathbf{y}}$:

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\operatorname{argmax}} \left[P(\mathbf{y}|\mathbf{x}) \right]. \tag{3.48}$$

Conditional Random Fields have great importance in feature-based opinion mining. Several approaches have been postulated that use this machine learning method (Choi et al., 2005; Zhao et al., 2008; Choi and Cardie, 2010; Qi and Chen, 2010; Miao et al., 2010; Qi and Chen, 2011). In the following two important methods are described.

Qi and Chen (2010) postulated a CRF based method to extract entities from reviews. They employed three types of tags to define each word: entity tag, position tag and opinion tag. Given the observations, a sequence of words $\mathbf{w} = w_1, w_2, \ldots, w_n$ with their corresponding POS-tags $\mathbf{s} = s_1, s_2, \ldots, s_n$, the objective is to find an appropriate sequence of tags (hidden labels) $\mathbf{t}$, which maximize the conditional likelihood. Since the CRF is a linear chain, each feature involves only two consecutive hidden states t_i, t_{i-1}. Thus, the equation 3.48 can be written in the form:

$$\hat{\mathbf{t}} = \underset{\mathbf{t}}{\operatorname{argmax}} \left[P(\mathbf{t}|\mathbf{w}, \mathbf{s}) \right] = \underset{\mathbf{t}}{\operatorname{argmax}} \prod_{i=1}^{N} P(t_i|\mathbf{w}, \mathbf{s}, t_{i-1}). \tag{3.49}$$

Due to some transformations, the equation 3.49 can be written as:

$$P(\mathbf{t}|\mathbf{w}, \mathbf{s}) = \frac{1}{Z} \exp \left[\sum_{i,k} \lambda_k f_k(t_{i-1}, t_i, \mathbf{w}, \mathbf{s}, i) + \sum_{i,k} \lambda'_k f'_k(t_i, \mathbf{w}, \mathbf{s}, i) \right]. \tag{3.50}$$

Qi and Chen (2010) tested their system on two datasets, one obtained from the Yahoo shopping site[17] and the other was the shareable corpus from Hu and Liu (2004*a*). In total, the two datasets consisted of 476 reviews for three cameras and one cellphone. For the evaluation Qi and Chen (2010) compared their method to the Hidden Markov Model[18] presented by Jin et al. (2009). Experiments showed that the CRF approach scored significantly better results then the HMM method. Precision improved from 83.9% to 90.0% on the average value and the recall increased from 72.0% to 79.8%.

Another CRF based method was presented by Choi and Cardie (2010). They proposed a hierarchical sequential learning method to detect opinions in documents. Their approach comprises the identification of boundaries of opinion expressions as well as determining the two attributes: polarity and intensity. Choi and Cardie (2010) applied nine conjunctive values of polarity and intensity labels $y_i = \{0 \ldots 9\}$ as shown in Table 3.7.

[17] `http://shopping.yahoo.com/` (accessed on November, 10^{th}, 2011)

[18] cf. Section 3.2.5.1 - Hidden Markov Models (page 59)

Table 3.7 – Labels for opinion extraction with polarity and intensity (Choi and Cardie, 2010)

Label	0	1	2	3	4	5	6	7	8	9
Polarity	none	pos	pos	pos	neutral	neutral	neutral	neg	neg	neg
Intensity	none	high	medium	low	high	medium	low	high	medium	low

Based on these labels the feature functions for a vertex are defined as follows:

$$\lambda f(y_i, \mathbf{x}, i) = \lambda_\alpha g_O(\alpha, \mathbf{x}, i) + \lambda_\beta g_P(\beta, \mathbf{x}, i) + \lambda_\gamma g_S(\gamma, \mathbf{x}, i), \tag{3.51}$$

and for the edge as:

$$\lambda' f'(y_i, y_{i-1}, \mathbf{x}, i) = \lambda'_{\alpha,\hat{\alpha}} g'_O(\alpha, \hat{\alpha}, \mathbf{x}, i) + \lambda'_{\beta,\hat{\beta}} g'_P(\beta, \hat{\beta}, \mathbf{x}, i) + \lambda'_{\gamma,\hat{\gamma}} g'_S(\gamma, \hat{\gamma}, \mathbf{x}, i), \tag{3.52}$$

where g_O and g'_O are feature vectors defined for **O**pinion extraction, g_P and g'_P are feature vectors defined for **P**olarity extraction, and g_S and g'_S are feature vectors defined for **S**trength extraction. The values of the features are:

$$\begin{array}{rcl} \alpha, \hat{\alpha} & \in & \{\text{OPINION}, \text{NO-OPINION}\} \\ \beta, \hat{\beta} & \in & \{\text{POSITIVE}, \text{NEGATIVE}, \text{NEUTRAL}, \text{NO-POLARITY}\} \\ \gamma, \hat{\gamma} & \in & \{\text{HIGH}, \text{MEDIUM}, \text{LOW}, \text{NO-STRENGTH}\}. \end{array} \tag{3.53}$$

Example: For instance, if $y_i = 1$, then

$$\begin{array}{rcl} \lambda f(1, \mathbf{x}, i) & = & \lambda_{\text{OPINION}}\, g_O(\text{OPINION}, \mathbf{x}, i) \\ & + & \lambda_{\text{POSITIVE}}\, g_P(\text{POSITIVE}, \mathbf{x}, i) \\ & + & \lambda_{\text{HIGH}}\, g_S(\text{HIGH}, \mathbf{x}, i). \end{array}$$

The system was investigated by examining three options for jointly extracting opinion expressions with their attributes. The options are: (1) polarity-only and intensity-only, (2) joint sequential tagging without hierarchy and (3) joint sequential tagging with hierarchy. The evaluation results showed that the simple joint sequential tagging approach even without exploiting the hierarchy brings a better performance than combining two separately developed systems. Based on the results Choi and Cardie (2010) argue that an approach exploiting the hierarchical class structure improves the performance compared to a non-hierarchical baseline.

CHAPTER 4

Implementation

This chapter presents a detailed description of the practical part of the thesis. At first a general description of the approach is provided in Section 4.1. In the following the implementation issues are addressed, presenting the applied technologies and also discussing the problems and difficulties of processing the German language. Finally, the last Section 4.3 describes the implemented system is in detail.

4.1 Approach

Based on the theoretical background that is presented in Chapter 3, a feature-based sentiment analysis system has been developed, which is specified for reviews written in the German language. The objective of the system is the automatic identification of features and opinion words in a textual document and to assign the opinion words to the corresponding features. The proposed process consists of three stages.

In the first stage the opinion lexicon is created. For this purpose three seed lists containing opinion words with either positive, negative or neutral orientation are manually crafted. By accessing SentiWS[1] the word forms and the orientation scores are obtained and stored in the opinion lexicon. In the next stage the documents are analysed in order to find nouns that are potential product features. A noun is considered a frequent feature if the number of its appearances in the documents exceeds a predefined threshold. The frequent features and the opinion lexicon are then stored for further processing. Finally, in the last stage the actual opinion mining process is conducted. This is done by determining the negation words and opinion words of a sentence and by allocating them to the corresponding features.

[1] `http://asv.informatik.uni-leipzig.de/download/sentiws.html` (accessed on November 6^{th}, 2011)

4.2 Implementation Issues

The following section presents the crucial technologies and frameworks that have been used for developing the prototype of the opinion mining system. In addition, this section gives an overview of the problems and challenges, researchers face when working with the German language.

4.2.1 Technologies and Resources

The prototype is fully developed in Java. For implementation several third party frameworks and packages have been used that are listed below.

- **OpenNLP**[2] is a comprehensive toolkit for Natural Language Processing tasks. It is available under the Apache Software Foundation License.

  OpenNLP uses machine learning techniques (e.g. Maximum Entropy) for common NLP tasks such as sentence detection, tokenization, part-of-speech tagging, named entity extraction, chunking and parsing. OpenNLP supports different languages and provides pretrained models for these tasks.

- **Stanford Parser**[3] is another toolkit for Natural Language Processing. It was developed and is still maintained by the Natural Language Processing Group of the Stanford University and is available under the GNU General Public License.

  The Stanford Parser is a probabilistic parser, implementing machine learning techniques for parsing different languages. It uses Probabilistic Context Free Grammar (PCFG) models to produce the most likely analysis of a sentence. This parser supports the identification of the grammatical structure of sentences.

- **Sentiment Wortschatz**[4] (SentiWS) is a German-language resource for sentiment analysis. This dictionary consists of 1,650 negative and 1,818 positive words (Remus et al., 2010). The words are weighted within an interval of $[+1.0, -1.0]$. In addition to the words their part of speech tags and the different word forms are given. In total, the SentiWS in its current version (v1.8b) contains 16,406 positive and 16,328 negative word forms.

- **UIMA**[5] stands for Unstructured Information Management Architecture. UIMA was originally developed by IBM and is now available under the Apache Software Foundation License. The UIMA framework supports the analysis of unstructured information. Therefore, it provides pipelines that cover the whole NLP process, from reading a document, analysing it and generating the user output.

  UIMA uses core building blocks, so called Analysis Engines (AEs) which perform the document analysis and refer the results (*UIMA Overview & SDK Setup*, 2010). Instances

[2] `http://incubator.apache.org/opennlp/`, (accessed on October 20^{th}, 2011)

[3] `http://nlp.stanford.edu/software/lex-parser.shtml` (accessed on October 20^{th}, 2011)

[4] `http://asv.informatik.uni-leipzig.de/download/sentiws.html` (accessed on November 6^{th}, 2011)

[5] `https://uima.apache.org/` (accessed on October 20^{th}, 2011)

of this component are called annotators. The result of an annotator are annotations, which are mappings of meta information about an item to the actual item. In general, an annotation is identified by its begin and end index. A typical example of an annotation would be a token annotation for a word that states these indices. In addition, it is also possible to add further meta information to the annotation, for instance the POS-tag of the word. A simple annotator performs granular functions, like tokenization or POS-tagging. These functions typically address just one part of an overall analysis task. For more complex tasks UIMA allows to combine several Analysis Engines to a workflow. These more complex analysis engines are called Aggregate Analysis Engines (*UIMA Overview & SDK Setup*, 2010).

For creating the annotations and handling the data exchange between the single annotators UIMA defines the Common Analysis Structure (CAS), an object-based data structure whose purpose is the representation of objects, properties and values (*UIMA Overview & SDK Setup*, 2010). The CAS logically contains the document being analysed.

For being able to work with a set of documents UIMA provides the Collection Processing Architecture (CPA) (*UIMA Overview & SDK Setup*, 2010). The CPA consists of two modules, the Collection Reader and the CAS Consumer. The Collection Reader's job is to iterate through a source collection, acquiring documents and initializing CASes for further analysis. The CAS Consumer on the other hand resides at the end of the flow and conducts the final CAS processing. Therefore it consumes the updated CASes produced by the annotators and generates an output that can be saved in relational databases, into XML files for further processing, or graphically presented to the user.

All these UIMA modules are aggregated by the UIMA Collection Processing Engine (CPE) (*UIMA Overview & SDK Setup*, 2010). The CPE specifies a "source" to "sink" flow from a Collection Reader through a set of Analysis Engines and then to a set of CAS Consumers. The complete workflow is illustrated in Figure 4.1.

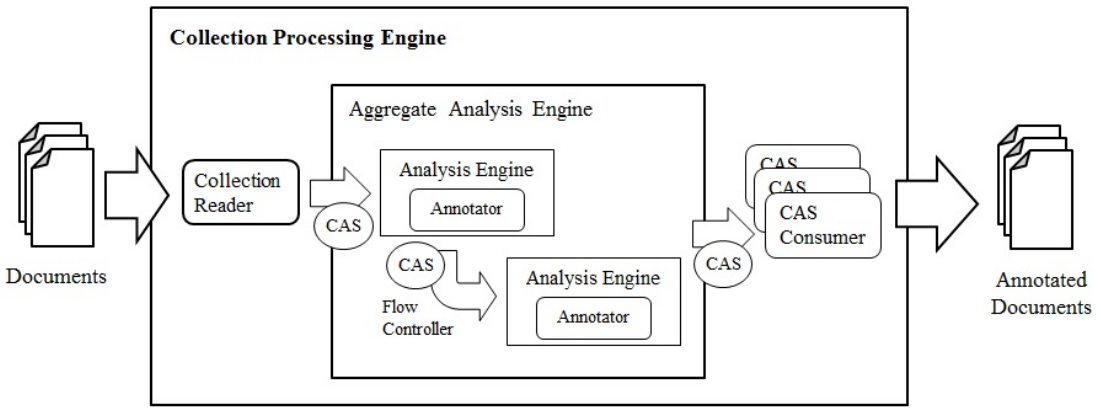


Figure 4.1 – UIMA component architecture from source to sink (*UIMA Overview & SDK Setup*, 2010)

4.2.2 Problems analysing the German Language

So far, most research in the field of opinion mining has been conducted for texts written in the English language. Even tough German and English are closely related and emerged from the Germanic branch of the Indo-European language family, several issues have to be considered when working with German texts (Shoebottom, 2010). In the following the most important differences for Natural Language Processing between English and German are presented.

- **Punctuation rules:** German has stricter punctuation rules than English. The usage of a comma has great significance for the semantic of a sentence. For example, every relative clause in German has to be indicated with a comma. Missing or wrongly placed punctuation marks lead to inaccuracy in the parser results.

- **Capital and small letters:** In German the first character of every noun and proper name has to be upper case, while in English only proper names start with an upper case. This has important consequences for the language parsing. The POS-tagger does not correctly classify words if their first letter is not written correctly. For example the POS-tagger identifies “iPhone” as an adverb, while “Iphone” is identified as a noun. Wrongly classified words might also falsify the output of the Stanford Parser and lead to inconsistent penntrees.

- **Syntax:** The syntax of German sentences is in some way more complex than the English language. Because of this complexity the OpenNLP parser and the Stanford parser do not support dependency parsing for German. These days only a few parsers[6] exist that do support dependency parsing for German.

- **Compound nouns:** A notable characteristic of the German language is the nearly unlimited possibility to combine nouns to build a new term or concept. For example the words “Sport” (sport) and “Stadion” (stadium) can be combined to “Sportstadion” (sports stadium). Another example is “Tastaturlayout” (layout of a keyboard). This characteristic makes it difficult to find feature terms. “Tastaturlayout” and “Tastatur” are handled as different words. Assume, that “Tastatur” is a frequent feature and “Tastaturlayout” is not. In this case only “Tastatur” can be identified. Instead in English, even though "keyboard" and “keyboard layout” are different noun phrases keyboard appears in both independently and thus can be extracted.

The former differences strongly influence the results of the language processing task for texts written in German. For correct tokenization and POS-tagging it is important that nouns start with a capital letter. In order to produce consistent penntrees the documents need to be well formed, meaning that the correct usage of commas, grammar and spelling has to be assured.

[6] A parser that supports dependency parsing for texts written in German is ParZu - The Zurich Dependency Parser. ParZu was developed by the Linguistics Group at the University of Zurich. So far ParZu is not available for Java. Thus, the Parser is not applicable for the opinion mining system presented in this thesis. More information about ParZu can be found at `https://github.com/rsennrich/ParZu` (accessed on October 11^{th}, 2011).

4.3 System Description

The postulated system utilizes a hybrid approach combining semi-supervised learning algorithms together with a large set of rules for determining the opinion and feature orientation. Following the three stages described in *Section 4.1* the proposed system consists of three main modules: (I) the *opinion lexicon* module, (II) the *feature extraction* module and (III) the actual *opinion mining* module. Figure 4.2 depicts the basic architecture of the system.

