Semantics-Driven Aspect-Based Sentiment Analysis

Semantics-Driven Aspect-Based Sentiment Analysis

Semantiek-gedreven Aspect-gebaseerde Sentiment Analyse

Thesis

to obtain the degree of Doctor from the
Erasmus University Rotterdam
by command of the
rector magnificus

Prof.dr. R.C.M.E. Engels

and in accordance with the decision of the Doctorate Board

The public defense shall be held on

November 16, 2018 at 13:30 hours

by

Kars Iddo Maik Schouten
born in Ede, The Netherlands.

Erasmus University Rotterdam

Doctoral Committee

Doctoral dissertation supervisors:	Prof.dr. F.M.G. de Jong
	Prof.dr.ir. R. Dekker
Other members:	Prof.dr. D. Fok
	Prof.dr. M. Thelwall
	Prof.dr. P. Buitelaar
Co-promotor:	Dr.ir. F. Frasincar

Erasmus Research Institute of Management - ERIM
The joint research institute of the Rotterdam School of Management (RSM)
and the Erasmus School of Economics (ESE) at the Erasmus University Rotterdam
Internet: http://www.erim.eur.nl

ERIM Electronic Series Portal: http://repub.eur.nl/

ERIM PhD Series in Research in Management, 453
ERIM reference number: EPS-2018-453-LIS
ISBN: 978-90-5892-527-5

Design: PanArt, `www.panart.nl`
Cover: K.I.M. Schouten

This publication (cover and interior) is printed by Tuijtel on recycled paper, BalanceSilk®.
The ink used is produced from renewable resources and alcohol free fountain solution.
Certifications for the paper and the printing production process: Recycle, EU Ecolabel, FSC®, ISO14001.
More info: www.tuijtel.com

SIKS Dissertation Series No. 2018-23
The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems. (`http://www.siks.nl`)

Table of Contents

List of Figures

Chapter 7

Chapter 8

Chapter 9

List of Tables

Chapter 6

Chapter 7

Chapter 8

Chapter 9

List of Algorithms

Chapter 1

Introduction*

Humans are social creatures. From the very beginning, humans thrived when in a community. In groups, each person offers a unique contribution to the success of the group while relying on others to do the same. Because an individual person has limited knowledge and few options, he or she needs the help of others. Typical human behavior thus includes sharing information and working towards a common goal. Therefore, sharing information, and communication in general, has been - and still is - the cornerstone of human civilization. This is illustrated by the Word Wide Web, a relatively recent, but already essential part of society.

The volume of social Web content is growing rapidly and is likely to increase even more in the near future. This is driven by the current generation of Web applications, the nearly limitless connectivity, and this innate desire for sharing information, in particular among younger generations. People using the Web are constantly invited to share their opinions and preferences with the rest of the world, which has led to an explosion of opinionated blogs, reviews of products and services, and comments on virtually everything. This type of web-based content is increasingly recognized as a source of data that has added value for multiple application domains.

For ages, governments and businesses alike have been struggling to determine the opinions of their target communities and audiences. Now, for the first time, people voluntarily publish their opinions on the World Wide Web for anyone to see. This social Web allows for timely retrieval of feedback on products, stocks, policies, etc., and many of the desired data, which was hard to come by in the past, is now readily available. This starkly contrasts with traditional surveys and questionnaires

*This chapter is partly derived from the article "K. Schouten and F. Frasincar. Survey on Aspect-Level Sentiment Analysis. *IEEE Transactions on Knowledge and Data Engineering*, volume 28, number 3, pages 813 - 830. IEEE, 2016."

that participants, who were often reluctant and unmotivated, had to fill in, resulting in sub-optimal information. Notwithstanding the easier retrieval of feedback, social Web data still has its challenges, such as accounting for representativeness of the responses, the anonymous nature of some platforms which influences the content of feedback, and the counterfeit nature of some feedback (e.g., spam reviews).

Many individuals are influenced by the opinionated materials they find on the Web. This is especially true for product reviews, which have been shown to influence buying behavior (Bickart and Schindler, 2001). Additionally, information provided by individuals on the Web is regarded as more trustworthy than information provided by the vendor (Bickart and Schindler, 2001). From a producer's point of view, every person is a potential customer. Hence, knowing their likes and dislikes can be of great help in developing new products (van Kleef et al., 2005), as well as managing and improving existing ones (Pang and Lee, 2008). Furthermore, understanding how the information in, for example, product reviews interacts with the information provided by companies enables the latter to take advantage of these reviews. For instance, businesses can control the amount of company-provided product information to improve sales (Chen and Xie, 2008). In fact, opinions on the Web have become a resource to be harnessed by companies, just like the traditional word-of-mouth (Goldsmith and Horowitz, 2006). Furthermore, there is an increasing interest in linking sentiment to financial indicators, such as stock prices (Cortis et al., 2017; Nguyen et al., 2015).

1.1 Aspect-Based Sentiment Analysis

While the large number of available reviews virtually ensures that all relevant parts of the entity under review are properly covered, manually reading each and every review is not feasible. Even when taking the time to read all reviews, thus avoiding the selection bias, it is practically impossible to retain every relevant piece of information and correctly summarize it. Sentiment analysis aims to solve this issue, as it is concerned with the development of algorithms that can automatically extract sentiment information from a set of reviews. The field of sentiment analysis operates at the intersection of information retrieval, natural language processing, and

artificial intelligence (Pang and Lee, 2008). This varied background has led to the use of different terms for similar concepts. A term often used is 'opinion mining', a denotation coming from the data mining and information retrieval community. The main task of opinion mining is to determine the opinions of a group of people regarding a particular topic. The term 'sentiment analysis' is also widely used and comes from the natural language processing domain. Here, the focus lies on determining the sentiment expressed in a given piece of text. The term subjectivity analysis is sometimes regarded as encompassing opinion mining and sentiment analysis, as well as related tasks (Tsytsarau and Palpanas, 2012), but also as a sub-task of opinion mining and sentiment analysis (Liu, 2012). Because sentiment is related to the human emotions communicated by a certain expression, it can be hard to describe, measure, and quantify. For this reason, most sentiment analysis is performed using sentiment polarity, i.e., reducing sentiment to a value on the positive-negative axis. This can range from a simple binary system where only the 'positive' and 'negative' values exist, or more fine-grained variants, such as using a real number between -1 and +1 to measure sentiment polarity. As this habit of using sentiment polarity is ubiquitous, the term 'sentiment' is used interchangeably with 'sentiment polarity' throughout this thesis.

Though possibly used for slightly different tasks or different angles, all these terms represent the same area of research. This field of research, labeled as opinion mining, sentiment analysis, or subjectivity analysis, studies the phenomena of opinion, sentiment, evaluation, appraisal, attitude, and emotion (Liu, 2012). For ease of reference, all these terms are often referred to as opinion or sentiment, even though they are not exact synonyms. An opinion can be defined as a "judgment or belief not founded on certainty or proof" (Collins English Dictionary, 2015). In this sense, 'opinion' is the opposite of 'fact', so statements expressing an opinion are subjective, while factual statements are objective. Sentiment, as a dimension, is orthogonal to the objective-subjective dimension (Kim and Hovy, 2004), as it is closely related to attitude and emotion and used to convey an evaluation of the topic under discussion. Because of this orthogonality, a given piece of text can be classified in one of four quadrants. It can be subjective or objective, as well as with or without sentiment. This is illustrated in Figure 1.1.

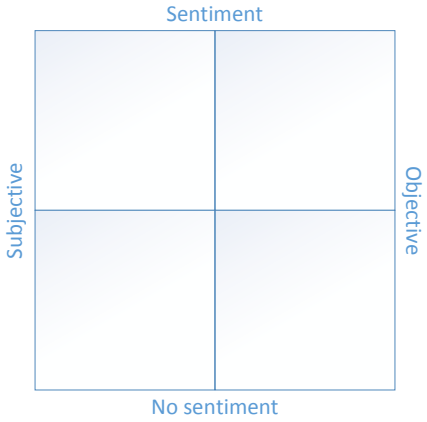

Figure 1.1: Visual representation of the two orthogonal dimensions: subjective vs. objective and sentiment vs. no sentiment

For example, people may have varying opinions, but no particular sentiment, on what color a certain dress is* in a subjective statement such as "Others think it looks like a blue and black dress, but it really is a white and gold dress." By contrast, the statement "Some persons looked at the dress and saw a blue and black one, while others were convinced it was white with gold" is purely objective and also without sentiment. Statements conveying sentiment can be both subjective and objective as well. For example "The blue and black dress is the most beautiful" is a subjective statement with sentiment, while "But... the dress is sold out!" is an objective statement that conveys a strong sentiment. In light of the above discussion, we will use the term sentiment analysis throughout this survey, as it best captures the research area under investigation.

With the above discussion in mind, finding sentiment can be formally defined as finding the quadruple (s, g, h, t) (Liu, 2012), where s represents the sentiment, g represents the target object for which the sentiment is expressed, h represents the holder (i.e., the one expressing the sentiment), and t represents the time at which

*See for instance http://www.wired.com/2015/02/science-one-agrees-color-dress/

the sentiment was expressed. Note that most approaches focus only on finding the pair (s, g). The target can be an entity, such as the overall topic of the review, or an aspect of an entity, which can be any characteristic or property of that entity. This decision is made based on the application domain at hand. For example, in product reviews the product itself is usually the entity, while all things related to that product (e.g., price, quality, etc.) are aspects of that product. Aspect-based sentiment analysis is concerned not just with finding the overall sentiment associated with an entity, but also with finding the sentiment for the aspects of that entity that are discussed. Some approaches use a fixed, predetermined list of aspects, while others freely discover aspects from the data.

Both sentiment and target can be expressed explicitly or remain implicit. When explicitly mentioned, a sentiment or target is literally in the text, while implicit expressions of sentiment or target have to be inferred from the text, which sometimes even requires additional context or domain knowledge. For example, "This hotel is fantastic" is an example of a sentence with an explicit entity and an explicit sentiment, while "The service is great" expresses a similar explicit sentiment, but with an explicit aspect of an entity as its target. On the other hand, "I could not sleep because of the noise" is an example that illustrates an implicit sentiment with an implicit target: one expects to be able to sleep well but according to the sentence, this expectation was not met. Therefore, this sentence can be seen as illustrating a negative sentiment.

The simplification of sentiment to the polarity of sentiment is illustrative for the field of sentiment analysis. While the concepts of sentiment, aspect, target, opinion, explicitness, and implicitness are rather refined, in practice these nuances are often lost as researchers tend to 'work with the available data'. This means that algorithms are not developed for capturing the phenomena described by these concepts, but for predicting the provided annotations, whatever they may be. It is all too often assumed that the provided annotations equal the concept of interest, which may or may not be true. Hence, it is better to define each algorithm by the data set it is trained on, rather than the label it receives (e.g., a sentiment analysis algorithm). For instance, the algorithms we developed earlier were labeled as being for 'implicit feature detection' (Schouten and Frasincar, 2014), where 'feature' is used to mean 'aspect'. Meanwhile, similar algorithms were later developed with the label 'aspect category detection' (Schouten et al., 2017c). When reading the subsequent chapters

of this thesis, one should take the notion of 'working with the available data' into account as it also applies to the various methods presented here.

1.2 Research Objectives

Whenever textual data is to be used, one cannot go without at least some form of natural language processing (NLP). In its most basic form, textual data is treated as a simple string of characters, where the characters themselves are represented by numbers according to the character encoding used. This low-level information is not particularly useful for most tasks, which is why the first, most basic NLP step is to tokenize the textual data, grouping the characters into words and making words the main unit of information. Words can then be grouped as sentences if necessary. Tokenization is straightforward for most languages and is usually not expensive in terms of computational time. Hence, the earlier models for many text-related tasks, including sentiment analysis, are based on which words appear in the text. An additional operation, often performed when working with words, is stemming or lemmatization. This simplifies words to their root (stemming) or to their dictionary form (lemmatization). And while this removes the information contained in the inflection of the word, it has the advantage of grouping semantically similar words together. After all, encountering a verb in the past tense or in the present tense, for instance, does not make a real difference in most situations.

Models using the frequency of words in a text are known as Bag-of-Words models, a term that allegedly first appeared in (Harris, 1954), albeit in a somewhat derogatory sense, where Dr. Harris notes that "language is not merely a bag of words". The Bag-of-Words model is indeed a simple representation of the text as the order of the words, and thus the grammar within a text, is lost. Still, it has shown to be a robust model that provides a hard-to-beat baseline for many text analysis tasks, including sentiment analysis. The Bag-of-Words model is often used as the basis for a Vector Space Model (Salton, 1979), where each position in the vector represents a word and the value at that position represents the frequency of that word in the text. The Vector Space Model is best known for its strong performance in document

classification for information retrieval, and is also related to popular metrics such as tf-idf (Salton and McGill, 1983) and BM25 (Robertson et al., 1995).

Meanwhile, the rise in computing power now makes it possible to process more data and use machine learning algorithms that are computationally more intensive. This has led to software packages that can perform advanced NLP tasks, such as grammatical parsing, discourse parsing, coreference resolution, and word sense disambiguation. The resulting information is typically not available when looking only at the Bag-of-Words model, but can be of great use when performing sentiment analysis. However, the harder the NLP task, the greater the risk of errors being propagated into the sentiment analysis algorithm. Therefore, while high-level linguistic information should intuitively improve performance, in practice this is often not the case. Another problem is that deep linguistic information is sometimes too specific, which leads to overfitting, and sometimes it is not specific enough, which means the information does not actually help when finding aspects or determining sentiment. This search for information within the text that is predictive for the problem at hand can be defined as the search for intra-textual discriminants.

Rather than trying to extract higher-level linguistic information from within the text, it is also possible to jump to the highest level of linguistic processing: pragmatics. "Pragmatics is concerned with the study of meaning as communicated by the speaker (or writer) and interpreted by a listener (or reader)" (Yule, 1996). Thus, for textual data, it is the message conveyed to the reader that is important, not the meaning of the words themselves. A good example of this difference is illustrated in sarcasm and irony detection, where the conveyed message is often in contrast with the literal meaning of the text. An example from the domain of sentiment analysis is implicit sentiment detection, where an objective statement such as "my pizza was cold when delivered" conveys a negative sentiment because you know the writer expected it to be hot. To interpret text on a pragmatic level, retrieving knowledge from outside the text is required, as well as formulating and encoding this knowledge in such a way that it can be used to successfully perform tasks such as aspect-based sentiment analysis. Searching beyond or outside the text itself for information that is predictive for a particular problem can be defined as the search for extra-textual discriminants. This leads to the overall research question that underlies this dissertation:

Which intra-textual and extra-textual discriminants are beneficial for performing aspect-based sentiment analysis?

The natural starting point for text mining in general, and aspect-based sentiment analysis specifically, is to examine a simple model and start delving into more complex ones from there. Hence, the initial questions deal with the use of word co-occurrences, moving towards more syntactic and semantic features, while the last question deals with the use of extra-textual information.

Question 1: How can the co-occurrence data between words and aspects be exploited to predict the presence of aspects within a text?

Simply looking at which words and aspects appear together can already be very informative, and this dissertation explores a simple method to find aspects using their co-ocurrences with regular words. To understand the extent to which this is useful, several extensions are also covered, including the use of an additional classifier to enable the prediction of multiple aspects per sentence, something that is not possible in the basic version of the method. Two more advanced methods that use co-occurrence data are also explored, one being an unsupervised method that uses spreading activation from a manual seed set based on co-occurrence data of words in the text, and the other being a supervised method that augments the use of co-ocurrences between words and aspects with grammatical dependencies.

Question 2: What is the information gained when using various intra-textual discriminants for aspect sentiment analysis?

Moving from aspect detection to sentiment analysis for aspects, this question investigates the added value of a wide range of possible intra-textual discriminants. By computing the Information Gain of each discriminant with respect to sentiment classification, and using that information to perform feature selection for a basic classification algorithm, the impact of each discriminant on the sentiment classification performance is determined.

Question 3: How can employing the discourse structure of a text lead to improved sentiment classification of aspects?

While most features are linked to words, or n-grams, it can be beneficial to take a step back and look at the structure of a text at a discourse level. This goes beyond the grammatical structure of sentences, as it looks at the structure of the argumentation, pointing out which phrases are essential in order to grasp the meaning of a text, and which phrases are in a supporting role. A natural way of utilizing discourse information is to give more weight to parts of a text that are relevant to classify the sentiment for a given aspect. A sentence can contain multiple aspects with differing sentiment values, so to properly classify the sentiment for each aspect, the sentence has to be segmented into parts that relate to the contained aspects. The discourse structure might be informative when doing that.

Question 4: In what way can extra-textual information be encoded and put to use for aspect-based sentiment analysis?

As humans have a large knowledge base to depend on when reading and interpreting texts, exploiting the use of external knowledge for both aspect detection and sentiment classification for aspects seems to be a promising research direction. By employing ontologies, it is possible to use its formal structure to reason over the concepts it contains and arrive at conclusions about sentiment that are not apparent from the text alone. This can be particularly useful for factual statements that carry sentiment by virtue of their deviation from standard expectations. Since manually creating domain ontologies is time consuming, the semi-automatic creation of these domain ontologies for sentiment classification will also be addressed.

1.3 Contributions

There are four main contributions of this dissertation. The first is a structured overview of the field of aspect-based sentiment analysis that includes all important methods in the field. Previously, surveys covering sentiment analysis devoted a section to aspect-based sentiment analysis, but the field has grown considerably such that a dedicated literature survey is desired. One of the main conclusions of the literature survey is that in order for the field to mature, common data sets need to be accessible to the whole community and evaluation metrics need to be standardized.

This enables a proper comparison of the various algorithms developed. Subsequently, the aim throughout this dissertation has been to always use freely available data sets and commonly used evaluation metrics, enabling a fair comparison against existing methods.

The second contribution is an in-depth exposition of a software framework design that enables the development and evaluation of text mining algorithms in general, and aspect-based sentiment analysis methods in particular. The framework uses common software design patterns and a discussion of the design choices is also presented. Furthermore, an implementation of the described software framework is provided freely on GitHub.

As a third main contribution, this dissertation provides experimental results for a wide range of approaches for aspect detection and sentiment analysis of aspects, ranging from straightforward co-occurrence-based methods to more advanced approaches such as using rhetorical structure theory for sentiment analysis of aspects and knowledge-driven methods for both aspect detection and sentiment analysis. In particular, this dissertation shows the value of complementing traditional machine learning approaches with external knowledge, encoded in, e.g., ontologies for aspect-based sentiment analysis.

Last, this dissertation shows how sources of external knowledge can be acquired using a semi-automatic ontology creation process. As knowledge-driven methods require less training data, they are already less dependent on human-labeled data sets. However, the need for extra-textual resources still requires a lot of manual labor. Limiting the human factor in creating these resources by using a semi-automatic method for ontology creation alleviates the need for manual labor even more.

1.4 Declaration of Contribution

This section lists the contribution of the dissertation author to each of the chapters.

Chapter 1: The work for this chapter was performed by the dissertation author, with feedback from promotors and co-promotor.

Chapter 2: The work for this chapter was performed by the dissertation author, with feedback from the co-promotor.

Chapter 3: The work for this chapter was performed by the dissertation author, with feedback from both co-promotor and one of the promotors.

Chapter 4: The work for this chapter was mainly performed by the dissertation author, with feedback from the co-promotor. The chapter includes notions from work done with the help of students as well as from work done with feedback from one of the promotors.

Chapter 5: The author of this dissertation has contributed significantly to this chapter by (1) programming the software environment in which the developed algorithms are running, taking care of things like algorithm setup, data preprocessing, and evaluation; (2) rewriting and summarizing the findings into a journal paper; (3) giving feedback together with the co-promotor to provide ideas to implement and guide the development process.

Chapter 6: The work for this chapter was mainly performed by the dissertation author, with feedback from the co-promotor. Some findings were taken from an initial implementation, with mixed results, by a group of students that were under the supervision of the dissertation author and the co-promotor.

Chapter 7: The author of this dissertation made a significant contribution to this chapter by (1) programming the software environment in which the developed algorithms are running, taking care of things like algorithm setup, data preprocessing, and evaluation; (2) giving feedback together with the co-promotor to provide ideas to implement and guide the development process; (3) editing the text for a conference publication.

Chapter 8: The work for this chapter was mainly performed by the dissertation author, with feedback from both one of the promotors and the co-promotor.

Chapter 9: The work for this chapter was mainly performed by the dissertation author, with feedback from the co-promotor. The ontology creation algorithm was co-developed with the second author.

Chapter 10: The work for this chapter was performed by the dissertation author, with feedback from promotors and co-promotor.

1.5 Outline

In the next chapter, a literature study is presented covering the field of aspect-based sentiment analysis. This provides a structured overview of the existing work and will serve as the backdrop for the various methods we have developed that are presented in the later chapters. After the literature study, the evaluation methodology is discussed, including the design and implementation of the software framework used to develop and evaluate the methods presented in the subsequent chapters.

Chapters 4 through 8 all present developed methods that address aspect detection, sentiment analysis of aspects, or both. In general, the earlier methods are more statistics-driven and word-based, while later methods use more high-level information, such as discourse information or an external knowledge base. In Chapter 4, a method based on co-occurrence data is given that addresses the problem of aspect detection, both explicit and implicit. Chapter 5 investigates different ways of using co-occurrence data to find aspects. Hence, chapters 4 and 5 address the first research question.

With aspects being determined, Chapter 6 continues with an in-depth look at which features of the textual data are important for sentiment analysis of aspects, addressing the issue raised in the second research question. This analysis provides experimental data that support the move to more semantic methods.

Following that, Chapter 7 steps away from the features covered in the previous chapter and discusses the third research question: the use of discourse information for sentiment analysis of aspects. The last research question, regarding the use of extra-textual information, is covered in chapters 8 and 9. Chapter 8 presents a hybrid method, complementing a traditional machine learning approach with the use of ontologies and logical reasoning for both aspect detection and sentiment analysis of aspects. Chapter 9 showcases a knowledge-driven method for sentiment analysis that complements the traditional Bag-of-Words model. Furthermore, a semi-automatic ontology creation algorithm is explored to mitigate the amount of effort required to build a domain ontology. The last chapter draws overall conclusions and provides an outlook for future research.

Chapter 2

Survey on Aspect-Level Sentiment Analysis*

THE *field of sentiment analysis, in which sentiment is gathered, analyzed, and aggregated from text, has seen a lot of attention in the last few years. This survey focuses on aspect-level sentiment analysis, where the goal is to find and aggregate sentiment on entities mentioned within documents or aspects of them. An in-depth overview of the current state-of-the-art is given, showing the tremendous progress that has already been made in finding both the target, which can be an entity as such, or some aspect of it, and the corresponding sentiment. Aspect-level sentiment analysis yields very fine-grained sentiment information which can be useful for applications in various domains. Current solutions are categorized based on whether they provide a method for aspect detection, sentiment analysis, or both. Furthermore, a breakdown based on the type of algorithm used is provided. For each discussed study, the reported performance is included. To facilitate the quantitative evaluation of the various proposed methods, a call is made for the standardization of the evaluation methodology that includes the use of shared data sets. Semantically-rich concept-centric aspect-level sentiment analysis is discussed and identified as one of the most promising future research directions.*

*This chapter is based on the article "K. Schouten and F. Frasincar. Survey on Aspect-Level Sentiment Analysis. *IEEE Transactions on Knowledge and Data Engineering*, volume 28, number 3, pages 813 - 830. IEEE, 2016."

2.1 Introduction

As discussed in Liu (2012), sentiment analysis has been studied mainly at three levels of classification. Sentiment is classified on either the document level, the sentence level, or the entity or aspect level. A focus on the first level assumes that the whole document expresses sentiment about only one topic. Obviously, this is not the case in many situations. A focus on the second level comes with a similar assumption in that one sentence should only contain sentiment about one topic. Within the same sentence, it is often the case that multiple entities are compared or that certain sentiment carrying opinions are contrasted. At both the document level and the sentence level, the computed sentiment values are not directly associated with the topics (i.e., entities or aspects of entities) discussed in the text. In a similar manner, sentiment can be computed over any arbitrary piece of text, even a complete corpus (e.g., a corpus of microblog entries, where each post is considered a document).

In contrast, aspect-level sentiment analysis aims to find sentiment-target pairs in a given text (i.e., this could range from sentences or smaller textual units, to complete corpora containing many documents). Within aspect-level sentiment analysis, the overall sentiment would generally refer to the entity, while aspect-level sentiment analysis would refer to the sentiment associated with aspects of the entity being discussed. This allows for a more detailed analysis that utilizes more of the information provided by the textual review. Therefore, this survey will focus on aspect-level analysis and its various sub-tasks. This also allows us to cover more recent developments, instead of repeating established insights that can be found in other surveys (Liu, 2012; Pang and Lee, 2008; Tang et al., 2009a; Tsytsarau and Palpanas, 2012).

A good survey and introduction into the field of sentiment analysis is the one by Pang and Lee (2008). Not only are various techniques and applications discussed, but also ethical, practical, and theoretical considerations are covered by their article. However, the coverage of the survey is restricted mostly to document-level machine learning approaches. There is a smaller survey by Tang et al. (2009a), and while it mainly focuses on document-level machine learning approaches as well, it specifically addresses the domain of consumer reviews. Tsytsarau and Palpanas (2012), while still focusing on document-level sentiment analysis, distinguishes between four different approaches for identifying the sentiment value of words: machine learning, dictionary-

based, statistical, and semantic. These four labels mainly describe how the sentiment value of a single word is determined. In Liu (2012), an updated overview of the entire field of sentiment analysis is given. The chapter dealing with aspect-level sentiment analysis is organized as a list of sub-problems that one encounters when implementing an actual solution: from definitions to aspect extraction, including various challenges that can be defined as part of aspect-level sentiment analysis, like dealing with implicit and explicit sentiment and entities, to how aspects and sentiment values can be identified and linked to one another. However, a systematic classification of approaches and reports of their accuracy are missing, a gap that the current survey is aiming to fill.

This survey is organized as follows. First, we discuss the evaluation methodology for aspect-level sentiment analysis. Then, we present various approaches for aspect detection and sentiment analysis in isolation as well as joint aspect detection and sentiment analysis approaches. After that, we discuss some interesting related problems that most approaches encounter and present some solutions dedicated to solve these issues. Then, the problem of aggregating sentiment scores is discussed, as well as the presentation of the aspect sentiment scores. We conclude this chapter with an informed outlook on the field of aspect-level sentiment analysis and highlight some of the most promising directions for future research.

2.2 Evaluation Methodology

Any maturing research area has to arrive at a common evaluation methodology that is generally accepted in the field. For aspect-level sentiment analysis, this is not yet the case, as evidenced by the wide variety of used evaluation measures and data sets.

In recent years, the International Workshop on Semantic Evaluation has embraced the task of aspect-level sentiment analysis (Pontiki et al., 2014, 2015), providing a controlled evaluation methodology and shared data sets for all participants. All competing systems get the same unannotated test data, which they will have to annotate with aspect tags and sentiment tags. This is sent to the organization which will perform a controlled evaluation of the provided data using the same procedures

for each competing system. The result is an overview of approaches that can be directly compared against each other.

Likewise, the GERBIL framework (Usbeck et al., 2015) also has the goal of directly comparing approaches with the same, controlled, evaluation methodology. To that end, it combines multiple data sets and many implementations of existing algorithms to compare against. Furthermore, the exact experimental setting is permanently stored and can be referred to so that readers can exactly see how the evaluation is performed. Unfortunately, at the time of writing, this system is only available for the task of entity annotation. However, the concept is applicable to many tasks, including aspect-level sentiment analysis.

Of course, many problems arise when research field standards are developed. For instance, the annotations needed differ for the various approaches since some methods classify sentiment in only positive or negative, while others use a five-star rating. In other cases, the specific focus of an evaluation may not be aspect-level sentiment analysis, like in Long et al. (2010) where the task of selecting comprehensive reviews is evaluated. The focus on different tasks also solicits the use of a wide variety of evaluation metrics.

2.2.1 Evaluation Measures

Currently, most of the surveyed work uses accuracy, precision, recall, and F_1 to measure quantitative performance, but some less common metrics are in use as well. To facilitate the proper interpretation of the reported performances, we will briefly discuss these less common metrics and present the general way of computing them.

For sentiment classification, multiple measures are in use: Ranking Loss, Mean Absolute Error, and Mean Squared Error. All of them assume that the sentiment value is at least an interval type variable. This assumption can be reasonable, even though in practice this is usually not the case.

Ranking Loss (Crammer and Singer, 2001), used in Moghaddam and Ester (2010), measures the average distance between the true rank and the predicted rank. For a sentiment classification problem with m sentiment classes (e.g., on a scale from one to five) and n test instances, Ranking Loss is defined in Equation 2.1 as the average deviation between the actual sentiment value y for instance i and the predicted

sentiment value $\hat{y}$ for that instance.

$$\text{Ranking Loss} = \sum_{i=1}^{n} \frac{|y_i - \hat{y}_i|}{m \times n} \tag{2.1}$$

An alternative to Ranking Loss is the macro-averaged Mean Absolute Error, which is particularly robust to imbalance in data sets. Used in Marcheggiani et al. (2014), it is computed as

$$\text{MAE}^M(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{m} \sum_{j=1}^{m} \frac{1}{|\mathbf{y}_j|} \sum_{y_i \in \mathbf{y}_j} |y_i - \hat{y}_i| \tag{2.2}$$

where $\mathbf{y}$ is the vector of true sentiment values, $\hat{\mathbf{y}}$ is the vector of predicted sentiment values, $\mathbf{y}_j = \{y_i : y_i \in \mathbf{y}, y_i = j\}$, and m is the number of unique sentiment classes in $\mathbf{y}$.

A similar measure is Least Absolute Errors (LAE), or L_1 error, which is used in Lu et al. (2011) to measure sentiment classification error. It is computed as

$$\text{LAE} = \sum_{i=1}^{n} |\hat{y}_i - y_i| \tag{2.3}$$

where $\hat{\mathbf{y}}$ is the vector of n sentiment predictions and $\mathbf{y}$ is the vector of true sentiment values.

Related to this is the Mean Squared Error (MSE), or the mean L_2 error, used in Wang et al. (2011) to evaluate the sentiment prediction error of the proposed method. This is a widely used metric, especially for regression, which is computed as

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 \tag{2.4}$$

where, again, $\hat{\mathbf{y}}$ is the vector of n sentiment predictions and $\mathbf{y}$ is the vector of true sentiment values.

For aspect detection, some algorithms return a ranked list of aspects. To compare rankings, multiple measures exist, one of which, the normalized Discounted Cumulative Gain, is used when reporting performance scores for the discussed work.

The normalized Discounted Cumulative Gain (nDCG) (Järvelin and Kekäläinen, 2002), also used in Wang et al. (2011), is particularly useful to evaluate relevance for lists of returned aspects. Furthermore, relevance does not have to be binary. The regular Discounted Cumulative Gain is computed as

$$\text{DCG@k} = \sum_{i=1}^{k} \frac{2^{\text{rel}(i)} - 1}{\log_2(i+1)} \tag{2.5}$$

where k represents the top k returned aspects that will be evaluated, and $rel(i)$ is the relevance score of aspect i. To normalize this score, and allow cross-query evaluation, the DCG score is divided by the ideal DCG. This is the DCG that would have been returned by a perfect algorithm. For most of the discussed approaches, nDCG cannot be computed, since it does not return a ranked list. However, if an algorithm produces rankings of aspects, for instance, based on how much these are discussed in a review, nDCG is an effective way of summarizing the quality of these rankings.

When dealing with generative probabilistic models, like topic models, where the full joint probability distribution can be generated, it is also possible to use the KullbackâĂŞLeibler divergence (Kullback and Leibler, 1951), or KL-divergence for short. This measures the difference between two probability distributions, where one distribution is the one generated by the model and the other is the distribution that represents the true data. How the KL-divergence is computed depends on the exact situation, for example whether the probability distributions are continuous or discrete. Characteristic for the KL-divergence, compared to other measures is that it is not a true metric, since it is not symmetrical: the KL-divergence of A compared to B is different than the KL-divergence of B compared to A.

2.3 Core Solutions

To provide insight into the large number of proposed methods for aspect-level sentiment analysis, a task-based top-level categorization is made, dividing all approaches into the following three categories: methods focusing on aspect detection, methods focusing on sentiment analysis, methods for joint aspect detection and sentiment

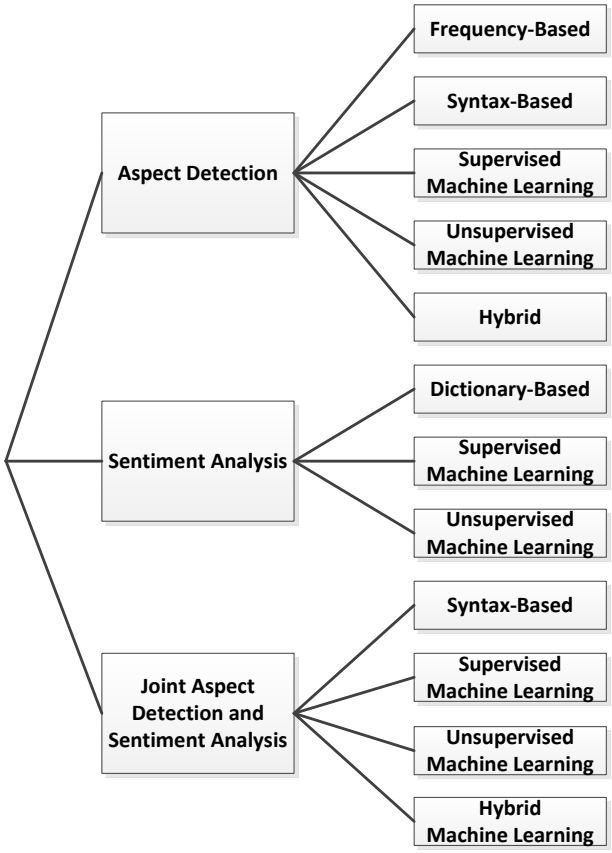

Figure 2.1: Taxonomy for aspect-level sentiment analysis approaches using the main characteristic of the proposed algorithm.

analysis. Within each task, a method-based categorization is made that is appropriate for that task (e.g., supervised machine learning, frequency-based, etc.). For each task, a table outlining all surveyed methods that cover that task is given. Each table lists the work describing the method, its domain (i.e., what kind of data it is evaluated on), a short description of the task that is evaluated, and the performance as reported by the authors. For the methods that perform sentiment analysis, the number of sentiment classes is also reported. Note that since evaluation scores are

taken from the original papers, experimental settings will be different for each work and as a consequence the methods cannot be compared using these evaluation scores. When multiple variants of an approach are evaluated and compared, we report only the results of the variant that yields the best performance. When the same method is evaluated over multiple data sets, the results are presented as the average or as a range.

Note that work describing both a method for aspect detection and a different method for sentiment analysis appears twice: the aspect detection method is discussed in Section 2.3.1, while the sentiment analysis method is discussed in Section 2.3.2. A tree overview of the classification system is shown in Figure 2.1, which is inspired by the organization of approaches that is used in the tutorial of Moghaddam and Ester (2013b).

2.3.1 Aspect Detection

All methods featuring an aspect detection method of interest are discussed in this section. A division is made between frequency-based, syntax-based (sometimes referred to as relation-based methods), supervised machine learning, unsupervised machine learning, and hybrid approaches. All the discussed approaches, together with their reported performance can be found in Table 2.1.

frequency-based	domain	evaluation task	performance
Hu and Liu (2004a)	product reviews	aspect detection	prec.: 72% recall: 80%
Long et al. (2010)	hotel reviews	comprehensive review selection	F_1: 70.6% - 93.3%
Hai et al. (2011)	cell phone reviews	implicit aspect detection	prec.: 76.29% recall: 72.71%
Liu et al. (2005)	product reviews	aspect detection (pros/cons)	prec.: 88.9% / 79.1% recall: 90.2% / 82.4%

Scaffidi et al. (2007)	product reviews	aspect detection	prec.: 85%-90% complexity: O(n)
Li et al. (2009)	product reviews	aspect detection	F_1: 74.07%
syntax-based			
Zhao et al. (2010b)	car, camera, and phone reviews (Zhao et al., 2008)	aspect detection	prec.: 73%, 66%, and 76% recall: 63%, 67%, and 68%
Qiu et al. (2009)	product reviews (Hu and Liu, 2004b)	aspect detection	prec.: 88% recall: 83%
Zhang et al. (2010)	cars & mattress reviews phone & LCD forum posts	aspect detection	prec.: 78% / 77% recall: 56% / 64% prec.: 68% / 66% recall: 44% / 55%
supervised machine learning			
Jakob and Gurevych (2010)	data from Kessler and Nicolov (2009); Toprak et al. (2010); Zhuang et al. (2006)	opinion target extraction	prec.: 61.4% - 74.9% recall: 41.4% - 66.1%
unsupervised machine learning			
Titov and McDonald (2008b)	product, hotel, and restaurant reviews	aspect detection	no quantitative evaluation

Lu et al. (2011)	hotel (Wang et al., 2010) & restaurant (Ganu et al., 2009) reviews	sentence labeling	accuracy: 79.4% F_1: 71.4% - 85.6%
Lakkaraju et al. (2011)	product reviews	aspect detection (2/5-class)	prec.: 83.33% / 82.52% recall: 81.12% / 80.72%
Zhan and Li (2011)	hotel (Baccianella et al., 2009) & restaurant (Ganu et al., 2009) reviews		not available (graphs only)
Wang et al. (2011)	hotel (Wang et al., 2010) & mp3 player reviews	aspect rating prediction	MSE: 1.234 nDCG: 0.901
Moghaddam and Ester (2013a)	product reviews (Jindal and Liu, 2008; Moghaddam and Ester, 2012; Wang et al., 2010)	item categorization (cold) item categorization (default)	accuracy: 79%-86% accuracy: 95%-97%
Hai et al. (2014)	product reviews	aspect detection	no quantitative analysis
hybrid			
Popescu and Etzioni (2005)	product reviews (Hu and Liu, 2004b)	aspect detection	prec.: 87.84% recall: 77.6%
Yu et al. (2011)	product reviews	aspect detection	F_1: 70.6% - 76.0%
Raju et al. (2009)	product reviews	aspect detection (incl. partial matches)	prec.: 92.4% recall: 62.7%

Blair-Goldensohn et al. (2008)	restaurant & hotel reviews	static aspect detection	prec.: 70.5% - 94.6% recall: 47.1% - 82.2%

Table 2.1: Approaches for aspect detection

Frequency-Based Methods

It has been observed that in reviews, a limited set of words is used much more often than the rest of the vocabulary. These frequent words (usually only single nouns and compound nouns are considered) are likely to be aspects. This straightforward method turns out to be quite powerful, a fact demonstrated by the significant number of approaches using this method for aspect detection. Clear shortcomings are the fact that not all frequent nouns are actually referring to aspects. Some nouns in consumer reviews, such as 'dollar' or 'bucks', are just frequently used. On the other hand, aspects that are not frequently mentioned, like very specific aspects that most people do not discuss, will be missed by frequency-based methods. To offset these problems, frequency-based methods can be supplemented with a set of rules to account for some of these issues. However, these manually crafted rules often come with parameters which have to be tuned.

The most well-known approach featuring a frequency-based method for aspect detection is Hu and Liu (2004a). The same authors describe the matching sentiment analysis method in Hu and Liu (2004b), which will be explained in Section 2.3.2. The aspect detection method described in Hu and Liu (2004a) only considers single nouns and compound nouns as possible aspects. First, the frequency of each combination of nouns is retrieved. For this, the nouns do not have to be next to each other, they should just appear in the same sentence. This helps to find aspects like 'screen size' when it is phrased as 'size of the screen'. The noun combinations that occur in at least 1% of the sentences are considered as aspects. Two rules are used to prune the result in order to lower the number of false positives. The first aims to remove combinations where the nouns never appear closely together, while the second aims to remove single-word aspects which appear only as part of a multi-word aspect. When

a sentence does not contain a frequent aspect but does contain one or more sentiment words, as indicated by the sentiment analysis method used in conjunction with the current approach, then the noun or compound noun nearest to the sentiment word is extracted as an infrequent aspect. This process, while sensitive to generating false positives, is able to increase the recall of the method. An improvement to this process can be found in Long et al. (2010), where grammatical dependencies are employed to find infrequent aspects instead of word distance. In this particular study, the goal is to find reviews that are most comprehensive with respect to a certain aspect, so that sentiment analysis can be performed on reviews that have a thorough discussion on that aspect.

Only explicit aspects are detected in Hu and Liu (2004a), but Hai et al. (2011) employs association rule mining to find implicit aspects as well. By restricting sentiment words to appear as rule antecedents only, and aspect words to appear as rule consequents, the generated association rules can now be used to find aspects based on already found sentiment words. Last, a major difference between the two methods is that, while Hu and Liu (2004a) generates the frequent item sets from a transaction file, Hai et al. (2011) generates its rules from the co-occurrence matrix of the bipartite of sentiment words and explicit aspects. One should note that these explicit features must therefore first be detected, before implicit features can be found using this method. The two methods can thus be thought of as complementary.

Similar to Hu and Liu (2004a) is Liu et al. (2005), where a supervised form of association rule mining is used to detect aspects. Instead of the full review text, Liu et al. (2005) targets pros and cons that are separately specified on some Web sites. Since pros and cons are known to be rich in aspect descriptions, this task is allegedly simpler than detecting aspects in the full text, and the obtained results are obviously better than those of Hu and Liu (2004a).

A major shortcoming of most frequency-based methods is the fact that nouns and noun phrases that naturally have a high frequency are mistakenly seen as aspects. Red Opal, a system introduced in Scaffidi et al. (2007), aims to address this issue by comparing the frequency of a prospective aspect with baseline statistics gathered from a corpus of 100 million words of spoken and written conversational English. To be considered as an aspect, a word or bigram has to appear more often in a review than is likely given its baseline frequency. This improves feature extraction and reduces

the number of non-features because these non-features are usually often occurring words that would be above a fixed threshold but are filtered out when using baseline statistics. As part of the evaluation, a small scale survey was conducted to assess the actual helpfulness of the extracted features, which suggested that users prefer bigram features over unigram features and specific features over more generic features. The same concept of baseline statistics is used in Li et al. (2009), where it used to filter the list of high-frequency noun phrases. Additionally, a part-of-speech pattern filter is also applied, such that every aspect needs to be followed by an adjective (note that this filter is designed to work with Chinese texts).

Syntax-Based Methods

Instead of focusing on frequencies to find aspects, syntax-based methods find aspects by means of the syntactical relations they are in. A very simple relation is the adjectival modifier relation between a sentiment word and an aspect, as in 'fantastic food', where 'fantastic' is an adjective modifying the aspect 'food'. A strong point of syntax-based methods is that low-frequency aspects can be found. However, to get good coverage, many syntactical relations need to be described.

To mitigate the low recall problem, a generalization step for syntactic patterns using a tree kernel function is proposed in Zhao et al. (2010b). Given a labeled data set, the syntactic patterns of all the annotated aspects are extracted. Then, for the unseen data, syntax trees of all sentences are obtained. Instead of directly trying to find an exact match between the aspect pattern and the syntax tree, both are split into several different substructures. Then the similarity between the pattern and a sentence can be measured as the number of matching substructures. The common convolution tree kernel is used to compute similarity scores for each pair of substructures, with a threshold determining whether a pair is a match or not.

In Qiu et al. (2009) (an extended version was published later in Qiu et al. (2011)), and its extension (Zhang et al., 2010), aspect detection and sentiment lexicon expansion are seen as interrelated problems for which a double propagation algorithm is proposed, featuring parallel sentiment word expansion and aspect detection. With each extra known sentiment word, extra aspects can be found, and with additional known aspect words, more sentiment words can be found, etc. The algorithm con-

tinues this process until no more extra sentiment words or targets can be found. To find sentiment words based on known aspect words, and the other way around, a set of rules based on grammatical relations from the employed dependency parser, is constructed. In this way, more sentiment-aspect combinations can be found and classified in a given text than with previous approaches. A big advantage of this method is that it only needs a small seed set to work properly compared to the large corpus most trained classifiers require.

Supervised Machine Learning Methods

There are not many supervised machine learning methods for aspect detection that are purely machine learning methods. Since the power of supervised approaches lies in the features that are used, feature construction often consists of other methods (e.g., frequency-based methods) in order to generate more salient features that generalize better than simple bag-of-words or part-of-speech features.

In Jakob and Gurevych (2010), aspect detection is cast as a labeling problem, which is solved by using a linear chain Conditional Random Field (CRF), common in natural language processing, to process a whole sequence (e.g., a sentence) of words. This automatically takes the context of a word into account when assigning it a label. Multiple features are used when determining the best label for a word, including the actual word, its part-of-speech tag, whether a direct dependency relation exists between this word and a sentiment expression, whether this word is in the noun phrase that is closest to a sentiment expression, and whether this word is in a sentence that actually has a sentiment expression. The ground-truth from a subset of the used data sets (Kessler and Nicolov, 2009; Toprak et al., 2010; Zhuang et al., 2006) is used to train the model. Four domains are covered in these review data sets: movies, web-services, cars, and cameras.

Unsupervised Machine Learning

In general, this class of models operates unsupervised, requiring only labeled data to test and validate the model. Nevertheless, a large amount of data is generally needed to successfully train these type of models. Most of the approaches in this section use LDA, which is a topic model proposed in Blei et al. (2003). Each document is

viewed as a mixture of topics that could have generated that document. It is similar to probabilistic Latent Semantic Analysis (Hofmann, 2000) but it utilizes a Dirichlet prior for the topic distribution instead of a uniform topic distribution. One of the main drawbacks of LDA is that the generated topics are unlabeled, preventing a direct correspondence between topics and specific aspects or entities. And while sometimes a quick glance at the words associated with a topic is enough to deduce which aspect it is referring to, not all topics are that clear cut. Because LDA utilizes a bag of words approach when modeling documents and topics, the contents of a topic (i.e., the words associated with it) are not required to be semantically related: it might be impossible to characterize a topic, making it much less suitable for interpretation.

Since LDA was designed to operate on the document level, employing it for the much finer-grained aspect-level sentiment analysis is not straightforward. Some critical issues that arise when implementing an LDA-based method for aspect-level sentiment analysis have been discussed in Titov and McDonald (2008b). The main argument is that since LDA uses a bag of words approach on the document level, it will discover topics on the document level as well. This is good when the goal is to find the document topic (i.e., this could be the entity, or some category), but not as useful when one is looking for aspects. The topics that LDA returns are simply too global in scope to catch the more locally defined aspects. One way to counter this would be to apply LDA on the sentence level, but the authors argue that this would be problematic since the bag of words would be too small, leading to improper behavior of the LDA model (cf. (Jin et al., 2011)). Although some solutions exist to this problem in the form of topic transitions (Blei and Moreno, 2001), the authors deem those computationally too expensive. Instead an extension to LDA is proposed called Multi-grain LDA (MG-LDA). Besides the global type of topic, MG-LDA models topics on two levels: global and local. The idea is to have a fixed set of global topics and a dynamic set of local topics, from which the document is sampled. To find the local topics, a document is modeled as a set of sliding windows where each window covers a certain number of adjacent sentences. These windows overlap, causing one particular word to be allowed to be sampled from multiple windows. This also solves the problem of too few co-occurrences: the bags of words are not too small in this case. The set of global topics act in a similar way to the background topic

of Mei et al. (2007) in Section 2.3.3, increasing the accuracy of the local topics that should represent the sought aspects.

A similar notion is demonstrated in Lu et al. (2011) where a distinction is made between global and local topics. Instead of the more complex construction of sliding windows, LDA is simply performed on the sentence level, with the exception that the document topics are modeled in conjunction with the sentence topics. In this way, the sentence topics can model the aspects with all non-relevant words modeled as a document topic.

While finding both global and local topics is useful to get coherent local topics that actually describe aspects, a different option is shown in Lakkaraju et al. (2011), where LDA is combined with a Hidden Markov Model (HMM) to distinguish between aspect-words and background words. This distinction is drawn by incorporating syntactic dependencies between aspect and sentiment. The same idea can be found in Li et al. (2010), a CRF model discussed in Section 2.3.3, although in Lakkaraju et al. (2011), it is employed in an unsupervised, generative manner.

Another way of adding syntactic dependencies is shown in Zhan and Li (2011), where the topic model employs two vocabularies to pick words from. One vocabulary holds the nouns, while the other holds all the words that are dependent on the nouns (e.g., adjectives, adjectival verbs, etc.). These pairs are extracted from the dependency tree as generated by a parser.

In Wang et al. (2011), the issue of coverage (cf. (Long et al., 2010) in Section 2.3.1) is addressed by estimating the emphasis placed on each aspect by the reviewer. This is done by modeling the overall rating of the product as the weighted sum of the aspect ratings. The inferred weights for the aspect can then be used as a measure of emphasis. However, where Long et al. (2010) returns the reviews which describe a certain aspect most comprehensively based on how much the reviewer is writing about it, Wang et al. (2011) determines the emphasis on a certain aspect in a review by its influence on the overall rating. This is an important difference, as the former will show the user reviews that talk much about a certain aspect, even when it is of no consequence to the overall rating, while the latter can output a list of reviews where a certain aspect greatly influences the rating, even when it is barely discussed.

Since LDA models are trained on a per-item basis, a significant number of data points is needed to infer reliable distributions. However, many products on the Web

have only a limited number of reviews. Continuing the work on aspect-level sentiment analysis and LDA models, a method to deal with this so-called cold start problem is proposed in Moghaddam and Ester (2013a). In addition to modeling aspects and sentiment values for products, it also incorporates product categories and the reviewers into the model. By grouping similar products into categories, aspects are associated to product categories instead of the individual products. Then instead of a distribution over all aspects, for each product, only a distribution over the aspects in the product category will have to be derived from the data. Furthermore, this distribution is influenced by the model of the reviewer, which is a distribution over the aspects this reviewer comments on mostly, and with what rating. Hence, a more accurate prediction can be made for products with little or no data.

In Hai et al. (2014), a supervised joint aspect and sentiment model is proposed to determine the helpfulness of reviews on aspect level. The proposed model is a supervised probabilistic graphical model, similar to supervised Latent Dirichlet Allocation. Just like similar LDA models in Section 2.3.3, this model separately and simultaneously models both aspect and sentiment words, to improve the quality of the found aspect topics. While the model is unsupervised with respect to aspect detection, it uses the helpfulness ratings provided for each review as supervision. Unfortunately, because the focus of this work is on the helpfulness prediction, the aspect detection part is not quantitatively evaluated.

Hybrid Methods

Every classification system has its exceptions, and the classification system used in this survey is no different. This section showcases work that falls in more than one of the above categories. When two types of methods are used, they are called hybrid methods and they come in two flavors: serial hybridization, where the output of one phase (e.g., frequency information) forms the input for the next phase (e.g., a classifier or clustering algorithm), and parallel hybridization, where two or more methods are used to find complementary sets of aspects.

Serial hybridization can be found in Popescu and Etzioni (2005), where Pointwise Mutual Information (Church and Hanks, 1990) is used to find possible aspects, which are then fed into a Naïve Bayes classifier to output a set of explicit aspects. Other

examples of serial hybridization include Raju et al. (2009), where the Dice similarity measure (Dice, 1945) is used to cluster noun phrases that are about the same aspect, and Yu et al. (2011) which targets pros and cons to find aspects using frequent nouns and noun phrases, feeding those into an SVM classifier to make the final decision whether it is an aspect or not.

Contrary to the above, a form of parallel hybridization can be found in Blair-Goldensohn et al. (2008), where a MaxEnt classifier is used to find the frequent aspects, for which there is ample data, and a rule-based method that uses frequency information and syntactic patterns to find the less frequent ones. In this way, available data is used to drive aspect detection, with a rule-based method that acts as back-up for cases where there is not enough data available.

2.3.2 Sentiment Analysis

The second part of aspect-level sentiment analysis is the actual sentiment analysis, which is the task of assigning a sentiment score to each aspect. The first proposed approaches generally use a dictionary to find the sentiment scores for the individual words followed by an aggregation and/or association step to assign the sentiment of the surrounding words to the aspect itself. The later approaches are all based on machine learning, either supervised or unsupervised. All the approaches that are discussed in this section can be found in Table 2.2, where their reported performance is also shown.

Dictionary-based

In Hu and Liu (2004b), a sentiment dictionary is obtained by propagating the known sentiment of a few seed words through the WordNet synonym/antonym graph. Only adjectives are considered as sentiment words here. Each adjective in a sentence will be assigned a sentiment class (i.e., positive or negative) from the generated sentiment dictionary. When a negation word appears within a word distance of five words starting from the sentiment word, its polarity is flipped. Then, a sentiment class is determined for each sentence using majority voting. Hence, the same sentiment class is assigned to each aspect within that sentence. However, when the number of positive and negative words is the same, a different procedure is used. In that

dictionary-based	domain	classes	evaluation task	performance
Hu and Liu (2004b)	product reviews	binary	sentiment classification	accuracy: 84.2%
Moghaddam and Ester (2010)	product reviews	5-star rating	sentiment classification	Ranking Loss: 0.49
Zhu et al. (2009)	restaurant reviews	ternary	aspect-sentiment extraction	prec.: 75.5%
supervised machine learning				
Blair-Goldensohn et al. (2008)	restaurant & hotel reviews	binary (pos/neg)	sentiment classification	prec.: 68.0% / 77.2% recall: 90.7% / 86.3%
Yu et al. (2011)	product reviews	binary	sentiment classification	F_1: 71.7%-85.1%
Choi and Cardie (2008)	MPQA corpus (Wiebe et al., 2005)	binary	sentiment classification	accuracy: 90.70%
Lu et al. (2011)	restaurant (Ganu et al., 2009) & hotel (Wang et al., 2010) reviews	5-star rating	sentiment classification	LAE: 0.560 - 0.790
Titov and McDonald (2008b)	product, hotel, and restaurant reviews	binary	sentiment classification	Ranking Loss: 0.669
unsupervised machine learning				
Popescu and Etzioni (2005)	product reviews (Hu and Liu, 2004b)	ternary	sentiment extraction	prec.: 76.68% recall: 77.44%
			sentiment classification	prec.: 84.8% recall: 89.28%

Table 2.2: Approaches for sentiment analysis

case, each sentiment bearing adjective is associated with the closest aspect within the sentence, in terms of word distance. Then majority voting is used among all sentiment words that are associated with the same aspect. In this case, having multiple polarities within the same sentence is a possibility.

In contrast to other dictionary methods, Moghaddam and Ester (2010) uses a set of adjectives provided by Epinions.com, where each adjective is mapped to a certain star rating. The unknown sentiment word, if it is not in this set, is then located in the WordNet synonymy graph. Employing a breadth-first search on the WordNet synonymy graph starting at the adjective with the unknown sentiment with a maximum depth of 5, the two closest adjectives which appear in the rated list of Epinions.com are found. Then, using a distance-weighted nearest-neighbor algorithm, it assigns the weighted average of the ratings of the two nearest neighbors as the estimated rating to the current adjective.

When performing sentiment analysis, some approaches, like the previously discussed Hu and Liu (2004b), compute one sentiment score for each sentence and then associate that sentiment with all the aspects that are mentioned in that sentence. However, this makes it impossible to properly deal with sentences that contain aspects with varying sentiment. A solution is proposed in Zhu et al. (2009), where all sentences are segmented with each segment being assigned to one of the aspects found in the sentence. Then, using a sentiment lexicon, the polarity of each segment is determined and an aspect-polarity pair is generated that reflects the overall polarity for this aspect within a particular review.

Supervised Machine Learning

While the methods in the previous section all use a dictionary as the main source for information, supervised machine learning methods usually learn many of their parameters from the data. However, since it is relatively easy to incorporate lexicon information as features into a supervised classifier, many of them employ one or more sentiment lexicons. In Blair-Goldensohn et al. (2008), the raw score from the sentiment lexicon and some derivative measures (e.g., a measure called purity that reflects the fraction of positive to negative sentiment, thus showing whether sentiment is conflicted or uniform) are used as features for a MaxEnt classifier. When

available, the overall star rating of the review is used as an additional signal to find the sentiment of each aspect (cf. (Scaffidi et al., 2007)).

In Yu et al. (2011), the short descriptions in the 'pros' and 'cons' section of a review are mined for sentiment terms. These sentiment terms are found using a dictionary (Wilson et al., 2005), with the location (i.e., either the 'pros' or 'cons' section) denoting their sentiment in that specific context. This information is then used to train a Support Vector Machine (SVM) that is able to classify sentiment terms as positive or negative. Given a free text review, for each aspect, the expression that contains its sentiment is found, which should be within a distance of five steps in the parse tree. Then, the SVM is used to determine the sentiment for that aspect.

While not exactly an aspect-level sentiment analysis method, Choi and Cardie (2008) is still interesting as it performs sentiment analysis on very short expressions, which can be associated to aspects (cf. (Zhu et al., 2009)). Since this method focuses solely on sentiment analysis, the expressions (i.e., short phrases expressing one sentiment on one aspect or entity) are given for this approach. The proposed method is a binary sentiment classifier based on an SVM. But while basic SVM approaches model the text using a simple bag-of-words model, the authors argue that such a model is too simple to represent an expression effectively. To solve this, the authors used the principle of compositional semantics, which states that the meaning of an expression is a function of the meaning of its parts and the syntactic rules by which these are combined. Applying this principle, a two-step process is proposed in which the polarities of the parts are determined first, and then these polarities are combined bottom-up to form the polarity of the expression as a whole. However, instead of using a manually-defined rule set to combine the various parts and their polarities, a learning algorithm is employed to cope with the irregularities and complexities of natural language.

The learning algorithm of the previous approach consists of a compositional inference model using rules incorporated into the SVM update method and a set of hidden variables to encode words being positive, negative, negator, or none of these types. The negator class includes both function-negators and content-negators. While function-negators are only a small set of words like "not" and "never", content-negators are words like "eliminated" and "solve", which also reverse the polarity of their surroundings. As machine learning approaches allow many features, they com-

bine multiple lexicons, adding sentiment information from both the General Inquirer lexicon as well as from the polarity lexicon from Wilson et al. (2005). With some simple heuristics and less sophisticated versions of the proposed method as a baseline, the above solution is evaluated on the MPQA corpus (Wiebe et al., 2005). Experiments show that using compositional inference is more beneficial than using a learning approach, but incorporating both clearly results in the highest accuracy.

Instead of a binary sentiment classifier, as is used in the above two methods (Choi and Cardie, 2008; Yu et al., 2011), a Support Vector Regression model is employed in Lu et al. (2011) to find the sentiment score for an aspect. This allows the sentiment score to be modeled as a real number in the zero to five interval, which is reminiscent of the widely used discrete 5-star rating system.

In Titov and McDonald (2008b), a perceptron-based online learning method called PRanking (Crammer and Singer, 2001), is used to perform the sentiment analysis, given the topic clusters that have been detected by an LDA-like model. The input consists of unigrams, bigrams, and frequent trigrams, plus binary features that describe the LDA clusters. For each sentence, a feature vector $\mathbf{x}$ is constructed consisting of binary features that signal the absence or presence of a certain word-topic-probability combination, with probabilities being grouped into buckets (e.g., 'steak', 'food', and '0.3-0.4'). The PRanking algorithm then takes the inner product of this vector ($\mathbf{x}$) and a vector of learned weights ($\mathbf{w}$) to arrive at a number, which is checked against a set of boundary values that divide the range a score can have into five separate ranges such that each range corresponds to a sentiment value (e.g. one to five). In the training phase, each misclassified instance will trigger an update where both the weights and the boundary values are changed. For example, if an instance is given a sentiment value which is too low, it will both increase weights and decrease threshold values.

Unsupervised Machine Learning

Another option is the use of an unsupervised machine learning method. In Popescu and Etzioni (2005), each explicit aspect is used to find a potential sentiment phrase by looking for an sentiment phrase in its vicinity, where vicinity is measured using the parsed syntactic dependencies. Each potential sentiment phrase is then examined,

and only the ones that show a positive or negative sentiment are retained. The semantic orientation, or polarity, is determined using an unsupervised technique from the computer vision area called relaxation labeling (Hummel and Zucker, 1983). The task is to assign a polarity label to each sentiment phrase, while adhering to a set of constraints. These constraints arise for example from conjunctions and disjunctions (Hatzivassiloglou and McKeown, 1997). The final output is a set of sentiment phrases with their most likely polarity label, be it positive or negative.

2.3.3 Joint Aspect Detection and Sentiment Analysis Methods

All approaches discussed until now either have a method or model dedicated to either aspect detection or sentiment analysis. Since the two problems are not independent, multiple approaches have been proposed that both extract the aspects and determine their sentiment. The main advantage is that combining these two tasks allows one to use sentiment information to find aspects and aspects to find sentiment information. Some methods explicitly model this synergy, while others use it in a more implicit way. We distinguish between syntax-based, supervised machine learning, unsupervised machine learning, and hybrid methods. In Table 2.3, all approaches discussed in this section are shown, together with their reported performance.

Syntax-Based Methods

Given the observation that it is much easier to find sentiment words than aspect words, syntax-based methods are generally designed to first detect sentiment words, and then by using the grammatical relation between a sentiment word and the aspect it is about, to find the actual aspect. A major advantage of this method is that low-frequency aspects can also be found, as the key factor here is the grammatical relation between the aspect and its sentiment word(s). This is also its greatest shortcoming, since patterns have to be defined that describe the set of possible relations between an aspect and a sentiment word. Unfortunately, a very specific set of relations will miss a lot of aspects leading to high precision, but low recall, while a more general set of relations will yield more aspects but also many more words that are not aspects, leading to low precision, but high recall. Additionally, the extraction of grammatical

relations (usually) requires parsing the text, which is both slow and usually not error-free.

An early syntax-based method is presented in Nasukawa and Yi (2003), where a shallow parser and an extensive set of rules is used to detect aspects and sentiment. The lexicon describes not just the sentiment for a given word, but also gives transfer patterns stating which words are affected by the sentiment. In this way, sentiment originates at a certain word, and is transferred by other words (e.g., verbs) to the aspect word. A good example would be the sentence "The automatic zoom prevents blurry pictures", where negative sentiment originates at 'blurry' and is reversed by the verb 'prevents', transferring the now reversed sentiment to the aspect 'automatic zoom'. Because the described relations are very specific, the result is a typical high-precision low-recall approach that, therefore, works best on large volumes of data.

While most of the previously described approaches focus on product reviews, in Zhuang et al. (2006), an aspect-level sentiment analysis approach is proposed for the movie review domain. This approach employs a lexicon for both the aspect detection and the sentiment analysis part. While the latter is common practice, the former is more of an exception. The intuition behind this is that a lexicon can capture all the domain specific cues for aspects. For example, this aspect lexicon includes a list of names of people involved in the movie that is under review. Dependency patterns that link the aspect and the sentiment word are used to find aspect-sentiment pairs. However, the described relations only cover the most frequent relations, so less frequent ones are missed.

syntax-based	domain	classes	evaluation task	performance
Nasukawa and Yi (2003)	general & camera reviews	binary	combined (general/camera)	prec.: 94.3% / 94.5% recall: 28.6% / 24%
Zhuang et al. (2006)	movie reviews	binary	aspect-sentiment pair mining	F_1: 52.9%
supervised machine learning				

Kobayashi et al. (2006)	product reviews	binary	sentiment extraction aspect-sentiment pair mining sentiment classification	prec.: 67.7% recall: 50.7% prec.: 76.6% recall: 75.1% prec.: 82.2% recall: 66.2%
Li et al. (2010)	product & movie reviews	binary	combined (movies/products)	prec.: 82.6% / 86.6% recall: 76.2% / 69.3%
Marcheggiani et al. (2014)	hotel reviews (annotated subset of (Wang et al., 2010))	ternary	aspect detection sentiment classification	F_1: 48.5% MAE^M: 0.5
Jin et al. (2009)	camera reviews	binary	aspect extraction sentiment sentence extraction sentiment classification	F_1: 78.8% - 82.7% F_1: 84.81% - 88.52% F_1: 70.59% - 77.15%
Zirn et al. (2011)	product reviews	binary	combined (pos/neg)	prec.: 66.38%/72.02% recall: 72.94%/65.34%
unsupervised machine learning				
Mei et al. (2007)	weblogs	binary	sentiment model (pos / neg)	KL-divergence: 21 / 19

Titov and McDonald (2008a)	hotel reviews	5-star rating	combined	avg. prec.: 74.5% - 87.6%
Moghaddam and Ester (2011)	product reviews	5-star rating	aspect detection	Rand Index: 0.83
			sentiment classification	Rand Index: 0.73
Jo and Oh (2011)	restaurant & product reviews	binary	sentiment classification	accuracy: 84% - 86%
Wang et al. (2011)	mp3 player & hotel (Wang et al., 2010) reviews	5-star rating	aspect rating prediction	MSE: 1.234 nDCG: 0.901
Sauper and Barzilay (2013)	restaurant reviews	binary	aspect cluster prediction sentiment classification	prec.: 74.3% recall: 86.3% accuracy: 82.5%
	medical summaries		aspect cluster prediction	prec.: 89.1% recall: 93.4%
hybrid machine learning				
Zhao et al. (2010a)	restaurant (Brody and Elhadad, 2010) & hotel (Baccianella et al., 2009) reviews	ternary + 'conflicted'	aspect identification sentiment identification	avg. F_1: 70.5% prec. @ 5: 82.5% prec. @ 10: 70.0%
Mukherjee and Liu (2012)	product reviews	binary	sentiment classification	prec.: 78% recall: 73%

Table 2.3: Approaches for joint aspect detection and sentiment analysis

Supervised Machine Learning

An evident problem is that in general, machine learning methods excel in classifying instances in a given number of classes. Since the number of possible aspects and the different words that can represent an aspect is practically unbounded, a default classification algorithm cannot be applied in a straightforward manner. In Kobayashi et al. (2006), both aspect detection and sentiment classification are cast as a binary classification problem. First, using a lexicon, all prospective aspect and sentiment words are tagged. Then, the problem of which aspect belongs to which sentiment word is solved using a binary classification tournament model. Each round of the tournament, two aspects are compared and the one that best matches the sentiment word proceeds to the next round. In this way, no direct relation between the aspect and sentiment is needed. The drawback is that no additional aspects can be found by exploiting this relation, but an advantage is that this method can effectively deal with ellipsis, a linguistic phenomenon where the aspect is not linked to the sentiment because it is either implicit or referred to using a co-reference. According to the authors, as much as 30% of the sentences feature ellipsis.

To address the issue of long-range dependencies, Li et al. (2010) encodes both syntactic dependencies between words and conjunctions between words into a CRF model. By introducing more dependencies between the hidden nodes in the CRF model, words that are not directly adjacent in the linear chain CRF, can now influence each other. Sentiment values and their targets are linked simply by minimizing the word distance and are extracted simultaneously. The model is then used to generate a list of sentiment-entity pairs as a summary of the set of texts, which are product and movie reviews in this case, grouped as positive or negative.

A strong limitation of the previous work is that each sentence is assumed to have only one aspect. In Marcheggiani et al. (2014), a CRF model is proposed that is able to deal with multiple aspects per sentence. Furthermore, when multiple aspects are mentioned in the same sentence, it is likely that they influence each other via certain discourse elements, which has an effect on the sentiment score for each aspect. Therefore, the model explicitly incorporates the relations between aspect-specific sentiments within one sentence. Last, the overall score of the review, which is often supplied by the users themselves, is taken into account as well. To do that, a

hierarchical model is proposed that simultaneously predicts the overall rating and the aspect ratings. This new model has an additional variable for the overall sentiment score, and pairwise factors that model the influence between the overall sentiment score and each aspect's sentiment score. A random subset of 369 hotel reviews from the TripAdvisor data set (Wang et al., 2010) is manually annotated for aspects to train and test the model.

An example of a method based on a lexicalized HMM is Jin et al. (2009). With HMM's, the context of a word can easily be taken into consideration by using n-grams. However, simply using higher n-grams (e.g., bigrams, trigrams, etc.) poses some problems. Because a lot of these n-grams are not likely to appear in the training corpus, their values have to be guessed instead of counted. Furthermore, computational complexity increases exponentially when using higher n-grams. This is the reason that in Jin et al. (2009) only unigrams are used. While this prevents the above mentioned problems, it also deprives the model of any context-sensitivity. To account for it, the part-of-speech of a word is also modeled, and in a way that makes it dependent on both the previous and the next part-of-speech tag, thereby introducing some form of context-awareness. A bootstrapping approach is proposed to make the model self-learn a lot of training examples, mitigating the dependence on labeled training data to some extent. The additional examples learned in this way proved to be beneficial when evaluating this approach, improving F_1-score for both aspect detection and sentiment classification.

A Markov logic chain is employed as the main learning method in Zirn et al. (2011). Within the Markov chain, multiple lexicons are incorporated, as well as discourse relations. The latter are acquired using the HILDA (duVerle and Prendinger, 2009) discourse parser which returns a coarse-grained set of discourse segments as defined in Soricut and Marcu (2003), which are based on the Rhetorical Structure Theory (Mann and Thompson, 1988). Since sentiment classification is done on the level of discourse segments, it is assumed each segment only expresses one sentiment, which is almost always the case. Entities, however, are not extracted in this method. The proposed classification in Zirn et al. (2011) is binary, which, according to the authors, results in problems with some segments that have no clear polarity. Their findings concerning the use of discourse elements were that using general structures that can be found in the text systematically improves the results. The fact that a

certain discourse relation describes a contrasting relation was encoded specifically, as it was expected to correlate with the reversing of polarity of the various segments it connects to. However, this correlation turned out to be not as strong as was expected beforehand. This means, according to the authors, that the classical discourse relations might not be the best choice to represent the general structure of the text when performing sentiment analysis. Nevertheless, the same authors believe that focusing on cue words to find discourse connectives in order to predict polarity reversals might still be worth investigating.

Unsupervised Machine Learning

The class of unsupervised machine learning approaches may be especially interesting, since these models are able to perform both aspect detection and sentiment analysis without the use of labeled training data. The first topic mixture model (Mei et al., 2007) is based on probabilistic Latent Semantic Indexing (PLSI) (Hofmann, 2000), a model similar to LDA, that is however more prone to overfitting and is not as statistically sound as LDA. In Mei et al. (2007), not only topics that correspond to aspects are modeled, but also a topic for all background words, causing the retrieved topics to better correspond to the actual aspects. Furthermore, the topics that correspond to aspects are again mixtures of sentiment topics. In this way, the end result is that both aspects and their sentiment are determined simultaneously with the same model. Leveraging a sentiment lexicon to better estimate the sentiment priors increases the accuracy of the sentiment classification.

In Titov and McDonald (2008a), Titov and McDonald extend the model they propose in Titov and McDonald (2008b) by including sentiment analysis for the found aspects. An additional observed variable is now added to the model, namely the aspect ratings provided by the author of the review. With the assumption that the text is predictive of the rating provided by the author, this information can be leveraged to improve the predictions of the model. A strong point is that the model does not rely on this information being present, but when present, it is used to improve the model's predictions. Besides utilizing the available aspect ratings, the model can extract other aspects from the text as well, and assign a sentiment score to them. While at least a certain amount of provided aspect ratings is needed for this

model to truly benefit from them, perhaps the biggest advantage is that the found aspects can be linked to actual aspects in the text. As mentioned earlier, generative models produce unlabeled clusters that are not associated with any particular aspect. This problem is solved by incorporating these aspect ratings into the LDA model, providing a link between the words in the document and the concrete aspects as annotated by the reviewer. Last, when put to the test against a MaxEnt classifier, a supervised method, the proposed method performed only slightly worse.

The main improvement of Moghaddam and Ester (2011), compared to previous topic models is that the sentiment class of an aspect is explicitly linked to the aspect itself. This makes the sentiment analysis more context-aware: in this way, a word that is positive for one aspect can be negative for another. The latter is generally true for models that couple the sentiment nodes to the aspect nodes in the graphical model, and this same idea is demonstrated in both Wang et al. (2011) and Jo and Oh (2011).

In Jo and Oh (2011), aspects are detected as topics by constraining the model to only one aspect-sentiment combination per sentence. By assuming that each sentence is about only one aspect and conveys only one sentiment, the model is able to find meaningful topics. This is a relatively simple solution compared to for example the sliding windows technique (Titov and McDonald, 2008b) or injecting syntactic knowledge into the topic model (Lakkaraju et al., 2011). Evaluation of the constructed topics revealed another interesting fact: in one particular case there were three topics that conveyed negative sentiment for the same aspect. While this may not seem ideal at first (i.e., one unique topic per aspect-sentiment combination is more logical), close inspection revealed that the three topics revealed three distinct reasons why the reviewers were negative about that aspect (i.e., the screen was too small, the screen was too reflective, and the screen was easily covered with fingerprints or dirt). This level of detail goes further than regular aspect-level sentiment analysis, providing not only the sentiment of the reviewers, but also the arguments and reasons why that sentiment is associated to that aspect.

In Sauper and Barzilay (2013), a probabilistic model is presented that performs joint aspect detection and sentiment analysis for the restaurant reviews domain and aspect detection alone for the medical domain. For the restaurant domain, it models the aspects in such a way that they are dependent on the entity (i.e., the restaurant),

instead of having a global word distribution for aspects like previous models. This allows the model to have different aspects for different kind of restaurants. For example, a steak house has different aspects than an Italian ice cream place and while the sentiment word distribution is global (i.e., the same sentiment words are used for all types of restaurants), a separate distribution that is different for each restaurant is used to model the link between aspects and sentiment words. Furthermore, an HMM-based transition function is employed to model the fact that aspects and sentiment words often appear in a certain order. Last, a background word distribution is determined on a global level to get rid of words that are irrelevant. A variant of the model is used to process dictated patient summaries. Since the set of relevant aspects is expected to be shared across all summaries, the aspects are modeled as global word distribution. The previous method operates in an unsupervised fashion, requiring only a set of sentiment seed words to bias the sentiment topics into a specific polarity. Furthermore, the proposed model admits an efficient inference procedure.

Hybrid Machine Learning

While LDA is designed to work with plain text, the above methods have shown that the right preprocessing can significantly improve the results of the generative model. This can be extended a bit further by already optimizing some of the input for the topic model by using a supervised discriminative method. Both methods presented in this section feature a MaxEnt classifier that optimizes some of the input for the LDA model.

The first method (Zhao et al., 2010a) uses a MaxEnt component to enrich the LDA model with part-of-speech information. In this way, the generative model can better distinguish between sentiment words, aspect words, and background words. The MaxEnt classifier is trained using a relatively small set of labeled training data, and the learned weights are now input for a hidden node in the topic model. This is done before training the LDA model, so while training the LDA model, the weights of the MaxEnt classifier remain fixed.

The second method (Mukherjee and Liu, 2012) that combines an LDA model with a MaxEnt classifier, uses the MaxEnt classifier to optimize the word priors that influence the generative process of drawing words. Again, part-of-speech information

is a major feature for the MaxEnt component. The fact that external information can be integrated into the generative process of an LDA model makes it a very powerful and popular method for aspect-level sentiment analysis.

2.4 Related Issues

While finding aspects and determining their sentiment value is the core of aspect-level sentiment analysis, there are more issues that play a role in developing an effective tool for aspect-level sentiment analysis. This section discusses some of these related issues. First, a set of sub-problems will be discussed, including how to deal with comparative opinions, conditional sentences, and negations and other modifiers. Then, a short discussion on aggregation of sentiment scores is given, followed by a concise exposition on presentation of aspect-level sentiment analysis results.

2.4.1 Sub-problems

Processing natural language in general, and performing aspect-level sentiment analysis, specifically, is a very complex endeavor. Therefore, it has been proposed, for example in Narayanan et al. (2009), that instead of focusing on a one-size-fits-all solution, researchers should focus on the many sub-problems. By solving enough of the sub-problems, the problem as a whole can eventually be solved as well. This line of thought has given rise to work specifically targeting a certain sub-problem in sentiment analysis, which is discussed below. The presented approaches are not solutions for aspect-level sentiment analysis and are therefore not in the tables together with the previously discussed approaches. However, when aspect-level sentiment analysis methods take the issues presented below into account (and some do to some extent), performance will increase.

Comparative Opinions

In comparative sentences, one entity or aspect is usually compared with another entity or aspect by preferring one over the other. Detecting comparative sentences and finding the entities and aspects that are compared, as well as the comparative words themselves is very useful (Jindal and Liu, 2006). However, for sentiment

analysis, one really needs to know which entity or aspect is preferred, a problem that is discussed in Ganapathibhotla and Liu (2008).

First, various categories of comparative sentences are defined and, for each category, it is shown how to process them. When possible, a comparator is reduced to its base form, and its sentiment is found using the sentiment word list generated from WordNet (Hu and Liu, 2004b). The comparators whose polarity cannot be determined in this way are labeled as context-dependent and are processed differently. For that, information in the pros and cons section is leveraged to compute an asymmetric version of the Pointwise Mutual Information association score between the comparative words and the words in the pros and cons. A set of rules then essentially combines the information about the entities, comparative words, and aspects being compared into one coherent outcome: either a positive or negative sentiment about the preferred entity.

A remaining problem in Ganapathibhotla and Liu (2008) is that when something is more positive than something else, the first is assumed to have a positive sentiment. This is not always the case. Also problematic is the negation of comparators, as stated by the authors themselves. Their example of "not longer" not necessarily being the same as "shorter" is illustrative. While the proposed method currently perceives the second entity as the preferred one when encountering negations, the authors admit that it could also be the case that the user did not specify any preference.

Conditional Sentences

As discussed in the previous section, conditional sentences pose a problem in that it is hard to determine whether they actually express some sentiment on something or not. In Narayanan et al. (2009), an approach dedicated to conditional sentences was proposed, which can be seen as an extension of the existing line of research based on Hu and Liu (2004b). First, the various types of conditionals were grouped into four categories, each with part-of-speech patterns for both the condition and the consequent in that category. Around 95% of the targeted sentences is covered by these patterns. The sentences found are then classified as either positive, negative, or neutral with respect to some topic in that sentence. For this study, the topic is assumed to be known beforehand. In contrast to previously described research, the

authors chose to use an SVM to classify these sentences as having either a positive or negative polarity.

Features used for the SVM are the basic ones like sentiment words and part-of-speech information, but also some common phrases and a list of words that imply the lack of sentiment. Also covered are negations by adding a list of negation keywords. This is, however, still based on a simple word distance metric. Other notable features are the fact whether the topic is in the conditional or the consequent and the length of both the condition and consequent phrases. Last, the sentiment words were weighted according to the inverse of their distance to the topic.

Multiple ways of training were proposed in Narayanan et al. (2009), but using the whole sentence instead of only the conditional or consequent part turned out to be the most successful. Interestingly, while the whole-sentence classifier gave the best results, the consequent-only classifier gave much better results than the conditional-only classifier, even approaching the results of the whole-sentence classifier, suggesting that most useful information to classify conditionals is in the consequent and not in the conditional part. The classifier was trained on a set of product reviews which were manually annotated and tested on both a binary and a ternary classification problem.

For the binary classification, the consequent-only classifier and the whole-sentence classifier yielded a similar performance while for the ternary classification, the whole-sentence approach performed clearly better. According to the authors, this signifies that to classify something as neutral, information from both the conditional and the consequent are needed. The best result the authors reported is an accuracy of 75.6% for the binary classification and 66.0% for the ternary classification. Unfortunately, no baseline was defined to compare these results against.

Negations and Other Modifiers

From amongst the set of modifiers that change the polarity or strength of some sentiment, negations are implemented most. This comes to no surprise given the effect negations can have on the sentiment of an aspect, sentence, or document. A theoretical discussion by Polanyi and Zaenen (2006) proposes some foundational considerations when dealing with these contextual valence shifters as they are sometimes

called. The authors distinguish between sentence-based contextual valence shifters and discourse-based ones.

Negations and intensifiers, which belong to the sentence-based group, are mostly single words influencing the polarity of words that are within their scope. Negations flip the polarity of a sentiment, while intensifiers either increase or decrease the sentiment value. Other sentence-based contextual valence shifters are: modals, where a context of possibility or necessity is created as opposed to real events (e.g., "if she is such a brilliant person, she must be socially incapable."); presuppositional items which represent certain expectation that are met or not (e.g., "this is barely sufficient"); and irony in which overly positive or negative phrases are turned on themselves to create a sentence with the opposite valence or polarity (e.g., "the solid and trustworthy bank turned to robbing their own customers").

The category of discourse-based contextual valence shifters is more complex in nature. While one group, the discourse connectors, are linked to some particular words, all other categories are much harder to identify. We will therefore only briefly discuss these discourse connectors, and refer the interested reader to Polanyi and Zaenen (2006) for more categories. Discourse connectors are words that connect two or more phrases in such a way that the combination is different in terms of sentiment than simply the sum of its parts. An example to illustrate this is "while he is grumpy each day, he is not a bad person", where we can see that the connector 'while' mitigates the effects of 'grumpy', resulting in an overall positive sentence.

An implementation of the above framework was described in Moilanen and Pulman (2007), where many of the ideas of Polyani and Zaenen are encoded in rules. The resulting pipeline, which also included a part-of-speech tagger and a parser, was evaluated to analyze where errors do occur. The results are rather interesting, as about two-thirds of the errors occur before the valence shifting module. Large contributions to errors are made by the parser and the tagger (around 14% each) and the lack of a word sense disambiguation module (25%). Errors made by the valence shifter module can roughly be attributed to three reasons: either the polarity reading was ambiguous (10%), more world knowledge was required (19%), or the polarity was modulated by phenomena more closely related to pragmatics than semantics (5%).

While Polyani and Zaenen did not really discuss the scope of a negation, this is actually a very important topic. Most approaches to sentiment analysis have at

least some handling of negations, but they usually employ only a simple word distance metric to determine which words are affected by a negation keyword (cf. (Hogenboom et al., 2011) for a comparison of different word distances). In Jia et al. (2009), the concept of the scope of a negation term is further developed. For each negation term, its scope is found by using a combination of parse tree information and a set of rules. The general idea is to use the parse tree to find the least common ancestor of the negation word and the word immediately following it in the sentence. Then all leaves descending from that ancestor that are to the right of the negation term are in the scope of the negation. This scope is then further delimited and updated by the set of rules to cover some exceptions to this general rule.

When looking at informal texts, such as microblog posts, additional modifiers need to be taken into account(Thelwall et al., 2012). Lexical variants that intensify the expressed sentiment include the use of repeated punctuation with exclamation marks and using repeated characters inside a word (e.g., 'haaaaaappy'). Other sources of sentiment that are employed in informal texts are emoticons, for which a custom list of emoticons with their sentiment score is usually needed.

2.4.2 Aggregation

Several of the discussed approaches aggregate sentiment over aspects, usually to show an aspect-based sentiment summary. Most methods aggregate sentiment by simply averaging or taking a majority vote. In contrast, methods that employ topic models, for example Titov and McDonald (2008a), aggregate naturally over the whole corpus, thereby computing sentiment for each topic or aspect based on all the reviews. A different approach is shown in Wang et al. (2010), where the topic model does not return the aggregated aspect ratings, but instead presents the aspect ratings for each individual review, as well as the relative weight placed on that aspect by the reviewer. The authors discuss that this enables advanced methods of aggregation, where aspect ratings can be weighted according to the emphasis placed on it by each reviewer.

In Basiri et al. (2014), multiple methods for aggregating sentiment scores are investigated. Even though this work focuses on combining sentence-level sentiment scores into a document-level sentiment score, the ideas can be naturally translated into the domain of aspect-level sentiment analysis. Next to a series of heuristic

methods, a formally defined method for aggregation based on the Dempster-Shafer Theory of Evidence (Shafer, 1976) is proposed. This is a theory of uncertainty that can be used to quantify the amount of evidence a certain source contributes to some proposition. In this case, the sources of evidence are the sentence sentiment scores, and the proposition to which these sources of evidence contribute is the final document-level sentiment score.

The following methods of aggregation are tested: randomly picking a sentence sentiment as the document sentiment, simply averaging all sentence sentiment scores, taking the absolute maximum score (e.g., when the strongest positive sentence is $+5$ and the strongest negative sentence is -4, the overall sentiment will be $+5$), summing the two maximum scores (e.g., in the previous example, summing $+5$ and -4 would result in a $+1$ document-level sentiment), scaled rate which is the fraction of positive sentiment words out of all sentiment words, and the discussed Dempster-Shafer method. As shown in Basiri et al. (2014), the proposed method clearly outperforms all heuristics. It is argued that this is caused by the fact that the Dempster-Shafer method takes all pieces of evidence into account, and the fact that it considers maximal agreements among the pieces of evidence. Of interest is the fact that this method is tested on two data sets that have also been used for already discussed methods that perform aspect-level sentiment analysis (cf. Tables 2.1, 2.2, and 2.3). Hence, methods for aspect-level sentiment analysis should be able to benefit from this research.

2.4.3 Presentation

As a final step in the process of aspect-level sentiment analysis, the results should be presented to the user. This can be done in several ways, the first of which is simply showing the numbers. In this case, for a certain product, a list of detected aspects is shown, together with the aggregated sentiment scores for each aspect. One can also imagine a table with the scores for multiple products in order to easily compare them.

In Liu et al. (2005), a visual format is advocated that shows bars that denote the sentiment scores. Clicking the bar would show more details, including relevant snippets of reviews. In this way, a user can quickly inspect the traits of several products and compare them, without getting overwhelmed by a table full of numbers.

When the timestamp of each review is available, a timeline (Ku et al., 2006) could also be generated to show the change in sentiment over time. This is important for services, which can change over time, or product characteristics which may only show after prolonged use.

Another possibility is to generate a summary of all the analyzed reviews. When done right, this will produce a readable review that incorporates all the available information spread over all reviews. In Carenini et al. (2006), an ontology is used to organize all the aspects into aspect categories and all sentences that express sentiment on an aspect are linked to the aspects in the ontology as well. Two methods for summary generation are tested: the first is to select representative sentences from the ontology, the second is to generate sentences with a language generator based on the aspects and their known sentiment scores. While the sentence selection method yields more variation in the language being used in the summary as well as more details, the sentence generation provides a better sentiment overview of the product. A variation of this method is contrastive summarization (Kim and Zhai, 2009), where the summary consists of pairs of sentences that express opposing sentiment on the same aspect.

2.5 Conclusions

From the overview of the state-of-the-art in aspect-level sentiment analysis presented in this survey, it is clear that the field is transcending its early stages. While in some cases, a holistic approach is presented that is able to jointly perform aspect detection and sentiment analysis, in others dedicated algorithms for each of those two tasks are provided. Most approaches that are described in this survey are using machine learning to model language, which is not surprising given the fact that language is a non-random, very complex phenomenon for which a lot of data is available. The latter is especially true for unsupervised models, which are very well represented in this survey.

We would like to stress that transparency and standardization is needed in terms of evaluation methodology and data sets in order to draw firm conclusions about the current state-of-the-art. Benchmark initiatives like SemEval (Pontiki et al., 2014,

2015) or GERBIL (Usbeck et al., 2015) that provide a controlled testing environment are a shining example of how this can be achieved.

When considering the future of aspect-level sentiment analysis, we foresee a move from traditional word-based approaches, towards semantically rich concept-centric aspect-level sentiment analysis (Cambria et al., 2013). For example, in "This phone doesn't fit in my pocket", it is feasible to determine that the discussed aspect is the size of the phone. However, the negative sentiment conveyed by this sentence, related to the fact that phones are supposed to fit in one's pocket, seems extremely hard to find for word-based methods. Related to this problem, pointing to the need for reasoning functionality, is the still open research question of irony. In Wallace (2013), a conceptual model is presented that explicitly models expectations, which is necessary to effectively detect irony. This is also a step away from the traditional word-based approach towards a semantic model for natural language processing. While concept-centric, semantic approaches have only recently begun to emerge (e.g., ontologies are being used to improve aspect detection (Peñalver-Martinez et al., 2014)), they should be up to this challenge, since semantic approaches naturally integrate common sense knowledge, general world knowledge, and domain knowledge.