At first the feature extraction module is started that searches the documents for frequent items. Independent of this module SentiWS is used to generate an opinion lexicon, containing words expressing a positive, negative or neutral sentiment. The opinion mining module uses the frequent feature list, as well as the opinion lexicon to determine the orientation of each feature. The final result of the opinion mining process is a set of quadruples $\langle f, ow, *nw*, p(f)\rangle$ extracted for each feature. A quadruple states the actual feature f, one corresponding opinion word ow, an optional negation word[7] $*nw*$ and the resulting orientation score $p(f)$.

In the following each module, its architecture and functionality is discussed in detail.

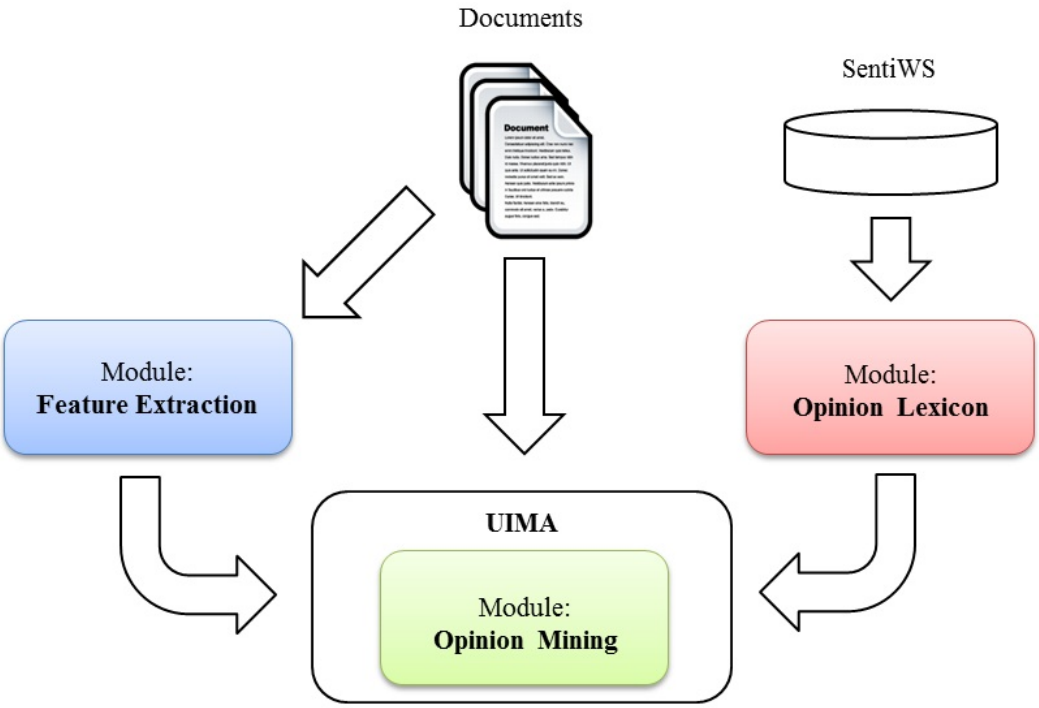


Figure 4.2 – Architecture of the opinion mining system

[7]The optionality of the negation word is indicated by the stars (*).

4.3.1 Module I: Opinion Lexicon

In this thesis an opinion is understood as a judgemental view, attitude or appraisal on a feature. The basic component to represent an opinion is an opinion word ow, which is typically an adjective. But also verbs, adverbs and nouns can hold an opinion. Each opinion word has a polarity score $p(ow)$ indicating its orientation. The polarity score is set within the interval $[+1.0, -1.0]$, $+1.0$ being totally positive and consequently -1.0 being totally negative. Thus, an opinion word having a score higher than zero is called *positive opinion word* ow^+, e.g. "ausgezeichnet"[8] (excellent) $[1.0]$. Vice versa an opinion word having a score below zero is referred to as *negative opinion word* ow^-, e.g. "schlecht" (bad) $[-0.7]$. Since not all opinion words also express an explicit polarity a third class needs to be considered. The orientation of these so called *neutral opinion words* ow^o depends on the context they appear in, e.g. "schnell" (fast) $[|0.7|]$. Nonetheless, these words also have a predefined polarity score which is an absolute value. Once the orientation of an neutral opinion word is determined, the score can then be either positive or negative . Single opinion words can be combined to opinion phrases. These n-grams are usually compositions of adjectives and adverbs. A typical example is the phrase "sehr gut" (very good).

For generating the opinion lexicon Sentiment Wortschatz (SentiWS) is used, which categorizes opinion words into two classes: "positive" and "negative". Because of the missing class "neutral", the original SentiWS lists are not applicable for this opinion mining system. In addition, many words listed in SentiWS are not used for expressing opinions in the area of product reviews. Therefore, an own comprehensive opinion lexicon has been created by applying a bootstrapping method. For this purpose three seed lists S_{pos}, S_{neg} and S_{neu} have manually been crafted. Using the seed words SentiWS has been searched for the inflected word forms, the polarity score as well as the POS-tag. An excerpt of the seed lists is depicted below:

1. $S_{pos} = \{$ausgezeichnet, gut, schön, toll, genial, super,...$\}$
2. $S_{neg} = \{$schlecht, schirch, mangelhaft, defekt, miserabel...$\}$
3. $S_{neu} = \{$groß, klein, schnell, langsam, enorm, absolut...$\}$

The opinion words combined with the additional information obtained from SentiWS are stored in the appropriate opinion word lists. Table 4.1 depicts the structure of the lexicon by means of the two adjectives "schön" (beautiful) and "schnell" (fast). The column *Score* shows the polarity score of the entries. Since, "schnell" is a neutral opinion word the polarity score is given as an absolute value. The endings are used to infer the different word forms.

4.3.2 Module II: Feature Extraction

The feature extraction module utilizes an association rules mining algorithm for finding frequent itemsets. An itemset is considered to be frequent when its count exceeds a predefined threshold, which is called *minimum support*. The itemsets that fulfil this condition are called *frequent features*. Besides the frequent features other items might appear in the documents, whose counts

[8] Since the system is based on analysing sentiment in documents written in German the examples are given in German. However, an English translation is also stated.

Table 4.1 – Sample entries in the opinion lexicon

Base form	Tag	Endings	Score
schnell	ADJ	e,er,es,en,em,ere,erer,eres,eren,ste,ster,stes,sten	$\|0.5\|$
schön	ADJ	e,er,es,en,em,ere,erer,eres,eren,ste,ster,stes,sten	0.75

do not exceed the minimum support. The feature extractor presented in this thesis does not extract these so called *infrequent features*. Following the argumentation form Hu and Liu (2004*b*) the features that are most commented on are also those features, people are most interested in. Therefore, the system is only interested in frequent features.

Two ways to express a feature can be differentiated. A feature can explicitly be expressed by a set of words or phrases. Another way is to implicitly refer to a feature by means of feature indicators (Liu, 2010). Features that are implicitly referred to are called *implicit features*. The following sentence shows an example of such an implicit feature.

> "*Das Mobiletelefon ist sehr dünn, passt aber nicht gut in die Hosentasche.*"
> (The mobile phone is really thin, but it does not really fit into the pocket.)

In this example "passt nicht gut in die Hosentasche" implicitly refers to the size of the mobile phone. Thus, the explicit feature size is indicated by the feature indicator "passt". The proposed system just supports the identification of explicit features. The identification and understanding of implicit features is subject to future research and development.

The feature extraction module is realized as a three step process as depicted in Figure 4.3. First the documents need to be preprocessed in order to produce the necessary POS tags. In the second step an association rules mining algorithm is executed to find the frequent features. Finally, the generated feature list needs to be pruned in order to remove inconsistencies. In the following outline each of the three steps is presented in detail.

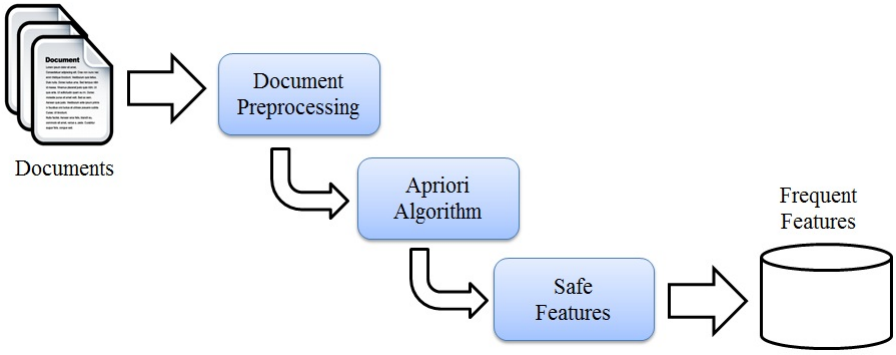


Figure 4.3 – Architecture of the feature extraction module

4.3.2.1 Document Preprocessing

Before the actual features can be extracted it is necessary to prepare the document for analysis. First a collection of documents D is loaded from the file system. Each document $d \in D$ is represented by its sentences $S = \{s_1, s_2, \ldots, s_n\}$. Te sentence detector provided by the OpenNLP library is used to retrieve the sentences. Once each single sentence is identified the special characters (e.g. §, $, &, #, ...), parentheses and *emoticons*[9] are removed. The removal of these special characters is necessary because they strongly influence the outcome of the further steps.

Subsequently each sentence has to be split into its single words $\{w_1, w_2, \ldots, w_m\}$. Using the OpenNLP tokenizer and the POS-tagger the words are identified and their corresponding POS-tags $\{t_1, t_2, \ldots, t_m\}$ are generated. For the German language, OpenNLP uses the Stuttgart Tübingen Tag Set (STTS). This tag set consists of eleven main categories, which are listed below. An overview of all tags used in STTS can be found in the Appendix A.1 of this thesis. For a detailed description of each tag see (Schiller et al., 1999).

(1) N - Nomina (Nouns)
(2) V - Verben (Verbs)
(3) ART - Artikel (Articles)
(4) ADJ - Adjektive (Adjectives)
(5) P - Pronomina (Pronouns)
(6) CARD - Kardinalzahlen (Cardinal)
(7) ADV - Adverbien (Adverbs)
(8) KO - Konjunktionen (Conjunctions)
(9) AP - Adpositionen (Adposition)
(10) ITL - Interjektionen (ITL)
(11) PTK - Partikeln (Particles)

The result of the POS-tagging step is a set of tuples $\{\langle w_1, t_1\rangle, \langle w_2, t_2\rangle, \ldots, \langle w_m, t_m\rangle\}$ for each sentence. An example is shown below

> "*Das neue IPhone, das ich gestern gekauft habe, hat einen schnellen Prozessor.*"
> (The new IPhone, which I bought yesterday has a fast processor.)

```
Das(ART) neue(ADJA) IPhone(NN) ,($,) das(PRELS) ich(PPER)
gestern(ADV) gekauft(VVPP) habe(VAFIN) ,($,) hat(VAFIN)
einen(ART) schnellen(ADJA) Prozessor(NN) .($.)
```

Once the word, tag tuples are generated the next step is to create the transaction file, which is subject to further processing. Generally, explicit product features are expressed by using nouns, noun phrases and cardinals. Thus, only those have to be stored in the transaction file. Each line of the transaction file represents one transaction and contains the nouns, noun phrases and cardinals of one sentence.

[9] An emoticon is a facial expression pictorially represented by punctuation and letters, usually to express a writer's mood, e.g. :), :-). The word "emoticon" is a combination of the English words "emotion" and "icon". Emoticons are commonly used on the Internet; cf. `https://en.wikipedia.org/wiki/Emotion` (accessed on October 8^{th}, 2011)

4.3.2.2 Extracting Frequent Features

For association rules mining a modification of the Apriori algorithm (Agrawal and Srikant, 1994) was implemented. The algorithm proposed by Agrawal and Srikant (1994) consists of two parts. In the first part, the itemsets that exceed the minimum support are found. In the second part the association rules are generated from the frequent itemsets (Gupta, 2006). In the context of this thesis only the first part is of interest.

In order to be able to handle sentences with diverging length the original Apriori algorithm was modified in a way that instead of using a matrix representation each transaction is represented by its actual words.

The algorithm works as follows:

- $k = 1$: Scan all transactions in the transaction file and generate a list of candidate features C_1. Each candidate, whose appearance is set above the predefined minimum support is added to the temporary frequent feature list T_1.
- $k = 2$: Each frequent feature in T_1 is combined with all other frequent features in T_1 to generate the candidates C_2. Scan the transaction file to find those candidates that exceed the minimum support. The candidates are added to T_2.
- $k > 2$: Pairs of frequent features in T_{k-1} are used to generate the candidates C_k in a way that each pair has the first respectively last $k - 2$ itemsets in common. The frequent candidates are added to T_k.

  > **Example:** For $k = 3$, given the frequent features "Samsung Galaxy" and "Galaxy S" the candidate "Samsung Galaxy S" is generated because the word "Galaxy" appears in the first itemset on the last position and in the second itemset on the first position. Thus, the two itemsets overlap.

The process terminates when no further candidates can be generated. The temporary frequent feature lists T_i with $i = \{1, 2, \ldots, k\}$ are merged to one list and saved to the file system. The basic functionality of the Aprior Algorithm is illustrated in Algorithm 1.

4.3.2.3 Feature Pruning

Not all features that have been generated by the Apriori algorithm are also useful. Thus, some feature pruning has to be conducted in order to get only genuine features. The feature list is manually checked for inconsistencies and for items that got wrongly classified. Those entries are deleted from the feature list. The final feature list is stored to the file system and can be applied to the opinion mining module.

Algorithm 1 - Apriori Algorithm

```
1:  procedure APRIORIALGORITHM
2:      T                                          ▷ T = temporary frequent feature list
3:      k ← 1                                      ▷ k = counter
4:      repeat
5:          C ← call GENERATECANDIDATES(k, T)      ▷ C = candidates list
6:          T ← call FREQUENTFEATURES(C)
7:          add T to F                             ▷ F = frequent feature list
8:          k ← k + 1
9:      until C.size() < 1
10: end procedure

11: procedure GENERATECANDIDATES(k, T)             ▷ Generate Candidates
12:     if k = 1 then                              ▷ Generate k = 1 itemsets
13:         for all item ∈ TransactionFile do
14:             if C not contain item then add item to C
15:             end if
16:         end for
17:     else if k = 2 then                         ▷ Generate k = 2 itemsets
18:         for i = 1 → T.size() do
19:             for j = i + 1 → T.size() do
20:                 add item_i + item_j to C
21:             end for
22:         end for
23:     else if k > 2 then                         ▷ Generate k > 2 itemsets
24:         for i = 1 → T.size() do
25:             for j = i + 1 → T.size() do
26:                 if k − 2 words match then add item_i + item_j to C
27:                 end if
28:             end for
29:         end for
30:     end if
31:     return C
32: end procedure

33: procedure FREQUENTFEATURES(C)                  ▷ Generate frequent features
34:     for all item ∈ C do
35:         if item.counter > minSupport then add item to T
36:         end if
37:     end for
38:     return T
39: end procedure
```

4.3.3 Module III: Opinion Mining

The opinion mining module represents the core application of the presented system. This module is integrated into the UIMA framework. It uses six Analysis Engines (AEs) implementing the annotators. The AEs are aggregated in an Aggregated Analysis Engine (AAE) and cover the whole workflow. The frequent feature list as well as the opinion mining lexicon are used by the analogous annotators to find the features and opinion words in the document. The AEs are executed in a fixed flow, starting with the sentence annotator. The workflow is illustrated in Figure 4.4.