Combining concept-centric approaches with the power of machine learning will give rise to algorithms that are able to reason with language and concepts at a whole new level. This will allow future applications to deal with complex language structures and to leverage the available human-created knowledge bases. Additionally, this will enable many application domains to benefit from the knowledge obtained from aspect-level sentiment analysis.

Chapter 3

Heracles: a Framework for Developing and Evaluating Text Mining Algorithms in Business

---

MANY *of today's businesses are driven by data, and while traditionally only quantitative data is considered, the role of textual data in our digital world is rapidly increasing. Text mining allows to extract and aggregate numerical data from textual documents, which in turn can be used to improve key decision processes. In this paper, we propose Heracles, a framework for developing and evaluating text mining algorithms, with a broad range of applications in industry. In contrast to other frameworks, Heracles supports both the development and evaluation stages of text mining algorithms. Several use cases show the general applicability and ease-of-use of the proposed framework, while a qualitative user study shows the strong points, as well as the aspects that need further work.*

3.1 Introduction

With text mining becoming ever more popular, numerous companies, labs, and research groups are committed to developing the next generation of text mining algorithms (Aggarwal and Zhai, 2012). As textual data abound on the Web, there are many applications in which text mining plays a key role. A prolific example of text mining is sentiment analysis on financial news, a topic that is popular in both academia (Chan and Chong, 2017; Schumaker et al., 2012), and business (Bloomberg, 2017), due to its challenging nature and potential profitability. Another use of text mining can be found in modeling and managing customer satisfaction (Farhadloo et al., 2016), a topic that is all the more relevant now that many major companies have social media divisions that monitor and engage with customers on various platforms. Some social media, such as Twitter, offer a unique possibility to harness the power of millions of personal expressions for problems ranging from predicting soccer matches (Schumaker et al., 2016) to natural disaster management (Spence et al., 2015).

While in academia, there are mature evaluation practices that ensure scientifically valid performance reports, these practices are not as ubiquitous in business, where focusing on rapid prototypes, fast deployment, and customer satisfaction is often more important than measuring the exact performance of the developed product. Even so, we argue that for academics and business researchers alike, using a solid evaluation methodology is of great benefit. This is the main reason we developed a framework that supports both the development and the evaluation of text mining algorithms. Since text mining is at the intersection of machine learning and natural language processing, support is needed in the form of machine learning algorithms that can be used to solve text mining problems, as well as natural language components to draw upon when processing the raw, unstructured text. Hence, the advantage of Heracles is that it supports all three parts of the process: natural language processing to deal with the raw textual data, machine learning algorithms to perform the required text mining tasks on the processed data, and evaluation procedures to assess the performance of a developed algorithm. While many frameworks exist that address one or two of the previously mentioned parts, we argue that all three are needed for a system to be truly beneficial to text mining developers.

The framework proposed in this paper is born out of the need to develop and test algorithms for sentiment analysis, one of the more popular branches of text mining (Feldman, 2013). For our participation in the sentiment analysis task at the well-known benchmarking workshop SemEval, we incorporated the used evaluation methodology, which is the de facto standard for evaluating text mining algorithms, in our software. SemEval (Bethard et al., 2016; Kilgariff and Palmer, 1998) is a long running workshop where current problems in text mining are targeted by releasing an annotated data set and issuing a challenge for researchers to compete and build the best algorithm for the given task. Tasks range from fine-grained sentiment analysis to semantic taxonomy enrichment. Going through a series of generalization steps to accommodate for different text mining tasks on data sets other then just the SemEval data, we arrived at a unified, powerful suite of tools that will be of use to any developer wishing to create the next successful application in the text mining industry.

This paper aims to describe the design principles of our proposed framework, as well as a number of practical use cases to illustrate its benefits. While we implemented the framework in the Java programming language, including a specific set of built-in tools tailored to our specific needs, the design patterns and architecture presented in this paper are general enough to be implemented in any object-oriented programming language. With this in mind, we included technical details in the system description to aid in this endeavor. A Java implementation of the framework is available at `https://github.com/KSchouten/Heracles`.

The paper is structured as follows. In Section 3.2 we present the related work by giving an overview of the existing frameworks for developing and evaluating text mining algorithms. Section 3.3 describes the design principles and architecture of the proposed Heracles framework. Five selected use cases are given in Section 3.4, showing the versatility of the proposed framework. Section 3.5 shows the results of our user study to evaluate the usefulness of the framework, followed by the conclusions and suggestions for future work in Section 3.6.

3.2 Related Work

There are many software frameworks (i.e., workbenches, toolkits, libraries, packages, etc.) for text mining, both open and proprietary, and we can name only a few in this section. Because of the closed nature of proprietary systems we are limited to discussing only open and freely available frameworks. All the investigated frameworks are listed in Table 3.1, along with some characteristics for each entry. The three main dimensions we use to evaluate existing frameworks are (1) their ability to support the natural language processing (NLP) part of developing a text mining algorithm, (2) their ability to support the machine learning (ML) part of developing a text mining algorithm, and (3) their ability to promote and enable rigorous evaluation of the developed algorithms. Furthermore, we look at whether a graphical user interface (GUI) or application programming interface (API) is available, whether parallel processing is supported, whether multiple input formats are possible, and, for NLP packages only, whether support for multiple languages is implemented. Last, we check if there are certain requirements for using the software, such as having the Java Virtual Machine running.

Many software packages integrate or provide wrappers for other pieces of software. This can confuse the license under which the software is made available. Hence, the license column in Table 3.1 refers to the license under which the core software is made available. Nevertheless, using third-party libraries will usually invoke the license of that component. For example, NLTK is licensed under the Apache license, which gives users a lot of freedom to use and modify the software, including commercial use. However, linked components might have their own license, so using, e.g., CoreNLP from within NLTK will invoke the stricter GPL license that CoreNLP uses.

The discussed software packages are divided into four groups. The first group consists of software that mainly provides NLP functionality. The second group is formed by software packages that provide support for machine learning algorithms. The third category is a singleton group, containing Gerbil, an independent evaluation-only Web framework. The last group is a set of frameworks that combine ML and NLP to support the development of new algorithms, in a fashion similar to our proposed Heracles framework. Last, we will present a brief feature overview of Heracles, to put it in perspective with the other discussed software packages.

	GUI	API	NLP Development	Multiple Languages	ML Development	Evaluation	Parallel Computing	Multiple Input Formats	License	System Requirements
Stanford CoreNLP	*	V	V	-		V			GPL v3	Java
NLTK	-	V	V	V	*	*			Apache v2	Python
OpenNLP	-	V	V	*	V	*			Apache v2	Java
xTAS	-	V	V	-		V	-		Apache v2	Linux, Python, and Java
Weka	V	V	-		V	V	V		GPL v3	Java
Scikit-learn	-	V	-		V	V	V		BSD	Python
TensorFlow	-	V	-		V	*	V		Apache v2	Python
Apache Spark	-	V	-		V	V	V		Apache v2	Java
Apache Mahout	-	V	-		V	*	V		Apache v2	Java
Mallet	-	V	-		V	V	*	V	CPL v1	Java
Watchmaker	*	V	-		V	-	V		Apache v2	Java
Java-ML	-	V	-		V	V	-	V	GPL v2	Java
GERBIL	V	V	-		-	V		-	LGPL v3	public REST interface
GATE	V	V	V	*	*	*	-	V	LGPL v3	Java
LingPipe	-	V	V	V	V	V	*	-	custom	Java
@Note	V	V	*	V	V	V	-	V	GPL v3	Java
CLULAB	-	V	V	*	V	V	*		Apache v2, GPL v3	Scala
Heracles	**-**	**V**	**V**	**V**	**V**	**V**	**V**	**V**	**GPL v3**	**Java**

Table 3.1: An overview of existing software packages and some of their characteristics.

3.2.1 Natural Language Processing Software

Currently, there are various software packages available that provide extensive NLP functionality. One well-known and widely used library is the Stanford CoreNLP (Manning et al., 2014). It can perform various NLP tasks like tokenization, part-of-speech tagging, lemmatization, but also named entity recognition, sentiment analysis, and grammatical parsing. It is written in Java, but it also has a server setup, where one can use a Web API to access its features from any language. CoreNLP is also integrated in several other software packages and frameworks, illustrating its positive qualities. CoreNLP can easily process sentences in parallel by setting the appropriate option to the number of threads that can be used. While licensed under the strictly open source GPL license, commercial licenses are available upon request.

The Apache OpenNLP (Apache OpenNLP, 2017) project has a different take on NLP by providing the developer with both pre-trained models as well as an easy way to train and evaluate the models on custom data. All algorithms are internal to the library instead of linking to other software packages. On the other hand, the NLTK (Bird et al., 2009) package aims to provide a full range of tools for NLP, and includes links to other packages to extend its functionality, such as Weka and CoreNLP. It also comes with a number of basic machine learning algorithms a developer can use to build a custom algorithm. However, advanced evaluation functionality like the cross-validation are absent. Both OpenNLP and NLTK do not explicitly provide parallel processing capabilities, although key components are thread-safe, enabling the developer to use them in a custom parallel computing setup. NLTK can be parallelized with third-party libraries, such as Execnet (Krekel, 2017), which uses the Message Passing Interface to safely communicate between processes that otherwise do not share any resources. Both OpenNLP and NLTK are licensed under the Apache license, giving the developer full freedom in how to use these packages.

A general text mining framework called xTas is proposed in (de Rooij et al., 2012), that includes both wrappers for popular packages as well as custom-built modules coming out of research. It uses Elasticsearch for distributed document storage and retrieval. While it is possible to run xTas locally as a Python module, it can also be run as a service, distributing tasks over multiple worker processes as necessary. Developers are encouraged to extend xTas by defining custom tasks, which will au-

tomatically run in a distributed fashion within the xTas framework. The xTas core software is licensed under the liberal Apache license, but several wrappers call upon software that is under stricter licenses. The system is developed for python, but currently runs only on linux operating systems. Java is required to use some of the provided wrappers, such as the one for CoreNLP.

3.2.2 Machine Learning Software

Next to the previously discussed NLP solutions, there are multiple frameworks for machine learning (ML), each with its own characteristics. A well-known framework in the scientific community is Weka (Witten and Frank, 2005), a general purpose, Java-based, machine learning framework that comes packaged with many algorithms. It provides proper evaluation support and includes also methods for feature selection and meta-optimization. Weka has specialized classes that can be used to utilize parallel computing, meaning that by default it will be single threaded. Its popularity is illustrated by the fact that various frameworks include wrappers for Weka (e.g., (Abeel et al., 2009; Cunningham et al., 2011)). An extra benefit is that Weka has a full-fledged graphical user interface where all implemented algorithms can be run over a dataset, which can be in any of the supported file formats. While no new algorithms can be developed in the graphical interface, it makes the existing ones accessible to a more general public. Weka itself is licensed under the GPL license, with various algorithms under different licenses, but commercial licenses may be available upon request.

Scikit-Learn (Pedregosa et al., 2011) could be considered the Python counterpart to Weka. It has multiple algorithms for the various machine learning tasks, such as classification, regression, and clustering. It is built on the popular NumPy, SciPy, and matplotlib libraries and is licensed under the liberal BSD license, which also allows commercial use. It can read files in the libsvm (Chang and Lin, 2011) format, as well as anything that is compatible with NumPy arrays or SciPy sparse matrices. Scikit-Learn has the permissive BSD license.

Also for Python is the Google TensorFlow (Abadi et al., 2015) project, a framework for deeply learned neural networks. It has APIs for various languages besides Python although not every API is as stable and as complete. TensorFlow supports

parallel processing, both on muliple CPUs and multiple GPUs with CUDA (Nickolls et al., 2008) support. While TensorFlow can evaluate the performance of a trained model, more advanced methods like support for cross-validation is not available. TensorFlow is licensed with the Apache license, allowing a full range of application.

Other frameworks that explicitly support parallel processing are Apache Spark (Zaharia et al., 2016) and the older Apache Mahout (Lyubimov and Palumbo, 2016). As Apache projects, both software packages have the Apache license. While Spark is a general purpose cluster computing framework, its built-in MLib (Meng et al., 2016) library provides a good number of machine learning algorithms that can be used in custom applications. In the past, Mahout used to run on Hadoop MapReduce (White, 2009), but nowadays it supports different engines, including the much faster Apache Spark. One of its main attractive features is its ability to perform linear algebra on big data sets.

For Java, another mention is Mallet (McCallum, 2002), which provides machine learning algorithms that are widely used in language processing. For example, it provides an implementation of Latent Direchlet Allocation (Blei et al., 2003) and graphical models in general, which is absent from, e.g., Weka. Sequence tagging is another area where Mallet complements Weka, by providing an implementation of Conditional Random Fields (Lafferty et al., 2001). Mallet is licensed under the CPL, a precursor to the Apache license that is similar in nature to the latter.

The Watchmaker (Watchmaker, 2017) framework provides evolutionary computing, like genetic algorithms, that can be used to optimize an arbitrary algorithm. The developer will need to translate the parameters to be optimized into a numeric parameter array and a specific function to compute the performance, given some input, needs to be implemented as well. Watchmaker will then be able to change, or evolve, the parameter array and get the performance for each. Keeping the good ones around and combining them into a new generation of possible solutions, etc., will result in a good set of parameters. Since this is a heuristic method, an optimal result cannot be guaranteed. While Watchmaker does not have a full graphical user interface, it does provide UI elements that can be integrated in a custom application that uses Watchmaker. For instance, a graphical tracker of the population, generations, and performance can be used to show progress of the optimizer. Watchmaker

can be incorporated in all kinds of software systems, as it is distributed under the Apache license.

As an honorable mention, we would like to list Java-ML (Abeel et al., 2009) here too, as it is one of the earlier Java frameworks for machine learning. It includes wrappers for Weka, making it easy to use any of its machine learning implementations. Other strong points include unified interfaces for each type of algorithm, and many examples and reference implementations.

3.2.3 Independent Evaluation Framework

A system that has scientific testing at its core is GERBIL (Usbeck et al., 2015). It successfully abstracts away from specific algorithms by specifying a common REST interface any algorithm should adhere to for using this framework. All algorithms implementing this interface can be easily linked up with the core benchmark module. Data sets are handled in a similar way, being stored in a different location (i.e., DataHub in this case). While GERBIL is primarily designed for the entity annotation task, the concept should apply to any task that produces annotations in text, which is the case for practically all text mining problems. A strong point of the framework is that experiments done using the REST API are permanently stored using a persistent URL, so previously performed experiments are logged and can be accessed and redone at the click of a button. In our current day and age where reproducibility is much talked about but often still a near impossible task, this is a great feature to have.

3.2.4 Text Mining Frameworks

In the text mining field, GATE (Cunningham et al., 2011) (General Architecture for Text Engineering) is a well established software package. With its GUI, it provides an easy way of building a custom NLP pipeline, stringing various text processing components together as needed. By means of various plugins it is possible to perform NLP tasks on text in languages other than English. It is also possible to create custom components for such a pipeline using the API, thereby extending the framework. When gold data is given (Biemann and Mehler, 2014), GATE can evaluate the created annotations, providing a graphical overview of which annotations were incorrect next to the standard performance metrics. By means of the Learning Framework Plugin,

it also provides support for machine learning components. In particular, the plugin provides wrappers for machine learning software, such as Mallet (McCallum, 2002), libSVM (Chang and Lin, 2011), parts of Weka (Witten and Frank, 2005), and Scikit-Learn (Pedregosa et al., 2011). Using more elaborate evaluation schemes such as k-fold cross-validation is not included in GATE. Its LGPL license allows developers to link their software to the GATE libraries without having to publish their own source code.

Another, well-known, framework in the field of text mining and NLP is LingPipe (Alias-i, 2017), which supports a variety of NLP and ML tasks. Examples include named entity recognition, topic modeling, classification, basic sentiment analysis, and spelling correction. It also supports the evaluation of algorithms, providing performance measures as well as options such as cross-validation. LingPipe does not provide built-in support for parallel computing, but it has thread-safe models and decoders for concurrent-read exclusive-write synchronization. While not licensed under a general license, it provides a royalty free license, not unlike the GPL, as well as various commercial licenses.

A framework that targets the biomedical domain, called @Note, is presented in (Lourenço et al., 2009). It encapsulates external libraries, such as GATE and Weka, and provides easy access to biomedical data by crawling the PubMed bibliographic catalogue (PubMed, 2017). It includes some functionality for model validation. In terms of architecture, @Note is built on top of a Java application development framework called AIBench (Fdez-Riverola et al., 2012), and hence inherits its patterns, such as the Model-View-Controller pattern. Furthermore, it is a modular framework allowing developers to build plugins for the system. @Note does not seem to support multiple languages by default, but given its link to GATE, which is multilingual, it might be possible to perform text mining tasks in other languages by building a plugin. Unfortunately, @Note does not support parallel processing, but it is completely open source.

The Computational Language Understanding Lab (CLULAB) at the University of Arizona has written several pieces of software around natural language processing and text mining. Their `processors` project contains components for event extraction and Rhetorical Structure Theory discourse parsing (Surdeanu et al., 2015), but also wrappers for Stanford's CoreNLP and MaltParser (Nivre et al., 2007). Furthermore,

it contains implementations for many common machine learning algorithms, supporting both classification and ranking tasks. The software is written in Scala under the Apache license, with the exception of the CoreNLP wrapper, which is under the GPL due to CoreNLP, which is integrated in that component, being under GPL. While CLULAB does not provide parallel processing support, the creators have made an effort to make the software thread-safe. Some of the provided components support multiple languages, but others, for instance the discourse parsers, do not.

The proposed Heracles framework, has been developed over the past four years to support both educational and research efforts at the Erasmus University Rotterdam. It does not have a graphical user interface, but focuses solely on providing a development environment to create new text mining algorithms. To that end, it supports various natural language processing components, mostly by wrapping the Stanford CoreNLP (Manning et al., 2014) package. Hence, it has the same support for multiple languages as CoreNLP does. Machine learning is supported by incorporating the Weka (Witten and Frank, 2005) toolkit. In that capacity, it is possible for developers to draw upon the parallel computing options that Weka provides. The Heracles framework itself does not provide specific features for parallel computing, but its components are thread-safe. Heracles can also be linked to the Watchmaker (Watchmaker, 2017) framework to provide evolutionary computing capabilities. Evaluation of algorithms is supported in various forms, including the basic random split into training and test set, but also cross-validation is available. Multiple algorithms can be run in sequence and tested against each other for statistically different results using a 2-sided paired t-test. Heracles supports a number of different input formats, including raw text, JSON, and XML formats. For specific data set formats, this can be tweaked so existing annotations are correctly loaded. Heracles is written in Java and due to wrapping CoreNLP, is licensed under GPL.

3.3 Design and Architecture of Heracles

In this section, we describe Heracles, our proposed framework and discuss its design principles and overall architecture. With the overview of related work in mind, the

goal is to present a framework that supports the development process of new text mining algorithms, including evaluating their performance.

When designing any framework, the target audience is one of the key concepts that determines the design choices. For the framework we developed, the targeted group of users is developers. For developers, the big advantage is that existing NLP and ML methods are easily available when developing a custom text mining algorithm, with the whole process of development and testing being performed within a streamlined evaluative workflow. This is achieved by providing boilerplate code and templates that ensure the developer can utilize the workflow, while still providing flexibility to work on different types of text mining algorithms.

There are three major classes of algorithms that Heracles is designed to support. The first is *classification*, where a given word or sequence of words has to be labeled with a value from a preset list of labels. Examples of this class of algorithms are word sense disambiguation, and positive vs. negative sentiment analysis. The second class of supported algorithms is *regression*, where a given word or sequence of words has to be labeled with a numerical, real value. Examples of regression in text mining are predicting sentiment as a value between -1 and 1, or predicting stock prices. The third class is *sequence labeling*, which is the problem of finding and labeling sequences of words within a text. Examples of this are named entity recognition, and explicit aspect detection for sentiment analysis, but also the basic NLP task of splitting a text into sentences can be considered sequence labeling.

The workflow, as supported by the framework is illustrated in Figure 3.1. It starts by reading a certain raw dataset and performing various NLP tasks on that dataset. The processed dataset is then optionally saved in a JSON format that preserves all the information as computed by the NLP components, so that the often slow NLP procedure does not have to be repeated with every run. Then the developer sets up an experiment, assigns it the processed data, adds the empty algorithm that will be developed, possibly also adds existing algorithms to compare against, and specifies the evaluation conditions (e.g., cross-validation or not). Now the developer is ready to start working on the algorithm, and, while in the development process, whenever a test run is required, the experiment can simply be executed. Optionally the results of the test run can be saved, either as just a report of the last experiment, or as an annotated data set that includes the predictions so it can be evaluated externally.

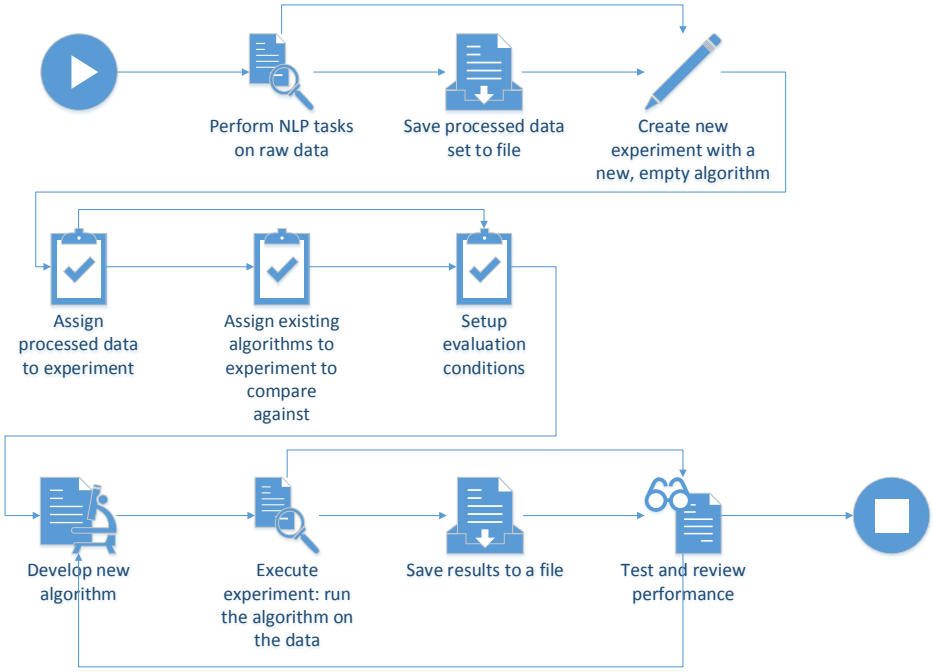

Figure 3.1: The basic developer workflow within Heracles

After inspecting the results, the developer can continue working on the algorithm, for example by adjusting for possible errors that were encountered during the last test run.

3.3.1 Design Patterns

Adhering to software engineering best practices, we have designed the framework using several existing architecture patterns. The simplified UML class diagram in Figure 3.2 shows how the main components of the framework are organized according to various design patterns. The first major pattern is a Layer pattern (Taylor et al., 2009), separating the various functionalities within the framework. One can distinguish between three layers, each having its own function and within one layer, various

implementations can easily be swapped for one another. The first layer is the *data layer*, where datasets are read from various raw sources and processed data models are written back to various file formats. Different implementations of this *data layer* allow for data sets in different formats to be loaded into the Heracles framework. Currently, implementations exist for reading and writing various XML and JSON formats. The second layer is the *natural language processing layer*, where all the required natural language processing is performed so that the developer has access to high level linguistic features when developing a text mining algorithm. Again, implementations of this layer may vary, depending on, for example, the natural language of the dataset. The third layer is the *algorithm layer*, which is the developed algorithm. Naturally, many different algorithms can be created that will all match the signature of this layer. The first layer is defined as an interface which any implementation of that layer needs to adhere to, to ensure compatibility between the layers. The second and third layer are defined using a Template Method pattern (Gamma et al., 1994) (i.e., an abstract class), which provides not only the interface but also additional functionality that is shared across all implementations of this layer.

The communication between the three main layers is defined using the internal data model, which in turn can be viewed as an API the developers can use in the *algorithm layer* to access data features. In that capacity, the data model, once finished, becomes read only and, to that end, implements the Iterator pattern (Taylor et al., 2009), exposing the data using iterators to prevent access to the underlying data representation. The communication between the layers is as follows. First, the *data layer* reads a raw data file and creates an internal data model that reflects as much of the original information as possible. It then passes the partially constructed data model to the *natural language processing layer*, which will complete the data model and make it read only, before passing it on to the *algorithm layer*, where it will be used to train and test the given algorithm implementation. For example, if the dataset already provides the text split in sentences, these sentence splits will be used when creating the data model. Otherwise, the *natural language processing layer* will execute a component that will try to determine the best places to split the text into proper sentences. Thus, utilizing information provided in the data file is favored over retrieving that same information using natural language processing techniques. Hence, the usually human-annotated information already in the data file is strictly

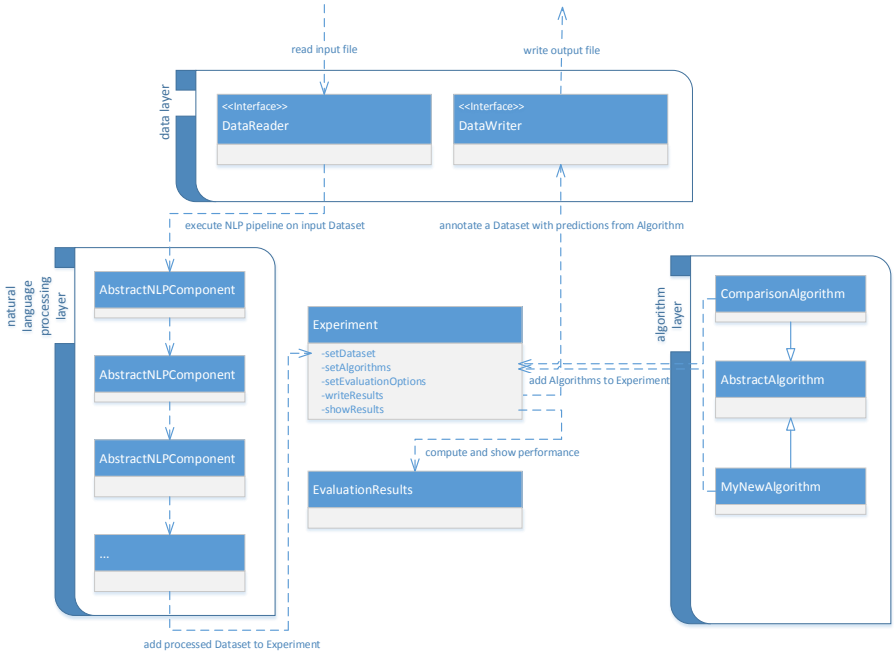

Figure 3.2: The main components of the framework, organized by layer

complemented with linguistic information from the *natural language processing layer*. It is up to the `data layer` implementation to properly execute this behavior.

Benefits of using this Layer pattern are that implementations can vary within a given layer. This makes the framework flexible and extendable. Different language-specific implementations for the *natural language processing layer* can easily be swapped in, just like different data readers, so practically any textual data set can be loaded in the framework. For the *algorithm layer*, the main benefit is that any algorithm, developed according to the specifications of the layer, can be automatically executed and tested without any additional effort from the developers side. The downside of using a Layer pattern is that in the rare instance of having to update the communication between the layers, some manual refactoring is needed.

Within the *natural language processing layer* and the *algorithm layer*, a Pipeline pattern (Gamma et al., 1994) can be discerned. All natural language processing components are modular and they can be executed over a given data model. Each module is labeled with the task it performs as well as with a list of prerequisite tasks that have to be performed before this module can be executed. In this way, multiple implementations of the same task can exist within the framework, and depending on the overall goal, a specific module can be selected. When a module has been executed, the data model is labeled with that task, enabling the framework to automatically check whether it is possible (or necessary) to run a certain module on a given data model.

For the *algorithm layer*, the Pipeline pattern is more static, with a general set of steps that is the same for all algorithms: preprocessing of the data, training of the algorithm, predicting for unseen data using the trained algorithm, and evaluating the predicted annotations. The series of steps that together comprise an algorithm's execution is discussed in more detail in a subsequent section. The main benefit of the Pipeline pattern for the *algorithm layer* is that it allows for rapid prototyping. Every algorithm will have predefined methods (i.e., stubs) which will be replaced with the logic of the algorithm. Methods that are not relevant for a given scenario can simply be left as stubs. For example, algorithms that do not require a supervised training phase can leave the `train` method empty. To provide machine learning support in this layer, two machine learning libraries have been integrated into Heracles: Weka (Witten and Frank, 2005), which supports many statistical machine learning methods, and Watchmaker (Watchmaker, 2017), for evolutionary computing support.

To utilize all three layers, a Builder pattern (Bloch, 2008) is used that creates an *experiment* using method chaining. Building up an experiment using method chaining is a convenient way to provide all the necessary information, such as which data set to use, which algorithms to run and compare, how many runs to make, whether to use cross-validation or not, etc. Depending on the options set by the developer, an experiment ends by providing the performance results or by providing an annotated dataset. The latter is useful when evaluation is done externally. Encapsulating experiments in this manner allows the framework to record all run experiments for archival purposes, and since experiments are independent from each other, multiple experiments can be run, both in a serial and parallel manner.

3.3.2 Class Architecture

The central component in the simplified UML class diagram in Figure 3.2 is the `Experiment` class. This class has methods to assign a `Dataset` and one or more `Algorithm`s to it. It also has various evaluation options, such as methods to use cross-validation, or methods to repeat the experiment a number of times. The `Experiment` class has two output modes: one is to use an `Algorithm` to annotate a dataset, the other uses an already annotated dataset to test and compare the performance of one or more `Algorithm`s. The latter has built-in support for significance testing, using a paired t-test. The data model, contained in the `Dataset` is passed around and is not shown in this diagram.

A `Dataset` object is created using an implementation of the `DataReader` interface. An object implementing the `DataReader` interface is expected to provide a read method that creates a `Dataset`, most likely by reading some raw textual data from a (set of) files. It will then count on a series of components in the *NLP layer* to do the necessary natural language processing so that a completely processed `Dataset` can be created. Since some data sets already provide a certain amount of structure or annotations, the `DataReader` is expected to create as much of the `Dataset` as it can, letting the components in the *NLP layer* do the rest. Using the interface design principle, an array of different `DataReader`s can be used to allow data sets in different formats to be processed in the framework. If a `DataReader` does not yet exist for a given data set, it is always possible to build one and plug it in.

The same modular approach is chosen for the NLP layer, where multiple subclasses of `AbstractNLPComponent` can be chained to form an NLP pipeline. Each subclass of `AbstractNLPComponent` can have its own requirements which are checked before attempting to execute this component. For instance, before being able to perform lemmatization, which is labeling each word with its dictionary form, each word first needs to labeled with its part-of-speech tag, which is the word type (e.g., noun, verb, etc.). Hence, the lemmatization component will check whether the part-of-speech tagging component has already been executed and will throw an error if that is not the case. Thus, the developer can freely choose the components that are going to be used as long as the individual components' requirements are met. Currently, Heracles has wrappers for the majority of the components of the Stan-

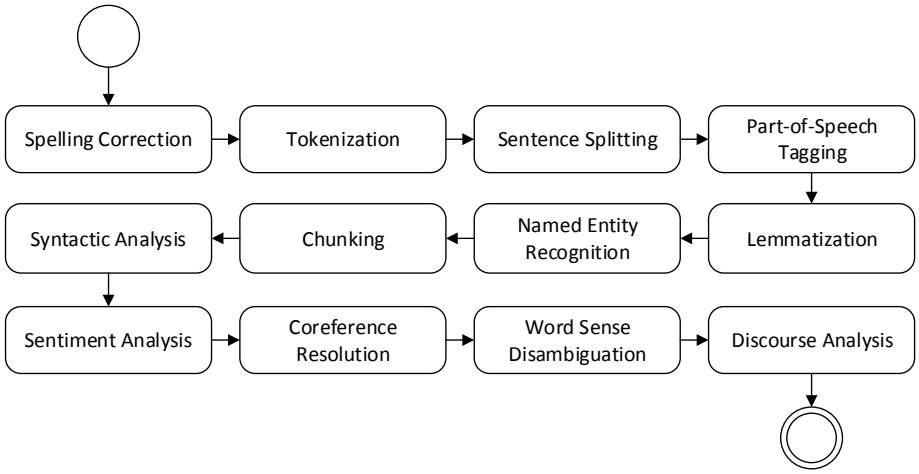

Figure 3.3: Activity diagram of the complete English NLP layer.

ford CoreNLP (Manning et al., 2014) library, a wrapper for the CLULAB Rhetorical Structure Theory parser (Surdeanu et al., 2015), and a component for matching text with concepts in a knowledge base (i.e., ontologies) using the Jena (Apache Jena, 2017) library for ontology communication. A visualization of all the components in the implementation of the *NLP layer* for English is given in Figure 3.3.

Given a loaded `Dataset`, one can execute an `Algorithm` object on that data, which will result in one or more `EvaluationResults` objects being returned. Since one `Algorithm` can perform multiple tasks that are evaluated separately, it is possible that multiple `EvaluationResults` are generated, one for each task. The `Algorithm` class is an abstract class, providing a template for all new algorithms that are to be developed. The main task of the user is to build their own algorithm and run it in the framework. By following the given template in `Algorithm`, the framework is able to automate many processes, including giving the right data to the right method (e.g., provide just the training data to the train method) and evaluating the output of the algorithm. The predictions that an algorithm makes are stored locally within

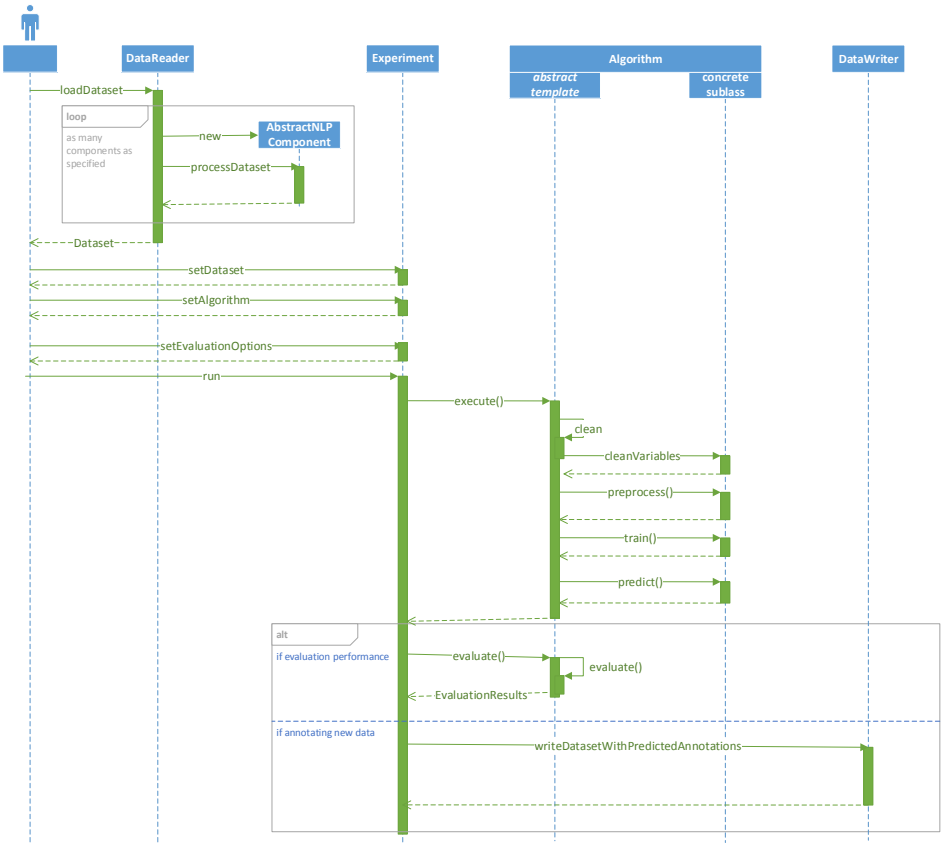

Figure 3.4: Sequence diagram of running an algorithm in the framework

the algorithm instance itself, so the loaded data model does not change. In fact, it is read-only to prevent data leakage or data corruption.

3.3.3 Algorithm Evaluation Procedure

As mentioned before, the framework provides a template to use when developing custom algorithms to ensure every algorithm can be run and evaluated automatically by the framework. The process of running an algorithm is shown as a UML sequence

diagram in Figure 3.4. The first step is to load a `Dataset` using a `DataReader`, which can be one provided with the framework or a custom built one. Then the developed algorithm is instantiated. To distinguish between custom functionality and functionality provided in the template, the `Algorithm` has two distinct lifelines, one for the processes described in the template and one for the processes described in the developed instantiation, or subclass, of the template. Both algorithm and data set are provided to the `Experiment`, which keeps track of those. Then, after setting the necessary evaluation options, the experiment is run.

The experiment splits the dataset in training and test set, depending on the evaluation options, and provides it to the algorithm. The `execute()` method, defined in the template, will go through all the required steps in order to successfully evaluate the performance of this algorithm. The first step is to clean the algorithm data, making sure that there are no results from previous runs leaking into this run. This is of special importance since the predictions of an algorithm are stored locally within the `Algorithm` object so, if not for this cleaning step, predictions would spill over from one run to the next, polluting the evaluation results. One part of the cleaning is to call the `cleanVariables()` method in the developed subclass. The developer is responsible for making sure that any variables that are trained on the data are initialized in this method, so they are reset when a new execution cycle begins. If any parameter of the model used by the developed algorithm is not properly reset, the algorithm could potentially be using information from previous runs which it should have no access to.

The next step is to call the `preprocess(allData)` method. In this method the developer gets access to the whole dataset to do some preprocessing of the data, if necessary. This preprocessing should not use any of the information that this same algorithm tries to predict, since that would defeat the purpose of out-of-sample testing. One can think of activities like counting word frequencies, or performing and caching lookups in knowledge bases. Any information gained in this procedure should be stored locally in the developed algorithm. The next phase is to train the algorithm using the provided training data. Here, the algorithm is supposed to look at the information it needs to predict in order to tweak any parameters, set model weights, etc.

Then, the algorithm is ready to process the test data, which is that part of the data which was excluded from the training data to perform out-of-sample testing. The developed algorithm should predict whatever information it is targeted to predict (i.e., a sentence-level sentiment analysis algorithm will predict a sentiment value for each sentence). These predictions are stored locally in a variable that is defined in the template. Because of that, the `evaluate()` method, which is defined in the template, will have access to these predictions and it will compare those with the provided human annotations in order to compute a performance score for the developed algorithm. For each task the developed algorithm provides predictions for, the evaluation method will create a `EvaluationResults` object to record the performance. All `EvaluationResults` are returned to the user at the end of the process.

Alternatively, the algorithm can use the `predict()` method to annotate data that does not have manually assigned labels. The output will then consist of the automatically annotated data set, and no performance evaluation is performed as the predicted annotations cannot be compared against a gold standard.

3.3.4 Internal Data Model

The data model that is created by the `DataReader` and NLP Pipeline components contains all the linguistic data needed for an algorithm to perform the text mining task at hand. In Figure 3.5, a UML class diagram is given for the data model as created by the English NLP Pipeline. Note that the data model itself is extendable and that not every component is mandatory. There are three main subclasses of `DataEntity` that are contained in a `Dataset` object: `Word`s, `Span`s, and `Relation`s. A `Word` object represents a single word in the text, a `Span` represents a sequence of `Word`s, and a `Relation` object models an arbitrary, directed relation between two instances of `DataEntity`.

While there can be many types of `Span` objects (e.g., sentences, documents, named entities, etc.), the `Dataset` object knows which type of `Span` is the top-level `Span`. This top-level `Span` is the textual unit, modeling pieces of text that are independent from the other pieces of text that exist in this `Dataset`. For example, a `Dataset` might contain user reviews about a certain product. Then each review is independent from each other review and thus the review is the textual unit. The sentences within

each review, however, are semantically related to each other and hence cannot be the top-level `Span` or textual unit.

Since `Span`s are sequences of `Word`s, and there can be many types of, potentially overlapping, `Span`s, the framework provides many methods to get the `Span`s of interest, based on type, which textual unit they belong to, and whether or not they overlap or touch a specific `Span`. `Span`s implement the `Iterable<Word>` interface which ensures that they can be used with the `for...each` construct in a natural way.

`Relation`s can be used to create relations between `Word`s, such as grammatical dependencies, or to create relations between `Span`s, such as discourse relations between parts of the text. Furthermore, `Relation`s can be used to model a relation between a `Word` and a `Span`, for example to denote the head word of a parse tree constituent.

This data model is flexible enough to model any natural language phenomenon without the need for additional classes. This makes serialization a practical endeavor, as the provided readers and writers for the data model will cover any use of this data model without the need for programmatic modification.

3.4 Use Cases

The framework we propose has already been used in several concrete applications. The most prominent use case has been the development of new sentiment analysis algorithms. Sentiment analysis is a popular research area (Feldman, 2013; Liu, 2012; Pang and Lee, 2008), both in academia and in business, because it is a challenging problem with many useful business applications. The central problem is to find the sentiment expressed in a certain text, with varying degrees of granularity. Sentiment can be determined for complete documents or for full sentences, but even more interesting is the task of aspect-level sentiment analysis (Schouten and Frasincar, 2016), where sentiment is computed with respect to the entities and aspects of entities that are actually discussed within the text. This task naturally breaks down into aspect detection and sentiment analysis for aspects. Aspects can be explicitly mentioned in a sentence, which means that there is a sequence of words (e.g., “glass of wine”) that

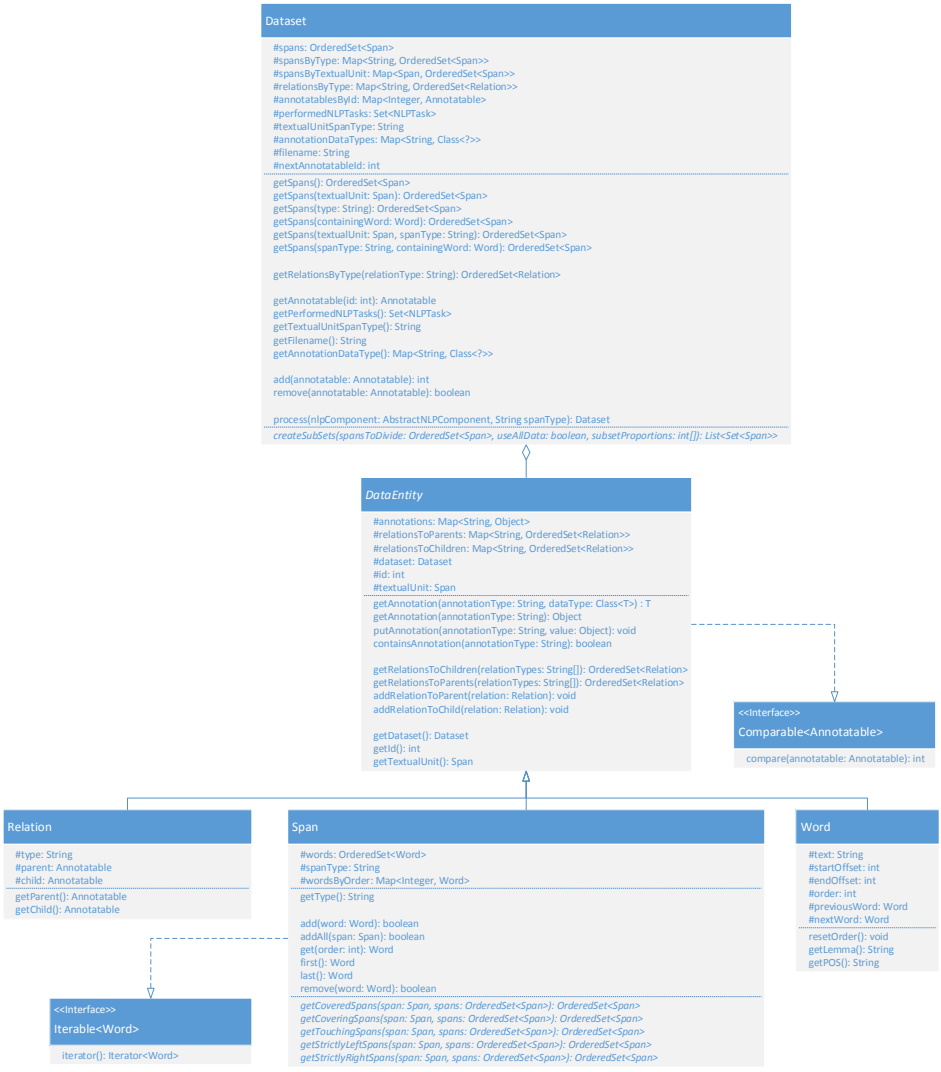

Figure 3.5: Class diagram of the data model used within the framework.

denotes that aspect, but they can also be implied by the sentence as a whole (e.g., "they threw the dishes on our table!", implying the `service` aspect). Each aspect has its own associated sentiment score that has to be determined.

One of the more recent projects that has been developed using Heracles is an ontology-enhanced algorithm (Schouten et al., 2017b) for both the aspect detection subtask and the subtask of classifying sentiment per aspect. In this work, an ontology, which is a formally specified knowledge base that holds domain-specific information, is used to improve the performance of a classical machine learning approach to both sub-tasks. To that end, the framework is enriched with an ontology component that uses the Jena (Apache Jena, 2017) library to connect with an ontology and this component provides functionality to search an ontology, add concepts, and get inference results from it. This component is used in a custom-built pipeline module, which is in the NLP layer, that labels the text with all the matching concepts found in the ontology. The ontology itself has concepts that cover aspects as well as sentiment.

A major advantage of using ontologies is the support for reasoning, which allows certain information to be inferred rather than having to explicitly specify all facts. For example, the ontology states that `delicious` is a sentiment expression for `Sustenance`. Then, because the `Soup` concept is a subclass of `Food`, which is in turn a subclass of `Sustenance`, we can infer that `delicious` is also a sentiment expression for `Soup`. This allows us to use the ontology to better link sentiment expressions to aspects, as we know that `delicious` cannot express sentiment about the location of a restaurant, for instance. This is also useful for aspect detection, because when a sentiment expression that is exclusively linked to an aspect, is found in a certain sentence, one can conclude that its linked aspect has to be present. This is especially useful for implicit aspects, where the aspect itself is not literally mentioned.

The machine learning component comes from the linked Weka (Witten and Frank, 2005) library, and this means that the various input features have to be specified in terms of Weka objects. Input features are things like which words are present in a sentence, which ontology concepts are found in the sentence, etc. While it can be considered an extra burden, having to transform between the data model used in Heracles, and the data model used by Weka, there are many advantages. The biggest one is that once transformed, there is no cost in transitioning to a different machine learning model. For the ontology-enhanced aspect-based sentiment analysis project,

we used a Support Vector Machine (SVM), but with just a small number of changes a different model could be used, such as a Random Forest, or a Logistic Regression. Furthermore, Weka has built-in options for feature selection and meta-parameter optimization (like the c and γ for an SVM). By taking the effort to link to Weka and transform to its data model, all this functionality comes at almost no extra costs in terms of development time.

In a different project, we look at whether using the discourse structure of a review can lead to a better defined scope for sentiment classification of aspects (Hoogervorst et al., 2016). Since multiple aspects can appear within the same sentence and some of them may be positive, while others may be negative, it is not correct to simply use the whole sentence to determine the sentiment of an aspect. Hence, looking at the discourse structure of a review is hypothesized to allow us to select only that part of the review that is relevant for a given aspect. To that end, an NLP module has been created that wraps around the CLULAB Rhetorical Structure Theory parser (Surdeanu et al., 2015), which divides a text into blocks called Elementary Discourse Units (EDU) and determines the rhetorical roles between those blocks. This information is then added to the data model using `Span` objects to denote the EDUs, and `Relation` objects to denote the rhetorical roles between the EDUs. This enables us to use this discourse information in the developed algorithm.

To properly use the discourse information, a weight is assigned to each rhetorical role and that weight is used to discount or emphasize certain parts of the review. These weights are optimized using the Watchmaker Framework (Watchmaker, 2017), and the developed algorithm implements the necessary Watchmaker interface which allows Watchmaker to call the algorithm with a certain set of weights for the rhetorical roles and receive the performance that would yield. These performance numbers then guide the evolutionary algorithm employed by Watchmaker to create better solutions. After a given number of trials, by specifying population size and generation limit, the best solution found thus far is returned. Since this is a heuristic method, this solution does not have to be the optimal one, and running Watchmaker again will most likely yield a different solution. However, because the algorithm does not have to convert to a certain data model, like when using Weka, one can optimize practically any type of algorithm using this framework, even those that are not easily converted to a classical machine learning approach.

Another use case demonstrates how sentiment analysis can also be seen as a regression type problem instead of a classification problem (Schouten et al., 2017a). In this setup, the data consists of news headlines, which are all labeled with the company of interest and a sentiment value between -1 and 1. Since this is a numeric value, it is no longer a classification but a regression number. Fortunately, we built the annotations object to have a dynamic casting procedure, so we can store practically anything as an annotation value. The data model knows what kind of data we store for each key and when retrieving annotation values, the stored objects are automatically typecast into the proper type. Again, using the link with Weka, we can use various regression solving algorithms, and in this particular case we chose a Support Vector Regression model. This model included input features that checked an ontology with company names, business concepts, and sentiment expressions, besides the regular bag of words features.

Last, in another project Boon et al. (2016) we investigate the performance of several word sense disambiguation algorithms. These algorithms are tasked with linking a word to a word sense in the WordNet (Fellbaum, 1998) semantic dictionary, given the context the word is in. This task played a central role in the original SensEval (Kilgariff and Palmer, 1998) workshops, the precursor to the already mentioned SemEval workshop series, that is still ongoing (Bethard et al., 2016). An example of word sense disambiguation is mapping the word 'Turkey' in "Turkey at Thanksgiving was delicious" to the sense that represents the animal instead of the country. To test the various algorithms that exist in the literature, the SemCor (Mihalcea, 2016) benchmarking data set is used, where each word has been manually annotated with the correct sense. While this is quite a different task than sentiment analysis, the Heracles framework can be used for tasks like this without any difficulties, as both the machine learning component from Weka is generic, as is the evaluation procedure in Heracles, which can evaluate any kind of annotation in text.

3.5 Feedback

During the development of Heracles, it has been used by several groups of students for assignments, as well as individual students for thesis projects. Over the last four

years we have gathered feedback from around 10 teams of roughly 4 students each, and from about 10 individual students. The individual students were oftentimes also involved in one of the groups, but as they have additional experience with the system, their input has been particularly valuable. Most students are in their senior year of the econometrics bachelor but some are doing an econometrics master program. Both programs are part of the Erasmus School of Economics at the Erasmus University Rotterdam. Asking the students directly for feedback has yielded useful information which has lead to further improvements to the system's architecture and implementation. A summary of the received feedback is provided in the next subsection, followed by a discussion of these remarks.

3.5.1 Remarks Received

What was most clear was that students appreciated the framework as a starting point, allowing them to use many functions that were already provided. In particular, the methods for reading and writing data sets, running data through a natural language processing pipeline, and the automatic evaluation where regarded as a major time saver. Furthermore, the data model was considered to be easy to work with, mostly due to the fact that there are only a handful of classes involved.

While developing with any API or existing framework requires a significant time investment in order to get acquainted with the ins and outs of the code, the Heracles framework does not have a very steep learning curve. It took individual bachelor students, who did only an introductory Java course, only about two weeks to get up to speed. For groups of students, this was obviously a bit quicker.

Another positive aspect, as noted by many students, is the fact that many example implementations are provided that show how to perform certain tasks. For example, the framework comes with a number of algorithms already implemented so users can see how various concepts are realized.

Over the years, students have remarked that certain design choices were not always clear. This resulted in not being able to find a specific piece of code in the most logical place. Subsequent iterations have addressed this issue, and this comment was not heard with respect to the latest version. Another change compared to previous versions is the fact that while previous versions where specifically tailored

to sentiment analysis, the current version is more generic. This comes however at the cost of a higher level of abstraction, which makes the code a bit harder to understand.

3.5.2 Discussion

From the feedback we received, it clearly shows that Heracles is considered an asset when developing text mining algorithms. Not only does it help to jump start the development process with all the existing code, it also ensures that the outcomes are scientifically valid, so developers can immediately see what works and what does not.

While starting development with Heracles requires an initial time investment, the required time to get up to speed is reasonable. This is partly due to a clear organization of the code, proper documentation, and the many provided examples.

While the framework is designed to ensure scientifically valid results, it is always possible to change the code of the framework and break this functionality. Currently, development is done within the Heracles software project, so any piece of code can be changed. Putting the Heracles core in a library format, which is thus no longer easily changed, might mitigate this potential problem. While protecting future developers against themselves can be a good thing, it also limits the possible contributions to the framework itself. Such a trade off would need to be carefully considered, and the exact choice likely depends on the exact scenario in which Heracles is used. Our advice is to keep everything open and accessible, and provide clear instructions so users do not inadvertently change important code. With the various student projects, we did not encounter difficulties with this approach. As a more external evaluation, we would like to point to the various publications (cf. Section 3.4) that have resulted from using the Heracles framework. This shows the flexibility of the software because of the different problems and approaches, it also illustrates the fact that Heracles is not a theoretical construct, but a software framework that has been put to the test with all the refinement and improvements that come with that.

3.6 Conclusions and Future Work

In this paper we have proposed a versatile framework for developing and evaluating text mining algorithms. With the importance of text mining in various business

domains (e.g., Fan et al. (2006); Kumar and Ravi (2016); Netzer et al. (2012)), our framework, which is open source, allows for easy experimentation with existing and future text mining algorithms.

In conclusion, we can state that the framework fulfills its objectives in aiding the development process and providing a solid scientific validation procedure for text mining algorithms. The framework is designed to be flexible and extendable, and it has shown to possess these qualities by virtue of the different use cases it has already been applied to. Hence, we believe that anyone who develops algorithms in the field of text mining, be it in a business or in academia, can benefit from the framework we propose. Users can opt to use the Java implementation we have built ourselves, or they can choose to use the design principles laid out in this work to build or improve their own text mining framework.

As with any software, there are multiple possibilities for improvement. One of the things that would help new users is an active community of users where developers can ask questions and get help. This aspect of software development is an important consideration when deciding which framework or language to use. In terms of the software itself, the most desirable feature that is not yet implemented is support for parallel computing. Since natural language processing and text classification can take quite a long time, providing tools to speed up this process are most welcome. This could range from relatively simple multi-threading to support for cluster computing paradigms like Spark (Zaharia et al., 2016).

Chapter 4

Using Co-occurrence Data to Find Aspects*

With the explosion of e-commerce shopping, customer reviews on the Web have become essential in the decision making process for consumers. Much of the research in this field focuses on explicit aspect extraction and sentiment extraction. However, implicit aspect extraction is a relatively new research field. Whereas previous works focused on finding the correct implicit aspect in a sentence, given the fact that one is known to be present, this research aims at finding the right implicit aspect without this pre-knowledge. Potential implicit aspects are assigned a score based on their co-occurrence frequencies with the words of a sentence, with the highest-scoring one being assigned to that sentence. To distinguish between sentences that have an implicit aspect and the ones that do not, a threshold parameter is introduced, filtering out potential aspects whose score is too low. Using restaurant reviews and product reviews, the threshold-based approach improves the F_1-measure by 3.6 and 8.7 percentage points, respectively.

---

*This chapter is based on "Kim Schouten and Flavius Frasincar. Finding Implicit Features in Consumer Reviews for Sentiment Analysis. In *14th International Conference on Web Engineering (ICWE 2014)*, volume 8541 of *Lecture Notes in Computer Science*, pages 130-144. Springer, 2014." complemented with notions from "Nikoleta Dosoula, Roel Griep, Rick den Ridder, Rick Slangen, Kim Schouten, and Flavius Frasincar. Detection of Multiple Implicit Features per Sentence in Consumer Review Data. In *12th International Baltic Conference on Databases and Information Systems (DB&IS 2016)*, volume 615 of *Communications in Computer and Information Science*, pages 289-303. Springer, 2016." and "Kim Schouten, Flavius Frasincar, and Franciska de Jong. In 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 203-207. ACL, 2014."

4.1 Introduction

With the explosion of online shopping at e-commerce companies like Amazon (US), Bol (NL), Alibaba (CN), etc., the use of consumer product reviews has become instrumental in the decision making process of consumers. In fact, potential consumers trust reviews from other consumers more than information on the vendor's website (Bickart and Schindler, 2001). As a result, the number of reviews for a single product can be quite high, especially for a popular product. When a consumer is interested in the overall sentiment of a product, (s)he must first read through many of the reviews to come to a conclusion. Since reading through these reviews is a tedious process, this may hinder decision making. Therefore an efficient way of displaying the overall sentiment of a product based on costumer reviews is desirable.

Much of the current research in the analysis of product reviews is concerned with classifying the overall sentiment for a certain product. To better describe the overall sentiment of a product, it is useful to look at the sentiment per product aspect, sometimes referred to as product features. Sentiment classification per aspect can be difficult as a customer review does not have a standard structure and may include spelling errors and synonyms for product aspects. Although a consumer might explicitly mention an aspect for a product, many of the important aspects are mentioned implicitly as well. For example:

"The battery of this phone is quite good."
"The phone lasts all day."

In the first sentence, the battery is explicitly mentioned and the second one refers to the battery lasting all day. Notice that while in the second sentence the battery is not explicitly mentioned, we can infer that the comment is about the battery. This inference is based on the other words in the sentence that direct the reader towards the actual aspect being described. This mapping from words in the sentence to the implied aspect must be shared between writer and reader of a text in order for the reader to understand what the writer meant to imply. Because of this, it is usually a small group of well-known, coarse-grained aspects that is used implicitly. Examples include generic aspects like price, size, weight, etc., or very important product-specific aspects like the already mentioned battery, sound quality, ease of use, etc. Since it

is this class of aspects that is often implied, it is important to include them in any sentiment analysis application, as they represent key aspects for consumers. This research presents a method to both determine whether an implicit aspect is present in a sentence, and if so, which one it is. After describing some of the related work that inspired this research, the proposed method will be presented, followed by the evaluation of its performance. This will lead to the conclusions and suggestions for future work in the last section.

4.2 Related work

While many methods have been proposed to find aspects for the task of aspect-level sentiment analysis, most of them focus on explicit aspects only. This is logical, given that the vast majority of the aspects in consumer reviews is mentioned explicitly. However, as discussed in the previous section, it is often the important aspects that are mentioned implicitly. Alas, only few works focus on this task. One of the first to address the problem of detecting implicit aspects is Su et al. (2008). An interesting solution is presented in the form of semantic association analysis based on Pointwise Mutual Information. However, since no quantitative results are given, it is impossible to know how well this method performs.

In Hai et al. (2011), a method based on co-occurrence Association Rule Mining is proposed. It is making use of the co-occurrence counts between opinion words and explicit aspects. The latter can be extracted from labeled data, or can be provided by an existing method that finds explicit aspects. Association rule mining is used to create a mapping from the opinion words to possible aspects. The opinion word then functions as the antecedent and the aspect as the consequent in the rules that are found. When an opinion word is encountered without a linked aspect, the list of rules is checked to see which aspect is most likely implied by that opinion word. On a custom set of Chinese mobile phone reviews, this method is reported to yield an F_1-measure of 74%.

Similar to Hai et al. (2011), the same idea of association rule mining is used in Wang et al. (2013). With association rule mining being used to find a set of basic rules, three possible ways of extending the set of rules are investigated: adding

substring rules, adding dependency rules, and adding constrained topic model rules. Especially the latter turned out to be a successful way of improving the results. By constraining the topic model (e.g., Latent Dirichlet Allocation (Blei et al., 2003) in this case), to include one of the aspect words and build the topic around that word, meaningful clusters are generated. Thus, a different way of finding co-occurrences between aspects and other words in the text is used, and it is reported that this complements the association rule mining method. The best reported result is an F_1-measure of 75.51% on a Chinese data set of mobile phone reviews.

Instead of using annotated explicit aspects, Zhang and Zhu (2013) uses the idea of double propagation (Qiu et al., 2011) to find a set of explicit words and a set of opinion words. An advantage is that the found explicit aspects are already linked to appropriate opinion words. Then a co-occurrence matrix is created, not between only opinion words and explicit aspects, but between the words in the sentences and the found explicit aspects. In this way, the right implicit aspect is chosen, not based on just the opinion words in the sentence, but based on all words in the sentence. The opinion words in the sentence are used to constrain the number of possible aspects from which the right one must be chosen: only aspects that have co-occurred with the encountered opinion word before, are eligible to be chosen.

In the previously introduced method, for each eligible explicit aspect, a score is computed that represents the average conditional probability of a aspect being implied, given the set of words in the sentence. The aspect with the highest score is chosen as the implicit aspect for this sentence. This method is reported to yield an F_1-measure of 0.80 and 0.79 on a Chinese corpus of mobile phone reviews, and a Chinese collection of clothes reviews, respectively. Like Wang et al. (2013), it uses all words to find implicit aspects instead of only opinion words as in Hai et al. (2011), and, apart from a small seed set of opinion words, it operates completely unsupervised.

However, there are several drawbacks that are apparent, both in Hai et al. (2011), Wang et al. (2013), and in Zhang and Zhu (2013). The first problem is that only aspects that have been found as explicit aspects somewhere in the corpus can be chosen as implicit aspects. This assumes that the same aspects are present in reviews, both explicitly and implicitly. However, as we have discussed before, well-known or important aspects are implied more often than aspects that are less

important or less described. Furthermore, by counting the co-occurrence frequencies between a aspect that is mentioned explicitly and the words in the sentence, it is assumed that when the aspect is used implicitly, the same sentential context is present. We argue, however, that this is not necessarily the case. For example, when saying that 'this phone is too expensive', the word 'expensive' prevents the word 'price' from being used. Either one uses the word 'expensive', or one uses the word 'price'. Because of that, there is no real co-occurrence between 'expensive' and 'price', even though the first definitely points to the latter as its implicit aspect.

4.3 Method

In this section the issues discussed in the previous section are addressed and an algorithm is presented that improves upon previous work in the given, more realistic, scenario. This scenario entails the following:

- Sentences can have both explicit and implicit aspects;
- Sentences can have zero or more implicit aspects;
- Implicit aspects do not have to appear explicitly as well;
- The sentential context of explicit aspects does not have to be the same as the sentential context for implicit aspects.

The algorithm first scans the training data and constructs a list F of all unique implicit aspects, a list O of all unique lemmas (i.e., the syntactic root form of a word) and their frequencies, and a matrix C to store all co-occurrences between annotated implicit aspects and the words in a sentence. Hence, matrix C has dimensions $|F|$ x $|O|$.

When F, O, and C have been constructed, processing the test data goes as follows. For each potential implicit aspect f_i, a score is computed that is the sum of the co-occurrence of each word in the sentence divided by the frequency of that word:

$$score_{f_i} = \frac{1}{v} \sum_{j=1}^{v} \frac{c_{i,j}}{o_j}, \tag{4.1}$$

where v is the number of words, f_i is the ith aspect in F for which the *score* is computed, j represents the jth word in the sentence, $c_{i,j}$ is the co-occurrence frequency of aspect i and lemma j in C, and o_j is the frequency of lemma o in O. Subsequently, for each sentence the highest scoring aspect is chosen.

However, since there are many sentences without any implicit aspect, a threshold is added, such that the highest scoring aspect must exceed the threshold in order to be chosen. If the computed score does not exceed the threshold, the considered implicit aspect is not assigned to that sentence. The pseudocode for the whole process is shown in Alg. 4.1, where the training process is shown (i.e., constructing co-occurrence matrix C and lists O and F), and in Alg. 4.2, where the processing of new sentences using the trained algorithm is shown.

Algorithm 4.1: Training the algorithm with annotated data.

```
Initialize list of unique word lemmas with frequencies O
Initialize list of unique implicit aspects F
Initialize co-occurrence matrix C
foreach sentence s ∈ training data do
    foreach word w ∈ s do
        if ¬(w ∈ O) then
            add w to O
        end
        O(w) = O(w) + 1
    end
    foreach implicit aspect f ∈ s do
        if ¬(f ∈ F) then
            add f to F
        end
        foreach word w ∈ s do
            if ¬((w, f) ∈ C) then
                add (w, f) to C
            end
            C(w, f) = C(w, f) + 1
        end
    end
    Determine optimal threshold.
end
```

Algorithm 4.2: Executing the algorithm to process new sentences.

```
foreach sentence s ∈ test data do
    currentBestAspect = empty
    scoreOfCurrentBestAspect = 0
    foreach aspect f ∈ F do
        score = 0; foreach word w ∈ s do
            score = score + C(w, f)/O(w)
        end
        if score > scoreOfCurrentBestAspect then
            currentBestAspect = f
            scoreOfCurrentBestAspect = score
        end
    end
    if scoreOfCurrentBestAspect > threshold then
        Assign currentBestAspect to s as its implicit aspect
    end
end
```

The optimal threshold is computed based on the training data only, and consists of a simple linear search. A range of values is manually defined, all of them which are then tested consequently. The values ranged from 0 to 1, with a step size of 0.001. The best performing threshold is then used when evaluating on the test data. Since there is only one parameter to train and the range of possible values is rather limited, more advanced machine learning techniques were not deemed necessary to arrive at a good threshold value.

A limitation of this method is the fact that it will choose at most one implicit aspect for each sentence. Both of our data sets, as can be seen in the next section, contain sentences that have more than one implicit aspect. In these cases, chances are higher that the chosen implicit aspect is in the golden standard, but all aspects beyond the first will be missed by the algorithm. There are several options to address this issue. For instance, we can predict not just the best performing aspect that exceeds the threshold, but all aspects that exceed the threshold. However, given the fact that aspects are different in terms of their frequencies, this is not likely to work well as is. Instead, we can train an aspect-specific weight to model the different

Algorithm 4.3: Generating exhaustive list of weights to test

```
1  Function GetAllWeightsToTry(nrAspects, weights)
       Input  : nrAspects: the number of unique aspect values
       Input  : weights: the array of weights to try
2      Initialize list of weight lists weightsToTry
3      nrWeights = length(weights)
4      for i = 0; i < nrWeights^nrAspects; i++ do
5          Initialize weights list trial
6          for d = 0; d < nrAspects; d++ do
7              weightIndex = floor(i / nrWeights^d % nrWeights)
8              Add weights[valueIndex] to trial
9          end
10         Add trial to weightsToTry
11     end
       Output : weightsToTry
```

aspect frequencies. This leads to the following augmented score function:

$$score_{f_i} = \frac{w_i}{v} \sum_{j=1}^{v} \frac{c_{i,j}}{o_j}, \tag{4.2}$$

where w_i is the weight for aspect f_i and all other variables are the same as in Equation 4.1.

The weights w_i are trained using a linear grid search. Preliminary experiments have shown that the weight for the `food` aspect will always be the lowest, so the weight for `food` remains fixed at 1.0. This is most likely due to `food` being the dominant aspect in the used data set. The following weights are tested for the remaining aspects: $\{1.25, 1.5, 1.75, 2.0, 2.25, 2.5\}$. Since the number of different aspects might be different for other data sets, the following algorithm creates an exhaustive enumeration of values to try for each aspect. In our case, with 3 aspects to assign weights to and 6 different weights, a total number of $6^3 = 636$ different weight combinations are tested.

Another option to deal with multiple aspects is to train a threshold for each aspect. This is one of the more straightforward extensions, and this is the method

applied in Schouten et al. (2014). The prediction method that incorporates this option is shown in Algorithm 4.4

Algorithm 4.4: Executing the algorithm to process new sentences.

```
foreach sentence s ∈ test data do
    foreach aspect f ∈ F do
        score = 0; foreach word w ∈ s do
            score = score + C(w, f)/O(w)
        end
        if score > threshold_f then
            Assign currentBestAspect to s as its implicit aspect
        end
    end
end
```

The last option for multiple aspect prediction that is investigated is the use of a separate classifier to determine whether a sentence contains more than aspect. This is the method used in Dosoula et al. (2016). When this classifier, a simple logistic regression, predicts that there is at most one imlicit aspect in the sentence, the method will use the original method of assigning the highest scoring implicit aspect. However, when the classifier predicts that there are multiple implicit aspects in this sentence, every aspect that exceeds the trained threshold will be assigned to the sentence.

The classifier calculates a score based on a number of sentence characteristics that are related with the number of implicit features k_s within a sentence s. When the score for a sentence exceeds another trained threshold, the classifier predicts multiple implicit features to be present. The score function uses the following variables: (i) number of nouns ($\texttt{\#NN}_s$), (ii) number of adjectives ($\texttt{\#JJ}_s$), (iii) number of commas ($\texttt{\#Comma}_s$), and (iv) the number of 'and' words ($\texttt{\#And}_s$). In order to determine the relation between these predictor variables and the number of implicit features, we estimate the following logistic regression equation by maximum-likelihood:

$$Score_{k_s} = \log\left(\frac{p_s}{1-p_s}\right) = \beta_0 + \beta_1 \texttt{\#NN}_s + \beta_2 \texttt{\#JJ}_s + \beta_3 \texttt{\#Comma}_s + \beta_4 \texttt{\#And}_s, \quad (4.3)$$

Algorithm 4.5: Algorithm training using annotated data.

```
Construct list F of unique implicit features
Construct list O of unique lemmas with frequencies
Construct co-occurrence matrix C
foreach sentence s ∈ training data do
    foreach word w ∈ s do
        if ¬(w ∈ O) then
            Add w to O
        end
        O(w) = O(w) + 1
    end
    foreach implicit feature f ∈ s do
        if ¬(f ∈ F) then
            Add f to F
        end
        foreach word w ∈ s do
            if ¬((w, f) ∈ C) then
                Add (w, f) to C
            end
            C(w, f) = C(w, f) + 1
        end
    end
end
Train logistic regression classifier
Train threshold for the feature detection algorithm through linear search
```

where p_s is the probability that sentence s contains multiple implicit features. The coefficients are estimated using the full data set.

This extended algorithm is now trained in two steps, only using the training data. First, the threshold for the classifier is trained in terms of prediction performance. Second, the threshold for the feature detection algorithm, now using the prediction of the classifier, is trained to optimize the feature detection performance.

We estimate the $Score_{k_s}$ function (4.3) using logistic regression. This is done using the training data only, and when using cross-validation, a different classifier is trained for each fold using the available training data.

Algorithm 4.6: Algorithm execution on new sentences in the test data.

```
Input: trained threshold threshold and trained logistic regression
  multiClassifier
Construct list NN with the number of nouns per sentence
Construct list JJ with the number of adjectives per sentence
Construct list CM with the number of commas per sentence
Construct list A with the number of 'and' words per sentence
Obtain β̂'s from logistic regression using the full data set
foreach sentence s ∈ test data do
    predictMultiple = multiClassifier.predict(NN_s, JJ_s, CM_s, A_s)
    currentBestFeature = empty
    scoreOfCurrentBestFeature = 0
    foreach feature f ∈ F do
        score = 0
        foreach word w ∈ s do
            score+ = C(w, f)/O(w)
        end
        if predictMultiple then
            if score > threshold then
                Assign feature f to s
            end
        else if score > scoreOfCurrentBestFeature then
            currentBestFeature = f
            scoreOfCurrentBestFeature = score
        end
    end
    if ¬(predictMultiple) then
        if scoreOfCurrentBestFeature > threshold then
            Assign currentBestFeature to s
        end
    end
end
```

4.4 Data Analysis

This section presents an overview of the two data sets that are used to train and evaluate the proposed method and its variants. The first data set is a collection of product reviews Hu and Liu (2004b), where both explicit and implicit features are

labeled. The second data set consists of restaurant reviews Ganu et al. (2009), where explicit aspects are labeled, as well as implicit aspect categories. Each sentence can have zero or more of these coarse-grained aspect categories. The restaurant set features five different aspect categories: 'food', 'service', 'ambience', 'price', and 'anecdotes/miscellaneous'. Since these aspects are implied by the sentence instead of being referred to explicitly, they function as implicit features as well. However, since there are only five options to choose from, it is much easier to obtain good performance on the restaurant set compared to the product set, where there are many different implicit features. Because of this, results for both data sets are not directly comparable. Even so, it is interesting to see how the proposed method performs on different data.

4.4.1 Product Reviews

The collection of product reviews are extracted from amazon.com, covering five different products: Apex AD2600 Progressive-scan DVD player, Canon G3, Creative Labs Nomad Jukebox Zen Xtra 40GB, Nikon Coolpix 4300, and Nokia 6610. Because the primary purpose of this data set is to perform aspect-level sentiment analysis, it is the case that features are only labeled as a feature when an opinion is expressed about that feature in the same sentence. In the example below, both sentences have a feature 'camera', but only in the second sentence is 'camera' labeled as a feature since only in the second sentence it is associated with a sentiment word.

"I took a picture with my phone's *camera*."
"The *camera* on this phone takes *great* pictures."

Because the product data set contains a lot of different, but sometimes similar, features, a manual clustering step has been performed. This makes the set of features more uniform and reduces unnecessary differences between similar features. It also removes some misspellings that were present in the data set. In total, the number of unique implicit features is reduced from 47 to 25.

As can be seen in Fig. 4.1, there are not many sentences with an implicit feature. This only stresses the need for a good selection criterion to distinguish the ones with an implicit feature from the ones that do not have one. There is also a small number

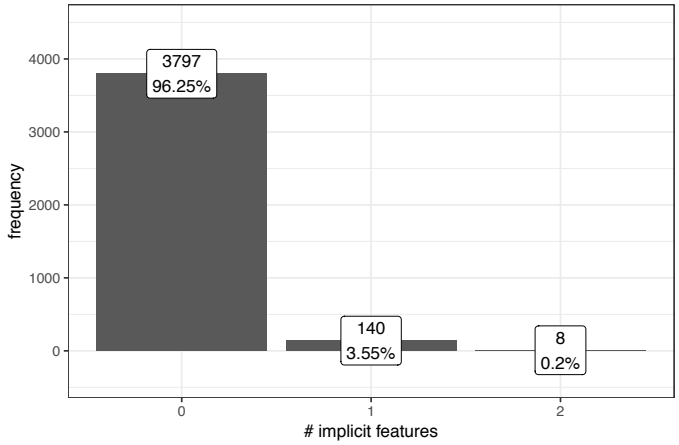

Figure 4.1: Distribution of sentences in the product review data set, according to the number of implicit features they contain.

of sentences (0.2%) that have two implicit features. Since the algorithm will only choose zero or one implicit feature for each sentence, this can potentially impact performance in a negative way. The second implicit feature will always be missed, leading to a lower recall. This is however slightly mitigated by the fact that it is easier to pick a correct feature, as it is checked against both annotated features in the sentence.

In Fig. 4.2, the frequency distribution of the set of implicit features is given. Frequency is measured as the number of sentences a certain implicit feature appears in. As can be seen, there are quite a few implicit features which appear in only a couple of sentences. Fifteen out of the 25 feature appear in less than 5 sentences, with eight features occurring in only one sentence. This makes it extremely difficult to learn a classifier that is able to find these features. In case of the features that appear only once, it is completely impossible to devise a classifier, since they cannot both appear in the test and in the training set.

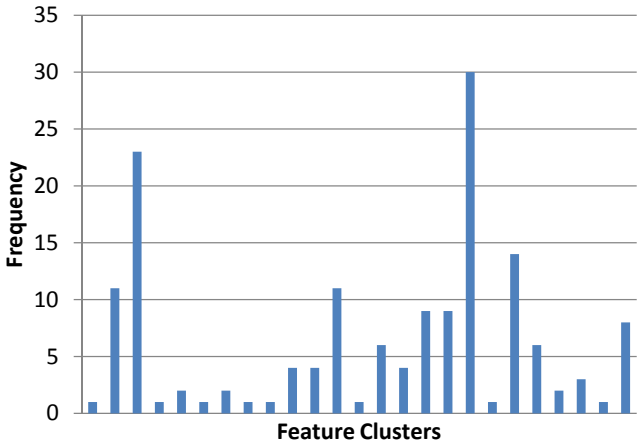

Figure 4.2: Frequencies for all 25 unique feature clusters in the product review data set.

4.4.2 Restaurant Reviews

Compared to the product reviews, the restaurant review data set has clearly different statistical characteristics, as shown in Fig. 4.3. Where the product review set has only a few sentences that contain an implicit feature, in the restaurant set, all of them have an aspect category, which we will regard as an implicit feature in this research. The much bigger size, together with the already mentioned fact that there are only five different implicit features in this data set, makes for a much easier task. To measure the influence of the threshold parameter, the fifth category of 'anecdotes/miscellaneous' is removed from the data set. Since this category does not really describe a concrete implicit feature, removing it leaves us with sentences that do not have any implicit feature, allowing the performance of the threshold to be assessed on this data as well.

Compared to the product reviews data set, the frequency distribution of the implicit features in the restaurant reviews set, shown in Fig. 4.4 is more balanced. Every features has at least a couple of hundred sentences in which it is appearing. The one outlier is the 'food' category, which appears twice as much as the second

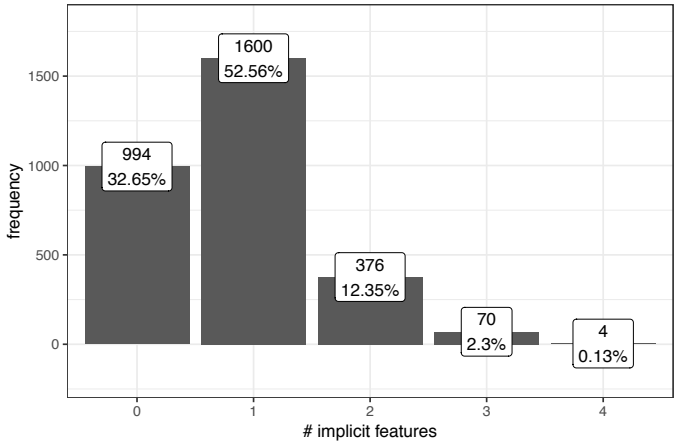

Figure 4.3: Distribution of sentences in the restaurant review data set, according to the number of implicit features they contain.

largest feature which is 'service'. Still, the difference between the feature that appears the most ('food') and the one that appears the least ('price') is only a factor of three, whereas for the product features, this would be much higher (i.e., around 30).

4.5 Evaluation

All evaluations are performed using 10-fold cross-evaluation. Each tenth of the data set is used to evaluate an instance of the algorithm that is trained on the other 90% of the data. Both the co-occurrence frequencies and the threshold parameter are determined based on the training data only. When evaluating the algorithm's output, the following definitions are used:

- *truePositives* are the aspects that have been correctly identified by the algorithm;
- $falsePositives$ are those aspects that have been annotated by the algorithm, that are not present in the golden standard;

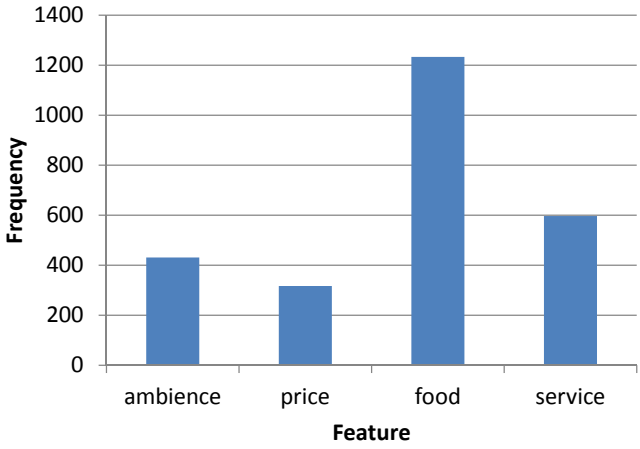

Figure 4.4: Frequencies for all 4 unique features in the restaurant review data set.

- $falseNegatives$ are those aspects that are present in the golden standard, but that have not been annotated by the algorithm;
- $trueNegatives$ are aspects that are not present in the golden standard, and are correctly not annotated by the algorithm.

When evaluating, an aspect always has to be the same one as in the golden aspect to count as a true positive. Simply stating that there is some implicit aspect in a sentence, which might be true, is not enough. In order to count as a true positive, it has to be the right implicit aspect. From this follows that, given a sentence with only one annotated implicit aspect and one golden implicit aspect, when the algorithm correctly identifies that a sentence contains an implicit aspect, but it chooses the wrong one, the wrongly assigned aspect will count as a false positive and the annotated one will count as a false negative. As such, both precision and recall will be lower. In general the algorithm can make three kinds of mistakes:

- State that a sentence contains an implicit aspect, while actually it does not: precision will be lower;

- State that a sentence does not contain an implicit aspect, while actually it does: recall will be lower;
- Correctly stating that a sentence contains an implicit aspect, but picking the wrong one: both precision and recall will be lower.

Because of the ten-fold cross-validation, the reported scores are computed on the sum of the ten confusion matrices (i.e., derived from the ten folds). For example, precision would be computed as:

$$precision = \frac{\sum_{fold=1}^{10} truePositives_{fold}}{\sum_{fold=1}^{10} truePositives_{fold} + falsePositives_{fold}}. \tag{4.4}$$

Recall is computed in a similar way, leaving the F_1-measure, being the harmonic mean of precision and recall, to be computed as usual. In the end, each sentence will be processed exactly once, but will be used nine times as training instance.

The proposed algorithm is tested both with and without the proposed threshold, to assess the benefit of training such a threshold. Furthermore, both versions are evaluated using a Part-of-Speech filter. The latter is used to filter out words in the co-occurrence matrix that may not be useful to find implicit aspects. Besides evaluating using all words (i.e., including stopwords), both algorithms are evaluated using an exhaustive combination of four word groups, namely nouns, verbs, adjectives, and adverbs.

Since the algorithm without a threshold will generally choose some implicit aspect for every sentence, any trained threshold is expected to surpass that score. To provide more insight in this problem, a maximum score is also provided. This maximum score is computed by filtering out all sentences without any implicit aspect and then letting the algorithm simply pick the most appropriate aspect. This situation reflects a perfect threshold that is always able to make the distinction between the presence or absence of an implicit aspect. Obviously, in reality, the trained threshold does not come close to this ideal performance, but including this ideal line allows the separation of errors due to threshold problems from errors due to not picking the right aspect. The latter is an intrinsic problem of the algorithm, not of the threshold. With this in mind, one can see that the gap between the ideal line and the bars represents errors that can be attributed to the threshold, while the gap between 100% performance

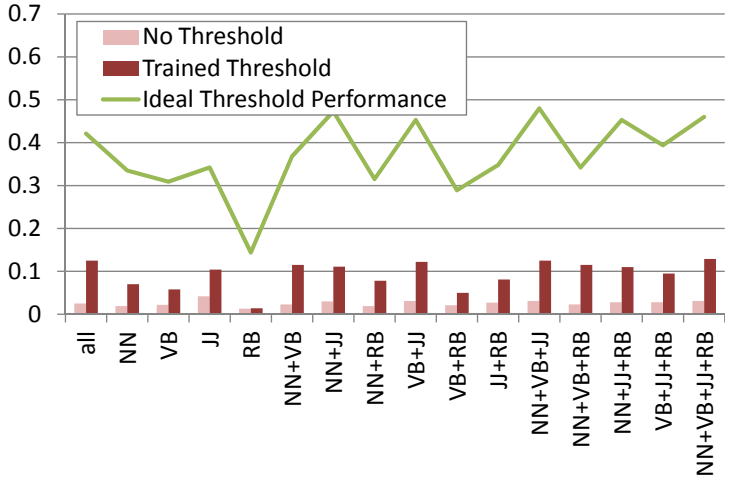

Figure 4.5: The performance on the product review data set in F_1-measure for the various PoS-filters.

and the ideal line represents errors that can be attributed to the method of using co-occurrence frequencies to find the right aspect.

The results on the product review data set are presented in Fig. 4.5, whereas the results on the restaurant review data set are presented in Fig. 4.6. In each graph there are two grouped bars for each Part-of-Speech filter, where the first bar shows the performance without a threshold and the second bar the performance with the trained threshold. The line above the bars represents the ideal, or maximum possible, performance with respect to the threshold, as discussed above. There are 16 different Part-of-Speech filters shown in both graphs. The first `all`, simply means that all words, including stopwords, are used in the co-occurrence matrix. The other fifteen filters only allow words of the types that are mentioned, where NN stands for nouns, VB stands for verbs, JJ stands for adjectives, and RB stands for adverbs.

For the product set, it is beneficial to keep as many words as possible, something that is probably caused by the small size of the data set. However, removing stopwords results in a slightly higher performance: the `NN+VB+JJ+RB` filter scores highest. Looking at the four individual categories, it is clear that adjectives are most important to find implicit aspects. For the restaurant set, the situation is a bit different.

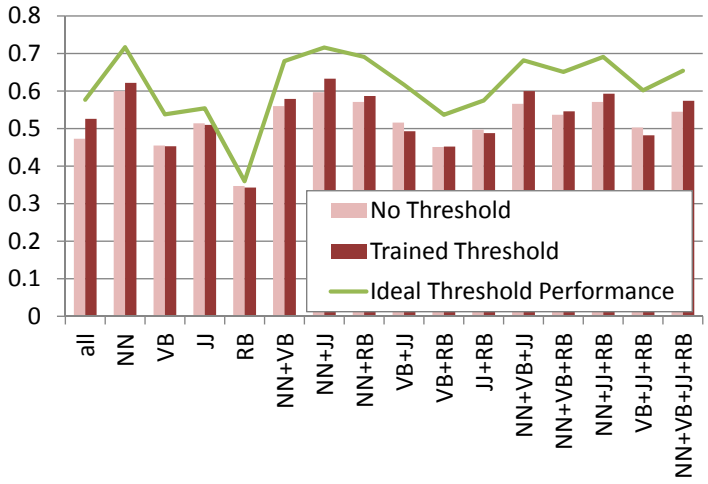

Figure 4.6: The performance on the restaurant review data set in F_1-measure for the various PoS-filters.

Here, nouns are the most important word group, followed by adjectives. Because of its larger size, it is possible to remove verbs and adverbs without any detrimental effects. Hence, the `NN+JJ` filter yields the best performance.

Another observation we can draw from comparing Fig. 4.5 and Fig. 4.6 is that the restaurant set is in general much easier for the algorithm to process. Not only are the ideal performances higher on the restaurant set, also the gap between the ideal and the realized performance is smaller. The most likely reason for this difference is the fact that in the restaurant set there are roughly 2000 sentences that contain at least one of the four possible implicit aspects, whereas in the product set, there are 140 sentences that contain at least one of 25 possible implicit aspects. Not only does this render the task of picking the right aspect more difficult, it also increases the complexity of judging whether a sentence contains one of these aspects.

The fact that the vast majority of the product set has no implicit aspect at all makes the utilization of a threshold all the more important. This is in contrast to the restaurant set, where two-thirds of the sentences have an implicit aspect. Again, this is shown clearly in Fig 4.5 and Fig 4.6: the relative improvement of the threshold is much higher for the product data than the restaurant data.