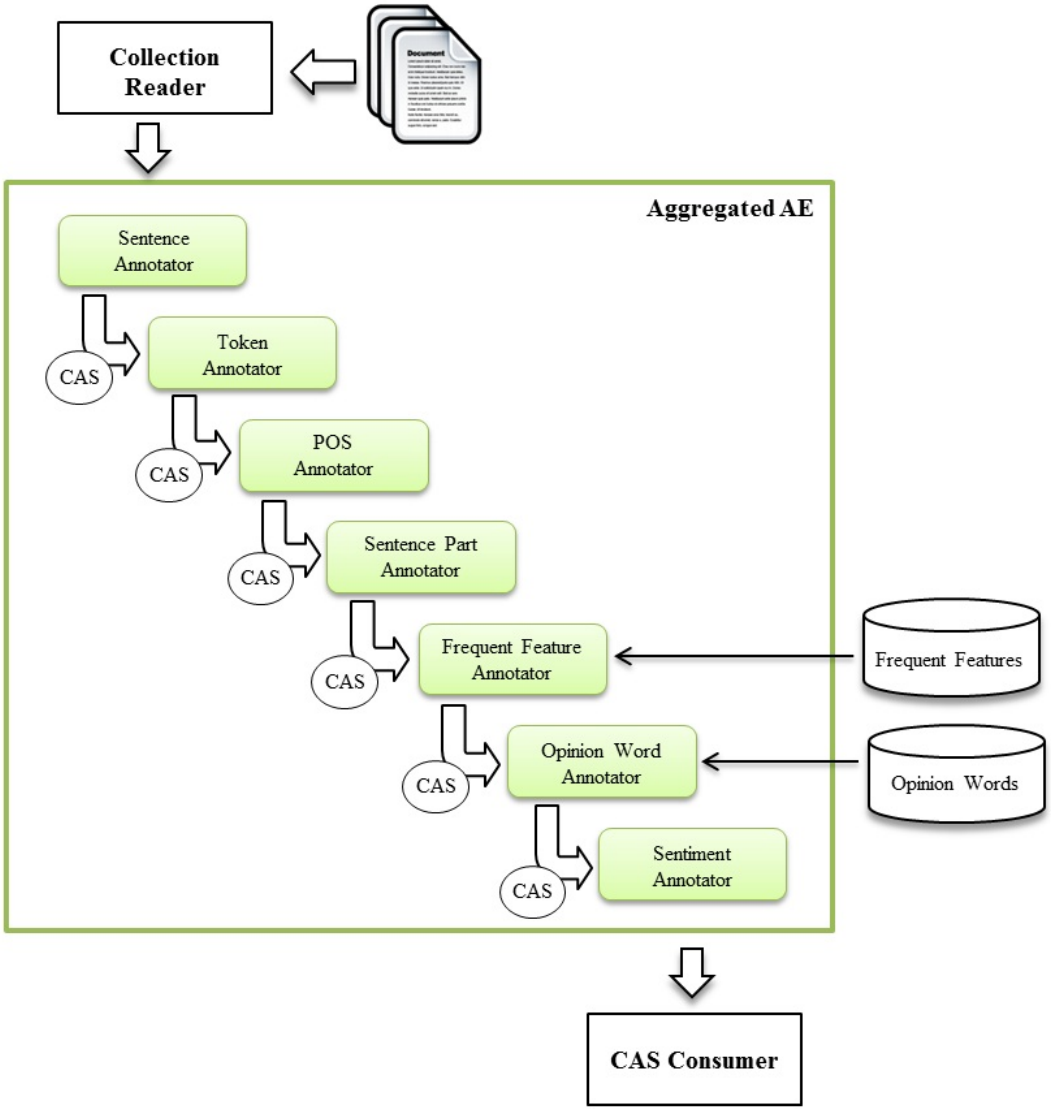


Figure 4.4 – Schematic model of the workflow and architecture of the opinion mining module

4.3.3.1 Document Preprocessing Annotators

The basic annotators for sentence splitting, tokenization and POS-tagging are provided within the UIMA framework. They are based on the the OpenNLP implementation of these tasks. Each task is integrated into a particular Analysis Engine[10].

Firstly, the Collection Reader iterates over a document collection D stored on the file system. For each document d a Common Analysis Structure (CAS) is generated. This structure is forwarded to the sentence annotator, which implements the OpenNLP sentence detector. Once a sentence is determined, the annotator produces a SENTENCE[11] annotation and maps it to the corresponding sentence $s_i \in d$. The updated CAS is then forwarded to the tokenizer. For each word w_{i_j} a TOKEN annotation is created, where w_{i_j} is the j^{th} word of the i^{th} sentence. Finally, the part of speech annotator creates the POS-tag for each word and updates the TOKEN annotation by adding the POS-tag. Figure 4.5 shows an example of a SENTENCE and a TOKEN annotation. An annotation is identified by its starting and ending indices. Additionally, other attributes can also be added to an annotation. In the case of the TOKEN annotation the POS-tag is added. Another important characteristic of UIMA is the aspect that any item of the document can be part of several annotations. As shown in Figure 4.5 the word "neue" (new) is annotated by a TOKEN annotation but it is also part of the SENTENCE annotation.

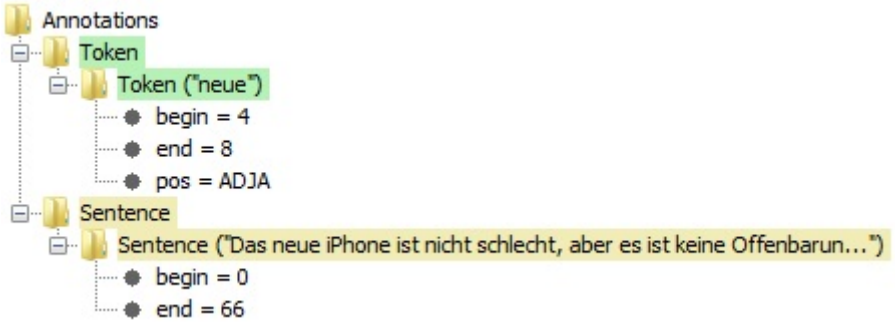


Figure 4.5 – SENTENCE and TOKEN annotations for the sentence:
"*Das neue iPhone ist nicht schlecht, aber es ist keine Offenbarung.*".
(The new iPhone is not bad, but it is not ground breaking.)

4.3.3.2 Sentence Part Annotator

The sentence part annotator splits a sentence into its parts and creates the corresponding SENTENCEPART annotations. For generating the sentence parts the Stanford parser is used. Each sentence is parsed in order to create the penntree. In the next step the penntree is recursively traversed. For each chunk indicating a sentence part the beginning and ending indices are determined and the sentence part annotation is created.

[10] Since the implementation of these annotators was not part of this thesis just a brief overview is given. For more detailed informations about the functionality and implementation of these annotator components please see the official UIMA project page; cf. `http://uima.apache.org/` (accessed on October 26^{th}, 2011)

[11] For better differentiation the annotation of an item is written in upper-case.

An example for a prenntree is depicted in Figure 4.6.

```
(CS
  (S (NP-SB (ART Das) (ADJA neue) (ADJA iPhone))
    (VAFIN ist)
    (AP (PTKNEG nicht) (ADJD schlecht)))
  ($, ,) (KON aber)
  (S (PPER-SB es) (VAFIN ist)
    (NP (PIAT keine) (NN Offenbarung)))
  ($. .)))
```

Figure 4.6 – Example of a penntree as generated by the Stanford parser for the sentence: "*Das neue iPhone ist nicht schlecht, aber es ist keine Offenbarung"*. (The iPhone is not bad, but it is not ground breaking.)

Based on this penntree the following two sentence parts are generated. Note that once the sentence structure is resolved the punctuation marks can be neglected.

(1) *"Das neue iPhone ist nicht schlecht"*
(2) *"aber es ist keine Offenbarung"*

Algorithm 2 shows the pseudo code of the sentence part annotator.

Algorithm 2 - Sentence part annotator

```
1:  procedure GETSENTENCES
2:      for all s ∈ sentenceList do                    ▷ Iterate all sentences s
3:          call GETSENTENCEPARTS(s)
4:      end for
5:  end procedure

6:  procedure GETSENTENCEPARTS(s)                      ▷ Get the sentence parts
7:      penntree ←LexicalizedParser(s)                 ▷ Parse sentence
8:      call RESOLVESENTENCESTRUCTURE(penntree)
9:      for all sp ∈ sentencePartList do               ▷ Iterate all sentence parts sp
10:         new SENTENCEPART← Annotation(sp)           ▷ Create SENTENCEPART annotation
11:     end for
12: end procedure

13: procedure RESOLVESENTENCESTRUCTURE(tree)           ▷ Resolve sentence structure
14:     for all child of tree do
15:         call RESOLVESENTENCESTRUCTURE(child)       ▷ Recursive call
16:         if Chunk of child is "S" then add child to sentencePartList
17:         end if
18:     end for
19: end procedure
```

Figure 4.7 illustrates an example of a SENTENCEPART annotation.

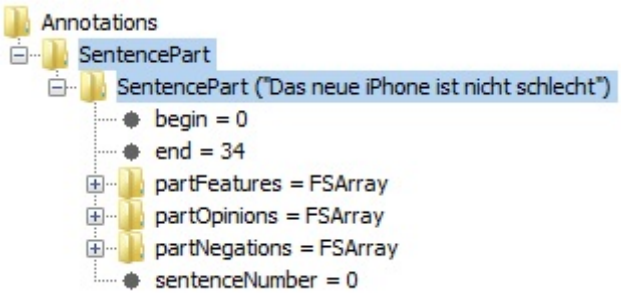


Figure 4.7 – SENTENCEPART annotation for the sentence part:
"Das neue iPhone ist nicht schlecht" (The new iPhone is not bad)

4.3.3.3 Frequent Feature Annotator

Once the SENTENCEPART annotations are created, the features need to be identified. This task is carried out by the frequent feature annotator.

In a first step the frequent feature annotator iterates all single TOKEN annotations and compares them with the k=1 entries in the feature list. In case of a match the next token is analysed. Both tokens together are compared with the k=2 entries in the feature list. By using recursive calls this step is repeated for k>2 until no more matches can be found.

Once a frequent feature is identified, a corresponding STOKEN (Sentiment Token) annotation is created that encompasses each matching token. In addition to creating the STOKEN annotation it is also required to allocate it to the correct sentence part. Therefore, the beginning and ending indices of the STOKEN annotation and the SENTENCEPART annotation are compared and the feature is added to the right sentence part. The POS-tag of the STOKEN annotation is set to "FF" to indicate a frequent feature.

The functionality of the frequent feature annotator is depicted in the following Algorithms 3 and 4.

Algorithm 3 - Frequent feature annotator - Part I

```
1: procedure FINDFREQUENTFEATURES
2:     for all t ∈ tokenList do                          ▷ Iterate all tokens t
3:         f ← DETERMINEFEATURE(t, tokenList)            ▷ f = feature
4:         if f is not NULL then
5:             new STOKEN ← Annotation(f)                ▷ Create STOKEN annotation for f
6:             call ADDTOSENTENCEPART(STOKEN)
7:         end if
8:     end for
9: end procedure
```

Algorithm 4 Frequent feature annotator - Part II

```
10: procedure DETERMINEFEATURE(t, tokenList)                    ▷ Determine feature
11:     if t matches entry in featureList then
12:         f ← Feature(t)                                      ▷ Create feature for t
13:         return f + DETERMINEFEATURE(tokenList.next(), tokenList)   ▷ Recursive call
14:     else
15:         return NULL
16:     end if
17: end procedure

18: procedure ADDTOSENTENCEPART(STOKEN)                         ▷ Add to sentence part
19:     for all sp ∈ sentencePartList do                        ▷ Iterate all sentence parts sp
20:         if STOKEN.Indices between the indices of sp then
21:             add STOKEN to sp                                ▷ Add STOKEN to the sentence part
22:         end if
23:     end for
24: end procedure
```

An example of the results of the frequent feature annotator is depicted in Figure 4.8. The sentence part has one frequent feature, the "iPhone". The STOKEN annotation is appended to the corresponding SENTENCEPART annotation. Besides the basic annotation properties (beginning and ending index) the STOKEN annotation also lists the additional properties: POS-tag (part of speech), orientation and position (showing the position of the word in the sentence).

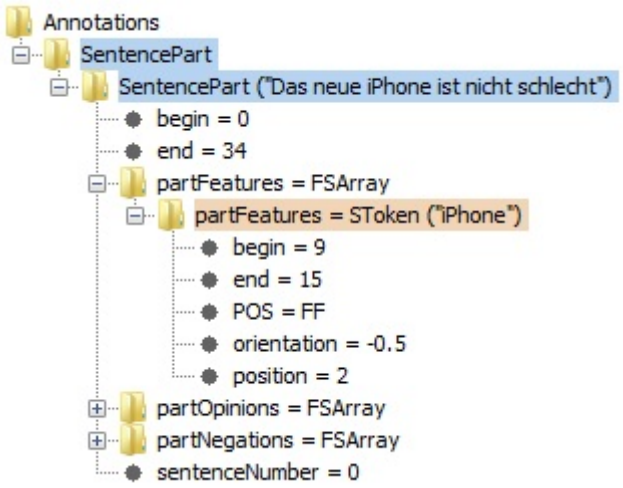


Figure 4.8 – STOKEN annotation for the frequent feature "iPhone".

4.3.3.4 Opinion Word Annotator

The opinion word annotator works very similar to the frequent feature annotator. In a first step the annotator iterates all tokens of a sentence part and checks their POS-tags. For adjectives, verbs and adverbs the tokens are looked up in the opinion word lexicon. In addition the nouns that could not be identified as frequent features are also looked up in the lexicon; e.g. "Offenbarung" (ground breaking). In case of a match the next token is recursively determined.

In contrast to the frequent feature annotator it is necessary to do some processing at this stage. The polarity score and also the orientation an opinion word can change due to other opinion words that appear in the same context. Hence, the single opinion words are aggregated to an opinion phrase. For example the phrase "sehr gut" (very good) expresses a strong positive orientation, while "gut" (good) on its own just states a medium positive orientation. Thus, the adverb "sehr" (very) has an impact on the polarity score of the following opinion words. The following equation depicts the calculation of the polarity score $p(ow)$ of the opinion phrase:

$$p(ow) = \left(\prod_{i}^{N} p(ow_i) \right) \times 2^N, \tag{4.1}$$

where $p(ow_i)$ is the polarity score of an opinion word ow_i that is part of the opinion phrase. N is the total number of single opinion words subsumed by the opinion phrase.

For each opinion phrase[12], consequently for each opinion word if it is not part of a phrase a STOKEN annotation is created. For an opinion word having a positive orientation the POS-tag: "POW" (positive opinion word) is assigned. Vice versa for a negative opinion word the tag: "NOW" is used. In this case, it is necessary to consider the neutral opinion words, which constitute a special case. Since their polarity depends on the context they appear in, no polarity score can be assigned at this point. An annotation of a neutral opinion word gets the POS-tag "OW" (Opinion Word), to indicate that the orientation is not yet determined. The final orientation is calculated in the next annotator. In the last step, analogous to the frequent feature annotator, the corresponding sentence part is determined for each opinion word by comparing the beginning and the ending indices of the annotations.

In addition to the opinion words, the opinion word annotator also detects negation words nw in each sentence part. If a token is identified as a negation word a STOKEN annotation is created having "NW" (Negation Word) as its POS-tag. A negation word is also added to its corresponding SENTENCEPART annotation.

The Algorithm 5 presents the implementation of the opinion word annotator.

[12]For simplicity, in the remainder of this thesis the concept "opinion word" always means both, a single opinion word or an opinion phrase.