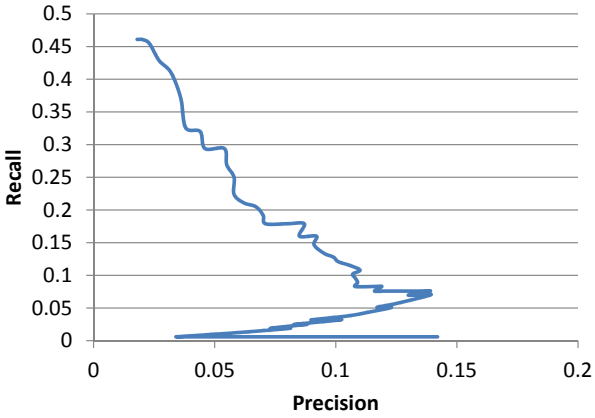

Figure 4.7: The precision-recall trade-off on the product review data set, when manipulating the threshold variable (using the `NN+VB+JJ+RB` filter).

In Fig. 4.7 and Fig. 4.8, the precision-recall trade-off is shown for the best scoring Part-of-Speech filter. The restaurant set yields a well-defined curve, which is to be expected due to the large quantity of available data. Note that as in all other graphs, two tasks are being evaluated: determine whether or not there is an implicit aspect in a sentence, and if so, determine which one it is. This is the reason that, even with a threshold of zero, the recall will not be 100%: while it does state that every sentence will have an implicit aspect, it still has to pick the right one in order to avoid a lower recall (and precision for that matter).

A comparison with the method of Zhang and Zhu (2013) is given in Table 4.1. To increase comparability, both methods are tested with all sixteen possible Part-of-Speech filters (only the best one is reported). To be fair, the original method is also tested with a threshold added, using the same procedure as for the proposed method, even though this contributes little to its performance.

Interestingly, the effect of the threshold is bigger on the product set compared to the restaurant set. This might point to the fact that training this parameter can partly mitigate the negative effect of having a small data set. Consequently, when the data set is larger, the algorithm on its own already performs quite well, leaving less room for improvement by other methods, like adding a threshold.

Since the product review data set does not have many sentences with multiple implicit aspects, the three variants to deal with this issue are tested only on the

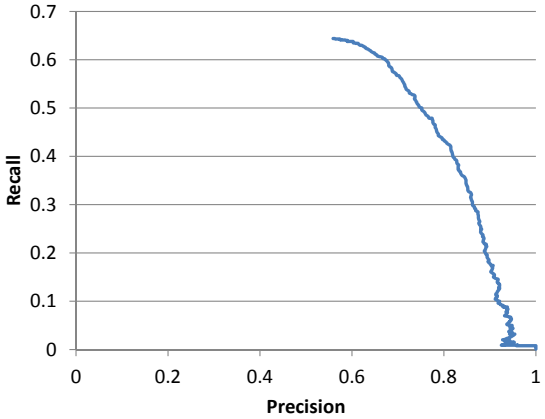

Figure 4.8: The precision-recall trade-off on the restaurant review data set, when manipulating the threshold variable (using the `NN+JJ` filter).

restaurant data set. All variants are tested using 10-fold cross-validation, with average F_1-scores being reported. A two-sided paired t-test is performed over the 10 sub results to determine which methods differ from each other in a statistically significant manner.

Clearly training a threshold for each aspect is the better option, and since each threshold is independent of all other thresholds, it does not become a multidimensional grid search but remains a simple linear search and thus reasonable to train.

4.6 Conclusion

Based on the diagnosed shortcomings in previous work, we proposed a method that directly maps between implicit aspects and words in a sentence. While the method effectively becomes a supervised one, it is not flawed in its assumptions as previous work, and performance is reported to increase on the two used data sets. Furthermore, a more realistic scenario is implemented wherein the proposed method not only has to determine the right implicit aspect, but also whether one is actually present or not.

The proposed algorithm shows a clear improvement with respect to an existing algorithm on the two data sets considered, as it is better in distinguishing between

Table 4.1: Comparison of results with Zhang and Zhu (2013), with and without the proposed threshold. Reported scores are F_1-measures for the best scoring Part-of-Speech filter. Differences between scores are expressed in percentage points (pp.), the arithmetic difference between two percentages.

product review data set

method	no threshold	trained threshold	difference
Zhang and Zhu (2013)	1.2% (`all`)	1.4% (`NN+VB+JJ+RB`)	+0.2 pp.
proposed method	4.2% (`JJ`)	12.9% (`NN+VB+JJ+RB`)	+8.7 pp.
difference	+3 pp.	+11.5 pp.	

restaurant review data set

method	no threshold	trained threshold	difference
Zhang and Zhu (2013)	31.5% (`all`)	32.4% (`all`)	+0.9 pp.
proposed method	59.7% (`NN+JJ`)	63.3% (`NN+JJ`)	+3.6 pp.
difference	+28.2 pp.	31.1 pp.	

sentences that have an implicit aspect and the ones that do not. Both for product reviews and restaurant reviews, the same general improvement is observed when implementing this threshold, even though the actual performance differs much between the two data sets.

Analysis of the performance of the algorithm in relation to the characteristics of the two data sets clearly shows that having less data, but more unique implicit aspects to detect severely decreases performance. While the proposed algorithm is much better in dealing with this lack of data, the results for that particular data set are still too low to be useful in practice. On the set of restaurant reviews, being of adequate size and having only four unique implicit aspects, the proposed algorithm yields promising results. Adding a threshold further boosts the performance by another 3 percentage points, which is highly desirable for this kind of user generated content.

Table 4.2: Comparison of the single-aspect method with the three multi-aspect variants on the restaurant review data set.

			p-values		
	avg. F_1	st.dev.	W	T	C
Single aspect only	0.6243	0.0266	<0.0001	<0.0001	0.0003
Aspect-specific weights (W)	0.6705	0.0195		0.0002	<0.0001
Aspect-specific thresholds (T)	0.7165	0.0175			<0.0001
Multiplicity classifier (C)	0.6335	0.0287			

Using aspect-specific thresholds is shown to be the best option, out of the three investigated possibilities, to enable the prediction of multiple implicit aspects per sentence. This addresses some of the future work specified in Schouten and Frasincar (2014). The weights are slightly less optimal, and using the classifier is performing worse than both.

As future work, the use of stronger machine learning algorithms might yield higher performance, although usually at the cost of explainability. Last, a move from word based methods, like this one, toward concept-based methods, as advocated in Cambria et al. (2013), would be interesting as well. For example, cases like:

> "This phone doesn't fit in my pocket."

are very hard to process based on words alone. It is probably feasible to determine that the implicit aspect here is 'size', if enough training data is at hand, but determining that this sentence represents a negative sentiment, since mobile phones are *supposed* to fit in ones pocket, seems extremely hard for word-based methods. While concept level methods are still in their infancy, they might be up to this challenge, since common sense knowledge, world knowledge, and domain knowledge are integrated in such an approach.

Chapter 5

More Ways of Using Co-occurrence Data to Find Aspects*

USING *on-line consumer reviews as Electronic Word of Mouth to assist purchase-decision making has become increasingly popular. The Web provides an extensive source of consumer reviews, but one can hardly read all reviews to obtain a fair evaluation of a product or service. A text processing framework that can summarize reviews, would therefore be desirable. A subtask to be performed by such a framework would be to find the general aspect categories addressed in review sentences, for which this work presents two methods. In contrast to most existing approaches, the first method presented is an unsupervised method that applies association rule mining on co-occurrence frequency data obtained from a corpus to find these aspect categories. While not on par with state-of-the-art supervised methods, the proposed unsupervised method performs better than several simple baselines, a similar but supervised method, and a supervised baseline, with an F_1-score of 67%. The second method is a supervised variant that outperforms existing methods with an F_1-score of 84%.*

*This chapter is published as "K. Schouten, O. van der Weijde, F. Frasincar, and R. Dekker. Supervised and Unsupervised Aspect Category Detection for Sentiment Analysis with Co-occurrence Data. *IEEE Transactions on Cybernetics*, volume 48, number 4, pages 1263-1275. IEEE, 2018."

5.1 Introduction

Word of Mouth (WoM) has always been influential on consumer decision-making. Family and friend are usually asked for advice and recommendations before any important purchase-decisions are made. These recommendations can both have short as well as long term influence on consumer decision-making (Bone, 1995).

With the Web, word of mouth has greatly expanded. Anyone who wishes to share their experiences, can now do so electronically. Social media, like Twitter and Facebook allow for easy ways to exchange statements about products, services, and brands. The term for this expanded form of word of mouth is Electronic Word of Mouth (EWoM).

Over the last few years, EWoM has become increasingly popular (Feldman, 2013). One of the most important forms of EWoM communication are product and service reviews (Sen and Lerman, 2007) posted on the Web by consumers. Retail Companies such as Amazon and Bol have numerous reviews of the products they sell, which provide a wealth of information, and sites like Yelp offer detailed consumer reviews of local restaurants, hotels and other businesses. Research has shown these reviews are considered more valuable for consumers than market-generated information and editorial recommendations (Bickart and Schindler, 2001; Smith et al., 2005; Trusov et al., 2009), and are increasingly used in purchase decision-making (Adjei et al., 2009).

The information that can be obtained from product and service reviews is not only beneficial to consumers, but also to companies. Knowing what has been posted on the Web can help companies improve their products or services (Pang and Lee, 2008).

However, to effectively handle the large amount of information available in these reviews, a framework for the automated summarization of reviews is desirable (Liu et al., 2012). An important task for such a framework would be to recognize the topics (i.e., characteristics of the product or service) people write about. These topics can be fine-grained, in the case of aspect-level sentiment analysis, or more generic in the case of aspect categories. For example, in the following sentence, taken from a restaurant review set (Pontiki et al., 2014), the fine-grained aspects are ‘fish’, ‘rice’, and ‘seaweed’ whereas the aspect category is ‘food’.

> "My goodness, everything from the fish to the rice to the seaweed was absolutely amazing."

As one can see, aspect categories are usually implied, that is, the names of the categories are not explicitly mentioned in the sentence. The same holds for fine-grained aspects: while most of them are referred to explicitly in a sentence, some are only implied by a sentence. For example, in the sentence below, the implied fine-grained aspect is 'staff', whereas the implied aspect category is 'service'.

> "They did not listen properly and served me the wrong dish!"

When the aspect categories are known beforehand, and enough training data is available, a supervised machine learning approach to aspect category detection is feasible, yielding a high performance (Kiritchenko et al., 2014). Many approaches to find aspect categories are supervised (Brychcín et al., 2014; C. Brun, 2014; Castellucci et al., 2014; Kiritchenko et al., 2014). However, sometimes the flexibility inherent to an unsupervised method is desirable.

The task addressed in this chapter stems from a subtask of the SemEval 2014 Challenge (Pontiki et al., 2014), which purpose is to identify aspect categories discussed in sentences, given a set of aspect categories. The sentences come from customer reviews and should be classified into one or more aspect categories based on its overall meaning. For example, given the set of aspect categories (*food*, *service*, *price*, *ambience*, *anecdotes/miscellaneous*), two annotated sentences are:

> "The food was great." → (*food*)

> "It is very overpriced and not very tasty." → (*price, food*)

As shown in the above examples, aspect categories do not necessarily occur as explicit terms in sentences. While in the first sentence 'food', is mentioned explicitly, in the second sentence it is done implicitly. In our experiments all sentences are assumed to have at least one aspect category present. Because it may not always be clear which category applies to a sentence, due to incomplete domain coverage of the categories and the wide variation of aspects a reviewer can use, a 'default' category is used. An example of a sentence where a 'default' category is used, is presented

below. Here, the second part of the sentence ("but everything else ... is the pits.") is too general to classify it as one of the other categories (i.e., food, service, price, ambience).

> "The food is outstanding, but everything else about this restaurant is the pits." → (*food*, *anecdotes/miscellaneous*)

In this work, both an unsupervised and a supervised method are proposed that are able to find aspect categories based on co-occurrence frequencies. The unsupervised method uses spreading activation on a graph built from word co-occurrence frequencies in order to detect aspect categories. In addition, no assumption has to be made that the implicit aspects are always referred to explicitly, like it is done in Hai et al. (2011). The proposed unsupervised method uses more than just the literal category label by creating a set of explicit lexical representations for each category. The only required information is the set of aspect categories that is used in the data set. The supervised method on the other hand uses the co-occurrences between words, as well as grammatical relation triples, and the annotated aspect categories to calculate conditional probabilities from which detection rules are mined.

This chapter is structured as follows. First, in Section 5.2, an overview of the related work that inspired this research is presented, then Section 5.3 gives the details of the proposed unsupervised method, while Section 5.4 discusses the details of the supervised method. Section 5.5 contains the evaluation of both methods, comparing them to several baselines and to two state-of-the-art methods. Last, in Section 5.6, conclusions are drawn and some pointers for future work are given.

5.2 Related Work

Since most aspect categories are left implicit in text*, methods for detecting implicit fine-grained aspects might be used for aspect categories as well. As such, some works on implicit aspect detection that inspired this research are discussed below.

An early work on implicit aspect detection is Su et al. (2006). The authors propose to use semantic association analysis based on Point-wise Mutual Information (PMI)

*In the restaurant data set Pontiki et al. (2014) that is used for evaluation, around 77% of the aspect categories was not literally mentioned in sentences.

to differentiate implicit aspects from single notional words. Unfortunately, there were no quantitative experimental results reported in their work, but intuitively the use of statistical semantic association analysis should allow for certain opinion words such as 'large', to estimate the associated aspect ('size').

In Su et al. (2008) an approach is suggested that simultaneously and iteratively clusters product aspects and opinion words. Aspects/opinion words with high similarity are clustered together, and aspects/opinion words from different clusters are dissimilar. The similarity between two aspects/opinion words is measured by fusing both the homogeneous similarity between the aspects/opinion words (content information), calculated by traditional approach, and the similarity by their respective heterogeneous relationships they have with the opinion words/aspects (link information). Based on the product aspect categories and opinion word groups, a sentiment association set between the two groups is then constructed by identifying the strongest n sentiment links. This approach, however, only considered adjectives as opinion words which are not able to cover every opinion, yet the approach was capable of finding hidden links between product aspects and adjectives. Unfortunately, there were no quantitative experimental results reported, specifically for implicit aspect identification.

A two-phase co-occurrence association rule mining approach to identify implicit aspects is proposed by Hai et al. (2011). In the first phase of rule generation, association rules are mined of the from [*opinion word* $\rightarrow$ *explicit aspect*], from a co-occurrence matrix. Each entry in the co-occurrence matrix represents the frequency degree of a certain opinion-word co-occurring with a certain explicitly mentioned aspect. In the second phase, the rule consequents (i.e., the explicit aspects) are clustered to generate more robust rules for each opinion word. Implicit aspects can then be found by identifying the best cluster for a given sentiment word with no explicit aspect, and assigning the most representative word of that cluster as the implicitly mentioned aspect. This method is reported to yield an F_1-score of 74% on a Chinese mobile phone review data set. However, this frequency-based method requires a very good coverage of opinion words with explicit aspects. It assumes that explicit feature annotations are given and that an implicit feature has to relate to an explicit feature. In our work we do not use these assumptions, providing for more generality of the proposed solution.

In Zheng et al. (2014) a semi-unsupervised method is proposed that can simultaneously extract both sentiment words and product/service aspects from review sentences. The method first extracts Appraisal Expression Patterns (AEPs), which are representations of how people express opinions regarding products or services. The set of AEPs is obtained by selecting frequently occurring Shortest Dependency Paths between two words in a dependency graph. Next the authors propose an AEP Latent Dirichlet Allocation model for mining the aspect and sentiment words. The model does however assume that all words in a sentence are drawn from one topic. This method is reported to yield at best an F_1-score of 78% on a Restaurant review data set.

Association rule mining is also employed in Wang et al. (2013), where first the candidate aspect indicators are extracted based on word segmentation, part-of-speech (POS) tagging, and aspect clustering. After that, the co-occurrence degree between these candidate aspect indicators and aspect words are calculated, using five collocation extraction algorithms. These five algorithms use frequency, PMI, frequency*PMI, t test, and χ^2 test, respectively, out of which frequency*PMI is the most promising. Rules are then mined of the form [*aspect indicator* $\rightarrow$ *aspect word*], and only the best rules from the five different rule sets are chosen as the basic rules. The basic set of rules is then extended by mining additional rules from the lower co-occurrence aspect indicators and non-indicator words. The authors propose three methods for doing so: adding dependency rules, adding substring rules, and adding constrained topic model rules. This method is reported to yield at best an F_1-score of 76% on a Chinese mobile phone review data set.

Association rule mining is also the main technique in Zhang and Zhu (2013). Unlike Hai et al. (2011) and Wang et al. (2013), no annotated explicit aspects are required, instead the double propagation algorithm from Qiu et al. (2011) is employed to identify the explicit aspects. An advantage of this double propagation method is that it links explicit aspects to opinion words. This is used later, to restrict the set of possible implicit aspects in a sentence to just those that are linked to the opinion words present in that sentence. The notional words are then used to further investigate which of these aspect is most likely the implicit aspect mentioned in this sentence. Their method yielded an F_1-score of 80% on a Chinese mobile phone review

data set, and apart from a small seed set of opinion words, it operates completely unsupervised.

The SemEval 2014 competition has given rise to a number of proposed methods to find aspect categories. The first to mention, because of its similarity with the currently proposed approaches, in that it is also co-occurrence based, is Schouten and Frasincar (2014), where co-occurrence frequencies are recorded between annotated aspect categories and notional words. This enables the direct association of words with categories. However, this does come at the cost of making the method supervised. Furthermore, its reported performance is one of the lowest in the SemEval-2014 rankings.

A high performing supervised method for category detection is presented in Brychcín et al. (2014). The authors use a set of binary Maximum Entropy classifier with bag-of-words and TF-IDF features for each aspect category. With a reported F_1-score of 81% this method was one of the best submitted constrained methods (i.e., no additional training resources were used apart from the official training data).

Another high performing supervised aspect category detection is proposed in Kiritchenko et al. (2014). Instead of a MaxEnt classifier, five binary (one-vs-all) SVMs are employed, one for each aspect category. The SVMs use various types of n-grams (e.g., stemmed, character, etc.) and information from a word clustering and a lexicon, both learned from YELP data. The lexicon directly associates aspects with categories. Sentences with no assigned category went through the post-processing step, where the sentence was labeled with the category with maximum posterior probability. The lexicon learned from YELP data significantly improved the F_1-score, which was reported to be 88.6% and ranked first among 21 submissions in SemEval-2014 workshop. However, for fair comparison, the score obtained without using the lexical resources derived from the YELP data, which is an F_1-score of 82.2%, is reported in the evaluation, as our proposed supervised method to which it is compared also does not use external knowledge.

As far as it goes for unsupervised approaches in the SemEval 2014 competition, the one presented in Garcia-Pablos et al. (2014) performs best, which reported an F_1-score of 60.0%. Garcia-Pablos et al. (2014) proposes a basic approach that first detects aspects (another subtask of the SemEval competition), which would then be compared with the category words using the similarity measure described by Wu and

Palmer (1994). The category with the highest similarity measure is then selected, if it surpasses a manually set threshold.

5.3 Unsupervised Method

The proposed unsupervised method (called the Spreading Activation Method) uses co-occurrence association rule mining in a similar way as Hai et al. (2011), by learning relevant rules between notional words, defined as the words in the sentence after removing stop words and low frequency words, and the considered categories. This enables the algorithm to imply a category based on the words in a sentence. To avoid having to use the ground truth annotations for this and to keep this method unsupervised, we introduce for each category a set of seed words, consisting of words or terms that describe that category. These words or terms are found by taking the lexicalization of the category, and its synonyms from a semantic lexicon like WordNet. For example, the *ambience* category has the seed set {*ambience, ambiance, atmosphere*}.

With the seed words known, the general idea of implicit aspect detection can be exploited to detect categories as well. The idea is to mine association rules of the form [*notional word* $\rightarrow$ *category*] from a co-occurrence matrix. Each entry in this co-occurrence matrix represents the frequency degree of two notional words co-occurring in the same sentence. Stop words, like '*the*' and '*and*', as well as less frequent words are omitted because they add little value for determining the categories in review sentences.

The reason why we choose to mine for rules similar to that of Hai et al. (2011)'s, and do not consider all notional words in the sentence at once to determine the implied categories, like Zhang and Zhu (2013), is based on the hypothesis that categories are better captured by single words. If we have for example categories like '*food*' and '*service*' all it takes to categorize sentences is to find single words like '*chicken*', '*staff*', or '*helpful*'.

Association rules are mined when a strong relation between a notional word and one of the aspect categories exists, with the strength of the relation being modeled using the co-occurrence frequency between category and notional word. We distin-

guish between two different relation types: *direct* and *indirect* relations. A direct relation between two words A and B is modeled as the positive conditional probability $P(B|A)$ that word B is present in a sentence given the fact that word A is present. An indirect relation between two words A and B exists when both A and B have a direct relation with a third word C. This indicates that A and B could be substitutes for each other, even though their semantics might not be the same. Without checking for indirect relations, substitutes are usually not found since they do not co-occur often together. A visual example of an indirect relation can be found in Figure 5.1.

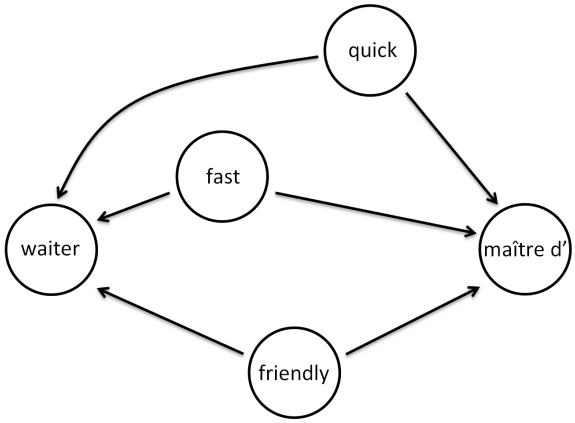

Figure 5.1: Example of an indirect relation: '*waiter*' and '*maître d'*' are indirectly related by having the same set of directly related notional words.

To exploit the direct, as well as the indirect relation information between notional words and seed words, the spreading activation algorithm (Crestani, 1997) is utilized, which is a method to search for associative networks. Spreading activation has been successfully applied in various fields, e.g., (Bagchi et al., 2000; Katifori et al., 2010), etc. For that, a network data structure is needed, consisting of vertices connected by links, as depicted in Figure 5.1. The vertices are labeled and the links may receive direction and/or weights to model the relations between vertices. The search process of finding an associative network is initiated by giving each vertex an activation

value. These initial values determine the area of the search as the activation values are iteratively spread out to other, linked, vertices.

In our case we want to use spreading activation to find, for each category, a network of words associated with the category's set of seed words. To do this, a network data structure is created, having vertices for all notional words and edges to model the direct relations between these words. In the network data structure all notional words receive an initial activation value of zero except for the category's seed words, which receive positive activation values. In the first iterative step of the spreading activation algorithm, these positive activation values are spread out to other words directly related to the seed words, based on the strength of the direct relation. In this way, words that have strong direct relations with the seed words receive high association values. The following iterative steps will be looking for words with high association values that are then activated and will spread out their activation value to other words directly related to them. In this way, notional words that are indirectly related to one of the seed words are also identified. The end result will be a network of notional words, each with their own activation value, the higher the activation value, the more related the notional word will be to the category.

The data network structure used for the spreading activation algorithm will consist of vertices that represent the notional words, and links between two vertices representing a strictly positive co-occurrence frequency. Each link represents the direct relation between two notional words and receives weight equal to the conditional probability that word A co-occurs with word B, given that B appears in a sentence. This also means that the links receive direction as the conditional probability is not symmetric, making the data network structure a co-occurrence digraph.

Once each category has its own associative network, rules can be mined of the form [*notional word* $\rightarrow$ *category*] from vertices in these networks, based on the activation value of the vertex. Since the same word can be present in multiple associative networks, one word might trigger multiple aspect categories. Based on the words in the sentence, a set of rules is triggered and their associated aspect categories are assigned to the sentence. Figure 5.2 illustrates how the unsupervised method works on a simple example corpus, with a decay factor of 0.9 and firing threshold of 0.4. The example shows how an associative network for the category 'food' is found and rules are extracted.

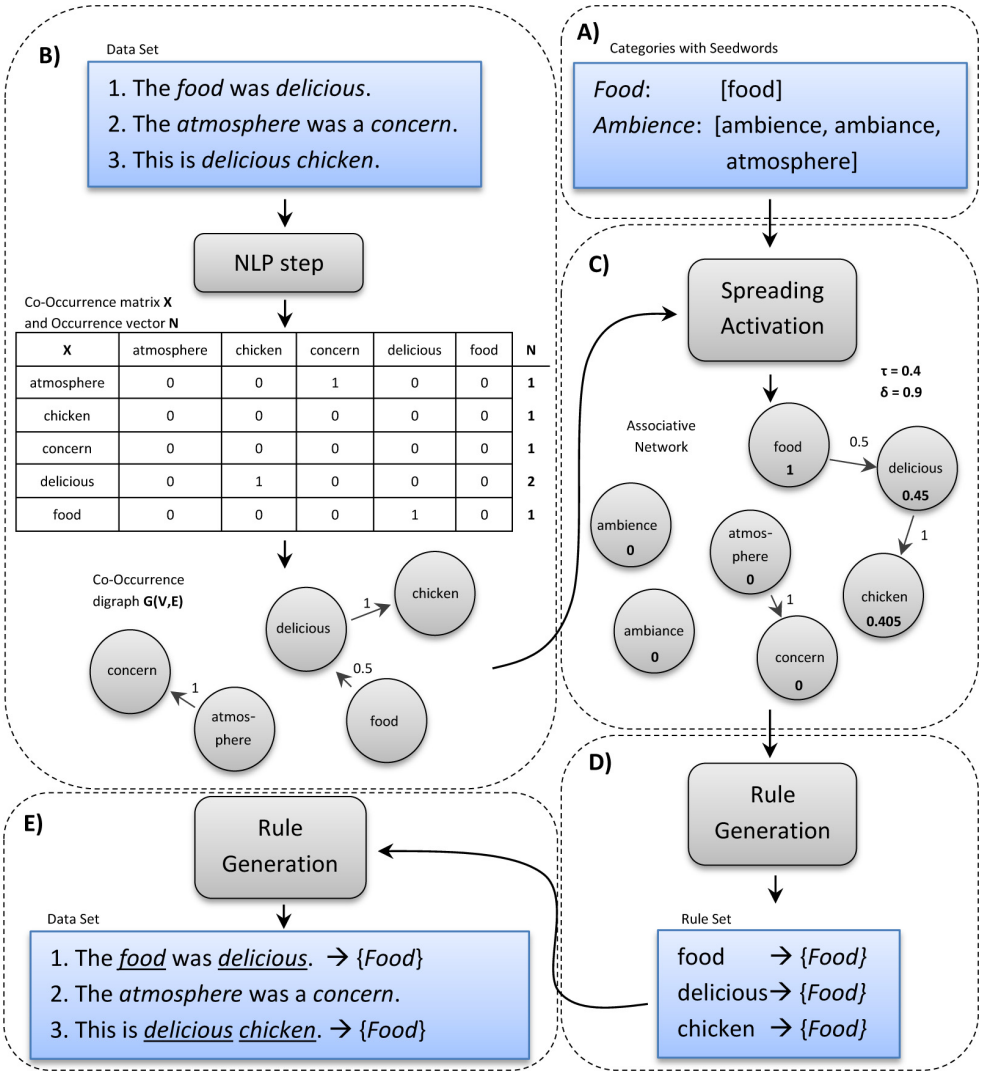

Figure 5.2: Example flowchart of the unsupervised method.

5.3.1 Algorithm

The method can best be described according to the following steps:

Identifying category seed word sets S_c

First, we identify for each of the given categories $c \in C$ a set of seed words S_c containing the category word and any synonyms of that word. This first step is represented by step A in Figure 5.2.

Determine co-occurrence digraph $G(V, E)$

Next, as a natural language preprocessing step, both training and test data are run through the lemmatizer of the Stanford CoreNLP (Manning et al., 2014). We keep track of all lemmas in the text corpus and count their occurrence frequencies. Stop words and lemmas that have an occurrence frequency lower than a small degree α are discarded, while the rest of the lemmas and corresponding frequencies are stored in the occurrence vector N. The parameter α is used to filter out low occurring lemmas. Each lemma in N is now considered to be a notional word. A co-occurrence matrix X is then constructed where each entry represents how often notional word from N_i appeared before N_j the same sentence.

From X and N the co-occurrence digraph $G(V, E)$ is constructed with nodes V and edges E. Each notional word $i \in N$ receives its own node $i \in V$. A directed edge $(i, j) \in E$ between nodes i and j exists if and only if the co-occurrence frequency $X_{i,j}$ is strictly positive. The weight of each edge $(i, j) \in E$ is denoted by $W_{i,j}$ and represents the conditional probability that notional word i co-occurs with notional word j in a sentence after it, given that j is present in that sentence. This formula is shown in Equation 5.1:

$$W_{i,j} = \frac{X_{i,j}}{N_j}, \tag{5.1}$$

where $X_{i,j}$ is the co-occurrence frequency of words i and j (word i after word j) and N_j is the frequency of word j. Step B in Figure 5.2 illustrates this step.

Apply spreading activation

Once the co-occurrence digraph $G(V, E)$ is obtained, we apply for each category $c \in C$ the spreading activation algorithm to obtain for each vertex $i \in V$ an activation value $A_{c,i}$. Each activation value has a range of $[0, 1]$, and the closer it is to 1 the stronger the notional word is associated with the considered category.

The process of obtaining these activation values for category $c \in C$ is initiated by giving all vertices $i \in V$ an activation value $A_{c,i}$. Vertices that are labeled as one of the category's seed words $s \in S_c$ receive the maximum activation value of 1, while the rest of the vertices receive the minimum activation value of 0.

After this initialization step, the iterative process of spreading the activation values starts. The actual spreading of activation values is done by 'firing' or 'activating' vertices. A vertex that is fired, spreads its activation value to all vertices directly linked to the fired vertex. The activation value added to the linked words depends on the activation value of the fired vertex and the weight of the link between the fired vertex and the vertex receiving the added activation value. The formula for the new activation value for one of the vertices j linked to the fired vertex i is shown in Equation 5.2.

$$A_{c,j} = \min\{A_{c,j} + A_{c,i} \cdot W_{i,j} \cdot \delta, 1\} \tag{5.2}$$

The parameter δ in Equation 5.2 models the decay of the activation value as it travels further through the network, ranging from 0 to 1. The closer this decay factor gets to 0 the more the firing activation value will have decayed (i.e., it will be closer to 0). Furthermore, any activation value $A_{c,j}$ can have a maximum value 1. Firing vertices is only allowed if its activation value reaches a certain firing threshold τ_c, depending on the category $c \in C$. Once a vertex has been fired it may not fire again. The sets M and F keep track of which vertex may be fired and which vertex has already been fired, respectively.

A single step in the iterative process of spreading the activation values starts by searching for vertices $i \notin F$ with activation value $A_{c,i}$ greater than firing threshold τ_c. These vertices are temporarily stored in M. Then for vertex $i \in M$ we look for vertex j linked to this vertex with edge $(i, j) \in E$, and modify its activation value $A_{c,j}$ according to Equation 5.2. This is done for each vertex $j \in V$ linked to vertex

Algorithm 5.1: Spreading activation algorithm

input : category c, vertices V, seed vertices S_c, weight matrix W, decay factor δ, firing threshold τ_c
output : activation values $A_{c,i}$ for category c

foreach $s \in S_c$ **do**
  $A_{c,s} \leftarrow 1$
end
foreach $\in V \setminus S_c$ **do**
  $A_{c,i} \leftarrow 0$
end
$F \leftarrow S_c$
$M \leftarrow S_c$
while $M \neq \emptyset$ **do**
  foreach $i \in M$ **do**
    foreach $j \in V$ **do**
      $A_{c,j} \leftarrow \min\{A_{c,j} + A_{c,i} \cdot W_{i,j} \cdot \delta, 1\}$
    end
  end
  $M \leftarrow \emptyset$
  foreach $i \in V \setminus F$ **do**
    if $A_{c,i} > \tau_c$ **then**
      add i to F
      add i to M
    end
  end
end

i with edge $(i, j) \in E$, after which vertex i is removed from M and stored in F, the same process is then executed for the remaining vertices $i \in M$. This concludes one iterative step, that is repeated until no more vertices $i \notin F$ with activation value $A_{c,i}$ greater than firing threshold τ_c exists. The pseudocode for the spreading activation algorithm can be found in Algorithm 5.1, and an illustration of this complete step can be found in Step C of Figure 5.2.

Rule mining

Once spreading activation is applied to all categories $c \in C$, matrix $A_{c,i}$ is obtained, containing, for each notional word $i \in N$, activation values for each category $c \in C$. From these associations values, rules are mined, based on the magnitude of these values. Vertices that have fired are seen as part of the associative network and from each vertex in that network, a rule is mined. Any vertex whose activation value $A_{c,i}$ is higher than parameter τ_c produces a rule [*notional word* $i \rightarrow$ *category* c] that is stored in rule set R. All notional words are allowed to imply multiple categories except for seed words, which can only imply the category they belong to. This step is depicted as Step D of Figure 5.2.

Assign aspect categories

In the last step we predict categories for each unprocessed sentence, using the rule set R obtained from the previous step. For each unprocessed sentence we use lemmatisation, and look if any word matches a rule, after which that rule is applied. Since multiple rules can be fired, it is possible to predict multiple aspect categories per sentence. This last step corresponds to Step E in Figure 5.2.

5.3.2 Parameter Setting

Three parameters, α, δ, and τ_c need to be set manually. For α, the minimal occurrence threshold, a value of $0.005 \times$ number of sentences in the dataset is used. In this way, low-frequency words are excluded from the co-occurrence matrix. The decay factor δ is set at 0.9 to increase the number of indicators (recall).

The τ_c parameter is set differently for each category c. With parameters α and δ fixed, the algorithm is run for each category using a range of values for τ_c. For each τ_c, the method constructs an association network, counting the number of notional words in it. The decision for the best value for τ_c can be made based on a plot of the activated word count relative to the total number of words in the network. The plots for categories 'service' and 'food' (cf. Section 5.5 for a description of the used data set) are shown in Figure 5.3 and 5.4, respectively.

Figure 5.3 shows that having high value for τ_c results in only seed words indicating the presence of a category (i.e., these are the explicitly mentioned categories). This is shown by the long flat tail to the right. On the other hand, having $\tau_c = 0$ results in all words being indicators, producing much noise. To find the optimal, or at least a good, value for τ_c, we use the breakpoint heuristic, where we find the breakpoint in the graph for relative word count, having the flat part of the graph to the right and the sloped part of the graph on the left. This is shown as the dashed vertical line. For most categories this results in a near-optimal choice for τ_c.

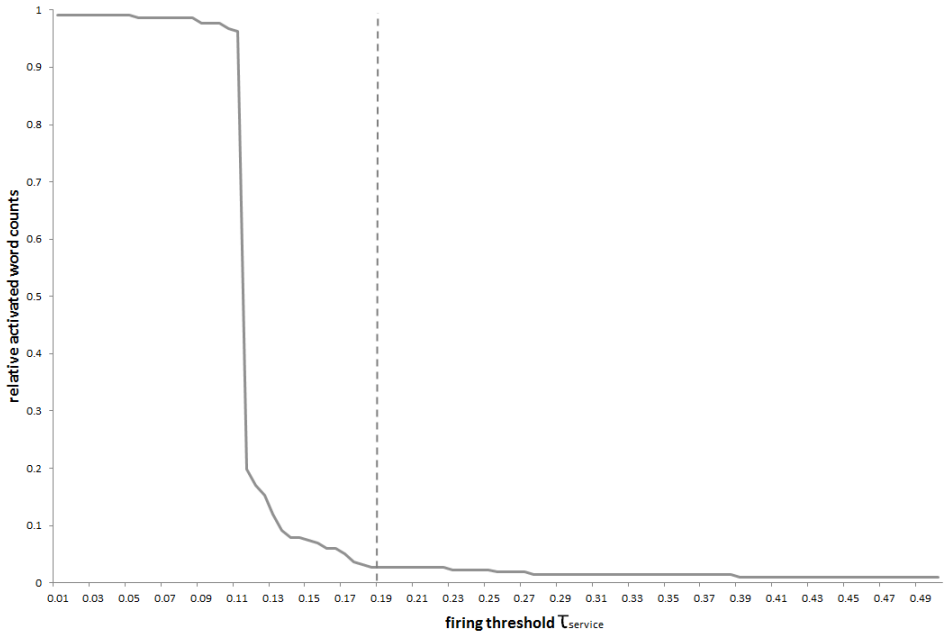

Figure 5.3: Graph displaying the relative activated word counts for different values of firing threshold $\tau_{service}$ together with the threshold chosen by the heuristic.

One exception is the '*food*' category, as shown in Figure 5.4. Here, we choose to have more words as indicators, because '*food*' is by far the largest of the aspect categories we aim to detect. Hence, it is reasonable to have a larger associative network, with more words pointing to the '*food*' category. Given the fact that many different words, such as all kinds of meals and ingredients point to 'food', it is rather

intuitive to have a bigger associate network for this category. Hence, when dealing with a dominant category like '*food*', the τ_c should be lower than the one given by the heuristic, for example by setting it similar to Figure 5.4.

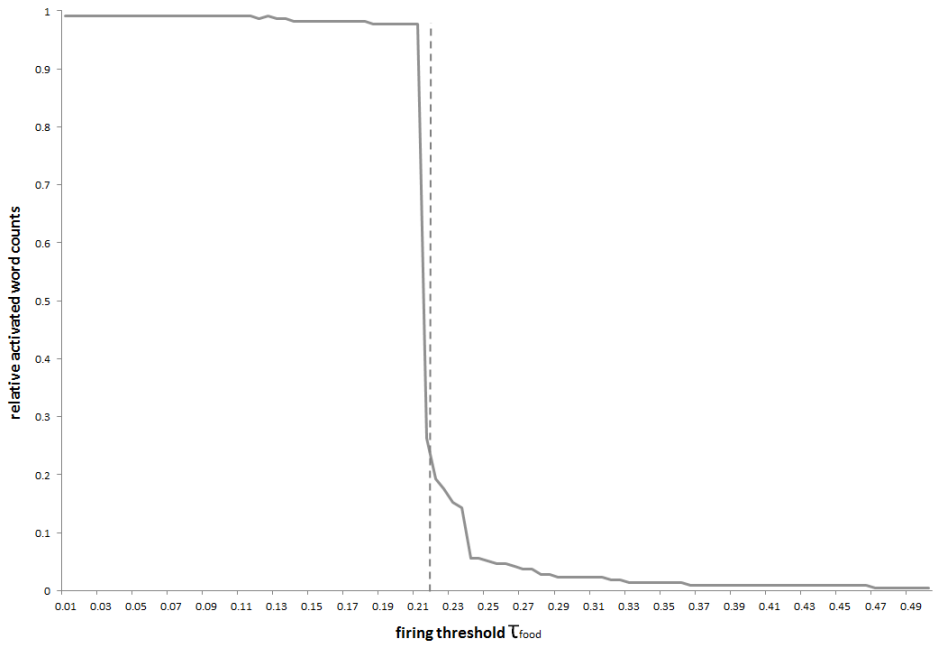

Figure 5.4: Graph displaying the relative activated word counts for different values of firing threshold τ_{food} together with the threshold chosen by the heuristic.

5.3.3 Limitations

A practical limitation of this unsupervised method is that it requires tuning for multiple parameters. Although one can implement a training regime to learn these parameters, this would render the method supervised, removing one of its key advantages. Another shortcoming, albeit a minor one, is the requirement of determining a seed set up front for each aspect category one wants to find. Using the lexical representation of the category complemented by some synonyms is an easy way of retrieving a suit-

able seed set words, but abstract or vague categories like 'anecdotes/miscellaneous' cannot be dealt with effectively in this way.

5.4 Supervised Method

Similar to the first method, the supervised method (called the Probabilistic Activation Method) employs co-occurrence association rule mining to detect categories. We borrow the idea from Schouten and Frasincar (2014) to count co-occurrence frequencies between lemmas and the annotated categories of a sentence. However, low frequency words are not taken into account in order to prevent overfitting. This is achieved using a parameter α_L, similar to the unsupervised method. Furthermore, stop words are also removed.

In addition to counting the co-occurrences of lemmas and aspect categories, the co-occurrences between grammatical dependencies and aspect categories are also counted. Similar to lemmas, low frequency dependencies are not taken into account to prevent overfitting, using the parameter α_D. Dependencies, describing the grammatical relations between words in a sentence, are more specific than lemmas, as each dependency has three components: governor word, dependent word, and relation type. The added information provided by dependencies, may provide more accurate predictions, when it comes to category detection. Knowing whether a lemma is used in a subject relation or as a modifier can make the difference between predicting and not predicting a category. To illustrate the value of dependencies, a small example is provided in Figure 5.5 below.
Assuming that the category 'food' exists, and that its category word is a good indicator word for this category, most of the time, the word 'food' will actually indicate the category 'food', as in the first sentence. However there are also sentences where the word 'food' does not indicate the category 'food', as shown in the second sentence. By using the word 'food' as indicator for the category 'food', both sentences will be annotated with the category 'food', but by looking at dependencies this does not have to be the case. In the first sentence 'food' is used in relation to 'good' as nominal subject, while in the second sentence 'food' is used to modify 'joint'. From

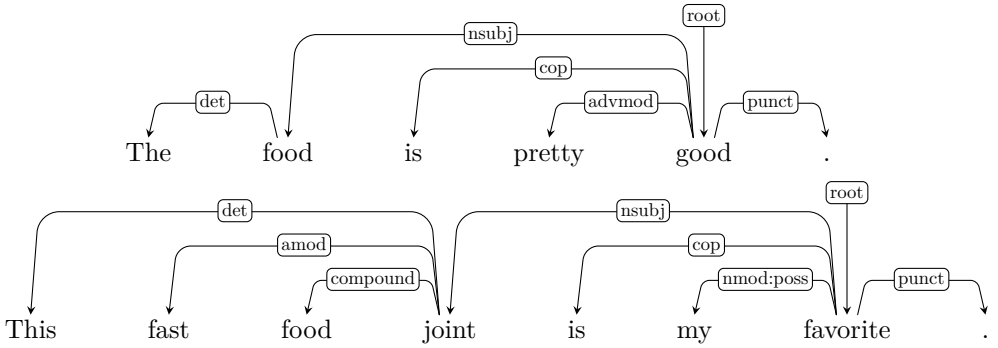

Figure 5.5: Two example sentences with their grammatical dependencies

these dependency relations we might learn that only when the word 'food' is used as a nominal subject, it implies the category 'food'.

The fact that dependencies are more specific than lemmas also has a disadvantage. With dependencies being triples, and hence more diverse than lemmas alone, they tend to have a much lower frequency count than single lemmas. This means that many dependencies would not occur frequently enough to be considered, since low frequency dependencies are omitted to mitigate overfitting. To cope with this problem, two variants of each dependency are added: the first is the pair of governor word and dependency type, and the second is the pair of depending word and dependency type. These pairs convey less information than the complete triples, but are still informative compared to having just lemmas. Since the frequency of these pairs is generally higher than that of the triples, more pairs are expected to pass the frequency filter. Hence, we extract, for each dependency, the following three forms: {dependency relation, governor, dependent} (D_1), {dependency relation, dependent} (D_2), and {dependency relation, governor} (D_3).

All the dependencies relations from the Stanford Parser (Manning et al., 2014) are used to build up the dependency forms, except for the determinant relation. For the previous first sentence, this would mean the following dependency sets: [{advmod, good, pretty}, {cop, good, is}, {nsubj, good, food}] (D_1), [{advmod, pretty}, {cop, is}, {nsubj, food}] (D_2), and [{advmod, good}, {cop, good}, {nsubj, good}] (D_3).

The co-occurrence frequencies provide the information needed to find good indicators (i.e., words or dependencies) for the categories. To determine the strength of an indicator, the conditional probability $P(B|A)$ is computed from the co-occurrence frequency, where category B is implied when lemma or dependency form A is found in a sentence. These conditional probabilities are easily computed by dividing the co-occurrence frequency of (B, A) by the occurrence frequency of A. The higher this probability, the more likely it is that A implies B. If this value exceeds a trained threshold, the lemma or dependency form indicates the presence of the corresponding category.

This threshold that the conditional probability has to pass is different for each category. It also depends on whether a dependency form or lemma is involved, since dependency forms generally have a lower frequency, requiring a lower threshold to be effective. Hence, given that there are three dependency forms and one lemma form, four thresholds need to be trained for each category in the training data. To find theses thresholds a simple linear search is performed, picking the best performing (i.e., on the training data) value from a range of values for each different threshold.

Once the conditional probabilities are computed and the thresholds are known, unseen sentences from the test set are processed. For each unseen sentence we check whether any of the lemmas or dependency forms in that sentence have a conditional probability greater than its corresponding threshold, in which case the corresponding category is assigned to that sentence. Figure 5.6 illustrates how the supervised method works on a very simple test and training set.

5.4.1 Algorithm

The method can best be described according to the following steps:

Determine lemmas/dependencies

As a natural language preprocessing step, both training and test data are run through the part-of-speech tagger, lemmatizer, and dependency parser (de Marneffe and Manning, 2008) of the Stanford CoreNLP (Manning et al., 2014). This results in all sentences having a set of lemmas, denoted by s_L, and three dependency form sets,

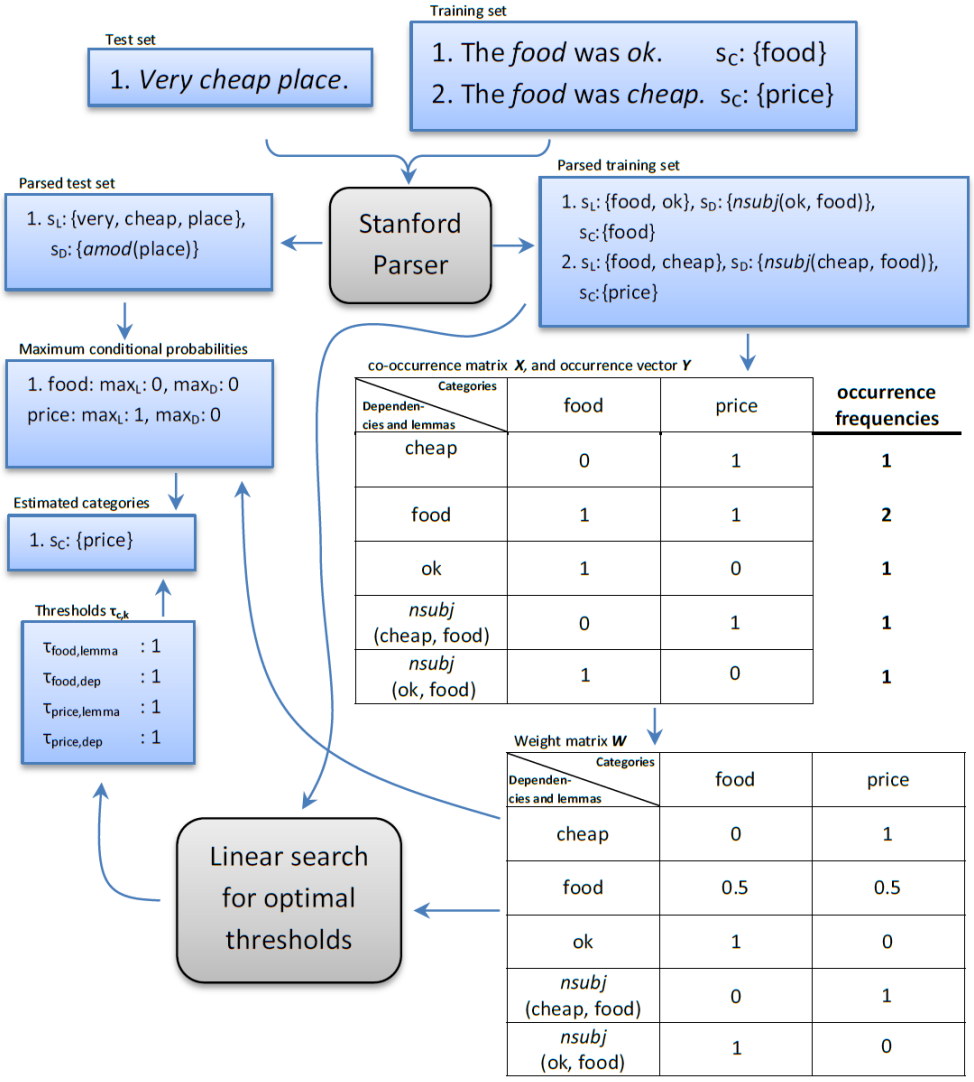

Figure 5.6: Example flowchart of the supervised method.

denoted by s_{D_1}, s_{D_2}, and s_{D_3}, respectively. The training set provides the annotated categories of each sentence s, which is denoted by s_C.

Determine weight matrix W

Next all unique categories are identified, storing them in category set C. Additionally, the occurrence frequencies of all lemmas and dependency forms are stored in vector Y, while the co-occurrence frequencies of all dependency form/lemma-category combinations, are counted and stored in matrix X, respectively. These three steps of gathering statistical information on the data are all performed on the training data alone.

After the occurrence vector Y and co-occurrence matrix X are obtained, we calculate for each co-occurrence entry $X_{c,j}$, with occurrence frequency Y_j greater than θ, its associated conditional probability $P(c|j)$, and store it in weight matrix W. The threshold θ prevents low occurring lemmas and dependency forms from becoming indicators. This way we aim to mitigate possible overfitting. The value of θ is, based on intuition, set to 4 for these experiments, however, this could be part of the training regime as well. The formula for calculating these conditional probabilities is shown in Equation 5.3. The pseudo-code for identifying the category set C, counting the occurrence and co-occurrence frequencies, and computing the weight matrix W, is shown in Algorithm 5.2.

$$W_{c,j} = \frac{X_{c,j}}{Y_j} \tag{5.3}$$

Find optimal thresholds $\tau_{c,k}$

Next we execute a linear search for optimal thresholds $\tau_{c,k}$, $c \in C$, $k \in \{L, D_1, D_2, D_2\}$ on the training set. For each category $c \in C$ we optimize the four thresholds $\tau_{c,L}$, τ_{c,D_1}, τ_{c,D_2}, τ_{c,D_3}. Because the selection of one threshold influences the selection of the other three thresholds, all thresholds are optimized together.

The linear search uses Equation 5.4 to find the maximum conditional probability $max_{c,k}$. If the maximum conditional probability $max_{c,k}$ is higher than the corresponding threshold $\tau_{c,k}$, we predict category c.

Algorithm 5.2: Identify category set C and compute weight matrix W.

input : training set, occurrence threshold θ
output : category set C, Weight matrix W

$C, X, Y \leftarrow \emptyset$
foreach *sentence* $s \in$ *Training set* **do**
  // s_k are the lemmas/dependecies of s
  foreach $s_k \in \{s_L, s_{D_1}, s_{D_2}, s_{D_3}\}$ **do**
    foreach *dependency forms/lemmas* $j \in s_k$ **do**
      // count dependency form/lemma occurrence j in Y
      if $j \notin Y$ **then**
        add j to Y
      end
      $Y_j \leftarrow Y_j + 1$
      // s_C are the categories of s
      foreach *category* $c \in s_C$ **do**
        // Add unique categories in category set C
        if $c \notin C$ **then**
          add c to C
        end
        // count co-occurrence (c, j) in X
        if $(c, j) \notin X$ **then**
          add (c, j) to X
        end
        $X_{c,j} \leftarrow X_{c,j} + 1$
      end
    end
  end
end
// Compute conditional probabilities
foreach $(c, j) \in X$ **do**
  if $Y_j > \theta$ **then**
    $W_{c,j} \leftarrow X_{c,j} / Y_j$
  end
end

The training set is then evaluated for a range of values of thresholds $\tau_{c,k}$, and the thresholds which provided the highest evaluation metric are selected as thresholds for the test set. In our experiments we used as evaluation metric the F_1-score and as range $[0.5, 1)$ with a step of 0.01 for thresholds $\tau_{c,k}$.

$$\max_{c,k} = \max_{j \in s_k} W_{c,j} \tag{5.4}$$

Algorithm 5.3: Estimating categories for the test set.

input : training set, test set, occurrence threshold θ
output : Estimated categories for each sentence in the test set

```
W,C ← Algorithm 5.2(Training set, θ)
τ_{c,L}, τ_{c,D_1}, τ_{c,D_2}, τ_{c,D_3} ← LinearSearch (Training set, W, C)
// Processing of review sentences
foreach sentence s ∈ test set do
    foreach category c ∈ C do
        // Obtain maximum conditional probabilities P(c|j) = W_{c,j} per
           type, for sentence s
        max_{c,L} ← max_{l∈s_L} W_{c,l}
        max_{c,D_1} ← max_{d_1∈s_{D_1}} W_{c,d_1}
        max_{c,D_2} ← max_{d_2∈s_{D_2}} W_{c,d_2}
        max_{c,D_3} ← max_{d_3∈s_{D_3}} W_{c,d_3}
        if max_{c,L} > τ_{c,L} or max_{c,D_1} > τ_{c,D_1} or max_{c,D_2} > τ_{c,D_2} or
          max_{c,D_3} > τ_{c,D_3} then
            estimate category c for sentence s
        end
    end
end
```

Estimate categories

The final step is to predict the aspect categories for each unseen sentence $s \in$ test set. From all lemmas and dependency forms s_L, s_{D_1}, s_{D_2}, and s_{D_3} in sentence s we find the maximum conditional probability $P(c|j)$, as described in Equation 5.4, for each category $c \in C$. Then, if any of these maximum conditional probabilities surpasses

their threshold $\tau_{c,k}$, category c is assigned as an aspect category for sentence s. The pseudo-code for this step is shown in Algorithm 5.3.

5.4.2 Limitations

The main disadvantage of this method is that, contrary to unsupervised methods, this method requires a sufficient amount of annotated data in order to work properly. For a small annotated data set this method will be inaccurate. Especially the dependency indicators require enough training data in order to be effectively used to predict categories.

Another limitation stems from the use of dependency relations. These are found by using a syntactical parser, which relies on the grammatical correctness of the sentence. However, the grammar used in review sentences can be quite disappointing. If sentences have weird grammatical structures, the parser will not be able to extract relevant dependency relations from these sentences, and may even misrepresent certain dependencies.

Furthermore, because dependencies are triplets, and many different dependency relations exist, the number of different dependency triplets is huge, which makes it harder to find rules that generalize well to unseen data. While a sufficiently large training set will negate this issue, this might unfortunately not always be available.

5.5 Evaluation

For the evaluation of the proposed methods, the training and test data from SemEval 2014 (Pontiki et al., 2014) are used. It contains 3000 training sentences and 800 test sentences taken from restaurant reviews. Each sentence has one or more annotated aspect categories. Figure 5.7 shows that each sentence has at least one category and that approximately 20% of the sentences have multiple categories. With 20% of the sentences having multiple categories, a method would benefit from being able to predict multiple categories. This is one of the reasons why association rule mining is useful in this scenario as multiple rules can apply to a single sentence.

Figure 5.8 presents the relative frequency of each aspect category, showing that the two largest categories, ‘food’ and ‘anecdotes/miscellaneous’, are found in more

than 60% of the sentences. This should make these categories easier to predict than the other categories, not only because of the increased chance these categories appear, but also because there is more information about them.

Last, in Figure 5.9, the proportion of implicit and explicit aspect categories is shown. It is clear that using techniques related to implicit aspect detection is appropriate here, given that more than three quarters of the aspect categories is not literally mentioned in the text.

Because both the unsupervised and the supervised method work best for well-defined aspect categories, the last category in this data set, 'anecdotes/miscellaneous' poses a challenge. It is unclear what exactly belongs in this category, and its concept is rather abstract. For that reason, we have chosen not to assign this category using any of the actual algorithms, but instead, this category is assigned when no other category is assigned by the algorithm. The characteristics in Figure 5.7 also show that the use of 'anecdotes/miscellaneous' as a 'fallback' is justified given its large size and the fact that every sentence has at least one category.

As is done at the SemEval 2014 (Pontiki et al., 2014) competition, the methods are evaluated based on the micro averaged F_1-score defined as follows:

$$F_1 = \frac{2 \cdot P \cdot R}{P + R}, \tag{5.5}$$

where precision (P) and recall (R) are defined as follows,

$$P = \frac{TP}{FP + TP}, R = \frac{TP}{FN + TP}, \tag{5.6}$$

where TP, FP, and FN represent the true positives, false positives, and false negatives, respectively, of the estimated aspect categories with respect to the (gold) aspect category annotations.

5.5.1 Unsupervised Method

Table 5.1 displays, for each aspect category, the chosen firing threshold together with the resulting precision, recall, and F_1-score on the test set. The category '*anecdotes/miscellaneous*' is estimated when none of the other four categories are chosen in the sentence.

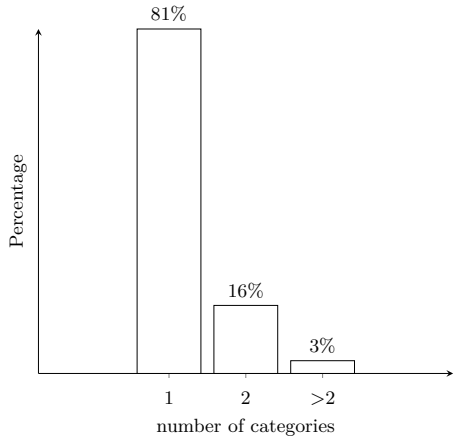

Figure 5.7: The distribution of number of aspect categories per sentence

With an overall F_1-score of 67.0% on the test set, the method seems to perform well, but the performance strongly depends on the choice of the parameters. If we would for example, had chosen to treat the category 'service' as a dominant category, like we did with the category 'food', and had lowered the firing threshold, then the precision of this category would have dropped significantly, while the recall would only increase slightly. Likewise, if we would not have treated the category 'food' as a dominant category, its recall would have severely dropped, while the precision would have increased. So certain domain knowledge about the dataset is required when choosing parameter values. Table 5.2 shows this sensitivity of the firing thresholds, where the relative change in terms of F_1-score is given when deviating from the chosen thresholds. As can be seen the proposed method is sensitive to threshold variations.

From Table 5.1, one can conclude that this approach has difficulty predicting the category '*ambience*'. This might be due to the nature of that particular category, as it is often not specified in a sentence by just one word, but is usually derived from a sentence by looking at the sentence as a whole. This can be illustrated with the following example

> "Secondly, on this night the place was overwhelmed by upper east side ladies perfume."

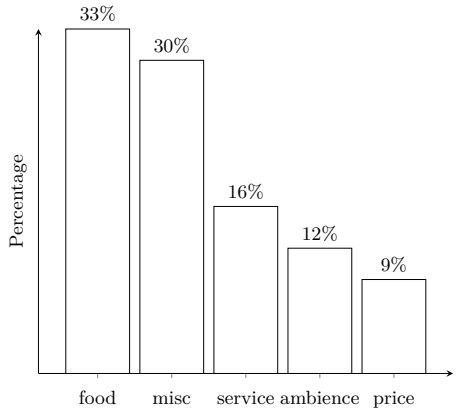

Figure 5.8: The relative frequency of the aspect categories

Category	TP's	FP's	FN's	τ_c	precision	recall	F_1
food	313	103	105	0.22	75.1%	74.4%	74.8%
service	100	4	72	0.19	96.2%	58.1%	72.5%
ambience	41	10	77	0.09	80.4%	34.8%	48.5%
price	52	16	31	0.09	79.0%	54.2%	64.3%
misc.	163	159	71	-	50.6%	70.9%	59.1%
all	852	157	173	-	70.0%	64.7%	67.0%

Table 5.1: Chosen firing thresholds and their evaluation scores on the test set.

where there is no particular word that strongly suggests that 'ambience' is the right category for this sentence.

5.5.2 Supervised Method

For the supervised method we use the training set to learn the parameters and co-occurrence frequencies, after which we evaluate the method on the test set. To see the impact the dependency indicators have, this method is executed separately for the dependency indicators, lemma indicators and a combined version where both lemma

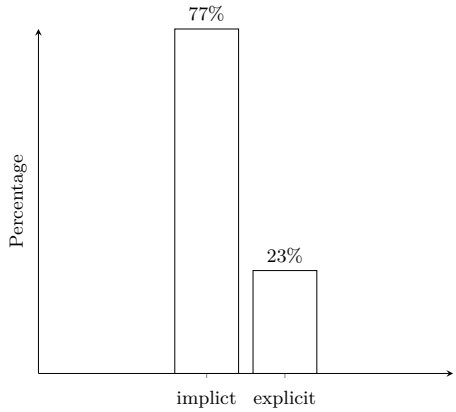

Figure 5.9: The ratio between implicit aspect categories and explicitly mentioned ones

Category	-0.05	-0.02	-0.01	0	0.01	0.02	0.05
food	-8	-7.9	-7.9	0	-1.9	-6.7	-25
service	-8.6	-3.3	-4.6	0	0	0	0
ambience	-47	3.1	8.9	0	0	0	-5.6
price	-72.1	-18.7	-11	0	0	0.1	1.6

Table 5.2: Relative change in F_1, when varying firing thresholds.

and dependency indicators are used, and evaluated on the test set. Tables 5.3, 5.4, and 5.5 show the results.

Comparing the results from Tables 5.3-5.5 shows that using both dependency and lemma indicators provides best results. However, these results are only slightly better than when only lemma indicators are used, which means that we cannot claim that dependency indicators are beneficial when predicting categories in terms of F_1 score. Table 5.5 does show that by themselves, dependency indicators do have predicting power, albeit less than lemma indicators. This was as expected, since dependency indicators consists of more than one component, which makes it harder to find rules that generalize well to unseen data, and, in addition, they also rely on the grammatical correctness of the sentence.

Category	TP's	FP's	FN's	precision	recall	F_1
food	371	51	47	87.9%	88.8%	88.3%
service	159	32	13	83.2%	92.4%	87.6%
ambience	83	28	35	73.8%	70.3%	72.5%
price	74	8	9	90.2%	89.2%	89.7%
anecdotes/misc.	165	38	69	81.3%	70.5%	75.5%
all	852	157	173	84.4%	83.1%	83.8%

Table 5.3: Evaluation scores of the supervised method with both dependency and lemma indicators on the test set.

Category	TP's	FP's	FN's	precision	recall	F_1
food	348	35	70	90.9%	83.3%	86.9%
service	153	13	19	92.2%	89.0%	90.5%
ambience	78	28	40	73.6%	66.1%	69.6%
price	79	9	4	89.9%	95.2%	92.4%
anecdotes/misc.	165	38	69	81.3%	70.5%	75.5%
all	823	123	202	87.5%	80.3%	83.5%

Table 5.4: Evaluation scores of the supervised method with only lemma indicators on the test set.

Using dependency indicators, in addition to lemma indicators, does seem to result into finding more categories, even though it is less precise in doing so. This is especially the case for the category 'food'. The main reason for this is that 'food' is by far the largest category, resulting in more available training data for this category, which makes it easier to find rules.

5.5.3 Comparison

To evaluate the quantitative performance of the proposed method, it is compared against several baseline methods and three successful methods from the SemEval-2014 competition. The four baseline methods are:

1. **Seed word baseline:** This baseline estimates a category if one of its seed words is present in the sentence.

Category	TP's	FP's	FN's	precision	recall	F_1
food	343	45	75	88.4%	82.1%	85.1%
service	152	27	20	84.9%	88.4%	86.6%
ambience	62	34	56	64.6%	52.5%	57.9%
price	61	5	22	92.4%	73.5%	81.9%
anecdotes/misc.	165	38	69	81.3%	70.5%	75.5%
all	783	149	242	84.0%	76.4%	80.0%

Table 5.5: Evaluation scores of the supervised method with only dependency indicators on the test set.

2. **Majority baseline:** This baseline predicts for every sentence the two most common categories (i.e., '*food*' and '*anecdotes/miscellaneous*') present in the data.

3. **Random baseline:** For this baseline we randomly select categories for each sentence. The chance of selecting a certain category depends on the appearance in the training set, just as the number of selected categories depends on the distribution of the number of categories per sentence in the training set.

4. **SemEval baseline:** The final baseline comes from Pontiki et al. (2014), and it is a simple supervised method. For every test sentence s, the k most similar to s training sentences are retrieved. Here the similarity between two sentences is measured by calculating the Dice coefficient of the sets of distinct words of two sentences. Then, s is assigned the m most frequent aspect categories of the k retrieved sentences. This baseline is clearly a supervised method and thus requires a training set.

The four methods from the literature are V3 (Garcia-Pablos et al., 2014), an unsupervised semantic similarity algorithm, Schouten and Frasincar (2014), a supervised co-occurrence based algorithm, UWB (Brychcín et al., 2014), the best performing submitted constrained (i.e., no external training data is used) method, and a constrained version (cf. end of Section 5.2) of Kiritchenko et al. (2014), the best constrained supervised machine learning approach at this particular task at SemEval-2014. The resulting overall precision, recall, and F_1-score are displayed in Table 5.6.

Method	precision	recall	F_1
Random baseline	30.8%	30.5%	30.6%
Majority baseline	38.8%	63.7%	48.2%
Seed word baseline	57.2%	46.4%	51.2%
Schouten and Frasincar (2014)	63.3%	55.8%	59.3%
V3 (Garcia-Pablos et al., 2014)	63.3%	56.9%	60.2%
SemEval-2014 baseline Pontiki et al. (2014)	-	-	63.9%
Proposed Unsupervised Method	69.5%	64.7%	67.0%
UWB (Brychcín et al., 2014)	85.1%	77.4%	81.0%
constrained Kiritchenko et al. (2014)	86.5%	78.3%	82.2%
Proposed Supervised Method	84.4%	83.1%	83.8%

Table 5.6: F_1-scores of different (constrained) methods.

Clearly, the supervised method, as well as many other (supervised) methods presented at SemEval-2014 perform better than the proposed unsupervised method. However, this is to be expected for an unsupervised method. Interestingly, it is able to outperform the basic bag-of-words supervised approach of the SemEval-2014 baseline, as well as the supervised co-occurrence based method from Schouten and Frasincar (2014). On the other hand, the second proposed method beats all constrained methods from the SemEval-2014 competition. Note however, that the full (unconstrained) method from Kiritchenko et al. Kiritchenko et al. (2014) outperforms our proposed method by a few percent, which is due to the fact that it enjoys an unconstrained training regime.

In Figure 5.10 F_1-scores are shown for different sizes of the training set, using a stratified sampling technique where the distribution of the categories remains similar to the original dataset. Each data point in the figure represents an incremental increase of 10% (300 sentences) in labeled data, for the supervised method, and unlabeled data for the unsupervised method. The supervised method always seems to outperform the unsupervised method, although larger training sizes for the unsupervised method seem to perform on par with the supervised method for which very small amounts of labeled data are available (F_1-score around 70%).

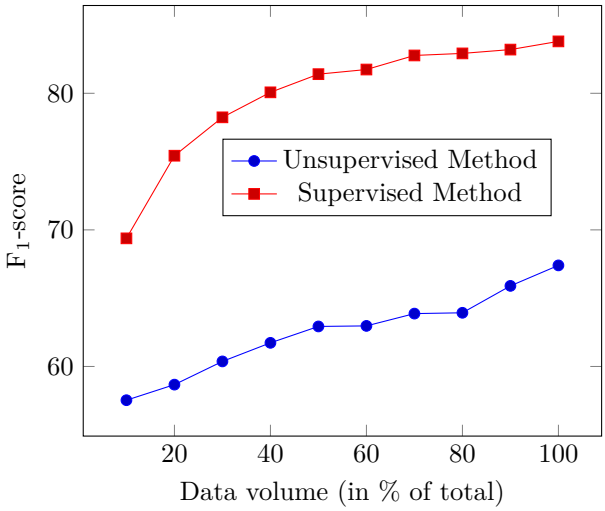

Figure 5.10: F_1-scores for different sizes of the training set (% of 3000 sentence).

5.6 Conclusion

In this work we have presented two methods for detecting aspect categories, that is useful for online review summarization. The first, unsupervised, method, uses spreading activation over a graph built from word co-occurrence data, enabling the use of both direct and indirect relations between words. This results in every word having an activation value for each category that represents how likely it is to imply that category. While other approaches need labeled training data to operate, this method works unsupervised. The major drawback of this method is that a few parameters need to be set beforehand, and especially the category firing thresholds (i.e., τ_c) need to be carefully set to gain a good performance. We have given heuristics on how these parameters can be set.

The second, supervised, method utilizes co-occurrence information in a more straightforward manner. Here, the co-occurrence frequency between annotated aspect categories and both lemmas and dependencies is used to calculate conditional probabilities. If the maximum conditional probability is higher than the associated, trained, threshold, the category is assigned to that sentence. Evaluating this

approach on the official SemEval-2014 test set Pontiki et al. (2014), shows a high F_1-score of 83%.

In terms of future work, we would like to investigate how injecting external knowledge would improve the results. While lexicons are a good way of doing that, as shown by Kiritchenko et al. (2014), we are especially interested in exploiting more semantic alternatives, like ontologies or other semantic networks. Also, as we are dealing with unbalanced data, we plan to explore machine learning techniques that address this problem Tang et al. (2009b).

Chapter 6

An Information Gain-Driven Feature Study for Aspect-Based Sentiment Analysis*

NOWADAYS, *opinions are a ubiquitous part of the Web and sharing experiences has never been more popular. Information regarding consumer opinions is valuable for consumers and producers alike, aiding in their respective decision processes. Due to the size and heterogeneity of this type of information, computer algorithms are employed to gain the required insight. Current research, however, tends to forgo a rigorous analysis of the used features, only going so far as to analyze complete feature sets. In this chapter we analyze which features are good predictors for aspect-level sentiment using Information Gain and why this is the case. We also present an extensive set of features and show that it is possible to use only a small fraction of the features at just a minor cost to accuracy.*

---

*This chapter is based on "Kim Schouten, Flavius Frasincar, and Rommert Dekker. An Information Gain-Driven Feature Study for Aspect-Based Sentiment Analysis. In *21st International Conference on Application of Natural Language to Information Systems (NLDB 2016)*, volume 9612 of *Lecture Notes in Computer Science*, pages 48-59. Springer, 2016."

6.1 Introduction

Nowadays, opinions are a ubiquitous part of the Web and sharing experiences has never been more popular (Feldman, 2013). Information regarding consumer opinions is valuable for consumers and producers alike, aiding in their respective decision processes. Due to the size and heterogeneity of this type of information, computer algorithms are employed to gain insight into the sentiment expressed by consumers and on what particular aspects that sentiment is expressed, and research into this type of algorithms has enjoyed increasingly high popularity over the last decade (Liu, 2012).

Research has led to a number of different approaches to aspect-level sentiment analysis (Schouten and Frasincar, 2016), that can be divided into three categories. The first group consists of methods that predominantly use a sentiment dictionary (e.g., (Hu and Liu, 2004b)). Sentiment values are then assigned to certain words or phrases that appear in the dictionary and using a few simple rules (e.g., for negation and aggregation), the sentiment values are combined into one score for each aspect. The second type is categorized by the use of supervised machine learning methods (e.g., (Choi and Cardie, 2008)). Using a significant amount of annotated data, where the sentiment is given for each aspect, a classifier can be trained that can predict the sentiment value for yet unseen aspects. Last, some methods based on unsupervised machine learning are also available, but these usually combine aspect detection and sentiment classification into one algorithm (e.g., (Titov and McDonald, 2008a)).

Supervised learning has the advantage of high performance, and given the fact that sentiment is usually annotated as a few distinct classes (i.e., positive, neutral, and negative), traditional statistical classifiers work remarkably well. Unfortunately, most of these methods are somewhat of a black box: once provided with enough input, the method will do its task and will classify aspect sentiment with relatively good accuracy. However, the inner workings are often unknown, and because of that, it is also not known how the various input features relate to the task. Since most classifiers can deal with large dimensionality on the input, one tends to just give all possible features and let the classifier decide which ones to use. While this is perfectly fine when aiming for performance, it does not give much insight into which particular features are good predictors for aspect sentiment. Knowing which features

are relevant is important for achieving insight into the performed task, but it also allows to speed up the training process by only employing the relevant features with possibly only a minor decrease in performance.

This focus on performance instead of explanation is most typically found in benchmark venues, such as the Semantic Evaluation workshops (Pontiki et al., 2015). Here, participants get annotated training data and are asked to let their algorithm provide the annotations for a non-annotated data set. The provided annotations are then centrally evaluated and a ranking is given, showing how each of the participating systems fared against each other. Scientifically speaking, this has the big benefit of comparability, since all of the participants use the same data and evaluation is done centrally, as well as reproducibility, since the papers published in the workshop proceedings tend to focus on how the system was built. The one thing missing, however, is an explanation of why certain features or algorithms perform so great or that bad, since there usually is not enough space in the allowed system descriptions to include this.

Hence, this chapter aims to provide insight into which features are useful, using a feature selection method based on Information Gain (Mitchell, 1997), which is one of the most popular feature filtering approaches. Compared to wrapper approaches, such as Forward Feature Selection, it does not depend on the used classification algorithm. Using Information Gain, we can compute a score for each individual feature that represents how well that feature divides the aspects between the various sentiment classes. Thus, we can move beyond the shallow analysis done per feature set, and provide deeper insight, on the individual feature level.

The remainder of this chapter is organized as follows. First, the problem of aspect-level sentiment analysis is explained in more detail in Section 6.2, followed by Section 6.3, in which the framework that is responsible for the natural language processing is described, together with the methods for training the classifier and computing the Information Gain score. Then, in Section 6.4, the main feature analysis is performed, after which Section 6.5 closes with a conclusion and some suggestions for future work.

6.2 Problem Description

Sentiment analysis can be performed on different levels of granularity. For instance, a sentiment value can be assigned to a complete review, much like the star ratings that are used on websites like Amazon. However, to get a more in-depth analysis of the entity that is being reviewed, whether in a traditional review or in a short statement on social media, it is important to know on which aspect of the entity a statement is being made. Since entities, like products or services, have many facets and characteristics, ideally one would want to assign a sentiment value to a single aspect instead of to the whole package. This challenge is known as aspect-level sentiment analysis (Schouten and Frasincar, 2016), or aspect-based sentiment analysis (Pontiki et al., 2015), and this is the field this research is focused on.

More precisely, we use a data set where each review is already split into sentences and for each sentence it is known what the aspects are. Finding the aspects is a task that is outside the scope of this chapter. Given that these aspects are known, one would want to assign the right sentiment value to each of these aspects. Most of the annotated aspects are explicit, meaning that they are literally mentioned in the text. As such, it is known which words in the sentence represent this aspect. Some aspects, however, are implicit, which means that they are only implied by the context of the sentence or the review as a whole. For these aspects, there are no words that directly represent the aspect, even though there will be words or expressions that point to a certain aspect. Both explicit and implicit aspects are assigned to an aspect category which comes from a predefined list of possible aspect categories.

For explicit aspects, since we know the exact words in the sentence that represent this aspect, we can use a context of n words before and after each aspect from which to derive the features. This allows for contrasting aspects within the same sentence. For implicit aspects, this is not possible and hence we extract the features from the whole sentence. Note that each aspect will have a set of extracted features, since it is the sentiment value of each aspect that is the object of classification. An example from the used data set showing both explicit and implicit aspects (i.e., `target="NULL"` for implicit features) is shown in Fig. 6.1.

```
<sentence id="1032695:1">
<text>Everything is always cooked to perfection, the service is
    excellent, the decor cool and understated.</text>
<Opinions>
<Opinion target="NULL" category="FOOD#QUALITY" polarity="" from="0"
    to="0"/>
<Opinion target="service" category="SERVICE#GENERAL" polarity="" from
    ="47" to="54"/>
<Opinion target="decor" category="AMBIENCE#GENERAL" polarity="" from=
    "73" to="78"/>
</Opinions>
</sentence>
```

Figure 6.1: A snippet from the used dataset showing an annotated sentence from a restaurant review.

6.3 Framework

In this section we present the steps of our framework. First, all textual data is preprocessed, which is an essential task for sentiment analysis (Haddi et al., 2013), by feeding it through a natural language pipeline based on Stanford's CoreNLP package (Manning et al., 2014). This extracts information like the lemma, Part-of-Speech (PoS) tag, and grammatical relations for words in the text. Furthermore, we employ a spell checker called JLanguageTool * to correct obvious misspellings and a simple word sense disambiguation algorithm based on Lesk (Lesk, 1986) to link words to their meaning, represented by WordNet synsets. Stop words are not removed since some of these words actually carry sentiment (e.g., emoticons are a famous example), and the feature selection will filter out features that are not useful anyway, regardless of whether they are stopwords or not.

The next step is to prepare all the features that will be used by an SVM (Chang and Lin, 2011), the employed classifier in this work. For example, if we want to use the lemma of each word as a feature, each unique lemma in the dataset will be collected and assigned a unique feature number, so when this feature is present in the text when training or testing, it can be denoted using that feature number. Note that, unless otherwise mentioned, all features are binary, denoting the presence or

*wiki.languagetool.org/java-api

absence of that particular feature. For the feature analysis, the following types of features are considered:

- **Word-based features:**
  - Lemma: the dictionary form of a word
  - Negation: whether or not one or more negation terms from the General Inquirer Lexicon[†] are present;
  - The number of positive words and the number of negative words in the context are also considered as features, again using the General Inquirer Lexicon;
- **Synset-based features:**
  - Synset: the WordNet synset associated with this word, representing its meaning in the current context;
  - Related synsets: synsets that are related in WordNet to one of the synsets in the context (e.g., hypernyms that generalize the synsets in the context);
- **Grammar-based features:**
  - Lemma-grammar: a binary grammatical relation between words represented by their lemma (e.g., "keep-nsubj-we");
  - Synset-grammar: a binary grammatical relation between words represented by their synsets which is only available in certain cases, since not every word has a synset (e.g., "ok#JJ#1-*cop*-be#VB#1");
  - PoS-grammar: a binary grammatical relation between words represented by PoS tags (e.g., "VB-nsubj-PRP"), generalizing the lemma-grammar case with respect to Part-of-Speech;
  - Polarity-grammar: a binary grammatical relation between synsets represented by polarity labels (e.g., "neutral-nsubj-neutral"). The polarity class is retrieved from SentiWordNet (Baccianella et al., 2010), with the neutral class being the default when no entry was found in SentiWordNet;

[†]`http://www.wjh.harvard.edu/~inquirer`

- **Aspect Features:**
  - Aspect Category: the category label assigned to each aspect is encoded as a set of binary features (e.g., "FOOD#QUALITY").

Note that with grammar-based features, we experiment with various sorts of triples, where two features are connected by means of a grammatical relation. This kind of feature is not well studied in literature, since n-grams are usually preferred by virtue of their simplicity. Apart from the effect of Information Gain, the benefit of using this kind of feature will be highlighted in the evaluation section.

With all the features known, the Information Gain score can be computed for each individual feature. This is done using only the training data. Information Gain is a statistical property that measures how well a given feature separates the training examples according to their target classification (Mitchell, 1997). Information Gain is defined based on the measure entropy. The entropy measure characterizes the (im)purity of a collection of examples. Entropy is defined as:

$$Entropy(S) = -\sum_{i} p(i|S) \log_2 p(i|S)$$

with S a set of all aspects and $p(i|S)$ the fraction of the aspects in S belonging to class i. These classes are either positive, negative, or neutral. The entropy typically changes when we partition the training instances into smaller subsets, i.e., when analyzing the entropy value per feature. Information Gain represents the expected reduction in entropy caused by partitioning the samples according to the feature in question. The Information Gain of a feature t relative to a collection of aspects S, is defined as:

$$Information\ Gain(S,t) = Entropy(S) - \sum_{v \in Values(t)} \frac{|S_v|}{|S_t|} Entropy(S_v)$$

where Values(t) is the set of all possible values for feature t. These values again are either positive, negative, or neutral. S_v is the subset of S with aspects of class v related to feature t. S_t is the set of all aspects belonging to feature t. $|\cdot|$ denotes the cardinality of a set.

In this work, we will analyze the optimal number of features with Information Gain, one of the most popular measures used in conjunction with a filtering approach for feature selection. This is executed as follows. First, the Information Gain is computed for each feature. Next, the IG scores of all the features are sorted from high to low and the top $k\%$ features are used in the SVM. This percentage k can either be determined using validation data or it can be manually set.

Afterwards, the training data is used to train the SVM. The validation data is used to optimize for a number of parameters: the cost parameter C for the SVM, the context size n that determines how many words around an explicit aspect are used to extract features from, and the value for k that determines percentage-wise how many of the features are selected for use with the SVM.

After training, new, previously unseen data can be classified and the performance of the algorithm is computed. By employing ten-fold cross-validation, we test both the robustness of the proposed solution and ensure that the test data and training data have similar characteristics.

6.4 Evaluation

Evaluation is done on the official training data of the SemEval-2016 Aspect-Based Sentiment Analysis task Pontiki et al. (2016). We have chosen to only use the data set with restaurant reviews in this evaluation because it provides target information, annotating explicit aspects with the exact literal expression in the sentence that represents this aspect. An additional data set containing laptop reviews is also available, but it provides only category information and does not give the target expression. The used data set contains 350 reviews that describe the experiences people had when visiting a certain restaurant. There were no restrictions on what to write and no specific format or template was required. In Table 6.1 the distribution of sentiment classes over aspects is given and in Table 6.2, the proportion of explicit and implicit aspects in this dataset are shown (cf. Section 6.2).

To arrive at stable results for our analysis, we run our experiments using 10-fold cross-validation where the data set is divided in ten random parts of equal size. In this setup, seven parts are used for training the SVM, two parts are used as a

Table 6.1: The sentiment distribution over aspects in the used data set

sentiment	nr. of aspects	% of aspects
positive	1652	66.1%
neutral	98	3.9%
negative	749	30.0%
total	2499	100%

Table 6.2: The distribution of explicit and implicit aspects in the used data set

type	nr. of aspects	% of aspects
explicit	1879	75.2%
implicit	620	24.8%
total	2499	100%

validation set to optimize certain parameters (i.e., the C parameter of the SVM, k, the exact percentage of features to be kept, and n, encoding how many words around an aspect should be used to extract features from), and the last part is used for testing. This procedure is repeated ten times, in such a way that the part for testing is different each round. This ensures that each part of the data set has been used for testing exactly once and so a complete evaluation result over the whole data set can be obtained.

From one of the folds, we extracted the list of features and their computed Information Gain. As shown in Fig. 6.2, the distribution of Information Gain scores over features is highly skewed. Only about 8.6% of the features actually receives a non-zero score.

Grouping the features per feature type, we can compute the average Information Gain for all features of a certain type. This plot, shown in Fig. 6.3, shows how important, on average, each of the feature types is. Given the fact that the y-axis is logarithmic, the differences between importance are large. Traditional feature types like 'Negation present' and 'Category' are still crucial to having a good performance, but the new feature type 'Polarity-grammar' also shows good performance. The new 'Related-synsets' and 'POS-grammar' category are in the same league as the

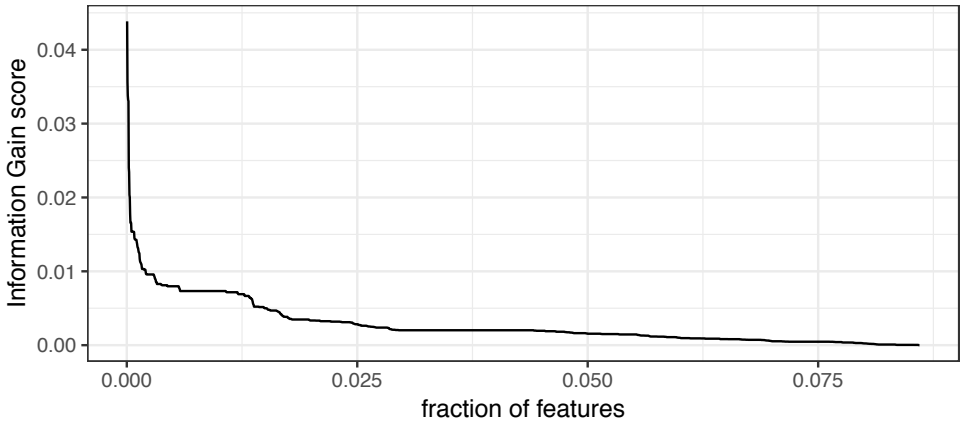

Figure 6.2: The Information Gain for all features with a non-zero score, in descending order of Information Gain.

traditional 'Lemma' category, having an average Information Gain. Feature types that are less useful are 'Lemma-grammar' and 'Synset-grammar', which are very fine-grained and are thus less likely to generalize well from training to test data.

In Fig. 6.4, the average in-sample accuracy, as well as the accuracy on the validation and test data are presented for a number of values for k, where k means that the top k percent ranked features were used to train and run the SVM. It shows that when using just the top 1% of the features, an accuracy of 72.4% can be obtained, which is only 2.9% less than the performance obtained when using all features. This point corresponds to a maximum in the performance on the validation data. Other parameters that are optimized using validation data are C, which is on average set to 1, and n, which is on average set to 4. Note that since all parameters that are optimized with validation data are optimized per fold, their exact value differs per fold and thus an average is given. The picture is split into five different levels of granularity on the x-axis, providing more details for lower values of k. In the first split, the method start at the baseline performance, since at such a low value of k, no features are selected at all. Then, in the second block, one can see that, because these are all features with high Information Gain, the performance on the test data closely tracks the in-sample performance on the training data. However, in the third split,

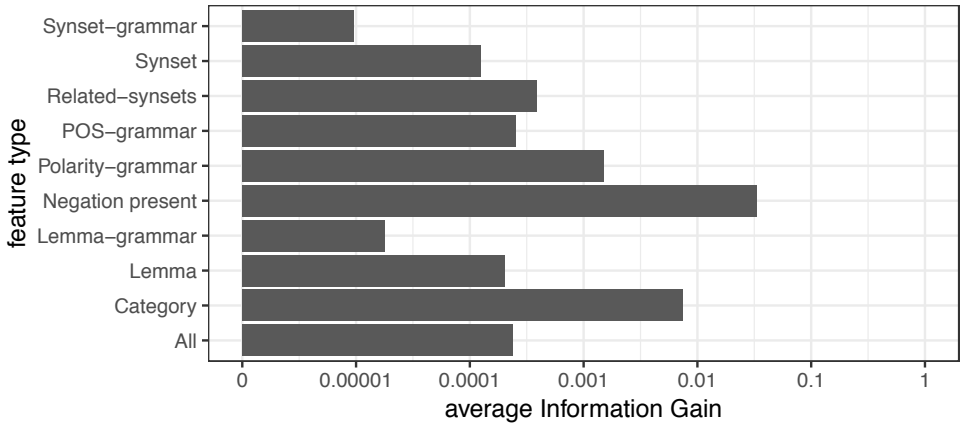

Figure 6.3: The average Information Gain for each feature type.

some minor overfitting starts to occur. The best features have already been used, so lesser features make their way into the SVM, and while in-sample performance goes up, out-of-sample performance does not grow as fast. This effect is illustrated even stronger in the fourth block, where performance on the training data goes up spectacularly, while performance on the test data actually goes down. The features that are added here all have a very low Information Gain.

The last block is slightly different because for almost all of these features, roughly 90% of the total number, the Information Gain is zero. However, as is made evident by the upward slope of the out-of-sample performance, these features are not necessarily useless and the SVM is able to use them to boost performance with a few percent. This is possible due to the fact that, while Information Gain is computed for each feature in isolation, the SVM takes interaction effects between features into account. Hence, while these features may not be useful on their own, given the features already available to the SVM, they can still be of use. This interaction effect accounts for the 2.9% penalty to performance increase when using feature selection based on Information Gain.