Algorithm 5 - Opinion word annotator

```
1:  procedure FINDOPINIONWORDS
2:      for all t ∈ tokenList do                                  ▷ Iterate all tokens t
3:          ow ← DETERMINEOPINION(t, tokenList)                   ▷ ow = opinion word
4:          if ow ≠ NULL then
5:              new STOKEN ← Annotation(ow)          ▷ Create STOKEN annotation for ow
6:              call ADDTOSENTENCEPART(STOKEN)
7:          else
8:              nw ← DETERMINENEGATION(t)                         ▷ nw = negation word
9:              if nw ≠ NULL then
10:                 new STOKEN ← Annotation(nw)       ▷ Create STOKEN annotation for nw
11:                 call ADDTOSENTENCEPART(STOKEN)
12:             end if
13:         end if
14:     end for
15: end procedure

16: procedure DETERMINEOPINION(t, tokenList)                  ▷ Determine opinion phrase
17:     if t matches entry in opinionLexicon then
18:         ow ← t                                            ▷ Create opinion word for t
19:         temp ← DETERMINEOPINION(tokenList.next(), tokenList)      ▷ Recursive call
20:         p̂(ow) = p(ow) ∗ temp(ow) ∗ 2^2                    ▷ Calculate polarity score
21:         return ow + temp
22:     else return NULL
23:     end if
24: end procedure

25: procedure DETERMINENEGATION(t)                            ▷ Determine negation word
26:     if t matches entry in negationLexicon then
27:         return nw ← t                                     ▷ Create negation word for t
28:     else return NULL
29:     end if
30: end procedure

31: procedure ADDTOSENTENCEPART(STOKEN)                       ▷ Add to sentence part
32:     for all sp ∈ sentencePartsList do                     ▷ Iterate all sentence parts sp
33:         if STOKEN.Indices between the indices of sp then
34:             add STOKEN to sp                              ▷ Add STOKEN to the sentence part
35:         end if
36:     end for
37: end procedure
```

Figure 4.10 illustrates the results of the opinion word annotator.

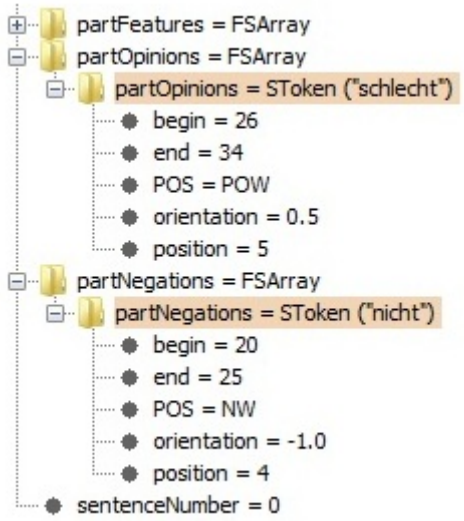


Figure 4.9 – STOKEN annotations for the opinion word "schlecht" (bad) and the negation word "nicht" (not)

4.3.3.5 Sentiment Annotator

The sentiment annotator builds the backbone of the application. Its objective is to append the opinion words to the corresponding features identified in the previous steps and to determine the aggregated orientation of the features. The result of this step is a set of so called SSENTIMENT annotations, each comprising a feature f, the appended opinion word ow, an optional negation word $*nw*$ and the final polarity score $p(f)$.

Before the opinion word can be appended to the feature its orientation needs to be determined. Since a negation word nw can change the orientation of an opinion word ow, both STOKEN categories need to be considered for this subtask. In general, the negation word is related to the closest opinion word. In case that more than one opinion word exist in a sentence part the distance between the negation word and each opinion word is calculated. The opinion word with the minimal distance is chosen. The distance measure is simply defined as the difference between the position of two words in the sentence. To get the final polarity score the opinion word is multiplied by the negation factor that is the polarity score of the negation word $p(nw) = -1$.

$$\hat{p}(ow) = p(ow) \times p(nw) \tag{4.2}$$

Furthermore, the POS-tag of the opinion word needs to be changed based on following rules: (1) in case that the tag is "POW" it is changed to "NOW", (2) vice versa, if the tag is "NOW" it is changed to "POW".

So far, no statement can be made about the orientation of the neutral opinion words. Following the approaches[13] postulated by Hatzivassiloglou and McKeown (1997); Kanayama and Nasukawa (2006) each sentence having neutral opinion words is searched for the appearance of conjunctions. As a reminder, an adjective that follows a coordinating conjunction (e.g. "und" (and)) shares the same orientation as the adjective before the conjunction. For adversative conjunctions (e.g. "aber" (but)) the adjectives share a different orientation. The application distinguishes between two cases: (1) a conjunction appearing within a sentence part and (2) a conjunction joining two parts. In case that no conjunction can be found within a sentence part the neighbouring sentence parts also need to be searched through. For that reason only sentence parts belonging to the same sentence are considered.

Based on the identified conjunction and related opinion words with already known orientation, the polarity of the neutral opinion word can be calculated as:

$$\hat{p}(ow) = p(ow) \times c \times m(ow_{rel}), \tag{4.3}$$

where $p(ow)$ is the initial polarity score of the neutral opinion word ow and c being the effect of the conjunction, e.g. $+1$ for "und" (and) -1 for "aber" (but). $m(ow_{rel})$ is a multiplier based on the polarity score of the related opinion word ow_{rel}:

$$m(ow_{rel}) = \begin{cases} +1 & \text{if } p(ow_{rel}) > 0 \text{ \& tag } \neq \text{ "OW"} \\ -1 & \text{if } p(ow_{rel}) < 0 \text{ \& tag } \neq \text{ "OW"} \\ 0 & \text{otherwise} \end{cases} \tag{4.4}$$

Example: Consider the following sentence:

> *Das Galaxy S2 hat einen schnellen Prozessor und ein brilliantes Display.*
> (The Galaxy S2 has a fast processor and a brilliant display.)

The neutral opinion word "schnellen" (fast) has a initial polarity score of 0.5, "und" (and) is a coordinating conjunction, and "brilliantes" (brilliant) is a positive opinion word. Thus, $c = +1.0$ and $m(\text{billiantes}) = +1.0$. The insertion into the Equation 4.3 leads to following result:

$$\hat{p}(ow) = 0.5 \times 1 \times 1 = 0.5.$$

Once the orientation for each opinion word is determined, the last step is to append the opinion word to the corresponding feature. Again, the distance measure is applied to find the opinion word with the minimal distance to the feature. For each sentence a set of SSENTIMENT annotations (quadruple) is created. For the sample sentence the following two quadruples are created:

- ⟨ Prozessor, schnellen, - , +0.5 ⟩
- ⟨ Display, brilliantes, - , +0.65 ⟩

[13] cf. 2.4.1.2 - Corpus Based Approaches (page 22)

The following Algorithm 6 presents the functionality of the annotator.

Algorithm 6 - SSentiment annotator

procedure FINDOPINIONWORDS
  for all $sp \in sentenceParts$ **do** ▷ Iterate all sentence parts sp
    if nw appears in $part$ **then** ▷ Check if a negation word appears in sp
      call HANDLENEGATION(nw, sp)
    end if
    if neutral ow appears in sp **then** ▷ Check if a neutral opinion word appears in sp
      call HANDLENEUTRAL(ow, sp)
    end if
    for all $ow \in sp.OpinionWords$ **do** ▷ Iterate all opinion words of sp
      call FEATUREORIENTATION(ow, nw, sp)
    end for
  end for
end procedure

procedure HANDLENEGATION(nw, sp) ▷ Handle negation
  for all $ow \in sp.OpinionWords$ **do** ▷ Iterate all opinion words of sp
    find ow with a minimum distance to nw
    $\hat{p}(ow) \leftarrow p(ow) \times p(nw)$ ▷ Calculate polarity score
  end for
end procedure

procedure HANDLENEUTRAL(ow, sp) ▷ Handle neutrals
  if con appears in part **then** ▷ Search for a conjunction within sp
    find ow_{rel} with a minimum distance to con ▷ ow_{rel} = related opinion word
  else if con appears in neighbouring parts **then** ▷ Consider neighbouring parts
    find ow_{rel} with a minimum distance to con
  else return
  end if
  $\hat{p}(ow) \leftarrow p(ow) \times c \times m(ow_{rel})$ ▷ Calculate polarity score
end procedure

procedure FEATUREORIENTATION(ow, nw, sp) ▷ Determine feature orientation
  for all $f \in sp.Features$ **do** ▷ Iterate all features of sp
    find f with a minimum distance to ow
    $p(f) \leftarrow p(ow)$ ▷ Determine polarity score of the feature
    new SSENTIMENT $\leftarrow$ Annotation($f, ow, nw, p(f)$) ▷ Create SSENTIMENT annotation
  end for
end procedure

Figure 4.10 illustrates the results of the annotator. For the sample sentence two SSENTIMENT annotations (quadruples) are identified. In both cases the opinion words are related to the frequent feature "iPhone":

- ⟨ iPhone, schlecht, nicht , +0.5 ⟩
- ⟨ iPhone, Offenbarung, keine , -0.5 ⟩

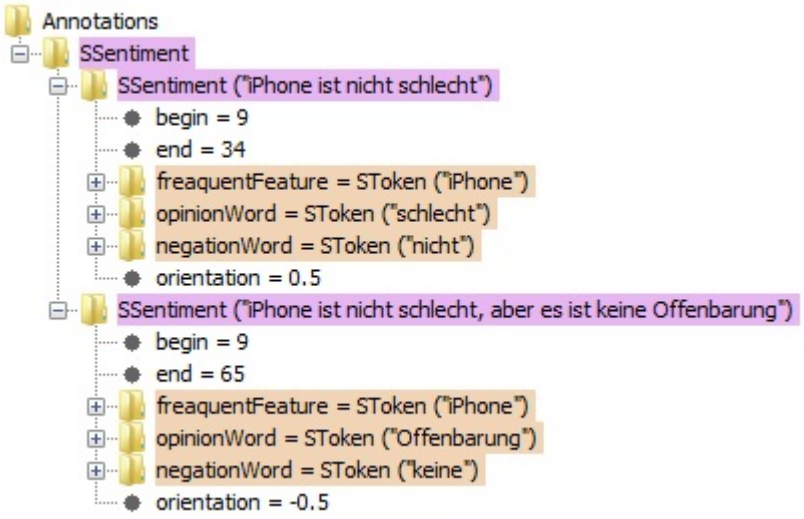


Figure 4.10 – SSENTIMENT annotations for the sentence: "*Das iPhone ist nicht schlecht, aber es ist keine Offenbarung.*" (The iPhone is not bad, but it is not ground breaking.)

4.4 System Output

As mentioned before UIMA provides the CAS Consumers, whose job is the final CAS processing. UIMA automatically generates the output based on the annotations created in prior steps. The output is usually stored in files (XML). In addition, UIMA also provides the graphical representation of the results. Therefore, the annotated text is depicted in a console and the relevant tokens and phrases are highlighted in colour.

Figure 4.11 shows the final output for the two sample sentences introduced above. For this example only the SSENTIMENT annotations are highlighted. Naturally, also the other annotation types can be chosen. Figure 4.12 depicts the TOKEN and SENTENCEPART annotations of a review obtained from Amazon.

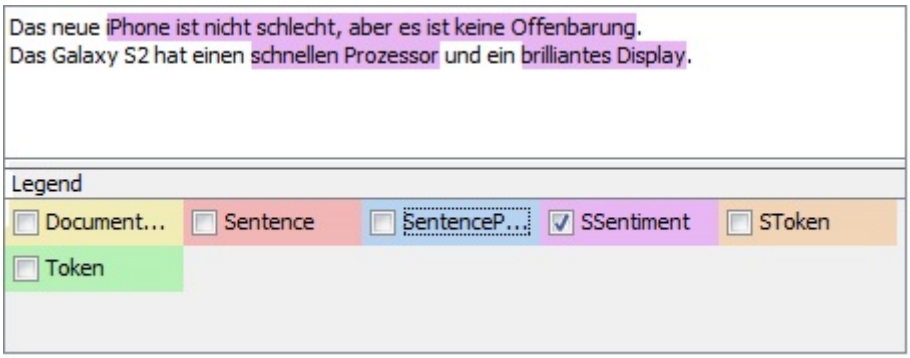


Figure 4.11 – Graphical representation of the SSENTIMENT annotation type

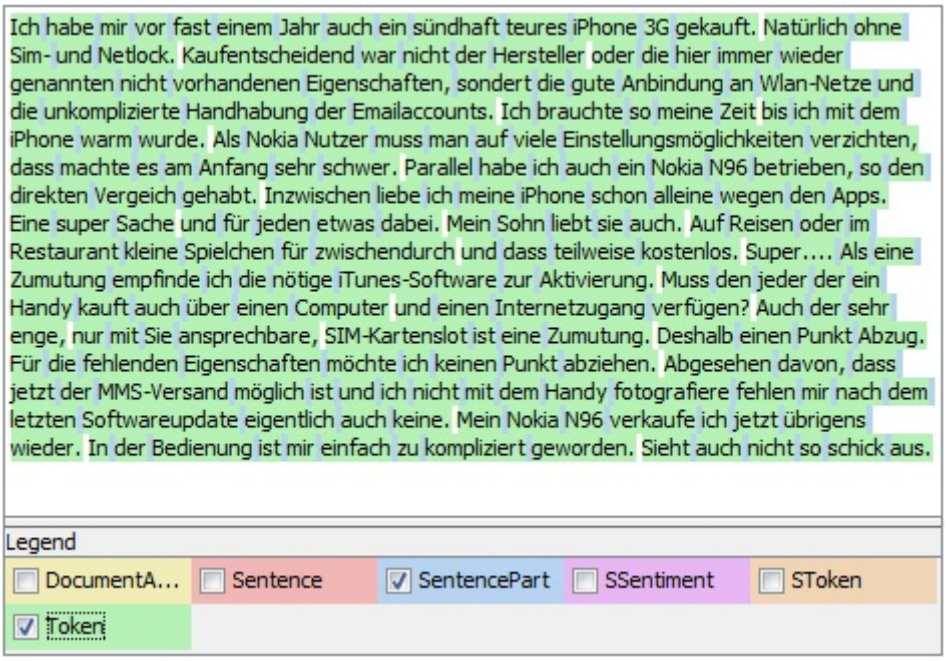


Figure 4.12 – Graphical representation of the SENTENCEPART and TOKEN annotation types

CHAPTER 5

Evaluation

This chapter presents the evaluation of the system as presented in the previous chapter (cf. Chapter 4). At first an overview of the methodology for evaluation in the research area of sentiment analysis are provided (cf. Section 5.1). In Section 5.2 a detailed description of the document corpus and the test sets is provided. Finally, the results of the evaluation of the feature extraction module as well as the opinion mining module are discussed in Section 5.3.

5.1 Evaluation in Sentiment Analysis

Usually, the effectiveness of opinion mining systems is evaluated in experiments, rather than their efficiency. Effectiveness is the ability to correctly classify an item (Tang et al., 2009), while efficiency describes the performance of an algorithm in respect to resources (e.g. time, processing power) needed to classify an item.

5.1.1 Precision and Recall

Precision and recall are used to measure the performance of information retrieval systems in general and opinion mining systems in particular. The confusion matrix shown in Table 5.1 depicts the dimensions of a classifier. The term "true positive" (tp) denotes the number of items that were correctly classified by the machine, while "true negative" (tn) states the correct absence of a result. "False positive" (fp) is the number of those items that were assigned to a class but actually do not belong to it. Vice versa "false negative" (fn) indicates the number of items belonging to a certain class which were not assigned to it.