The diminishing Information Gain is also clearly illustrated in Fig. 6.5. It follows roughly the same setup in levels of detail as Fig. 6.4, with this exception that the first split is combined into the first bar, since not enough features were selected in

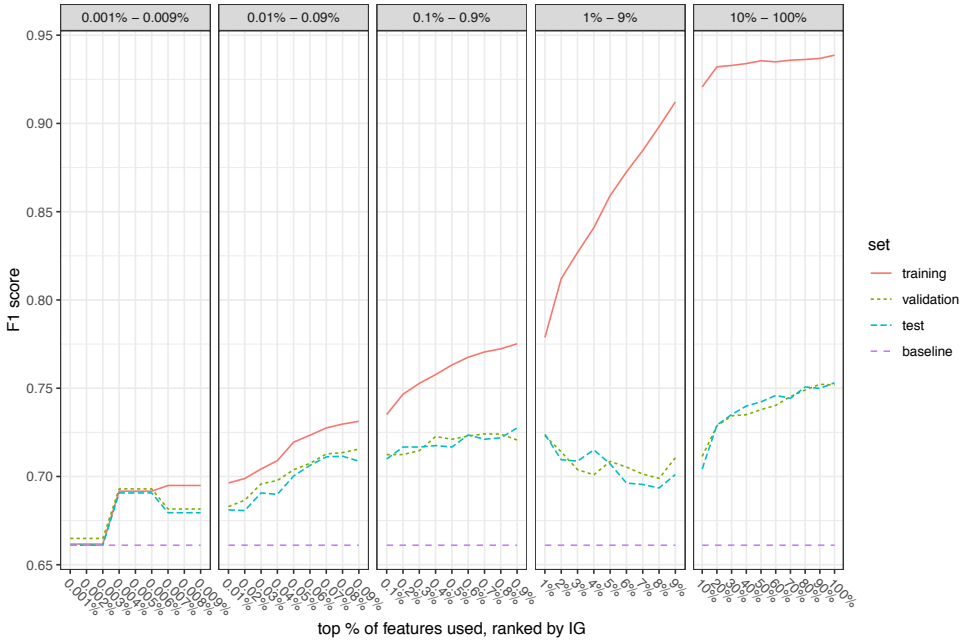

Figure 6.4: The average accuracy on training, validation, and test set for each of the subsets of features.

the first split to create a meaningful set of bars. Furthermore, while Fig. 6.4 has a cumulative x-axis, this figure does not, showing the average Information Gain of features that are added in each range (instead of all features added up to that point). Given that the lowest 90% of the features has an Information Gain of zero, there are no visible bars in the last split.

For each of the feature selections, we can also look at how well each type of feature is represented in that subset. Hence, we plot the percentage of features selected belonging to each of the feature types in Fig. 6.6. Features whose proportion decrease when adding more features are generally more important according to the Information Gain ranking, while features whose proportion increases when adding more features are generally less important since they generally have a lower Information Gain. This corresponds to the feature types with high bars in Fig. 6.3. It is interesting to see that 'Negation' and 'Category' are small but strong feature sets, and that

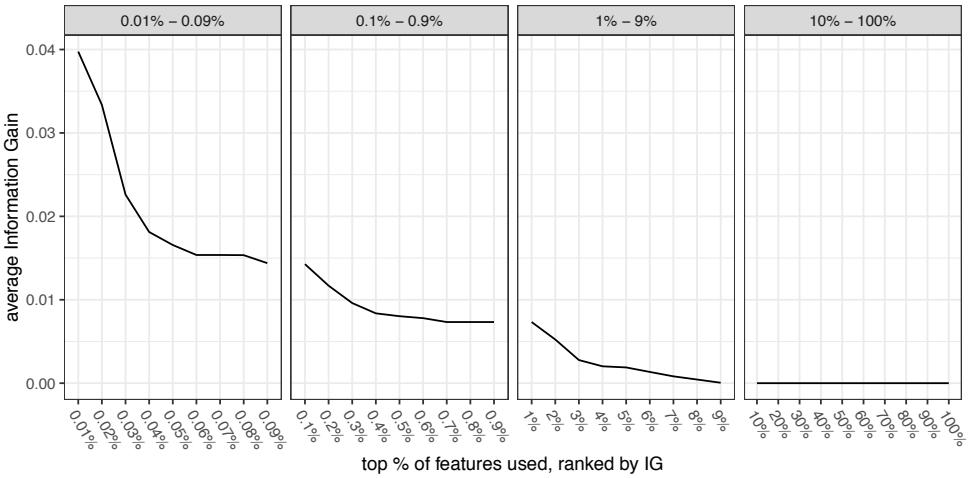

Figure 6.5: The average Information Gain score for each of the subsets of added features.

'Related-synsets', while not having many strong features, has many features that get a non-zero Information Gain, making it still a useful category of features.

Analyzing the top ranked features per feature type in Table 6.3, some of the features are easily recognized as being helpful to detect sentiment. For instance, the lemma 'not', as well as its corresponding synset in WordNet are good indicators of negative sentiment. Other features, like the 'SERVICE#GENERAL' category feature are not so self-evident. These more generic features, while not directly pointing to a certain sentiment value, are still useful by virtue of their statistics. Again looking at the 'SERVICE#GENERAL' category, we check the dataset and see that about 56% of all aspects with this category have a negative sentiment, whereas overall, only 30% of the aspects are negative. This is such a sharp deviation from the norm, that having this category label is a strong sign for an aspect to be negative. It seems that in the used data set, people tend to be dissatisfied with the provided service.

Sometimes features that appear in different categories might still represent (almost) the same information. For instance, the two top 'Lemma-grammar' features are basically the same as the two top 'Synset-grammar' features, corresponding to a

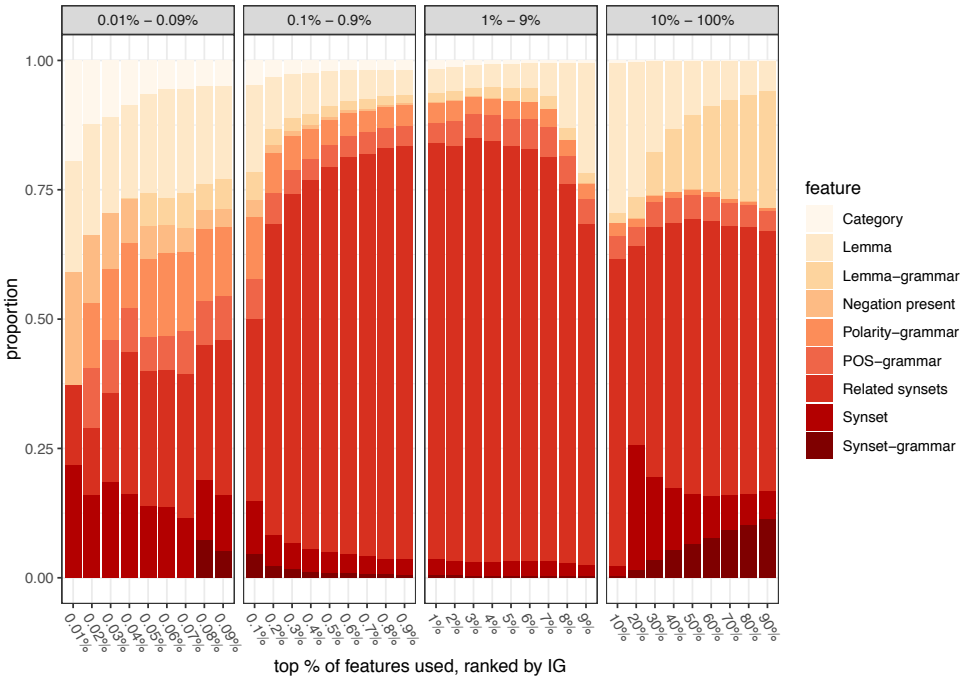

Figure 6.6: Proportion of each feature type for each of the cumulative subsets of features.

phrase like "*some aspect* is ok" or "*some aspect* is good". Another example of this is the lemma 'ok' and its corresponding synset 'ok#JJ#1'.

An interesting type of features is the 'Related-synsets' category. In Table 6.3, we seen that any synset that is similar to concepts like 'big', 'alarming', and 'satisfactory' are good predictors of sentiment, and this corresponds well with our intuition. Sometimes, a high ranked feature can give insight into how consumers write their reviews. A good example is the 'CD-dep-$' feature in the 'POS-grammar' category, which denotes a concrete price, such as "$100", and is predominantly used in conjunction with a negative sentiment. Apparently, when people are upset about the price of a restaurant, they feel the need to prove their point by mentioning the exact price.

	Category		Synsets		Polarity-grammar
1	SERVICE#GENERAL	3	not#RB#1	6	neutral-amod-positive
40	FOOD#QUALITY	27	ok#JJ#1	7	neutral-amod-neutral
42	RESTAURANT#PRICES	37	good#JJ#1	11	neutral-neg-negative

	Related-synsets		POS-grammar		Lemma-grammar
2	Similar To big#JJ#1	9	NN-amod-JJ	28	ok-cop-be
8	Similar To alarming#JJ#1	25	JJ-cop-VBZ	35	good-cop-be
10	Similar To satisfactory#JJ#1	34	CD-dep-$	374	good-punct-.

	Synset-grammmar		Lemma		Negation Present
29	ok#JJ#1-cop-be#VB#1	5	not	4	Negation present
45	good#JJ#1-cop-be#VB#1	22	do		
705	average#JJ#1-cop-be#VB#1	26	ok		

Table 6.3: Top 3 features for each feature type with their Information Gain based rank.

Last, the 'Polarity-grammar' features also score well in terms of Information Gain. The three top features in this category would match phrases such as "good service", "big portions", and "not returning", respectively. Even the 'neutral-amod-neutral' is used in a positive context about 80% of the time and is therefore a good predictor of positive sentiment. The first and third feature are obvious predictors for positive and negative sentiment, respectively.

In terms of computing time, if we define the training time when using all features to be 100%, we find that training with 1% of the features takes about 20% of the original time, whereas employing only 0.1% of the features requires just over 1% of the original time.

6.5 Conclusion and future work

In this chapter, filtering individual features using Information Gain is shown to provide good results. With only the 1% best features in terms of Information Gain, an accuracy is obtained that is only 2.9% below the accuracy obtained when using all features. Furthermore, training the SVM with 1% of the features takes only 20% of

the time required to train it using all features. Apart from feature selection, we have shown the effectiveness of a number of relatively unknown types of features, such as 'Related-synsets' and 'Polarity-grammar'. For future work, the set of features can be expanded even further to include a comparison of grammar based features against n-gram based features. Also of interest is the context of an aspect from which we compute the sentiment score. Currently, this is determined using a simple word distance around the aspect words, but this could be done in a more advanced way, for instance using grammatical relations or even Rhetorical Structure Theory (Mann and Thompson, 1988).

Chapter 7

Aspect-Based Sentiment Analysis on the Web using Rhetorical Structure Theory*

FINE-GRAINED *sentiment analysis on the Web has received much attention in recent years. In this chapter we suggest an approach to Aspect-Based Sentiment Analysis that incorporates structural information of reviews by employing Rhetorical Structure Theory. First, a novel way of determining the context of an aspect is presented, after which a full path analysis is performed on the found context tree to determine the aspect sentiment. Comparing the proposed method to a baseline model, which does not use the discourse structure of the text and solely relies on a sentiment lexicon to assign sentiments, we find that the proposed method consistently outperforms the baseline on three different datasets.*

---

*This chapter is based on "Rowan Hoogervorst, Erik Essink, Wouter Jansen, Max van den Helder, Kim Schouten, Flavius Frasincar, and Maite Taboada. Aspect-Based Sentiment Analysis on the Web using Rhetorical Structure Theory. In *16th International Conference on Web Engineering (ICWE 2016)*, volume 9671 of *Lecture Notes in Computer Science*, pages 317-334. Springer, 2016."

7.1 Introduction

Being an integral part of most people's lives, the Web is one of the primary outlets for consumers to express their opinions on products and services they feel engaged with. This engagement can stem from the fact that they purchased a certain product or service resulting in a glowing review of that product or service, but it can also come in the form of an outcry on social media against a product or service because of some shortcoming that prevents the consumer from actually buying it. The willingness to freely share these thoughts and emotions is a driving force behind the success of review sites and review sections on e-commerce sites.

The ubiquity of reviews in e-commerce has proven to significantly affect customer decisions (Bickart and Schindler, 2001), as well as to provide a valuable marketing tool for companies (Chen and Xie, 2008). However, to get a robust overview of a certain product or service, a large number of reviews needs to be covered. This calls for an automated method that performs sentiment analysis on consumer reviews.

For more than a decade (Pang et al., 2002; Turney, 2002), many different methods have been developed that aim to automatically extract consumer sentiment from reviews. Over the years, not only has the accuracy of these methods been improved, its level of detail has also increased. Whereas the first methods computed sentiment with respect to the whole review, later methods analyzed the text at a finer level of granularity, such as at the sentence level (e.g., (Kim and Hovy, 2004)) or even sub-sentence level (e.g., (Wilson et al., 2005)).

In contrast to methods that compute a sentiment value for a given piece of text, Aspect-Based Sentiment Analysis (ABSA) aims to extract sentiment values for a set of aspects (i.e., characteristics or traits) of the product or service being reviewed (Schouten and Frasincar, 2016). Hence, ABSA considers the joint problem of finding aspects (i.e., what characteristics of the entity under review are discussed?) and computing sentiment for each found aspect (i.e., what sentiment is expressed on that particular trait?). The second task also requires that one must identify the exact part of the text that covers a certain aspect. In this research we focus on the second task only, and thus the aspects that are discussed in the text are considered given. This assumption is valid for many review sites in which the aspects of interest

are predefined and the user sentiment is gauged for these particular aspects so that products or services can be easily compared.

An interesting new method that has been introduced at the review level (Heerschop et al., 2011) and sub-sentence level (Zirn et al., 2011) is that of using discourse relations in the text to compute sentiment values. More specifically, these approaches use a parser implementing the Rhetorical Structure Theory (RST) (Mann and Thompson, 1988) to find discourse relations in the text, exploiting those to assign weights to the different discourse elements in the text. Hence, parts that are important for the discourse can be emphasized with a high weight, while parts that are less relevant can be diminished with a lower weight. The application of discourse relations can lead to significant performance improvements, as evidenced by the 15% increase in F_1-score reported in Heerschop et al. (2011). Similar results are reported in Zirn et al. (2011), where using RST at the sub-sentence level is shown to lead to considerable improvements over a Support Vector Machine baseline.

While the application of discourse analysis for sentiment analysis has been successful at both the review and sub-sentence level, its application has, to the best knowledge of the authors, not been considered for ABSA. Unfortunately, a direct extension of the existing methods to the aspect level is non-trivial due to the fact that aggregating text units is not as natural for aspects as it is for text elements. Instead, a crucial step when applying discourse analysis to ABSA is to define a context for the investigated aspect, mapping the aspect to a certain set of text elements. This work aims to extend the application of discourse analysis to the aspect level, where we focus on finding discourse relations through the application of RST. A main contribution in this respect is a novel way of defining the aspect context based on the discourse relations found through the RST analysis. Furthermore, we suggest how to incorporate this new way of defining the aspect context into a larger framework for ABSA.

The organization of this chapter is as follows. In Section 7.2, we consider RST and its current application to sentiment analysis. The main processing framework is introduced in Section 7.3, where we present our method of finding aspect context, as well as the weighting scheme and natural language processing steps involved in computing the sentiments associated to the aspects. We then discuss the actual implementation of our framework in Section 7.4. The performance of the proposed

method is evaluated in Section 7.5, and in Section 7.6 conclusions and suggestions for future work are presented.

7.2 Related Work

While many of the suggested methods for sentiment analysis have proven to be successful (Schouten and Frasincar, 2016), a possible deficiency of these traditional methods is that they do not make use of structural elements in text. As has been shown by Asher et al. (2009), considering such semantic relations in text may have a positive impact on the sentiment mining task. As a result, we have recently seen the development of different sentiment mining methods that take the discourse structure of text into account. A common characteristic of these methods is that they tend to rely on the use of RST (Mann and Thompson, 1988) for finding the discourse relations in the text.

To find discourse relations in text, an RST analysis first splits the text into clauses, also called elementary discourse units (EDUs). It then postulates relations between these EDUs, selecting from a list of predefined rhetorical relations. One can distinguish between two types of relations: mononuclear relations and multinuclear relations. In mononuclear relations, the two discourse elements have an unequal status, with one element being the more prominent 'nucleus' and the other being the supporting 'satellite'. An example of a mononuclear relation is:

> 'I bought this laptop, because it has a good processor.'

In this sentence, the clause after the comma is ancillary (and therefore the satellite) to the first clause (the nucleus), as it gives an explanation to the first part. In multinuclear relations, all elements have the same status and are considered nuclei. An example of such a multinuclear relation is:

> 'This laptop is neither fast nor does it have large storage capacity.'

Here, none of the elements is more prominent than the other and they are thus both considered nuclei.

An important property of an RST analysis is that EDUs can be aggregated to form new clauses, for which we can then again determine the discourse relation they are in. By proceeding in this way, a complete hierarchical structure of the text is obtained, which can be represented as a tree. In RST, the hierarchical structure of a text is referred to as the discourse tree.

Due to the complexity of RST methods, many researchers thought in the 1990s that discourse analysis could only be performed in combination with fully specified clauses and sentence structures (Kamp and Reyle, 1993; Lascarides et al., 1992). However, in Marcu (1997), the first discourse parser that works with unrestricted text is presented. The proposed parser automatically determines the EDUs in unrestricted text as well as the discourse relations between these EDUs, using several lexicographical rules.

Based on the successful application of RST parsers for unstructured text, RST is a natural candidate for use with online consumer reviews and RST has been applied to sentiment analysis in several different ways. One of the first works to apply RST to sentiment analysis is Taboada et al. (2008), which suggests to rank words that are contained in a satellite differently than those that are contained in a nucleus. When determining whether a word is contained in a satellite or a nucleus, only the leaf level of the discourse tree is considered. Interestingly, this relatively simple split into nuclei and satellites already leads to an improved performance of their suggested sentiment orientation calculator.

This idea is extended by Heerschop et al. (2011), where, although still functioning at the leaf-level of the discourse tree, not just a distinction between nucleus and satellite is used, but also the rhetorical relations between the discourse elements. For this analysis, the eight relations that are most frequently found in the used dataset are considered, out of the 23 rhetorical relations from Mann and Thompson (1988). One of the findings is that words that are contained in some relation may be of more importance than others, or it may be the case that a relation indicates the presence of a negation. In the given evaluation setup where sentiment is classified per sentence, applying RST leads to a 15% improvement in F_1 score compared to a baseline which does not incorporate the discourse structure of the text. Sentence-level sentiment analysis is also the focus of Yang and Cardie (2014), where discourse information is used to formulate constraints for a Conditional Random Field model. It is shown that

these discourse-based constraints are especially important in improving sentiment prediction.

In Zirn et al. (2011), RST is applied at the sub-sentence level through an application of Markov Logic. The main focus is on dividing the relations into contrasting and non-contrasting relations, as a contrasting relation may potentially negate the sentiment found in an EDU. In an experimental evaluation of the proposed method, a considerable improvement is found, compared to a baseline model without discourse information.

Another method that operates at the sub-sentence level is presented in Zhou (2013), where sentiment is predicted for each EDU. An interesting part of this research is the comparison of RST with the Penn Discourse Treebank (PDTB). The main conclusions are that RST outperforms PDTB and that methods that include discourse information when predicting sentiment for an EDU outperform the baselines that do not have access to this information.

In Lazaridou et al. (2013), sentiment is also predicted for each EDU, however, the authors present a Bayesian model that jointly models sentiment, aspects, and discourse markers. Unfortunately, the method assumes that the overall document sentiment is given, which is not the case in ABSA.

The application of discourse analysis can bring significant improvements to the analysis of sentiment. However, it has not yet been applied to sentiment analysis at the aspect level. In order for that to be possible, it is crucial to find the context of a certain aspect within the text, since the sentiment should be computed from that particular context. Next to the actual sentiment analysis method itself, the proposed method for finding this aspect context is one of the main contributions of this work.

7.3 Framework

In this section we discuss the framework used to find the sentiment with respect to some predefined aspects in online customer reviews. Fig. 7.1 shows the main steps as proposed in the framework and these steps will be elaborated on one-by-one in the coming subsections.

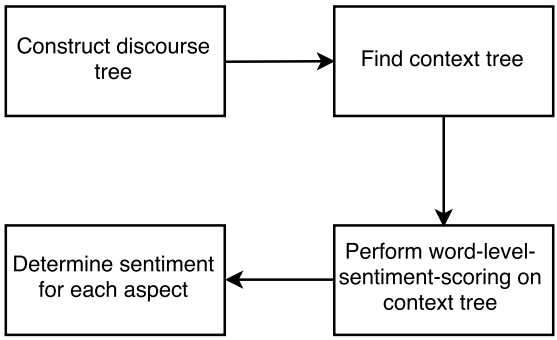

Figure 7.1: Outline of framework

7.3.1 Constructing the Discourse Tree

To incorporate the structural relations of a review into the analysis, our method relies on the application of RST to analyze the discourse structure in the review text. Two important elements of such an analysis are to determine the type of discourse parser used and the exact set of rhetorical relations considered in the analysis.

In this work, we propose the use of a document discourse parser to construct discourse trees, which has the advantage over sentence level parsers that one can also take inter-sentence relations into account. Since the context of an aspect is expected to be rather small, the use of inter-sentence relations may be advantageous considering that reviews tend to contain rather informal language in which sentences are short. An example would be:

> 'I like the speed of this laptop. But starting up is terribly slow.'

In such a sentence the actual aspect relating to the speed of the laptop is in the first sentence, but the very positive sentiment in this first sentence is actually reconsidered to some extent in the second sentence. Hence, to properly find the sentiment relating to the aspect, the inter-sentence relationship is of importance, which confirms the need for a document-level discourse parser.

Furthermore, a subset of the 23 discourse relations, as first introduced by Mann and Thompson (1988), is utilized to analyze the discourse structure of the text. For this work, we choose to use the eighteen most frequently found relations, as

```
<sentence id="1028246:1">
<text>Service was devine, oysters where a sensual as they come, and
    the price can't be beat!!!</text>
<opinions>
<opinion target="Service" category="SERVICE#GENERAL" polarity="" from
    ="0" to="7" />
<opinion target="oysters" category="FOOD#QUALITY" polarity="" from="
    20" to="27" />
<opinion target="NULL" category="RESTAURANT#PRICES" polarity="" from=
    "0" to="0" />
</opinions>
</sentence>
```

Figure 7.2: A snippet from the used dataset showing an annotated sentence from a restaurant review.

identified by Hernault et al. (2010), instead of the small subset of eight relations used by Heerschop et al. (2011), as we hypothesize that in the supervised setting the framework might still be able to find the correct impact of a relation, even when a relation is not often encountered. Moreover, we expect errors in estimating the impact values for these infrequent relations to have only a minor effect on performance.

7.3.2 Finding the Context Tree

In order to determine the sentiment associated with a certain aspect, it is important to define the context of that aspect within the text. To that end, a method is proposed that uses the found RST relations to define which parts of the text are covering which aspect. The method starts by finding the leaves in the RST tree that contain aspects. The aspects themselves are given for the data we use. An example of an annotated sentence from one of the used datasets is given in Fig. 7.2 below (including the obvious spelling mistakes, as is common in user generated content on the Web).

For each sentence, the aspects are given together with the position inside the sentence (i.e., with the `from` and `to` values). Some aspects are implicit and do not have a specific target inside the sentence (i.e., `target` is `NULL` and `from` and `to` are zero). In certain cases, including all instances of implicit aspects, the aspect is not contained in a single EDU, but its description is spread over multiple leaf nodes. In

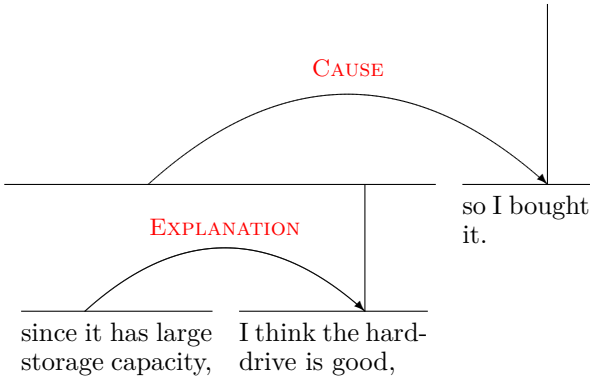

Figure 7.3: Full discourse tree of a simple review sentence (the curved lines denote a mononuclear relation between a nucleus and a satellite)

such a situation, all leaf nodes related to the aspect are considered separately and the final results are aggregated. The polarity values are empty, since that is the very thing our method predicts for each aspect.

After finding the leaf nodes that contain the aspect, we determine the relevant part of the review based on the fact that satellites tend to be complementary to nuclei. To illustrate this, consider the example:

> 'I think the hard-drive is good, since it has large storage capacity, so I bought it.'

Fig. 7.3 shows the rhetorical structure of this example. In this example the aspect sentiment related to the aspect of the overall quality of the hard-drive is contained in the EDU 'I think the hard-drive is good', but to fully determine the sentiment expressed on the hard-drive we need its corresponding satellite: 'since it has large storage capacity'.

Based on this complementary relation of satellites to nuclei, we argue that a satellite should be taken into account when determining aspect sentiment for the case where the satellite relates back to a nucleus containing the aspect. On the other hand, we argue that nuclei do not add to the understanding of the satellite. Hence, we have a natural definition of context through the asymmetry found in the relation between nucleus and satellite. Relating this back to the discourse tree, it follows that the context defined by this asymmetry is a sub-tree of the original discourse

tree. This subtree contains as root the lowest level satellite in which the aspect is contained. More specifically, for the example discussed in Fig. 7.3, this means that both the satellite and nucleus of the lower level of the tree should be considered, where this nucleus and satellite jointly correspond to the part of the sentence until the junction made by 'so'.

To find the context tree for an aspect, the algorithm looks for the lowest level satellite that contains an aspect. This is done by checking whether the leaf node containing the aspect is a satellite. If this is the case, we stop searching. Otherwise we check whether the parent node (i.e., the linked node one level higher than the leaf node in the discourse tree) is a satellite. If that is not the case, we move up to its parent node and repeat this procedure until we reach a satellite. One can easily verify that this procedure indeed returns the indicated subtree for the example considered in Fig. 7.3.

In some cases, one aspect can have multiple context trees. Consider the following example for which the discourse tree is given in Fig. 7.4.

> 'I like this laptop, because it starts up quickly and reacts immediately to my commands.'

In this example, the aspect relating to the speed of the laptop is described in the second part of the sentence 'because it starts up quickly and reacts immediately to my commands'. However, this sub-sentence consists of two leaves, and thus for both leaves a context-tree is found. In this small example, both leaf nodes will have the same context tree, since they share an immediate parent which is also a satellite. Hence the context tree for both leaves is the tree corresponding to this sub-sentence.

In other cases, the same aspect can be referred to in separate sentences. Since the RST analysis takes inter-sentence relations into account, these cases are naturally dealt with. An example of the RST analysis taking inter-sentence relations into account can be found in the following example, for which the discourse tree is given in Fig. 7.5.

> 'I was quite hopeful about the speed. However, it is truly atrociously slow, especially when starting up.'

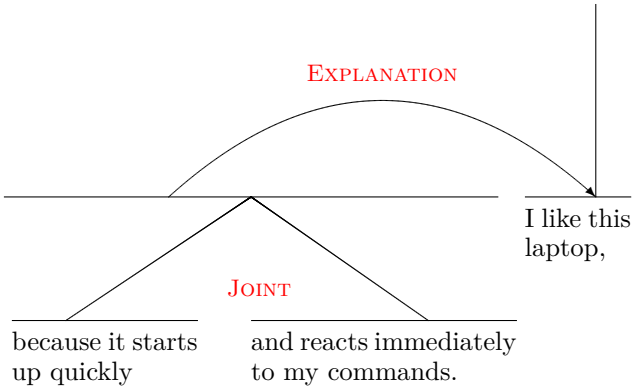

Figure 7.4: Full discourse tree for a sentence with multiple context trees (note that the straight lines of the 'Joint' relation denote the fact that this is a multinuclear relation)

The aspect 'speed' (e.g., of a laptop), is literally mentioned in the first sentence but without expressing any sentiment. In contrast to this, the second sentence expresses a strong negative sentiment on this aspect, but without literally mentioning the aspect again. The contrasting inter-sentence relation is exploited by our method to combine the sentiment from the second sentence with the aspect in the first sentence.

7.3.3 Performing Word-Level Scoring

After finding the context tree the next step is to determine the sentiment of the different EDUs contained in the context tree. Note that since EDUs get combined further up the tree, we should compute the sentiment for all leaf nodes in the context tree. To determine the sentiment for the leaves we use a sentiment lexicon-based approach that is constructed around the notion of synsets, which correspond to a set of cognitive synonyms.

In this approach all words in the considered reviews are first disambiguated since words can have different meanings in different context. To find a word's meaning, its part-of-speech (POS) and lemma are initially collected. As a lexicon can still have multiple entries for a POS of a word, the information regarding the POS of the word is then complemented by its corresponding word sense as determined by the

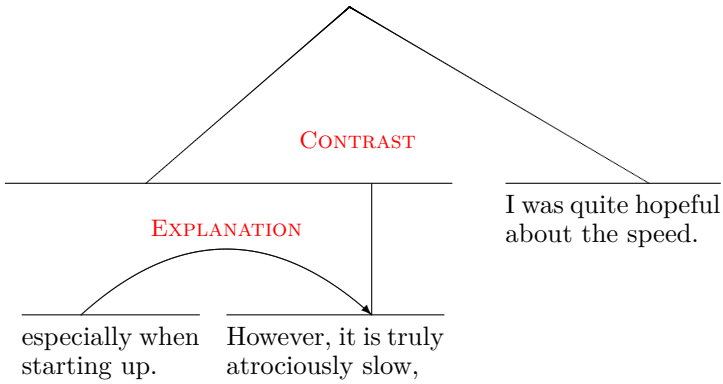

Figure 7.5: Full discourse tree showing inter-sentence relation

Lesk algorithm (Lesk, 1986). Using the POS, lemma, and word sense of a word, its sentiment score can be obtained from a sentiment lexicon.

7.3.4 Determining Aspect Sentiment

Combining the output from the previous steps leads to the calculation of the sentiment for each aspect based on its context tree and sentiment score per EDU. This can be done using either the full-path rhetorical structure model or the partial rhetorical structure model. When using the partial rhetorical structure model, only the relations at the leaf level of the discourse tree are utilized, as these are the most fine-grained. In contrast, using the full-path rhetorical structure will allow the use of the path from the root of the context tree to the leaf level. For this work, the full-path rhetorical structure model is employed, as it has been shown to better capture the expressed sentiment (Hogenboom et al., 2015).

To introduce this method more formally, we use some relevant notation. Let S be the set of leaf nodes of the context tree, $sent(s_i)$ be the sentiment score corresponding to leaf node $s_i \in S$, and P_{s_i} denote all nodes on the path from the root node of the context tree to leaf node s_i. Furthermore, let $sent(t_j)$ be the sentiment score for word $t_j \in s_i$. Last, let w_{r_n} denote the weight associated with the rhetorical role of node r_n. Then, the sentiment score of a leaf node $s_i \in S$ can be computed as

$$sent(s_i) = \sum_{t_j \in s_i} sent(t_j) \times \prod_{r_n \in P_{s_i}} w_{r_n}, \forall s_i \in S. \tag{7.1}$$

In addition to the context tree and the model described above, we also need a weighting scheme which determines for each rhetorical role r_n, the weight w_{r_n}. In our framework we apply the weighting scheme as proposed in Heerschop et al. (2011). It hypothesizes that relations only differentiate between the importance of different satellites and not of different nuclei. This is intuitively illustrated by the example as introduced in Fig. 7.3, where it seems plausible that the explanation relation as found in the lower level of the tree does not contribute to the nucleus in the leaf level of the tree. Hence, we obtain a single weight for all EDUs attached to a nucleus, whereas we consider a separate weight for each satellite relation. More formally, with R denoting the set of RST relations, the set of weights is defined as $W = \{(r, \text{Satellite}) | r \in R\} \cup \{\text{Nucleus}\}$. Here, a relation $r_n = (Nucleus, r)\ \forall r \in R$ is assigned the weight that corresponds to the relation $Nucleus$ as considered in W.

The weights are then optimized on training data using a genetic algorithm. Starting with an initial population of random weights, the fittest of these are combined in every step to produce a new set of weights. Moreover, each step mutations occur with a certain probability, to ensure variety in the population, lowering the risk of getting stuck in local optima. The chosen fitness measure is the F_1 value that is obtained when running the algorithm with the given weights.

The last step in our framework is to determine the sentiment for each aspect. To that end, the sum is taken of all the sentiment scores belonging to the context trees that we found to relate to the current aspect. A threshold ϵ is then used to categorize the sentiment score into the positive or negative class. As suggested by Heerschop et al. (2011), the threshold ϵ is set as the mean of (1) the average computed sentiment score for the aspects with positive sentiments and (2) the average computed sentiment score of the aspects with a negative sentiment, to avoid the sentiment bias in reviews. In its current form, the proposed algorithm does not predict neutral sentiment values and is limited to positive and negative only.

7.4 Implementation

Our implementation is done in the Java programming language, using the SentiWordNet (Baccianella et al., 2010; Esuli and Sebastiani, 2006) sentiment lexicon, the CLULAB (Surdeanu et al., 2015) processors for RST parsing, and Stanford CoreNLP (Manning et al., 2014) for various basic natural language processing tasks.

SentiWordNet is used to assign a sentiment score to each disambiguated word, represented as a WordNet synset. For each synset, three scores are available: objectivity, negativity, and positivity. These three scores always sum up to one, hence when knowing two scores, the third can be inferred. To arrive at a single sentiment score for each synsets, we ignore the objectivity score and subtract the negativity score from the positivity score, resulting in a real number in the $[-1, 1]$ interval (Hogenboom et al., 2015).

CLULAB is a text-level discourse parser which we use to construct a discourse tree for each review. It is primarily based on the HILDA discourse parser (Hernault et al., 2010) and it uses the same discourse relation set as the discourse parser developed by Feng and Hirst (2012). This means that CLULAB can be used to parse entire texts, whereas sentence level discourse parsers can only be used to construct discourse trees for separate sentences. To break down the input text into EDUs, also called discourse segmentation, and find the relations between them, support vector machines (SVMs) are used. To specify the different relations between EDUs, CLULAB uses a discourse relation set consisting of 18 different relations. Examples of the considered relations are attribution, condition, and cause. In addition, it specifies whether the specific relation is multinuclear or mononuclear.

7.4.1 Finding the Context Tree

After constructing the discourse trees, the algorithm presented in Algorithm 7.1 finds all leaves that contain the aspect under investigation, using as input the aspect to consider, together with the previously constructed discourse tree. The algorithm starts at the root of the discourse tree and recursively checks all the leaf nodes whether they contain the aspect or part of it. If so, that leaf node is added to the list which is the output of the algorithm.

Algorithm 7.1: *findLeaves(tree, aspect)*

input : The discourse tree and aspect under consideration
output : List with all leaves that contain the aspect

```
// Find leaves that contain aspects
if tree is leaf then
    if tree contains aspect then
        return tree;
    else
        return ∅;
    end
end
aspectLeaves ⟵ ∅;
for all children of tree do
    aspectLeaves ⟵ aspectLeaves ∪ findLeaves(child, aspect) ;
end
return aspectLeaves;
```

In the previous step we found all leaf nodes that contain the aspect, irrespective of them being a nucleus or a satellite. In the next step, the context tree of each of the selected leaf nodes is determined based on the asymmetry between nucleus and satellite. The algorithm for this task is given by Algorithm 7.2. The algorithm starts from the position of a leaf node, after which it iteratively evaluates the role of the parent node of the current node. If the parent node is a nucleus, the parent node becomes the current node and the algorithm moves up one step to evaluate its parent node. This procedure is repeated until the evaluated parent node is either a satellite or the root of the discourse tree. Then the algorithm stops and returns the tree that has that node as its root. Hence, every leaf node will get a context tree that is defined as its closest ancestor, that is a satellite, or the top ancestor (i.e., root of the discourse tree), together with all nodes that share this ancestor.

7.4.2 Performing Word-Level Scoring

The next step involves assigning scores to all the nodes that have been previously classified as leaf nodes in the context tree. First, punctuation is removed from the text in the EDUs that correspond to the leaf nodes in the context tree. Then the text

Algorithm 7.2: *defineContext(aspectLeaves)*

input : $aspectLeaves$, the set of leaves that contain the considered aspect
output : $contextTrees$, the set of all context trees for the considered aspect

```
// Find closest ancestor that is a satellite
contextTrees ⟵ ∅;
foreach leafNode ∈ aspectLeaves do
    node ⟵ leafNode;
    while hasParentNode(node) and typeOf(parentNode(node)) = Nucleus
     do
        node ⟵ parentNode(node);
    end
    // Context tree, defined by its root node, is added to set
    contextTrees ⟵ contextTrees ∪ {node}
end
```

is split into separate words. After disambiguating to get the meaning of a word, its sentiment score is retrieved from SentiWordNet. Summing up the sentiment scores yields the sentiment score for each EDU that is linked to a particular aspect.

7.4.3 Determining Aspect Sentiment

The last step of our algorithm is determining the sentiment score for each context tree. For this purpose, we sum the weighted sentiment of the leaf nodes of a context tree (the RST-based weights are here aggregated by multiplication from leaf to root). The pseudocode for this step is presented in Algorithm 7.3, where this algorithm should be called with the root node of the context tree as *node* and with a $weightNode$ equal to 0. In this algorithm we apply recursion to assign to all leaf nodes a score, which is then weighted as it returns back to the top of the tree. Here we make use of the function $getRelationWeight(node)$, which gives the weight based on the RST relation it is involved in and whether or not this node is a nucleus or satellite.

The last step is now to compare the obtained sentiment score for the aspect to the value of ϵ. If the sentiment score is smaller than ϵ, the returned sentiment for the aspect is negative, otherwise it is positive.

Algorithm 7.3: *assignScores(nodeTree, weights, weightNode)*

input : the context *tree* under consideration represented by a node, the *weights* for the RST relations, the weights so far built up in the tree *weightNode*

output : The sentiment score of the context tree

```
currentWeight ⟵ weightNode;
if node is not the root node of the context tree then
    currentWeight ⟵ currentWeight ∗ getRelationWeight(node);
end
if node is a leaf node then
    score ⟵ the sentiment for this EDU;
    return score ∗ currentWeight;
end
newScore ⟵ 0;
foreach child of node do
    newScore ⟵ newScore + assignScores(child, weights, currentWeight);
end
return newScore
```

7.5 Evaluation

This section first introduces the datasets that have been used to evaluate the performance of our proposed methods, after which a comparison of the performance of these methods is made. Additionally, this section introduces a baseline method which we benchmark our method against.

7.5.1 Data Description

In the evaluation we used three sets of reviews. All of these datasets are from the SemEval ABSA Task (Pontiki et al., 2014, 2015). We consider here the dataset on restaurant reviews of the SemEval 2014 and the datasets on restaurant and laptop reviews of the SemEval 2015. For the aspects considered for the dataset about laptops, one can think of among others the battery, the screen, and the hard-drive. For the restaurant datasets, aspects relate to for example the atmosphere, taste of the food, and level of service.

An important difference between these datasets is that the SemEval 2014 dataset consists of only single sentence reviews, while the SemEval 2015 datasets consider reviews with multiple sentences. Furthermore, the size of the datasets differs as the restaurants dataset of 2014 contains a total of 3041 reviews and a total of 3693 aspects. The dataset of laptops contains instead 277 reviews, 1739 sentences and 1974 aspects, while the dataset of restaurants of the SemEval 2015 contains a total of 254 reviews, 1315 sentences, and 1654 aspects. The datasets contain mostly positive and negative aspects, but a small number of aspects are annotated as neutral or conflicted. Since our method only predicts positive or negative, the performance with respect to these classes is not reported. Note that for the overall performance measures, these instances are considered incorrect classifications and hence are included in the computation of the performance measures.

7.5.2 Baseline Method

The acquired results are compared against a baseline model in order to evaluate the performance of the suggested method. In this case the baseline method considers a simple natural language processing approach that employs SentiWordNet and does not account for the discourse structure of the text. For each aspect we consider a fixed size context window of one to three words around the aspect. For aspects that do not have a specific target in the sentence but are implicit, we consider the whole sentence as the context window. Let the set of the words in this context for aspect i be given by c_i, then the baseline method computes the sentiment for aspect i as

$$\text{sent}(i) = \sum_{t_j \in c_i} \text{sent}(t_j). \tag{7.2}$$

where t_j are the words in the current review j in which i is an aspect. Similar to the proposed method, the sentiment score for each word is retrieved from SentiWordNet.

7.5.3 Comparison of Methods

The proposed method is evaluated on three datasets using the common 10-fold cross-validation technique. The performance of the baseline and the proposed method are

Table 7.1: Performance of baseline method on the laptops 2015 dataset

	Precision	Recall	F_1
Overall	0.31	0.31	0.31
Positive	0.39	0.34	0.36
Negative	0.24	0.32	0.27

Table 7.2: Performance of proposed method on the laptops 2015 dataset

	Precision	Recall	F_1
Overall	0.67	0.67	0.67
Positive	0.67	0.88	0.76
Negative	0.69	0.47	0.56

presented in Table 7.1 and Table 7.2 for SemEval 2015 laptop data. Since this data set does not provide any location within the sentence for aspects, the context window for the baseline is irrelevant.

These tables clearly show that the proposed method outperforms the baseline. An important observation is that the performance is somewhat lower for negative aspects than for positive ones. A possible explanation for this lower performance on negative polarities might be found in the fact that negative reviews are often written with many positive words (Pang and Lee, 2008). However, using discourse information mitigates this issue to some extent, as the proposed method is considerably less sensitive to this phenomenon than the baseline.

Table 7.3 and Table 7.4 show the results for the baseline method and the proposed method for the SemEval restaurant 2015 reviews, respectively. The results found here confirm to a large extent the observations made for the previous dataset. Again, the proposed method outperforms the baseline model.

Last, Table 7.5 and Table 7.6 show the performance of both methods on the SemEval restaurants 2014 dataset. These tables show that the performance of our method is lower than for the other two datasets. A possible reason for the decrease in performance is that this dataset only considers single sentence reviews. As we apply RST analysis at the review level, this implies that we can not use inter-sentence

Table 7.3: Performance of baseline method on the restaurants 2015 dataset for *context window* = 1 | 2 | 3, respectively

Category	Precision	Recall	F_1
Overall	0.57 | 0.54 | 0.50	0.57 | 0.54 | 0.50	0.57 | 0.54 | 0.50
Positive	0.70 | 0.69 | 0.68	0.74 | 0.67 | 0.61	0.72 | 0.68 | 0.64
Negative	0.15 | 0.16 | 0.16	0.15 | 0.20 | 0.23	0.15 | 0.18 | 0.19

Table 7.4: Performance of proposed method on the restaurants 2015 dataset

	Precision	Recall	F_1
Overall	0.74	0.74	0.74
Positive	0.80	0.86	0.83
Negative	0.52	0.47	0.49

relations in this dataset, as done for the restaurant 2015 dataset. However even in this scenario where the RST analysis cannot be used to its full potential, it is still the better option, yielding higher performance than the baseline method, both for negative and positive aspects.

7.6 Conclusion

While the application of Rhetorical Structure Theory (RST) in sentiment analysis has already been proven to obtain good performance at higher levels of text granularity, the method has not yet been explored for Aspect-Based Sentiment Analysis (ABSA). For this reason we propose a framework that uses RST for ABSA. In this framework, discourse trees are first created by using a document level discourse parser. The main contribution of this chapter is the definition of the aspect context through constructing a context tree based on the asymmetrical relation between satellites and nuclei in RST. This context tree is used for the sentiment scoring, for which we use sentiment lexicon-based word-scoring and a full-path based rhetorical structure processing scheme.