Table 5.1 – Confusion matrix (Prabowo and Thelwall, 2009)

		Machine	
		positive	negative
Human	true	tp	tn
	false	fp	fn

Given these dimensions, the concepts: precision and recall can be defined as follows: "Precision is the fraction of all correctly classified cases within the systems outputs" (Tang et al., 2009). "Recall is the ratio of correct cases that the system assigned compared to the base of all cases where a human analyst associated either positive or negative sentiment." (Tang et al., 2009). The precision and recall can be calculated using the following equations:

$$\text{Precision} = \frac{C}{A}, \tag{5.1}$$

$$\text{Recall} = \frac{C}{B}, \tag{5.2}$$

where

- A is the number of all cases in which the system classified an item; this comprises the correctly and wrongly classified items (tp, fp)
- B is the number of all cases in which a human analyst classified an item (tp, fn)
- C is the number of correct cases in the system output based on the manual judgement (tp)

The F-measure or balanced F-score combines precision and recall. It is the harmonic mean and referred to as F_1 measure, because recall and precision are evenly weighted:

$$F = \frac{2 \times \text{Precision} \times \text{Recall}}{(\text{Precision} + \text{Recall})}. \tag{5.3}$$

5.1.2 Correlation Coefficient

Another measure for evaluating the system performance is the correlation coefficient, which measures how accurate the sentiment classification is (Tang et al., 2009). It is defined as the relation between the prediction and the actual values. Therefore, a correlation coefficient of 1 indicates perfect linear relation, meaning that the prediction and the actual value or completely accordant. A coefficient of 0 means that the prediction is completely unrelated to the actual value. The correlation coefficient between two variables (x,y) is defined as the covariance $cov(x, y)$ of these variables divided by the product of their standard deviations $s(x)$ and $s(y)$:

$$r(x, y) = \frac{cov(x, y)}{s(x)s(y)}. \tag{5.4}$$

The covariance is expressed as:

$$cov(x, y) = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{N - 1}, \tag{5.5}$$

and the sample standard deviations are defined as:

$$s(x) = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N - 1}}, \quad s(y) = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \bar{y})^2}{N - 1}}. \tag{5.6}$$

In opinion mining x_i is the estimated value of instance i (document, sentence, phrase) and y_i is the actual value of instance i. The total number of instances is given by N. $\bar{x}$ and $\bar{y}$ are the sample means of x_i and y_i

$$\bar{x} = \frac{\sum_{i=1}^{N} x_i}{N}, \quad \bar{y} = \frac{\sum_{i=1}^{N} y_i}{N}. \tag{5.7}$$

5.2 Dataset

The opinion mining system is applied to a huge document corpus that consists of 1,348,006 documents. This document corpus was created during March 2010 and March 2011 and consists of reviews obtained from Amazon and postings extracted from different forums. A detailed overview of the different sources as well as the distribution of the documents can be found in Appendix A.2 of this thesis.

Note that not all documents of this corpus are also suited for sentiment analysis. Especially the forum posts contain a lot of noise. Some entries do not contain any sentiment at all or they are formulated in English. Since the opinion mining system is specified for analysing documents written in German these entries have not been analysed.

In order to be able to evaluate the accuracy of the opinion mining system, four test sets have been manually extracted containing documents focused on two different products: the Apple iPhone and the Blackberry Bold from Research in Motion (RIM). For each product ten Amazon reviews and fifteen forum posts have been extracted. In total the test sets consist of 512 sentences. The Amazon reviews amount to 300 sentences and the forum posts comprise of 212 sentences. Then, every document has manually been analysed and the correct opinions and features have been identified. In addition, the different features and their counts have been documented. In general, test sets are referred to as *gold standard*.

The analysis of the single corpora shows that the Amazon reviews are in general longer than the forum posts, having more than twice as much sentences. On average an Amazon review contains exactly 15 sentences, while a forum entry just embodies about 7.1 sentences. Obviously, the length of a review positively correlates with the number of opinions expressed in the document, making it more reasonable to analyse Amazon reviews.

5.3 Evaluation Results

The feature extraction module and the opinion mining module[1] have been evaluated with respect to accuracy by applying the three evaluation metrics: recall, precision and F-measure in which recall and precision have been equally weighed. In the following the approaches and the results of the modules are presented.

5.3.1 Evaluation of the Feature Extraction Module

For evaluation, the feature extraction module have been executed on the test sets. Several different minimum support (MinSup) values have been examined. The system outputs have then been compared to the manually crafted feature list that represents the reference values. In total 95 features are manually identified. Finally, based on the observations the recall as well as the precision and F-measure can be calculated.

5.3.1.1 Results

Figure 5.1 shows the number of correctly and wrongly extracted frequent features depending on the predefined minimum support (MinSup). The most features could be extracted when using a minimum support of 0.25% . In this case the number of wrongly extracted features is even higher than the number of correctly extracted features. With a minimum support of 1.00% no features have been wrongly identified, but on the other hand only 16 features could be extracted.

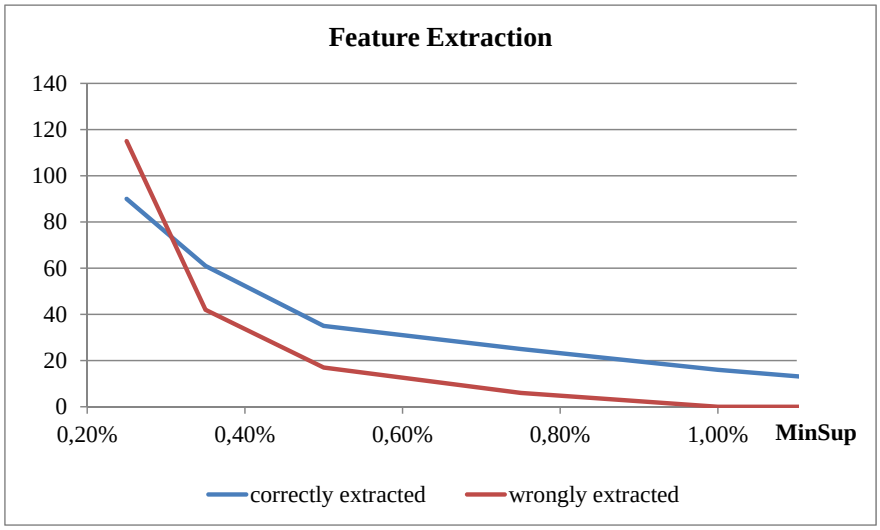


Figure 5.1 – Number of correctly and wrongly extracted features: shows the number of correctly and wrongly extracted features depending on the predefined minimum support.

[1] Note that the opinion lexicon module was not evaluated since it was mostly manually crafted.

Following the theoretical explanation, precision and recall can be calculated as:

$$\begin{aligned} \text{Precision} &= \frac{\text{correctly extracted features}}{\text{correctly extracted features + wrongly extracted features}}, \\ \text{Recall} &= \frac{\text{correctly extracted features}}{\text{manually extracted features}}. \end{aligned} \tag{5.8}$$

The resulting values for recall, precision and F-measure are depicted in Table 5.2.

Table 5.2 – Evaluation of the feature extraction module

MinSup	Recall	Precision	F-Measure
0.25%	0.9474	0.4390	0.6000
0.35%	0.6421	0.5922	0.6162
0.50%	0.3684	0.6731	0.4762
0.75%	0.2632	0.8065	0.3968
1.00%	0.1684	1.0000	0.2883
1.25%	0.0947	1.0000	0.1731

Figure 5.2 illustrates the development of the recall, precision and F-measure depending on the values of the minimum support.

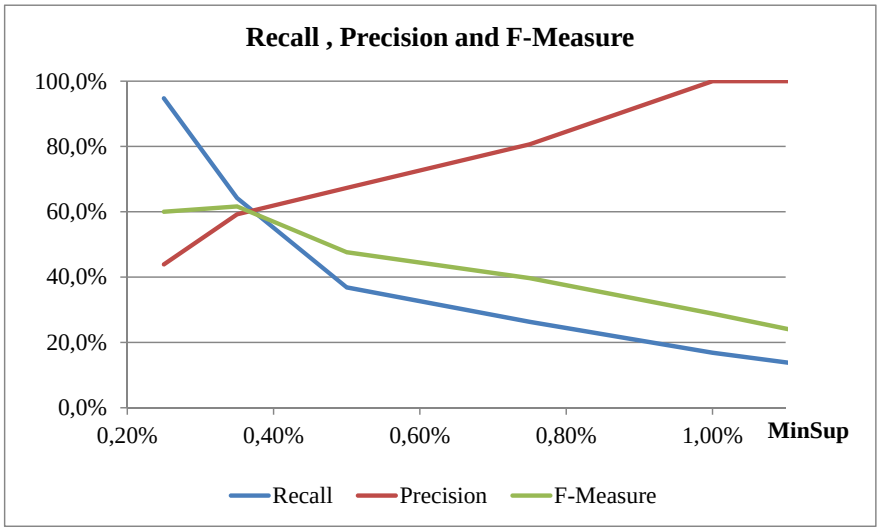


Figure 5.2 – Precision, recall and F-measure of the feature extractor: shows the recall, precision and F-measure values depending on the predefined minimum support.

5.3.1.2 Discussion

The experiments for feature extraction module show that a lower minimum support leads to more correctly extracted features but simultaneously the number of wrongly extracted features strongly increases. Exerting a minimum support of 0.25% the recall almost reaches 95%. But on the other hand the precision only accounts to 43.90%. A minimum support of 1.00% leads to a precision of 100%, meaning that no single feature was extracted by mistake, while the recall only reaches 16.84%. For every minimum support exceeding the threshold of 1% the precision has a constant value of 100%, while the recall constantly declines. By applying a minimum support of 0.35% the precision and recall balance each other, leading to the best F-measure of 61.62%. However, the right choice for the minimum support value depends on the application. In some cases it might be more important to retrieve the features and in other cases it might be more important to take care that not too many features are wrongly extracted.

5.3.2 Evaluation of the Opinion Mining module

The accuracy of the opinion mining module has been determined by comparing the system outputs (SSENTIMENT annotation) to the manually labelled gold standard. For this purpose three scenarios have been evaluated: (1) in the first scenario no preprocessing has been conducted for the documents, (2) in the second scenario the reviews have been preprocessed in order to remove special characters and emoticons, (3) the third scenario represents the ideal case, thus the reviews have been manually preprocessed in order to get well formed documents with correct spelling and punctuation.

5.3.2.1 Results

The following figures (5.3, 5.4, 5.5, 5.6) show the experimental results of the Blackberry and iPhone corpus. Note that the *total number* shows the number of SSENTIMENT annotations that have been identified by the human analyst. The term *Correctly classified* indicates how many annotations have been correctly identified by the opinion mining system, and *wrongly identified* presents the number of wrongly identified annotations. In total 409 annotations have been manually identified. Even though less Amazon reviews than forum entries have been analysed they almost contain twice as much opinions (270 to 139). This is quite understandable since on average, a review is much longer than a forum post and in addition it is fully dedicated to describing a certain object, while a forum is intended to represent a discussion about different topics.

Figure 5.3 illustrates the results of the Amazon reviews of the Blackberry corpus. It can be stated that scenario 3 has lead to the best results, with 90 out of 152 SSENTIMENT annotations correctly identified. In addition the number of wrongly identified features has turned out to be the lowest: 23.

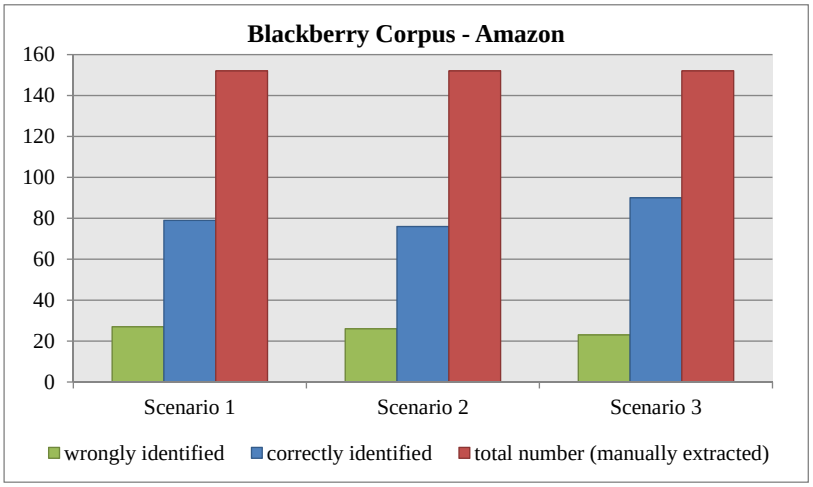


Figure 5.3 – Experimental results of the Blackberry corpus (Amazon reviews): shows the total number of manually identified annotations in contrast to the correctly and wrongly identified annotations by the opinion mining system.

The results of the forum entries can be seen in Figure 5.4. For the forum entries similar trends can be recognized as for the Amazon reviews. Scenario 3 performed the best with 33 out of 69 correctly identified annotations. But in contrast to the Amazon corpus less annotations have been correctly identified on an average level. An interesting observation of this corpus is the fact that in scenario 1 less annotations could correctly be identified than in scenario 2. For the other corpora this effect is reversed.

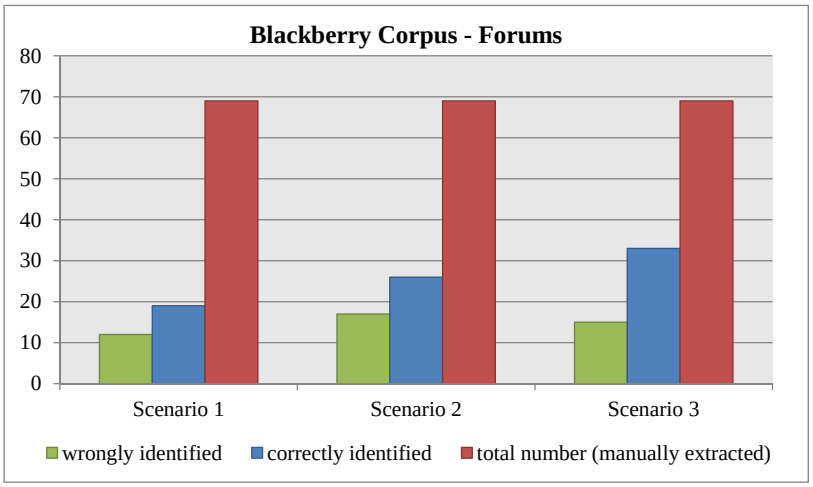


Figure 5.4 – Experimental results of the Blackberry corpus (forum entries): shows the total number of manually identified annotations in contrast to the correctly and wrongly identified annotations by the opinion mining system.

Figure 5.5 illustrates the results of the Amazon reviews of the iPhone corpus. Again the scenario 3 has lead to the best results. For the Amazon reviews 71 of 118 possible annotations have correctly been identified, with only 17 annotations wrongly identified.

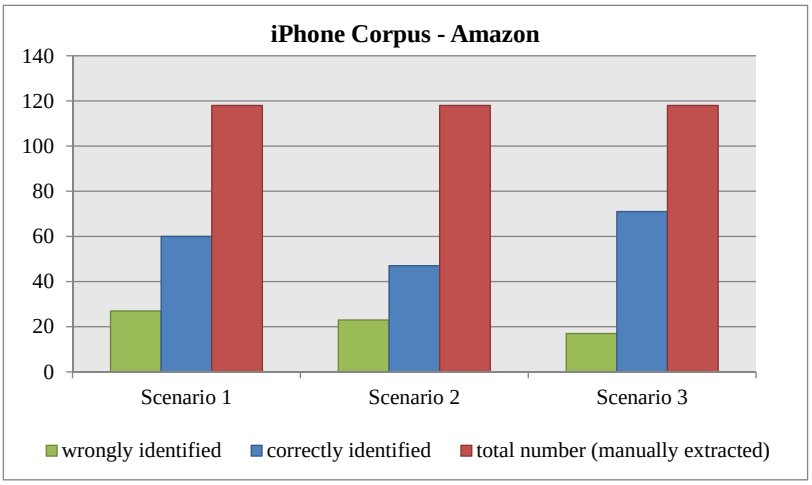


Figure 5.5 – Experimental results of the iPhone corpus (Amazon reviews): shows the total number of manually identified annotations in contrast to the correctly and wrongly identified annotations by the opinion mining system.

The results of the forum entries of the iPhone corpus are depicted in Figure 5.6. Corresponding to the overall trend the scenario 3 performed the best with 37 out of 69 correctly identified annotations.

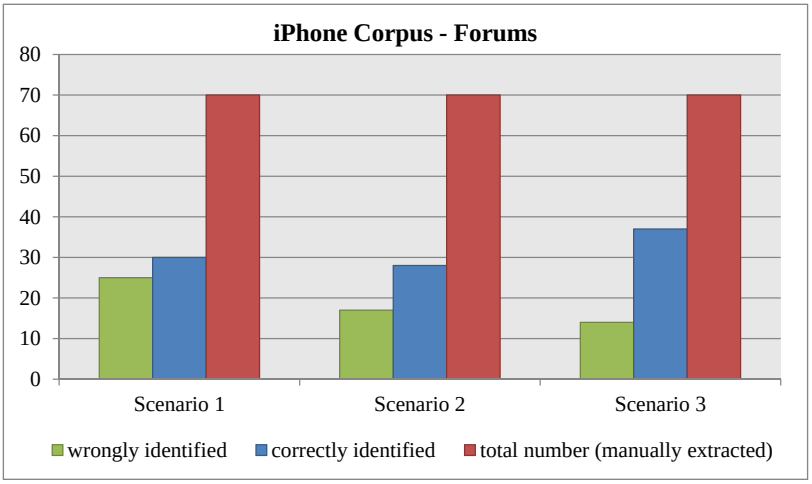


Figure 5.6 – Experimental results of the iPhone corpus (forum entries): shows the total number of manually identified annotations in contrast to the correctly and wrongly identified annotations by the opinion mining system.

Given the numbers presented above, the recall, precision and F-score can be calculated for each scenario. The following tables (5.3, 5.4, 5.5) depict these three evaluation metrics for each scenario: no preprocessing, computational preprocessing and manual preprocessing. In addition, the average values for each product are also presented to provide a better comparability.

Table 5.3 – Recall, precision and F-measure of scenario 1 (no preprocessing)

Product	Corpus	Recall	Precision	F-Measure
Blackberry	Amazon	0.5197	0.7453	0.6124
	Forums	0,2754	0.6129	0.3800
	Average	**0.3975**	**0.67913**	**0.4962**
iPhone	Amazon	0.6897	0.5085	0.5854
	Forums	0.4286	0.5455	0.4800
	Average	**0.4685**	**0.6176**	**0.5327**

Table 5.4 – Recall, precision and F-measure of scenario 2 (manual preprocessing)

Product	Corpus	Recall	Precision	F-Measure
Blackberry	Amazon	0.5000	0.7451	0.5984
	Forums	0.3768	0.6047	0.4643
	Average	**0.4384**	**0.6749**	**0.5314**
iPhone	Amazon	0.3983	0.6714	0.5000
	Forums	0.4000	0.6222	0.4870
	Average	**0.3992**	**0.6468**	**0.4935**

Table 5.5 – Recall, precision and F-measure of scenario 3 (computational preprocessing)

Product	Corpus	Recall	Precision	F-Measure
Blackberry	Amazon	0.5921	0.7965	0.6792
	Forums	0.4783	0.6875	0.5641
	Average	**0.5352**	**0.7420**	**0.6217**
iPhone	Amazon	0.6017	0.8068	0.6893
	Forums	0.5286	0.7255	0.6116
	Average	**0.5651**	**0.7662**	**0.6504**

5.3.2.2 Discussion

The experiments show that the system performs really weak for forum entries. This is true for both product corpora. For the iPhone corpus the average recall of all three scenarios has just reached 45.24% while the precision has scored 63.11%. The Blackberry corpus has even lead to worse results regarding the forum entries. In this case the recall remains beneath 38% and the precision is at 63.50%.

These weak results can be explained due to the structure of a forum. A forum is a platform where people can discuss things and exchange their thoughts about everyday matters. Based on the analysed documents a typical forum post just contains around 7.1 sentences or short statements. Often these statements do not contain any sentiment at all, or it is expressed in a very subtle manner by referring to other posts. Considering the forum structure it is common that a forum entry is related to previous entries. Instead of explicitly mentioning the object in every entry, the object is implicitly inferred by commenting other posts. For the proposed system this characteristic of forums presents a problem, since it lacks the ability to perform cross-document analysis. Instead, each document is considered as one information unit, which is independently analysed. In addition, a forum discussion in some way represents a conversation between "friends". Hence, the writers do not put much emphasis on correct spelling and grammar.

In contrast to the forum corpus the system has lead to better results for the product reviews obtained from Amazon. For the Blackberry corpus an average recall of 53.73% could be reached, while the precision reached 76.23%. If just considering the perfect case (scenario 3) the system even scored a recall of 59.21% and a precision of almost 80%. For this scenario the iPhone corpus has gained slightly better results, a recall of 60.20% and a precision of 80.68%. Although this seems rather low, it is necessary to consider the complex grammatical structure of the German language that makes it hard to analyse sentences consisting of more than ten words. In addition, as presented in Section 2.2.2 (page 12) it is even a challenging task for a human analyst to identify the opinions in a text.

The results imply that well formed documents are needed to produce good results. Regarding the Amazon reviews, without doing any preprocessing, the recall has only reached 50.85% for the iPhone corpus. For the Blackberry corpus the recall is a bit better and reaching 51.97%. But still, these results are quite unsatisfactory. The reason is that the Stanford parser uses probabilistic language models and requires well formed documents. Special characters or missing punctuation marks lead to wrong penntrees, which strongly affect the creation of the SENTENCEPART annotations. Another issue that has great impact on the results is the way a review is noted down. For instance, many Amazon reviews are formulated in format 3[2], combining free text with explicit listings of pros and cons. The sentence detector cannot recognize the listings. Instead, it is handled as one long sentence. As a consequence the sentence parser fails to parse such a sentence and no annotations can be created. Of course, this observation also applies to the forum entries. Without doing any preprocessing the system has scored a recall of only 27.54% for the Blackberry corpus and 42.86% for the iPhone corpus. For scenario 3 the recall has reached 47.83% and 52.86%.

[2] cf. Section 3.2.2 - Feature Extraction (page 51)

An unexpected observation, in some way contrary to the findings mentioned above is that the computational preprocessing of the documents leads to a worse recall than without doing any preprocessing. Even though the precision for scenario 2 on an average value is better, the total F-score is still lower (scenario 1: 51.44% and scenario 2: 51.24%). The experiments show that this effect can be explained due to the strict preprocessing rules, removing every special character from the text. However, further experiments have revealed that some special characters like parenthesis, colons and dashes are indeed needed for the Stanford parser to construct the correct sentence structure, while other characters (#,+,-,$, ...) negatively affect the outcome. In order to produce better results it would be necessary to investigate which special characters negatively affect the parser and which lead to better results. This is subject to further research and development.

CHAPTER 6

Summary and Future Work

Finally, this Master's thesis concludes with a Summary (cf. Section 6.1) and an outlook to future research and development (cf. Section 6.2) for the presented opinion mining system.

6.1 Summary

In this Master's thesis an approach for sentiment analysis is presented. Sentiment analysis, which is also called opinion mining is a research branch of Natural Language Processing (NLP). In contrast to traditional NLP opinion mining is not concerned with mining facts. Instead the objective is to automatically detect sentiment and subjectivity in text documents.

The research area of sentiment analysis itself can be roughly divided into two main research paradigms. *Sentiment classification* strives to determine whether a document, a sentence or just a single phrase expresses a positive respectively a negative sentiment. The objective of *feature-based sentiment analysis* on the other hand is to find product features in a text and to identify which opinion words are related to a specific feature. In some way *feature-based sentiment analysis* can be considered as a specialization of *sentiment classification*. Indeed, many *feature-based sentiment analysis* systems also use *sentiment classification* as an intermediate step. However, since *feature-based sentiment analysis* follows a different, more mature approach it is considered as an own paradigm in the area of sentiment analysis.

Given the enormous growth of user-generated content available in the Internet it is obvious to utilize this vast amount of information in a way to gain intelligence. As a result sentiment analysis has received great attention from both, the research community and the business world in the last years. Sentiment analysis systems can be utilized in many different application fields. Besides review analysis opinion mining applies to business intelligence and marketing. Understanding what people think of products and what they like or dislike is an important factor when it comes to competition. Sentiment analysis can be also used as an enabling technology to improve recommender systems or to detect opinion spam. Due to the growing popularity of social networks opinion mining is increasingly applied to these platforms.

In this thesis a comprehensive overview of the present research in the field of sentiment analysis is provided. Sentiment classification approaches utilize supervised machine learning algorithms (e.g. Naïve Bayes, Maximum Entropy, Support Vector Machines) or unsupervised methods (e.g. Pointwise Mutual Information) to determine the right class of a whole document, a sentence or a single phrase. In most cases the classes are corresponding to POSITIVE or NEGATIVE. *Feature-based sentiment analysis* systems implement probabilistic sequence models (Hidden Markov Models, Conditional Random Fields) or combine machine learning algorithms with comprehensive sets of rules, to identify opinion words and features and in a next step to set them in relation together.

The prototype that has been developed in course of this Master's thesis utilizes a hybrid approach for feature-based sentiment analysis of text documents written in the German language. The system consists of three stages that cover the whole opinion mining process. In the fist stage a comprehensive opinion lexicon is created that comprises positive, negative and neutral opinion words. By applying a bootstrapping method *Sentiment Wortschatz* is searched for opinion words that are relevant for analysing reviews. The second stage implements a feature extractor to find features people are most interested in. This extractor is based on association rules mining. With this in mind a modified Apriori algorithm is used to generate a list of candidate itemsets. The itemsets, whose counts exceed the minimum support are added to the frequent feature list. The last stage represents the core application and covers the actual opinion mining process. That is the identification of opinions and features in a document and the allocation of the opinion words to the related features. The opinion mining module is integrated into the UIMA framework. It consists of seven Analysis Engines that implement the annotators. Each annotator produces annotations for a specific part of a document. The Analysis Engines are executed in a fixed flow, starting with the sentence annotator. The result of this stage and consequently of the whole system is a set of quadruples for each feature. A quadruple compasses the feature, one related opinion word, an optional negation word and the feature orientation.

Finally, the results of the system evaluation are presented. The feature extraction module and the opinion mining module have been evaluated in respect to accuracy by calculating the three evaluation metrics: recall, precision and F-measure. Therefore, a test set of 50 documents has been manually extracted from a huge document corpus. The test set consists of 20 reviews obtained from Amazon and 30 posts collected from different technical forums. In a next step each document has been manually labelled (gold standard). For the feature extraction module several experiments with different values for the minimum support have been conducted. The system output has then been compared to the manually crafted feature list. For the opinion mining module three scenarios have been examined: no preprocessing of the input documents, computation preprocessed in order to remove special characters and manual preprocessing to generate well formed and grammatically correct documents. The experiments show that the system performs rather weak for forum posts. That is because of the characteristic of forums to represent conversations or discussions between users. For the Amazon reviews the system scores much better results, since generally a review is formulated better than a forum entry. However, the complexity of the German language makes it very difficult to automatically parse longer sentences. Hence, well written documents are essential for the system to score good results.

6.2 Future Work

In the future additional improvements and further refinements are needed to deal with the problems researchers face in the area of natural language processing in general and opinion mining in particular. Several potential improvements have been identified:

1. **Dependency parsing:** Probably the most interesting and expedient improvement can be yielded by implementing a dependency parsing system that allows to parse German sentences. By implementing a mature dependency parser the opinion words can be directly associated to the features. In this case it is not necessary to make assumptions about the affiliation to features.

2. **Spelling:** Since the system relies on probabilistic parsers for determining the POS-tags and chunks the performance of the system strongly depends on the quality of the written text. Spelling mistakes, wrong usage of upper-case and lower-case as well as missing punctuation marks strongly corrupt the mining results. Therefore, improvements can be achieved by reducing the effects of spelling mistakes.

3. **Special characters:** In addition to spelling mistakes, special characters like "$, §, &, . . ." and emoticons also negatively affect the outcome of the parsing step. Thus, by training a new parsing model that considers these special characters the system can be extended in a way to be able to analyse emoticons. This is an interesting research area because people use emoticons to express their feelings and opinions towards objects.

4. **Runtime improvements:** The system was constructed in respect to yielding high effectiveness rather than a good runtime performance. Additional effort can be undertaken to improve the runtime of the system.

5. **Identifying feature indicators:** So far, the system supports the identification of frequent features. However, people often use specific indicators like "es" (it), "er" (he) and so on to refer to a feature. A module for identifying these feature indicators and setting them in relation to already found features can strongly improve the accuracy of this system.

6. **Ontologies:** Another improvement can be achieved by implementing ontologies that list the parts and attributes of objects. The utilization of an ontology brings many improvements, because it allows to gain deeper understanding of the feature relationships and to infer which features belong to which product. This is important for analysing product comparisons.

7. **Additional dimensions:** Besides identifying product features and opinions other dimensions can also be of interest for analysis. For instance the identification of the opinion holder. Therefore, the scope of this system can be extended by implementing additional functionalities widening the application area of the system.

Bibliography

Abe, S. (2010), *Support Vector Machines for Pattern Classification*, second edn, Springer-Verlag New York. (Cited on pages 38, 39, and 40.)

Agrawal, R. and Srikant, R. (1994), Fast algorithms for mining association rules in large databases, *in* 'Proceedings of the 20^{th} International Conference on Very Large Data Bases', VLDB '94, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 487–499. (Cited on page 73.)

Ahkter, J. K. and Soria, S. (2010), Sentiment Analysis: Facebook Status Messages, Technical report, Stanford University, The Stanford Natural Language Processing Group. Available online at `http://nlp.stanford.edu/courses/cs224n/2010/reports/ssoriajr-kanej.pdf`; accessed on November 12^{th}, 2011. (Cited on pages 18 and 38.)

Andersson, A. (2000), 'The markov property'. Available online at `https://www.sics.se/~aeg/report/node12.html`; accessed on November 12^{th}, 2011. (Cited on page 59.)

Andreevskaia, A. and Bergler, S. (2006), Mining WordNet For a Fuzzy Sentiment: Sentiment Tag Extraction From WordNet Glosses, *in* 'Proceedings of the European Chapter of the Association for Computational Linguistics', EACL '06, Association for Computational Linguistics, Stroudsburg, PA, USA. (Cited on page 21.)

Archak, N., Ghose, A. and Ipeirotis, P. G. (2007), Show me the money!: Deriving the pricing power of product features by mining consumer reviews, *in* 'Proceedings of the 13^{th} ACM SIGKDD international conference on Knowledge discovery and data mining', KDD '07, ACM, New York, NY, USA, pp. 56–65. (Cited on page 16.)

Asur, S. and Huberman, B. A. (2010), Predicting the future with social media, *in* 'Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology', IEEE Computer Society, Los Alamitos, CA, USA, pp. 492–499. (Cited on page 18.)

Barbosa, L. and Feng, J. (2010), Robust sentiment detection on Twitter from biased and noisy data, *in* 'Proceedings of the 23^{rd} International Conference on Computational Linguistics: Posters', COLING '10, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 36–44. (Cited on page 18.)

Berger, A. L., Pietra, V. J. D. and Pietra, S. A. D. (1996), 'A maximum entropy approach to natural language processing', *Computational Linguistics* **22**(1), 39–71. (Cited on page 37.)

Bhuiyan, T., Xu, Y. and Josang, A. (2009), State-of-the-Art Review on Opinion Mining from Online Customers' Feedback, *in* 'Proceedings of the 9^{th} Asia-Pacific Complex Systems Conference', COMPLEX '09, pp. 385–390. (Cited on pages 11, 44, and 49.)

Brants, T. (2000), TnT - A Statistical Part-of-Speech Tagger, *in* 'Proceedings of the sixth Applied Natural Language Processing Conference', ANLP '00, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 224 – 231. (Cited on page 24.)

Brill, E. (1992), A simple rule-based part of speech tagger, *in* 'Proceedings of the 3^{rd} conference on Applied natural language', ANLC '92, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 152–155. (Cited on page 24.)

Brill, E. (1994), Some advances in transformation-based part of speech tagging, *in* 'Proceedings of the twelfth National Conference on Artificial Intelligence', Vol. 1 of *AAAI '94*, American Association for Artificial Intelligence, Menlo Park, CA, USA, pp. 722–727. (Cited on page 24.)

Cardie, C., Farina, C. and Bruce, T. (2006), Using natural language processing to improve eRulemaking: project highlight, *in* 'Proceedings of the 2006 international conference on Digital Government Research', dg.o '06, ACM, New York, NY, USA, pp. 177–178. (Cited on page 18.)

Chaovalit, P. and Zhou, L. (2005), Movie Review Mining: a Comparison between Supervised and Unsupervised Classification Approaches, *in* 'Proceedings of the 38^{th} Annual Hawaii International Conference on System Sciences', Vol. 04 of *HICSS '05*, IEEE Computer Society, p. 112 cont. (Cited on page 16.)

Choi, Y., Breck, E. and Cardie, C. (2006), Joint extraction of entities and relations for opinion recognition, *in* 'Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing', EMNLP '06, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 431–439. (Cited on page 51.)

Choi, Y. and Cardie, C. (2010), Hierarchical sequential learning for extracting opinions and their attributes, *in* 'Proceedings of the ACL 2010 Conference Short Papers', ACLShort '10, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 269–274. (Cited on pages 62 and 63.)

Choi, Y., Cardie, C., Riloff, E. and Patwardhan, S. (2005), Identifying sources of opinions with conditional random fields and extraction patterns, *in* 'Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing', HLT '05, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 355–362. (Cited on pages 8, 51, and 62.)

Conrad, J. G. and Schilder, F. (2007), Opinion mining in legal blogs, *in* 'In Proceedings of the 11^{th} International Conference on Artificial Intelligence and Law', ICAIL '07, ACM Press, New York, NY, USA, pp. 231–236. (Cited on page 18.)

Cortes, C. and Vapnik, N. V. (1995), 'Support-Vector Networks', *Machine Learning* **20**(3), 273–297. (Cited on page 38.)

Dave, K., Lawrence, S. and Pennock, D. M. (2003), Mining the peanut gallery: opinion extraction and semantic classification of product reviews, *in* 'Proceedings of the twelth international conference on World Wide Web', WWW '03, ACM, New York, NY, USA, pp. 519–528. (Cited on page 10.)

De Marneffe, M.-C. and Manning, C. D. (2008), Stanford typed dependencies manual, Technical report, Stanford University, The Stanford Natural Language Processing Group. Available online at `http://nlp.stanford.edu/software/dependencies_manual.pdf`; accessed on November 16^{th}, 2011. (Cited on pages 27 and 28.)

Della Pietra, S., Della Pietra, V. and Lafferty, J. (1997), 'Inducing Features of Random Fields', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19**(4), 380–393. (Cited on page 38.)

Ding, X., Liu, B. and Yu, P. S. (2008), A holistic lexicon-based approach to opinion mining, *in* 'Proceedings of the international conference on Web search and web data mining', WSDM '08, ACM, New York, NY, USA, pp. 231–240. (Cited on page 30.)

Esuli, A. and Sebastiani, F. (2005), Determining the semantic orientation of terms through gloss classification, *in* 'Proceedings of the 14^{th} ACM international conference on Information and knowledge management', CIKM '05, ACM, New York, NY, USA, pp. 617–624. (Cited on pages 20 and 21.)

Esuli, A. and Sebastiani, F. (2006), SentiWordNet: A publicly available lexical resource for opinion mining, *in* 'Proceedings of the fifth Conference on Language Resources and Evaluation', LREC '06, pp. 417–422. (Cited on pages 10 and 21.)

Etzioni, O., Cafarella, M., Downey, D., Popescu, A.-M., Shaked, T., Soderland, S., Weld, D. S. and Yates, A. (2005), 'Unsupervised named-entity extraction from the web: an experimental study', *Artifical Intelligence* **165**, 91–134. (Cited on page 55.)

Forrester Research (2011*a*), 'European online retail forecast, 2010 to 2015'. Available online at `http://www.forrester.com/rb/research`; accessed on July 04^{th}, 2011. (Cited on page 1.)

Forrester Research (2011*b*), 'Us online retail forecast, 2010 to 2015'. Available online at `http://www.forrester.com/rb/research`; accessed on July 04^{th}, 2011. (Cited on page 1.)

Gamon, M. (2004), Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis, *in* 'Proceedings of the 20^{th} international conference on Computational Linguistics', COLING '04, Association for Computational Linguistics, Stroudsburg, PA, USA. (Cited on page 41.)

Gamon, M. and Aue, A. (2005), Automatic identification of sentiment vocabulary: Exploiting low association with known sentiment terms, *in* 'Proceedings of the ACL Workshop on Feature Engineering for Machine Learning in Natural Language Processing', ACL '05, pp. 57–64. (Cited on page 44.)

Gamon, M., Aue, A., Corston-Oliver, S. and Ringger, E. (2005), Pulse: Mining Customer Opinions from Free Text, *in* 'Proceedings of the sixth International Symposium on Intelligent Data Analysis', Vol. 3646 of *IDA '06*, Springer-Verlag, Heidelberg, Berlin, Germany, pp. 121–132. (Cited on pages 45 and 48.)

Ghahramani, Z. (2004), Unsupervised Learning, *in* O. Bousquet, U. von Luxburg and G. Rätsch, eds, 'Advanced Lectures on Machine Learning', Vol. 3176 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 72–112. (Cited on page 32.)

Gindl, S., Weichselbraun, A. and Scharl, A. (2010), Cross-Domain Contextualization of Sentiment Lexicons, *in* 'Proceedings of the 19^{th} European Conference on Artifical Intelligence', ECAI '10, IOS Press, Amsterdam, The Netherlands, pp. 771–776. (Cited on page 3.)

Go, A., Huang, L. and Bhayani, R. (2009), Twitter Sentiment Analysis, Technical report, Stanford University, The Stanford Natural Language Processing Group. Available online at `http://nlp.stanford.edu/courses/cs224n/2009/fp/3.pdf`; accessed on November 12^{th}, 2011. (Cited on page 18.)

Gruhl, D., Guha, R., Kumar, R., Novak, J. and Tomkins, A. (2005), The predictive power of online chatter, *in* 'Proceedings of the eleventh ACM SIGKDD international conference on Knowledge Discovery in Data Mining', KDD '05, ACM, New York, NY, USA, pp. 78–87. (Cited on page 17.)

Gupta, G. (2006), *Introduction to Data Mining with Case Studies*, Prentice-Hall Of India Pvt. Ltd. (Cited on pages 29, 33, 53, and 73.)

Hatzivassiloglou, V. and McKeown, K. R. (1997), Predicting the semantic orientation of adjectives, *in* 'Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics', EACL '97, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 174–181. (Cited on pages 22 and 83.)

Henrich, V. and Hinrichs, E. (2010), Standardizing wordnets in the iso standard lmf: Wordnet-lmf for germanet, *in* 'Proceedings of the 23^{rd} International Conference on Computational Linguistics', COLING '10, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 456–464. (Cited on page 21.)

Hu, M. and Liu, B. (2004*a*), Mining and summarizing customer reviews, *in* 'Proceedings of the tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining', KDD '04, ACM, New York, NY, USA, pp. 168–177. (Cited on pages 19, 20, 30, 49, 50, 55, 56, 58, 60, and 62.)

Hu, M. and Liu, B. (2004*b*), Mining opinion features in customer reviews, *in* 'Proceedings of the 19^{th} national conference on Artifical intelligence', AAAI'04, AAAI Press, pp. 755–760. (Cited on pages 23, 30, and 71.)

Jin, W., Ho, H. H. and Srihari, R. K. (2009), 'OpinionMiner: A novel machine learning system for web opinion mining and extraction', *Human Factors* pp. 1195–1203. (Cited on pages 59, 60, and 62.)

Jindal, N. and Liu, B. (2008), Opinion spam and analysis, *in* 'Proceedings of the international conference on Web search and Web data mining', WSDM '08, ACM, New York, NY, USA, pp. 219–230. (Cited on pages 17 and 30.)

Joachims, T. (1998), Text categorization with Support Vector Machines: Learning with many relevant features, *in* 'Proccedings of the tenth EUROPEAN Conference on Machine Learning', Vol. 1398 of *ECML '98*, Springer-Verlag, Berlin, Heidelberg, Germany, pp. 137–142. (Cited on page 38.)

Kamps, J., Marx, M., Mokken, R. and de Rijke, M. (2004), Using wordnet to measure semantic orientations of adjectives, *in* 'Proceedings of the fourth International Conference on Language Resources and Evaluation', Vol. IV of *LREC '04*, pp. 1115–1118. (Cited on page 20.)

Kanayama, H. and Nasukawa, T. (2006), Fully automatic lexicon expansion for domain-oriented sentiment analysis, *in* 'Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing', EMNLP '06, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 355–363. (Cited on pages 22, 23, and 83.)

Kennedy, A. and Inkpen, D. (2006), 'Sentiment Classification of Movie Reviews Using Contextual Valence Shifters', *Computational Intelligence* **22**(2), 110 –125. (Cited on page 41.)

Kim, E., Gilbert, S., Edwards, M. J. and Graeff, E. (2009), 'Detecting Sadness in 140 Characters: Sentiment Analysis of Mourning Michael Jackson on Twitter', *Web Ecology Project* . Available online at `http://www.webecologyproject.org/wp-content/uploads/2009/08`; accessed on November 16^{th}, 2011. (Cited on page 18.)

Kim, S.-M. and Hovy, E. (2004), Determining the Sentiment of Opinions, *in* 'Proceedings of the 20^{th} International Conference on Computational Linguistics', COLING '04, Association for Computational Linguistics, Stroudsburg, PA, USA. (Cited on page 8.)

Kim, S.-M. and Hovy, E. (2006*a*), Automatic identification of pro and con reasons in online reviews, *in* 'Proceedings of the COLING/ACL on Main Conference', COLING-ACL '06, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 483–490. (Cited on page 45.)

Kim, S.-M. and Hovy, E. (2006*b*), Identifying and Analyzing Judgment Opinions, *in* 'Proceedings of the Main Conference on Human Language Technology of the North American Chapter of the Association of Computational Linguistics', HLT-NAACL '06, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 200–207. (Cited on pages 7, 8, 13, and 51.)

Kobayashi, N., Inui, K. and Matsumoto, Y. (2007), Extracting Aspect-Evaluation and Aspect-Of Relations in Opinion Mining, *in* 'Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning', EMNLP-CoNLL '07, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 1065–1074. (Cited on page 8.)

Koppel, M. and Schler, J. (2006), 'The Importance of Neutral Examples for Learning Sentiment', *Computational Intelligence* **22**(2), 100–109. (Cited on page 42.)

Kotsiantis, S. B. (2007), 'Supervised Machine Learning: A Review of Classification Techniques', *Informatica* **31**, 249–268. (Cited on page 32.)

Lafferty, J. D., McCallum, A. and Pereira, F. C. N. (2001), Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data, *in* 'Proceedings of the 18^{th} International Conference on Machine Learning', ICML '01, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 282–289. (Cited on page 61.)

Liddy, E. D. (2001), Natural Language Processing, *in* 'Encyclopedia of Library and Information Science', 2 edn, Marcel Decker, Inc. (Cited on page 9.)

Liu, B. (2008), Opinion Mining, *in* 'Proceedings of the 17^{th} international World Wide Web Conference', WWW '08, Department of Computer Science; University of Illinois Chicago, Beijing, China. (Cited on page 10.)

Liu, B. (2010), *Handbook of natural language processing*, Chapman & Hall/CRC machine learning & pattern recognition series, second edn, Taylor and Francis, chapter "Sentiment Analysis and Subjectivity". (Cited on pages 2, 7, 8, 19, 22, 23, 31, 32, 33, 42, 43, 49, 51, 53, 56, 57, 58, and 71.)

Liu, B., Hu, M. and Cheng, J. (2005), Opinion observer: analyzing and comparing opinions on the Web, *in* 'Proceedings of the 14^{th} International Conference on World Wide Web', WWW '05, ACM, New York, NY, USA, pp. 342–351. (Cited on pages 51, 53, and 54.)

Manning, C. D., Raghavan, P. and Schütze, H. (2008), *Introduction to Information Retrieval*, Cambridge University Press, chapter "Text classification & Naive Bayes". (Cited on pages 33, 34, and 35.)

Manning, C. D. and Schütze, H. (1999), *Foundations of Statistical Natural Language Processing*, first edn, The MIT Press, chapter "Part-of-Speech Tagging". (Cited on page 28.)

Marcus, M. P., Marcinkiewicz, M. A. and Santorini, B. (1993), 'Building a large annotated corpus of English: The Penn Treebank', *Computational Linguistics* **19**, 313–330. (Cited on page 24.)

Matsumoto, S., Takamura, H. and Okumura, M. (2005), Sentiment Classification Using Word Sub-sequences and Dependency Sub-trees, *in* 'Advances in Knowledge Discovery and Data Mining', Vol. 3518 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 21–32. (Cited on page 41.)

Miao, Q., Li, Q. and Zeng, D. (2010), Mining Fine Grained Opinions by Using Probabilistic Models and Domain Knowledge, *in* '2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology', pp. 358 – 365. (Cited on page 62.)

Mishne, G. and Glance, N. (2006), Predicting movie sales from blogger sentiment, *in* 'AAAI 2006 Spring Symposium on Computational Approaches to Analysing Weblogs', AAAI-CAAW '06, AAAI Press, Menlo Park, California, USA, pp. 155–158. (Cited on page 17.)

Mullen, T. and Collier, N. (2004), Sentiment analysis using support vector machines with diverse information sources, *in* 'Proceedings of Conference on Empirical Methods in Natural Language Processing', EMNLP' 04, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 412–418. (Cited on page 41.)

Niazi, R., Cercone, N. and An, A. (2011), A Computational Linguistics Approach to Feature-based Sentiment Analysis, Technical report, York University. Available online at `http://www.cse.yorku.ca/techreports/2011/CSE-2011-03.pdf`; accessed on November 16^{th}, 2011. (Cited on pages 23 and 56.)

Nigam, K., Lafferty, J. and McCallum, A. (1999), Using maximum entropy for text classification, *in* 'Workshop on Machine Learning for Information Filtering', IJCAI '99, pp. 61–67. (Cited on pages 37 and 38.)

Nigam, K., McCallum, A. K., Thrun, S. and Mitchell, T. (2000), 'Text Classification from Labeled and Unlabeled Documents using EM', *Machine Learning* **39**, 103–134. (Cited on page 47.)

Pak, A. and Paroubek, P. (2010), Twitter as a Corpus for Sentiment Analysis and Opinion Mining, *in* 'Proceedings of the seventh Conference on International Language Resources and Evaluation', LREC '10, European Language Resources Association (ELRA), pp. 1320–1326. (Cited on page 18.)

Pang, B. and Lee, L. (2004), A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts, *in* 'Proceedings of the 42^{nd} Annual Meeting on Association for Computational Linguistics', ACL '04, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 271–278. (Cited on pages 29, 30, 37, 41, 44, 45, 46, and 47.)

Pang, B. and Lee, L. (2005), Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales, *in* 'Proceedings of the 43^{rd} Annual Meeting on Association for Computational Linguistics', ACL '05, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 115–124. (Cited on pages 30 and 41.)

Pang, B. and Lee, L. (2008), 'Opinion mining and sentiment analysis', *Foundations and Trends in Information Retrieval* **2**(1-2), 1–135. (Cited on pages 1, 10, 11, 13, 15, 17, and 29.)

Pang, B., Lee, L. and Vaithyanathan, S. (2002), Thumbs up?: Sentiment *C*lassification using machine learning techniques, *in* 'Proceedings of the ACL Conference on Empirical Methods in Natural Language Processing', Vol. 10 of *EMNLP '02*, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 79–86. (Cited on pages 11, 12, 16, 29, 37, 38, and 41.)

Pew Internet & American Life Project (2010), 'Online product research'. Available online at `http://www.pewinternet.org/Reports/2010/Online-Product-Research.aspx`; accessed on July 04^{th}, 2011. (Cited on page 2.)

Popescu, A.-M. and Etzioni, O. (2005), Extracting product features and opinions from reviews, *in* 'Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing', HLT '05, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 339–346. (Cited on page 55.)

Prabowo, R. and Thelwall, M. (2009), 'Sentiment analysis: A combined approach', *Journal of Informetrics* **3**, 143–157. (Cited on page 88.)

Qi, L. and Chen, L. (2010), A linear-chain crf-based learning approach for web opinion mining, *in* 'Proceedings of the eleventh International Conference on Web Information Systems Engineering', WISE'10, Springer-Verlag, Berlin, Heidelberg, pp. 128–141. (Cited on pages 61 and 62.)

Qi, L. and Chen, L. (2011), Comparison of Model-Based Learning Methods for Feature-Level Opinion Mining, *in* '2011 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology', pp. 265 – 273. (Cited on page 62.)

Qiu, G., Liu, B., Bu, J. and Chen, C. (2009), Expanding domain sentiment lexicon through double propagation, *in* 'Proceedings of the 21^{st} international jont conference on Artifical intelligence', Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 1199–1204. (Cited on page 23.)

Remus, R., Quasthoff, U. and Heyer, G. (2010), SentiWS-a Publicly Available German-Language Resource for Sentiment Analysis, *in* 'Proceedings of the 7^{th} International Conference on Language Resources and Evaluation', LREC '10, p. 1168–1171. (Cited on pages 21 and 66.)

Rifkin, R. and Klautau, A. (2004), 'In defense of one-vs-all classification', *Journal of Machine Learning Research* **5**, 101–141. (Cited on page 42.)

Schiller, A., Teufel, S., Stöckert, C. and Thielen., C. (1999), Guidelines für das Tagging deutscher Textcorpora mit STTS (kleines und großes Tagset), Technical report, Universität Stuttgart - Institut für maschinelle Sprachverarbeitung; Universität Tübigen - Insitut für Sprachwissenschaft. Available online at `http://www.sfs.uni-tuebingen.de/`

resources/stts-1999.pdf; accessed on November 16^{th}, 2011. (Cited on pages 72 and 115.)

Senecal, S., Kalczynski, P. J. and Nantel, J. (2005), 'Consumer's decision-making process and their online shopping behavior: a clickstream analysis', *Journal of Business Research* **58**(11), 1599–1608. (Cited on page 1.)

Shoebottom, P. (2010), 'The differences between english and german'. Available online at http://esl.fis.edu/grammar/langdiff/german.htm; accessed on October 24^{th}, 2011. (Cited on page 68.)

Sutton, C. and Mccallum, A. (2006), *An Introduction to Conditional Random Fields for Relational Learning*, MIT Press. (Cited on page 59.)

Tang, H., Tan, S. and Cheng, X. (2009), 'A survey on sentiment detection of reviews', *Expert Systems with Applications* **36**(7), 10760–10773. (Cited on pages 3, 10, 11, 16, 29, 42, 45, 87, and 88.)

Tjong Kim Sang, E. F. and Buchholz, S. (2000), Introduction to the conll-2000 shared task: chunking, *in* 'Proceedings of the second workshop on Learning language in logic and the fourth conference on Computational natural language learning - Volume 7', ConLL '00, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 127–132. (Cited on page 26.)

Toutanova, K., Klein, D., Manning, C. D. and Singer, Y. (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, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 173–180. (Cited on pages 24 and 28.)

Toutanova, K. and Manning, C. D. (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 38^{th} Annual Meeting of the Association for Computational Linguistics', Vol. 13 of *EMNLP '00*, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 63–70. (Cited on pages 24 and 28.)

Tsur, O., Davidov, D. and Rappoport, A. (2010), A Great Catchy Name: Semi-Supervised Recognition of Sarcastic Sentences in Product Reviews, *in* 'Proceedings of the fourth International AAAI Conference on Weblogs and Social Media', number 9 *in* 'ICWSM '10', p. 162. (Cited on page 14.)

Tsuruoka, Y. and Tsujii, J. (2003), Training a naive bayes classifier via the EM algorithm with a class distribution constraint, *in* 'Proceedings of the seventh Conference on Natural language learning', Vol. 4 of *CONLL '03*, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 127–134. (Cited on page 47.)

Tumasjan, A., Sprenger, T., Sandner, P. and Welpe, I. (2010), Predicting elections with Twitter: What 140 characters reveal about political sentiment, *in* 'Proceedings of the fourth International AAAI Conference on Weblogs and Social Media', AAAI '10, pp. 178–185. (Cited on page 18.)

Turney, P. D. (2002), Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews, *in* 'Proceedings of the 40^{th} Annual Meeting on Association for Computational Linguistics', ACL '02, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 417–424. (Cited on pages 31, 42, 43, 44, and 56.)

Turney, P. D. and Littman, M. L. (2003), 'Measuring praise and criticism: Inference of semantic orientation from association', *ACM Transactions on Information Systems TOIS* **21**(4), 37. (Cited on page 20.)

UIMA Overview & SDK Setup (2010). Available online at `https://uima.apache.org/downloads/releaseDocs/2.3.0-incubating/docs/pdf/overview_and_setup.pdf`; accessed on October 26^{th}, 2011. (Cited on pages 66 and 67.)

Vapnik, V. N. (1979), 'Estimation of Dependences Based on Empirical Data [in Russian]', *Nauka, Moscow* . English translation: Springer Verlag, New York, 1982. (Cited on page 38.)

Vapnik, V. N. (1998), *Statistical Learning Theory*, John Wiley and Sons. (Cited on pages 38 and 41.)

Wang, D. and Liu, Y. (2011), A Cross-corpus Study of Unsupervised Subjectivity Identification based on Calibrated EM, *in* 'Proceedings of the second Workshop on Computational Approaches to Subjectivity and Sentiment Analysis', WASSA '11, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 161–167. (Cited on pages 45, 47, and 48.)

Wiebe, J. (1994), 'Tracking Point of View in Narrative', *Computational Linguistics* **20**(2), 233–287. (Cited on page 10.)

Wiebe, J. and Riloff, E. (2005), 'Creating subjective and objective sentence classifiers from unannotated texts', *Computational Linguistics and Intelligent Text Processing* **3406**, 486–497. (Cited on page 45.)

Wilson, T., Wiebe, J. and Hoffmann, P. (2005), Recognizing contextual polarity in phrase-level sentiment analysis, *in* 'Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing', HLT '05, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 347–354. (Cited on page 3.)

Yang, H. and Callan, J. (2005), Near-duplicate detection for eRulemaking, *in* 'Proceedings of the 2005 national conference on Digital Government Research', dg.o '05, Digital Government Society of North America, pp. 78–86. (Cited on page 18.)

Zhang, H. (2004), The Optimality of Naive Bayes, *in* 'Proceedings of the 17^{th} International Florida Artificial Intelligence Research Society Conference', FLAIRS '04, AAAI Press, Menlo Park, California, USA. (Cited on page 36.)

Zhao, J., Liu, K. and Wang, G. (2008), Adding redundant features for CRFs-based sentence sentiment classification, *in* 'Proceedings of the Conference on Empirical Methods in Natural Language Processing', EMNLP '08, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 117–126. (Cited on page 62.)

Zhao, L. and Li, C. (2009), Ontology Based Opinion Mining for Movie Reviews, *in* 'Proceedings of the 3^{rd} International Conference on Knowledge Science, Engineering and Management', KSEM '09, Springer-Verlag, Berlin, Heidelberg, pp. 204–214. (Cited on page 16.)

Zhuang, L., Jing, F. and Zhu, X.-Y. (2006), Movie review mining and summarization, *in* 'Proceedings of the 15^{th} ACM international conference on Information and knowledge management', CIKM '06, ACM, New York, NY, USA, pp. 43–50. (Cited on page 16.)

APPENDIX A

Resources

A.1 Stuttgart-Tübigen Tagset (STTS)

In total the Stuttgart-Tübigen Tagset (STTS) comprises 54 POS-tags. 48 of these are real POS-tags. The remaining tags are used for foreign language expressions (FM), non-words containing special characters (XY), first part of compositions (TRUNC) and for punctuation marks ($.,$,,$(). In the following Table A.1 each tag is described and examples are given.

Table A.1 – The Stuttgart-Tübigen Tagset (STTS) (Schiller et al., 1999)

POS	DESCRIPTION	EXAMPLES
ADJA	attributives Adjektiv	[das] große [Haus]
ADJD	adverbiales oder prädikatives Adjektiv	[er fährt] schnell, [er ist] schnell
ADV	Adverb	schon, bald, doch
APPR	Präposition; Zirkumposition links	in [der Stadt], ohne [mich]
APPRART	Präposition mit Artikel	im [Haus], zur [Sache]
APPO	Postposition	[ihm] zufolge, [der Sache] wegen
APZR	Zirkumposition rechts	[von jetzt] an
ART	bestimmter oder unbestimmter Artikel	der, die, das, ein, eine
CARD	Kardinalzahl	zwei [Männer], [im Jahre] 1994
FM	Fremdsprachliches Material	[Er hat das mit “] A big fish [” übersetzt]
ITJ	Interjektion	mhm, ach, tja

POS	DESCRIPTION	EXAMPLES
KOUI	unterordnende Konjunktion mit "zu" und Infinitiv	um [zu leben], anstatt [zu fragen]
KOUS	unterordnende Konjunktion mit Satz	weil, daß, damit, wenn, ob
KON	nebenordnende Konjunktion	und, oder, aber
KOKOM	Vergleichskonjunktion	als, wie
NN	normales Nomen	Tisch, Herr, [das] Reisen
NE	Eigennamen	Hans, Hamburg, HSV
PDS	substituierendes Demonstrativpronomen	dieser, jener
PDAT	attribuierendes Demonstrativpronomen	jener [Mensch]
PIS	substituierendes Indefinitpronomen	keiner, viele, man, niemand
PIAT	attribuierendes Indefinitpronomen ohne Determiner	kein [Mensch], irgendein [Glas]
PIDAT	attribuierendes Indefinitpronomen mit Determiner	[ein] wenig [Wasser], [die] beiden [Brüder]
PPER	irreflexives Personalpronomen	ich, er, ihm, mich, dir
PPOSS	substituierendes Possessivpronomen	meins, deiner
PPOSAT	attribuierendes Possessivpronomen	mein [Buch], deine [Mutter]
PRELS	substituierendes Relativpronomen	[der Hund ,] der
PRELAT	attribuierendes Relativpronomen	[der Mann ,] dessen [Hund]
PRF	reflexives Personalpronomen	sich, einander, dich, mir
PWS	substituierendes Interrogativpronomen	wer, was
PWAT	attribuierendes Interrogativpronomen	welche [Farbe], wessen [Hut]
PWAV	adverbiales Interrogativ- oder Relativpronomen	warum, wo, wann, worüber, wobei
PAV	Pronominaladverb	dafür, dabei, deswegen, trotzdem
PTKZU	"zu" vor Infinitiv	zu [gehen]
PTKNEG	Negationspartikel	nicht
PTKVZ	abgetrennter Verbzusatz	[er kommt] an, [er fährt] rad
PTKANT	Antwortpartikel	ja, nein, danke, bitte
PTKA	Partikel bei Adjektiv oder Adverb	am [schönsten], zu [schnell]
TRUNC	Kompositions-Erstglied	An- [und Abreise]
VVFIN	finites Verb, voll	[du] gehst, [wir] kommen [an]
VVIMP	Imperativ, voll	komm [!]
VVINF	Infinitiv, voll	gehen, ankommen
VVIZU	Infinitiv mit "zu", voll	anzukommen, loszulassen
VVPP	Partizip Perfekt, voll	gegangen, angekommen

POS	DESCRIPTION	EXAMPLES
VAFIN	finites Verb, aux	[du] bist, [wir] werden
VAIMP	Imperativ, aux	sei [ruhig !]
VAINF	Infinitiv, aux	werden, sein
VAPP	Partizip Perfekt, aux	gewesen
VMFIN	finites Verb, modal	dürfen
VMINF	Infinitiv, modal	wollen
VMPP	Partizip Perfekt, modal	gekonnt, [er hat gehen] können
XY	Nichtwort, Sonderzeichen enthaltend	3:7, H2O, D2XW3
$,	Komma	,
$.	Satzbeendende Interpunktion	. ? ! ; :
$(	sonstige Satzzeichen; satzintern	- [,]()

A.2 Document Corpus

In total the corpus consists of 1,348,006 documents, obtained from different sources. In the following Table A.2 the sources as well as the distribution of the documents is depicted. The sources are sorted by the quantity of documents in descending order .

Table A.2 – Sources and distribution of the document corpus

Source	Quantity
Reviews	
http://www.amazon.de	10573
Forum posts	
http://www.android-hilfe.de	745297
http://www.androidpit.de	265448
http://www.gulli.com	85111
http://www.computerbase.de	76773
http://www.hardwareluxx.de	41383
http://www.forum-3dcenter.org	30333
http://www.chip.de	22007
http://www.raidrush.ws	20858
http://www.geizhals.at	16342
http://www.xda-zone.de	4788
http://www.ubuntuusers.de	3982
http://www.hartware.de	3955
http://www.planet3dnow.de	2793
http://www.parents.at	2514
http://www.winfuture-forum.de	2352
http://www.tomshardware.de	2213
http://www.wcm.at	1800
http://www.administrator.de	1652
http://www.pocketnavigation.de	1342
http://www.mcseboard.de	1270
http://www.pcwelt.de	1091
http://www.notebookforum.at	1079
http://www.xps-forum.de	954
http://www.notebookjournal.de	916
http://www.heise.de	524
http://www.zdnet.de	248
http://www.prad.de	152
others	256