To evaluate the performance of the proposed framework a comparison between the suggested methods and a baseline model that does not incorporate the discourse

Table 7.5: Performance of baseline method on the restaurants 2014 dataset for *context window* = 1 | 2 | 3, respectively

Category	Precision	Recall	F_1
Overall	0.50 | 0.47 | 0.43	0.50 | 0.47 | 0.43	0.50 | 0.47 | 0.43
Positive	0.56 | 0.55 | 0.54	0.83 | 0.74 | 0.67	0.67 | 0.63 | 0.60
Negative	0.12 | 0.15 | 0.15	0.07 | 0.15 | 0.20	0.09 | 0.15 | 0.17

Table 7.6: Performance of proposed method on the restaurants 2014 dataset

	Precision	Recall	F_1
Overall	0.60	0.60	0.60
Positive	0.64	0.91	0.75
Negative	0.42	0.32	0.36

tree of the review is made. The comparison of the performances of the proposed method to the benchmark is made on the basis of three different datasets, comprised of laptop and restaurant reviews. We find that for all three datasets, the baseline model is clearly outperformed, but the proposed method seems susceptible to negative reviews that use many positive words.

Based on the success of using RST for ABSA as reported in this work, a next step would be to extend the proposed methodology with additional components. An interesting avenue of research would be to incorporate the found discourse structure and context tree into a classification algorithm such as a Support Vector Machine. This combines the raw power of machine learning with the finesse and detail of discourse analysis.

Chapter 8

Ontology-Enhanced Aspect-Based Sentiment Analysis*

With many people freely expressing their opinions and feelings on the Web, much research has gone into modeling and monetizing opinionated, and usually unstructured and textual, Web-based content. Aspect-based sentiment analysis aims to extract the fine-grained topics, or aspects, that people are talking about, together with the sentiment expressed on those aspects. This allows for a detailed analysis of the sentiment expressed in, for instance, product and service reviews. In this work we focus on knowledge-driven solutions that aim to complement standard machine learning methods. By encoding common domain knowledge into a knowledge repository, or ontology, we are able to exploit this information to improve classification performance for both aspect detection and aspect sentiment analysis. For aspect detection, the ontology-enhanced method needs only 20% of the training data to achieve results comparable with a standard bag-of-words approach that uses all training data.

*This chapter is based on "Kim Schouten, Flavius Frasincar and Franciska de Jong. Ontology-Enhanced Aspect-Based Sentiment Analysis. In *17th International Conference on Web Engineering (ICWE 2017)*, volume 10360 of *Lecture Notes in Computer Science*, pages 302-320. Springer, 2017."

8.1 Introduction

With many people freely expressing their opinions and feelings on the Web, much research has gone into modeling and monetizing opinionated, and usually unstructured and textual, Web-based content (Liu, 2012). A popular option to extract information from Web texts is to perform sentiment analysis. Given a certain unit of text, for instance a document or a sentence, the task of sentiment analysis is to compute the overall sentiment expressed by the author of the text. Text can be tagged for sentiment by using labels for emotions, or by assigning a polarity value to the processed unit of text, which is a more commonly adopted method. Aspect-based sentiment analysis goes a step deeper. Rather than labeling a document or a sentence, it aims to extract and tag semantic units. It captures the topics, or aspects, that are being talked about, together with the sentiment expressed about those aspects (Schouten and Frasincar, 2016). Relating the expressed sentiment directly to certain topics enables the extraction of opinions expressed in product and service reviews in a much more focused way. Instead of an overall score in which both positive and negative aspects are combined, a breakdown can now be provided, showing the aspects for which the reviewer said positive things and the aspects he or she was less enthusiastic about.

Generally speaking, we can define two sub-problems that together comprise aspect-based sentiment analysis: aspect detection and aspect sentiment classification. We define aspect detection as capturing the topics, or aspects, that are being talked about. This can be done within the textual unit of choice, for instance per sentence or per document. Most of the aspects will be explicitly mentioned in the textual unit, and the exact phrase that mentions the aspect is defined as the target expression of the aspect. Aspects without a target expression are not explicitly mentioned but rather are implied by a, usually larger, portion of text. Since target expressions are often very specific, it can be informative to group aspects together in aspect categories. Implicit aspects, even though lacking a specific target expression, can be categorized in the same manner. Even within sentences, multiple aspects, both explicit and implicit, can occur, as shown in Example 8.1. This sentence contains two explicit aspects: “chow fun” and “pork shu mai”, both belonging to the broader

'food' category, as well as an implicit aspect about sharing the table with a loud and rude family, which can be categorized under 'ambiance'.

> "Chow fun was dry; pork shu mai was more than usually greasy and had to share a table with loud and rude family."

Aspect sentiment analysis, the second sub-problem of aspect-based sentiment analysis, can be defined as determining the sentiment expressed on that sentiment in the text where the aspect is mentioned. For explicit aspects, the target expression indicates where in the text the aspect is mentioned and this information can be useful in determining the relevance of each piece of sentiment carrying text as textual units, such as sentences, can contain multiple aspects that have differing sentiment values. This is illustrated in Example 8.1 in which one sentence contains two aspects, one about the food and one about the service, but the expressed sentiment on these two aspects is completely different. For implicit aspects, where target expressions are not available, the complete textual unit can be relevant, but the aspect category usually provides some information on which part of the sentence might be relevant.

> "The food was great, if you're not put off by the rude staff."

Current approaches for aspect-based sentiment analysis rely heavily on machine learning methods because this yields top performance (Pontiki et al., 2015). Deep learning approaches are especially popular and include techniques such as word embeddings (Mikolov et al., 2013), convolutional neural networks (Kim, 2014), recursive neural tensor networks (Socher et al., 2013), and long short-term memory networks (Tai et al., 2015). While the above methods have shown very promising results in various natural language processing tasks, including sentiment analysis, there are some downsides as well. For example, while the methods learn their own features, often better than what an individual researcher could come up with, they do so at the cost of requiring much training data. While this may not be a problem for resource-rich languages, such as English, and resource-rich domains such as reviews or tweets, it is a real issue for many other languages and domains where training data are not abundant or simply unavailable.

With the previous argumentation in mind, we propose a knowledge-driven approach to complement traditional machine learning techniques. By encoding some

common domain knowledge into a knowledge repository, or ontology, we limit our dependence on training data (Cambria, 2016). The idea is that, compared to using only information from the text itself, relating the text to the concepts described in a knowledge repository will lead to stronger signals for the detection of aspects as well as the prediction of sentiment. Consequently, having stronger signals limits the amount of necessary training data, as the relation between input and desired output is easier to discover.

While knowledge repositories, such as ontologies, have to be adjusted for every domain and language, this is a less resource-intensive task than manually annotating a big enough corpus for the training of a deep neural network. Furthermore, since ontologies are designed to be reused, for example within the Linked Open Data cloud, it is easy to share knowledge.Last, with ontologies, being logically sound structures, it is possible to reason with the available data, and to arrive at facts not directly encoded in the ontology. For example, if meat is a type of food, and food is edible, we can state that meat is edible, even without directly specifying this. Furthermore, inferencing functionality can help to disambiguate sentiment carrying phrases given the context they appear in, for example by taking into account that "cold" is positive for "beer", but negative for "pizza". This opens up some exciting possibilities when performing sentiment analysis.

This chapter is structured as follows. In the next section, some of the related work is presented, followed by a discussion of the problem and the used data set in Sect. 8.3. In Sect. 8.4, an overview of the proposed method is given, and in Sect. 8.5, its performance is compared and evaluated. The chapter concludes with Sect. 8.6, providing both conclusions and possible avenues for future work.

8.2 Related Work

In Cambria (2016), a short overview is given of the field of affective computing and sentiment analysis, and the author makes a case for the further development of hybrid approaches that combine statistical methods with knowledge-based approaches. The leading idea in that field is that the intuitive nature and explanatory power of

knowledge-based systems should be combined with the high performance of machine learning methods, which also forms the research hypothesis of our current work.

Sentic computing is presented in Cambria and Hussain (2015), which combines statistical methods with a set of linguistic patterns based on SenticNet (Cambria et al., 2014). Each sentence is processed in order to find the concepts expressed in it. The discovered concepts are linked to the SenticNet knowledge repository, which enables the inference of the sentiment value associated to the sentence. If there are no concepts expressed in this sentence or if the found concepts are not in the knowledge base, then a deep learning method that only uses the bag of words is employed to determine the sentiment for that sentence. Note that this is a sentence level approach and not an aspect based approach.

In Dragoni et al. (2014), a multi-domain approach to sentence-level sentiment analysis is presented, where the task is to assign sentiment to each sentence, but where the sentences could come from a variety of domains. The proposed method is designed in such a way that sentiment words can be disambiguated based on the recognized domain the sentence originates from. Similar to what is typical of our approach, the used knowledge graph is split into two main parts: a semantic part in which the targets are modeled, and a sentiment part in which the links between concepts and sentiment are described. A big difference with our approach is that while we opt for a focused domain ontology, in Dragoni et al. (2014), due to the multi-domain nature of the problem, a very broad knowledge graph is created that combines several resources such as WordNet (Fellbaum, 1998) and SenticNet (Cambria et al., 2014). Another difference is the use of fuzzy membership functions to describe the relations between concepts and domains, as well as between concepts and sentiment. This gives more flexibility in terms of modeling, but makes it harder to reason over the knowledge graph.

In Reforgiate Recupero et al. (2015), an extended version of the Sentilo framework is presented that is able to extract opinions, and for each opinion its holder, expressed sentiment, topic, and subtopic. The framework converts natural language to a logical form which in turn is translated to RDF that is compliant with Semantic Web and Linked Data design principles. Then, concepts with relations that signal sentiment are identified. The sentiment words receive a sentiment score based on SentiWordNet (Esuli and Sebastiani, 2006) and SenticNet (Cambria et al., 2014).

A typical problem for which external knowledge can be very useful is the issue of sentiment ambiguity: a certain expression can be positive in one context and negative in the other (e.g., the cold pizza and cold beer example from the previous section). This problem is tackled in Xia et al. (2015) by means of a Bayesian model that uses the context around a sentiment carrying word, in particular the words that denote the aspect, to determine the polarity of the sentiment word. When there is not enough information in the context to make the decision, a backup method is to retrieve inter-opinion data, meaning that if the previous opinion was positive and there is a conjunction between that one and the current one, it is very likely that the current opinion is positive too.

In contrast to the previous approaches, high performing SemEval submissions on the aspect-based sentiment analysis task (Pontiki et al., 2015), are typically limited in their use of external knowledge or reasoning. For instance, in the top performing system for aspect category classification (Toh and Su, 2015), a set of binary classifiers is trained, one for each aspect category, using a sigmoidal feedforward network. It uses words, bigrams of words, lists of opinion target words extracted from the training data, syntactic head words, and Brown word clusters as well as k-mean clusters from word2vec (Mikolov et al., 2013). The highest performing system in the sentiment classification task (Saias, 2015) also exclusively focuses on using lexical features. Besides the information that can be directly extracted from the text, a number of lexical resources such as sentiment lexicons were used to detect the presence of negation and sentiment words. While lexical resources can be seen as being external knowledge, they are limited in functionality and do not, for example, support reasoning.

8.3 Specification of Data and Tasks

The data set used in this research is the widely known set of restaurant reviews from SemEval (Pontiki et al., 2015), with each review containing one or more sentences: it contains 254 reviews with in total 1315 sentences. Each sentence is annotated with zero, one, or multiple aspects, and each aspect is put into a predefined aspect category and is labeled as either positive, neutral, or negative. For explicit aspects, the target expression is also provided. Some statistics related to aspects and sentiment can

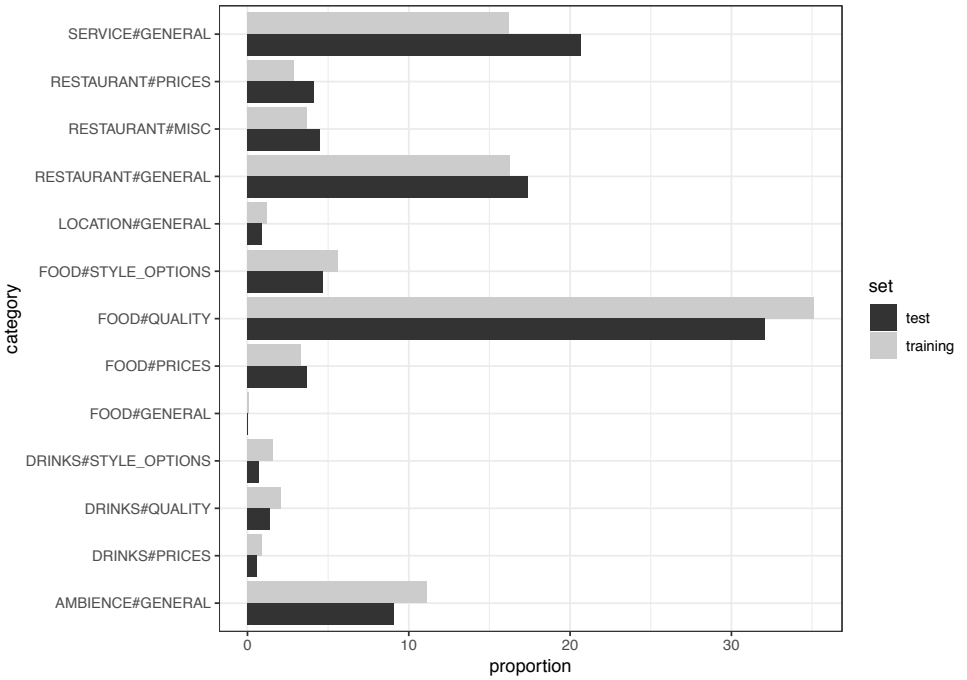

Figure 8.1: Relative frequencies of each aspect category label

be found in the figures below. In Fig. 8.1, the number of times each category label appears is presented and in Fig. 8.2, the proportion of sentences with multiple aspects is shown in which not all aspects have the same sentiment label. This is related to Fig. 8.3, which shows the distribution of aspects per sentence. Fig. 8.4 presents the distribution of sentiment values over aspects, showing that this data set, especially the training data set, is unbalanced with respect to sentiment.

A snippet of the used data set is shown in Fig. 8.5. The data set is already organized by review and by sentence, and each sentence is annotated with zero or more opinions, which represent the combination of an aspect and the sentiment expressed on that aspect.

For the aspect detection task, only the sentence is given. The task is to annotate the sentence with aspects but the `polarity` field can be left empty. While some

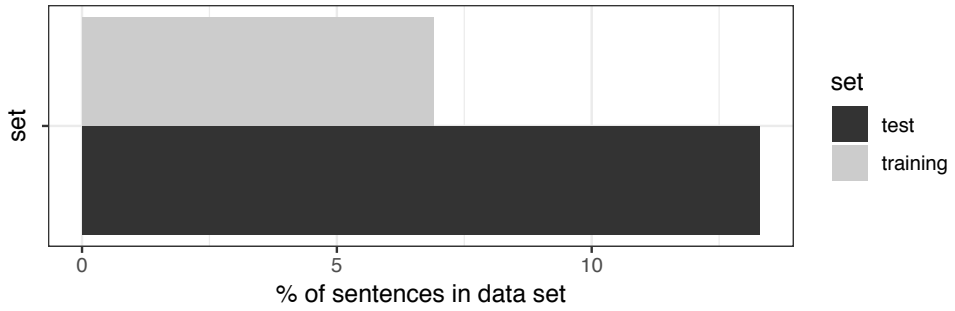

Figure 8.2: The proportion of sentences with multiple aspects in which not all have the same sentiment value

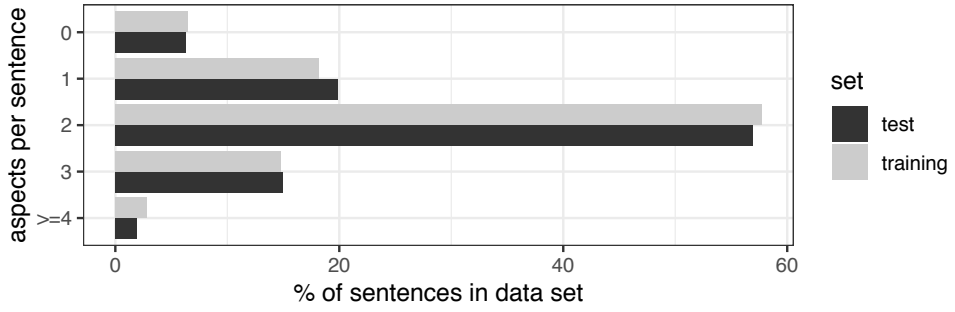

Figure 8.3: The distribution of aspects per sentence

variations of the task exist, we limit ourselves to predicting only the `category` field of each aspect. Hence, the `target` field and corresponding `to` and `from` fields are ignored. The category labels themselves consist of an entity and an attribute that are separated by a hash symbol. In this work, however, we regard the category as just a single label. In the evaluation, every category that is in the data and is also predicted is a true positive, every category that is predicted but is not in the data is a false positive and every category that is not predicted, even though it is in the data, is a false negative. From the number of positives and negatives, the standard evaluation metrics of precision, recall, and F_1 score can be computed.

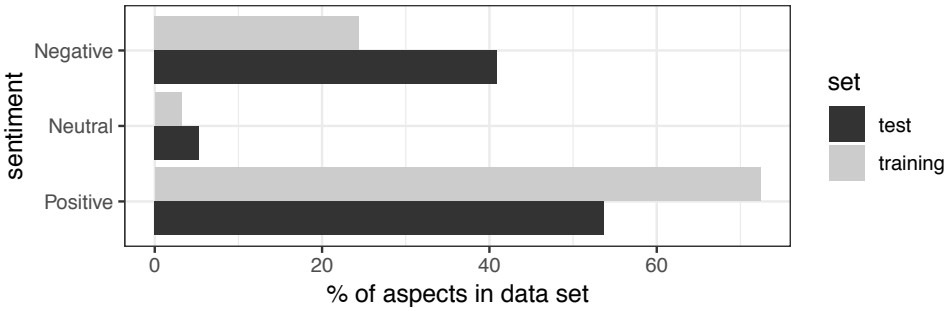

Figure 8.4: Relative frequencies of each sentiment label

```
<sentence id="1032695:1">
<text>Everything is always cooked to perfection, the service is
    excellent, the decor cool and understated.</text>
<Opinions>
<Opinion target="NULL" category="FOOD#QUALITY" polarity="positive"
    from="0" to="0"/>
<Opinion target="service" category="SERVICE#GENERAL" polarity="
    positive" from="47" to="54"/>
<Opinion target="decor" category="AMBIENCE#GENERAL" polarity="
    positive" from="73" to="78"/>
</Opinions>
</sentence>
```

Figure 8.5: A snippet from the used dataset showing an annotated sentence from a restaurant review.

For the sentiment classification task, the sentence with the aspects are given. Thus, we get everything in Fig. 8.5, except for the values of the `polarity` fields. Every correctly predicted polarity is a true positive and every incorrectly predicted polarity is both a false positive and a false negative, so precision, recall, and F_1 have the same value for this task.

8.4 Method

Since both detecting aspects and determining their sentiment can be seen as a classification task, we choose an existing classifier to work with. In this case, we choose to

use a linear Support Vector Machine (SVM), since it has shown good performance in the text classification domain, with relatively many input features compared to the number of training examples (Chang and Lin, 2011). For aspect detection, we train an independent binary classifier for each different aspect category. In this way, per sentence, each classifier will determine whether that aspect is present or not, enabling us to find zero, one, or more aspects per sentence. For sentiment classification, we train only one (multiclass) model, that is able to predict one of three outcomes: positive, neutral, or negative. We use the libsvm (Chang and Lin, 2011) implementation of the SVM classifier.

Using natural language processing (NLP) techniques, we gather information from the review texts that will comprise the input vector for the SVM. In Figure 8.6, the components of the NLP pipeline are shown. First, an automated spelling correction is performed, based on the JLanguageTool library (JLanguageTool, 2016). Given the nature of consumer-written reviews, this is a very helpful step. Next, the text is split into tokens, which are usually words, but also includes punctuation. Then, these tokens are combined into sentences. With sentence boundaries defined, the words in each sentence can be tagged with Part-of-Speech tags, which denote the word types (e.g., noun, verb, adjective, etc.). Once that is known, words can be lemmatized, which means extracting the dictionary form of a word (e.g., reducing plurals to singulars). The syntactic analysis then finds the grammatical relations that exist between the words in a sentence (e.g., subject, object, adjective modifier, etc.). All these components are provided by the Stanford CoreNLP package (Manning et al., 2014). The last step connects each word with a specific meaning called a synset (i.e., set of synonyms), given the context in which it appears, using a Lesk (Lesk, 1986) variant and WordNet semantic lexicon (Fellbaum, 1998). This particular version of the Lesk algorithm is provided by DTU (Jensen and Boss, 2008).

8.4.1 Ontology Design

The ontology consists of three main parts, modeled as top-level classes. The first is a *Sentiment* class, which has individuals representing the various values of sentiment. In our case, that is only positive, neutral, and negative, but one can imagine a more fine grained classification using basic emotion classes like anger, joy, etc. The

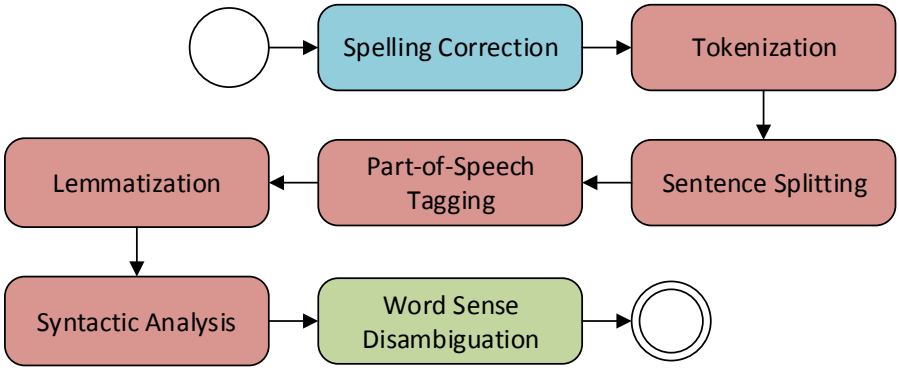

Figure 8.6: The NLP pipeline used at the basis of the methods

second major class is *Target*, which is the representation of an aspect. The higher level concepts correspond to aspect categories, while the subclasses are often target expressions of an aspect. Subclasses of *Target* are domain specific, and for our restaurant domain we use *Ambience, Sustenance, Service, Restaurant, Price, Persons*, and *Quality*. Some of these have only a handful of individuals, such as *Quality* since quality is more expressed in evaluative words and not in concepts, while *Sustenance*, unsurprisingly, has many subclasses and individuals. Because we want to use object relations, all subclasses of Target are modeled as having both the class role and the individual role, much like classes in OWL FULL. For every subclass of *Target*, there is an individual with the same (resource) identifier that is of that same type. Hence, there is a subclass *Beer*, and an individual *Beer* that is of type *Beer*. This duality allows us to use the powerful subclass relation and corresponding reasoning, as well as descriptive object relations. The latter are mainly used for the third part of the ontology, which is the *SentimentExpression* class.

The *SentimentExpression* class only has individuals describing various expressions of sentiment that can be encountered. Each sentiment expression is linked to a *Sentiment* value by means of an object relation called *hasSentiment*, and to a *Target*

with an object relation called *hasTarget*. In most cases, the *hasTarget* relation points to the top-level concept *Target*, since the word "good" is positive regardless of the target. However, the word "cold", when linked to the concept *Pizza* has the *negative* sentiment value, while it has the *positive* value when linked to *Beer*.

The ontology is lexicalized by means of a data property that is added to each concept. The targets have a *targetLexicalization* property, and the sentiment expressions have a *sentimentLexicalization* property. By means of these lexicalizations, which can be one or more words, the concepts in the ontology can be linked to words or phrases in the text.

In Figure 8.7, the sentiment expression for "cold beer" is shown with its related concepts. Note that the ellipse around the *Beer* class and the *Beer* individual denotes the fact that those are two roles of the same concept.

This ontology design allows us to perform two types of reasoning, one for aspect detection and one for sentiment classification. The first is that if we encounter a sentiment word, we know that its target is also in this sentence. For example, when we find the word "delicious", we will find the SentimentExpression with the same *sentimentLexicalization*. This concept has a target, namely the *Sustenance* concept. Because of this, we know that the sentence where we find "delicious", the target aspect is something related to food or drinks. The second type of reasoning is that when we encounter a sentiment word in a sentence and that word is linked to a *SentimentExpression* in the ontology, the aspect for which we want to determine the sentiment has to be of the same type as the target of that *SentimentExpression* in order for its sentiment value to be relevant. For example, we again find the word "delicious" in the text, but we want to determine the sentiment for the aspect `FOOD#PRICE`, we should not take the positive value of "delicious" into account, since it is not relevant to the current aspect we are classifying. This is especially useful when a sentence has more than one aspect.

The ontology is created manually, using the OntoClean methodolody (Guarino and Welty, 2002), so it is guaranteed to fit with the restaurant domain of the reviews. To keep the ontology manageable, we have deliberately opted for a relatively small, but focused, ontology. As such, it contains 56 sentiment expressions, 185 target concepts, and two sentiment concepts: positive and negative. The maximum depth of its class hierarchy is 5.

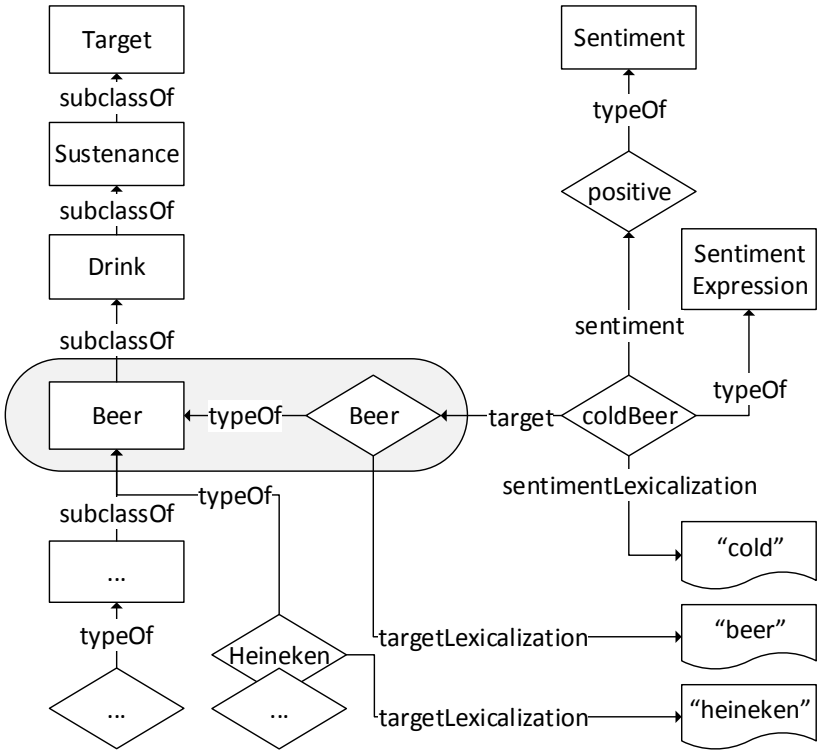

Figure 8.7: Snippet of the ontology showing a sentiment expression and its related concepts

8.4.2 Features

Since the aspect detection task is defined as predicting zero or more aspect category labels per sentence, we extract the following features from each sentence: the presence or absence of lemmatized words, the presence or absence of WordNet synsets, and the presence or absence of ontology concepts. For the latter, we use words and phrases in the sentence to find individuals of top-level class *Target* that have a matching *targetLexicalization*. When a concept is found, we include all its types as features. For example, when we find the concept *Steak*, we also include the concepts *Meat*,

Food, *Sustenance*, and *Target*. Furthermore, when a word or phrase matches with a *sentimentLexicalization*, we include the *target* of that SentimentExpression as being present as well. All these features are binary, so the input vector for the SVM will contain 1 for features that are present, and 0 otherwise. The same features are used for each binary aspect classifier

This process of gathering features can be formalized as follows.

$$L_W = \{l | l = lemma(w), w \in W\} \tag{8.1}$$

$$Z_W = \{z | z = synset(w), w \in W\} \tag{8.2}$$

$$\begin{aligned} C_W = \{c | k : c, (k, lemma(w)) : targetLexicalization, w \in W\} \cup \\ \{c | (k, c) : target, (k, lemma(w)) : sentimentLexicalization, w \in W\} \end{aligned} \tag{8.3}$$

where W represents a set of words, given as a parameter, $i : c$ represents an individual k of type c, and $(k, c) : target$ represents that k is related to c through relation type $target$. Then, let W' be the set of all words in the data set. Every word has its own unique representation in this set, so the same word appearing in three different places will have three entries in this set. And let S be the indexed set of all sentences in the data set, with functions $g : I \to S$, and $g' : S \to I$, so that $i \to s_i$, $s \to i_s$ and $I = \{i \in \mathbb{N} | i \geq 0, i < |S|\}$, resulting in a unique one-to-one mapping between I and S. Then W_i is defined as the set of words in sentence s_i.

Using W', we gather all possible features from the full data set into set $F_{W'}$.

$$F_{W'} = L_{W'} \cup Z_{W'} \cup C_{W'} \tag{8.4}$$

Similar to S, set $F_{W'}$ is indexed with a one-to-one mapping: $h : F_{W'} \to J$ and $h' : J \to F_{W'}$, so that $j \to f_j$ and $f \to j_f$ with $J = \{j \in \mathbb{N} | j \geq 0, j < |F_{W'}|\}$.

Given this mapping between J and F_{W_i}, the index numbers of only the features that are present in a given sentence i are retrieved through $h(F_{W_i})$. This leads to

defining the input matrix $\mathbf{X}$ as having

$$x_{ij} = \begin{cases} 1, \text{if } j \in h(F_{W_i}) \\ 0, \text{otherwise} \end{cases} \tag{8.5}$$

Where i specifies the row in the matrix, representing the current sentence and j specifies the column in the matrix, representing the current feature.

For sentiment classification, the process of gathering features is similar, so due to the page limit, the formalization is omitted here. The difference is that the scope here is a single aspect for which we want to determine the sentiment. An aspect already has the category information given, together with its position in the sentence, if applicable. Besides the features for lemmas, synsets, and ontology concepts, we also include the aspect category information of an aspect as a feature (e.g., `FOOD#QUALITY` or `FOOD#PRICE`. In addition, we include some sentiment information, using existing sentiment tools or dictionaries together with our own ontology. Utilizing the Stanford Sentiment tool (Socher et al., 2013), which assigns sentiment scores (decimals between -1 and 1) to every phrase in the parse tree of a sentence, we add a feature that represents the sentiment of the whole sentence, as well as a feature that represents the sentiment of the smallest phrase containing the whole aspect. The latter is only available for explicit aspects, whereas the former is always available. Since the sentence sentiment score is additional information that can be useful, for instance when the aspect sentiment, as determined by this tool, is incorrect, we chose to always add this feature.

Since the lowest level of the parse tree is comprised of the words in a sentence, we use the same tool to get sentiment values for each word. A special review sentiment dictionary (Kiritchenko et al., 2014) is used to retrieve sentiment values for some of the words as well, and as a third source of sentiment information we use the ontology to find sentiment information for any word that can be linked to a *SentimentExpression* in the ontology. As explained in the previous section, we only take the latter into account when the aspect for which we want to determine the sentiment can be linked to a concept in the ontology that matches with the target concept of the sentiment expression. The *positive* concept is translated to a value of 1, and the *negative* concept is translated to a value of -1. All these sentiment values are averaged to arrive

at a single sentiment value for a given word or phrase. However, when we do find an applicable sentiment expression in the ontology, preliminary experiments suggest to use double the weight for this value in the average computation. Assigning a higher weight is an intuitive course of action, since we are sure this is relevant information.

Some of the aspects have location information provided, so we know which words in the sentence are describing that particular aspect. When this information is available, we construct a scope around the aspect so words that are closer to the aspect are more valuable than words further away. The distance is measured in terms of grammatical steps between words. In this way, words that are grammatically related are close by, even though there might be many words in between them in the actual sentence. Based on some preliminary experiments, we compute the distance correction value as:

$$distanceCorrection = \max(0.1, 3 - grammaticalSteps) \tag{8.6}$$

Instead of having binary features (cf. Eq. 8.5), denoting just the presence of features, we multiply the distance correction value with the average sentiment value and use this as the weight for each word-based feature. The aspect category features, as well as the sentence sentiment and aspect sentiment features are not affected by this because distance and sentiment are irrelevant seeing that these features are not directly linked to specific words.

8.5 Evaluation

In this section, we will evaluate the proposed method and discuss the results. First, the used data sets are described, followed by a comparative overview of the performance of the method. Then, to test our hypothesis that less data is needed for knowledge-driven approaches, a series of experiments is performed with varying amounts of training data. This is followed by a feature analysis, showing which features are useful and demonstrating that the output of the algorithm can be explained.

Table 8.1: The performance on the aspect detection task

	avg. F_1	st.dev.	p-values of two-sided t-test base	+S	+O	in-sample F_1 (training data)	out-of-sample F_1 (test data)
base	0.5749	0.0057	-	-	-	0.803	0.5392
+S	0.6317	0.0039	<0.0001	-	-	0.896	0.5728
+O	0.6870	0.0026	<0.0001	<0.0001	-	0.858	0.6125
+SO	0.6981	0.004	<0.0001	<0.0001	<0.0001	0.920	0.6281

Table 8.2: The performance on the aspect sentiment classification task

	avg. F_1	st.dev.	p-values of two-sided t-test base	+S	+O	in-sample F_1 (training data)	out-of-sample F_1 (test data)
base	0.7823	0.0079	-	-	-	0.831	0.7372
+S	0.7862	0.0049	0.0294	-	-	0.847	0.7349
+O	0.7958	0.0069	0.0002	0.0008	-	0.863	0.7479
+SO	0.7995	0.0063	<0.0001	<0.0001	0.0029	0.884	0.7527

8.5.1 Performance

To test the performance of the proposed method, we compare the full ontology-enhanced method (+SO) against more basic versions of the same algorithm, having the exact same setup except for some missing features: a version without synsets, but with ontology features (+O); a version with synsets but without ontology features (+S); and a version without both synsets and ontology features (base). The two tasks of the algorithm are tested separately, so the aspect sentiment classification is performed with the gold input from the aspect detection task. This prevents errors in aspect detection from influencing the performance analysis of the sentiment classification. The two performances on the two tasks are given in Table 8.1 and Table 8.2, respectively. The reported F_1 scores are averages over 10 runs where each run is using 10-fold cross-validation with randomly assigned folds. The standard deviation is also reported, together with the p-values of the two-sided t-test.

For aspect detection the picture is most clear, showing that every step towards including more semantic features, starting with synsets and going to ontology concepts, is a significant improvement over not including those features. Note that most of the improvement with respect to the base algorithm comes from the ontology. The

Table 8.3: Ranks of the proposed methods in top of SemEval-2015 ranking

Aspect Detection		Aspect Sentiment Classification	
Team	Performance	Team	Performance
+SO	0.628	sentiue	0.787
NLANGP	0.627	ECNU	0.781
NLANGP	0.619	Isislif	0.755
UMDuluth-CS8761-12	0.572	**+SO**	0.752
UMDuluthTeamGrean	0.572	LT3	0.750
SIEL	0.571	UFRGS	0.717
sentiue	0.541	wnlp	0.714

synsets, while showing a solid improvement on their own, are able to increase the performance with much less when the ontology is also used (i.e., going from +O to +SO).

For aspect sentiment classification, the results are less pronounced than for aspect detection, but it still shows the same overall picture. Adding more semantic features improves the performance, and while the improvement is less, it is statistically significant. A key observation here is that for sentiment analysis, we already employ a number of features that convey sentiment values, and as such, it is a strong signal that in spite of all that information, the ontology is still able to boost the performance.

This work is mainly focused on showing how ontologies have an added value for aspect-based sentiment analysis. Nevertheless, our methods show a competitive performance. In Table 8.3, an overview of the top performances of SemEval submissions on the same task are given (Pontiki et al., 2015). These methods have been tested on the exact same test data, so their reported F_1 scores are directly comparable. For ease of reference, our +SO method is shown in bold, together with the top 6 out of 15 submissions for both tasks.

8.5.2 Data Size Sensitivity

Since we hypothesize that an ontology-enhanced algorithm needs less data to operate than a traditional machine learning method, we perform the following experiment.

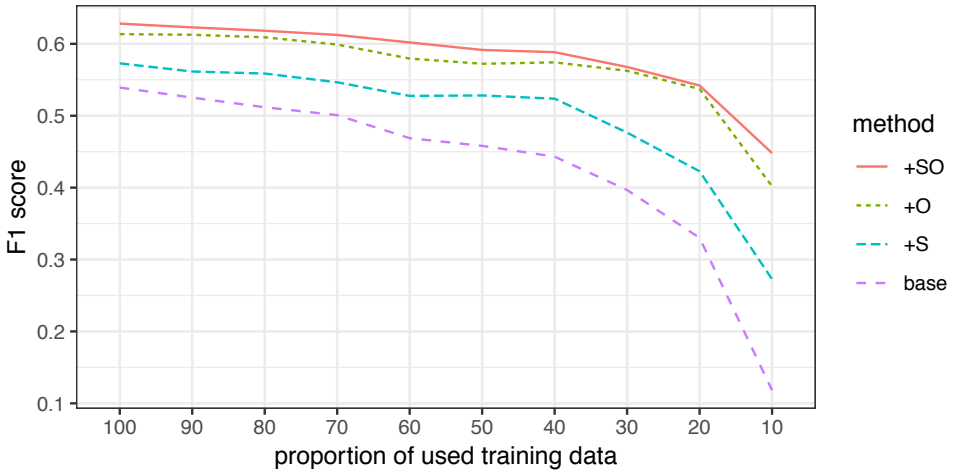

Figure 8.8: The data size sensitivity for the aspect detection task

Taking a fixed portion of the data as test data, we train on an ever decreasing part of the total available training data. In this way, the test data remains the same, so results can be easily compared, while the size of the training data varies. This maps the sensitivity of the algorithms to training data size and the results are shown in Fig. 8.8 for the aspect detection task and in Fig. 8.9 for the aspect sentiment classification task.

When looking at the sensitivity of the aspect detection method, we can see that the base algorithm is quite sensitive to the size of the data set, dropping the fastest in performance when there are fewer training data. With synsets, the performance is more stable, but with less than 40% of the original training data, performance drops significantly. The two versions that include ontology features are clearly the most robust when dealing with limited training data. Even at 20% of the original training data, the performance drop is less than 10%.

For sentiment classification, it shows that all methods are quite robust with respect to the amount of training data. This might be because all variants are using external information in the form of the sentiment dictionaries, already alleviating the need for training data to a certain extent. The gap between ontology-enhanced

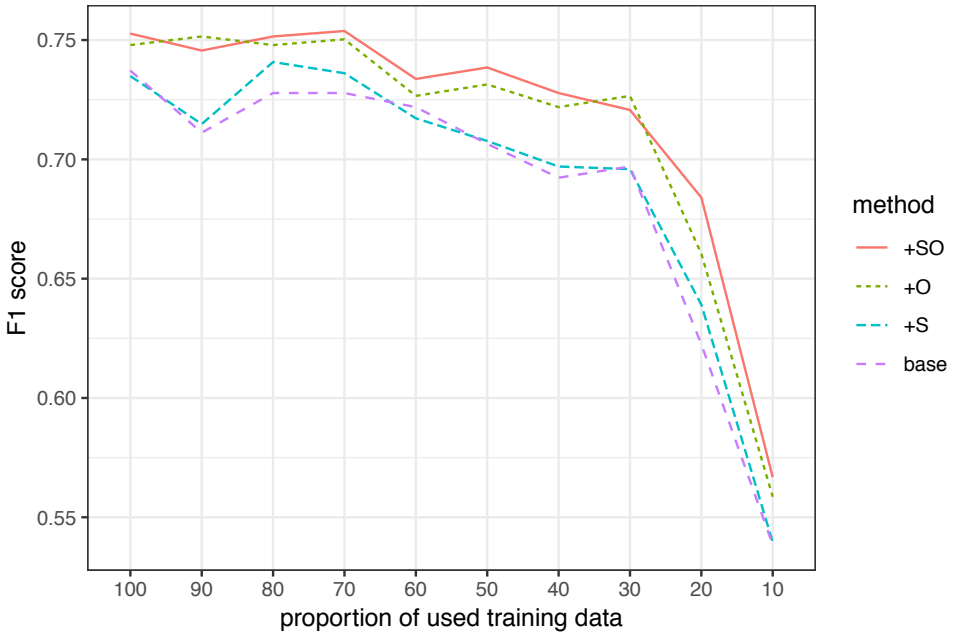

Figure 8.9: The data size sensitivity for the aspect sentiment classification task (note that the y-axis does not start at 0 to improve readability)

methods and the ones without ontology features does not widen, so contrary to our hypothesis, ontology features do not reduce the required number of training data for aspect sentiment analysis. On the other hand, the ontology-enhanced methods consistently outperform the other two methods, so the ontology features stay relevant even at smaller numbers of training data.

8.5.3 Feature Analysis

To investigate whether the ontology concepts are indeed useful for the SVM model, we take a look under the hood by investigating the internal weights assigned by the SVM to each feature. Since we use a binary classifier for each aspect category label, there are too many trained models to show these weights for all of them. Hence, we will only look at the top-weighted features for some of the more illustrative ones.

Table 8.4: Top 4 features for `DRINKS#STYLE_OPTIONS` according to weight assigned by SVM classifier

1.	0.369 -	Ontology Concept: Menu	3.	0.307 -	Synset: list
2.	0.356 -	Ontology Concept: Drink	4.	0.265 -	Synset: enough

Table 8.5: Top 10 features for `DRINKS#PRICES` according to weight assigned by SVM classifier

1.	0.428 -	Ontology Concept: Price	6.	0.180 -	Lemma: price
2.	0.303 -	Ontology Concept: Drink	7.	0.176 -	Synset: wine
3.	0.232 -	Synset: drink	8.	0.175 -	Lemma: wine
4.	0.204 -	Lemma: drink	9.	0.165 -	Ontology Concept: Wine
5.	0.184 -	Lemma: at	10.	0.157 -	Synset: value

A very nice example is the trained SVM for the `DRINKS#STYLE_OPTIONS` category, which deals with the variety of the available drinks. The four most important features are listed in Table 8.4.

Clearly, the top two features perfectly describe this category, but the synsets for "list" and "enough", as in "enough options on the list", are also very appropriate. Another good example is `DRINKS#PRICES`, for which the top 10 features are shown in Table 8.5. Here, again, the top two concepts are typical of this aspect category. However, we see that certain lemmas and synsets are also high on the list, but with a lower weight. Note the word "at" which is often used to denote a price, but which, being a function word, is not associated with a synset or an ontology concept.

An example where the ontology could make less of a difference is the category `RESTAU- RANT#MISC`, where, due to the miscellaneous nature of this aspect, no ontology concepts were applicable. Another category with interesting weights is `FOOD#QUALITY`, where next to the obvious ontology concept *Food*, a lot of adjectives were useful for the SVM since they convey the quality aspect. The fact that people write about quality is demonstrated by the strong use of sentiment words, such as "amazing" and "authentic". Hence, it is rather difficult to define an ontology concept *Quality*, and while this concept is present in the ontology, it is not very useful, being ranked the 44th most useful feature here.

Table 8.6: Top 10 features for `RESTAURANT#GENERAL` according to weight assigned by SVM classifier

1.	0.636 -	Lemma: worth	6.	0.402 -	Lemma: up
2.	0.505 -	Lemma: love	7.	0.399 -	Lemma: again
3.	0.487 -	Lemma: back	8.	0.368 -	Lemma: overall
4.	0.419 -	Lemma: wrong	9.	0.360 -	Lemma: favorite
5.	0.406 -	Lemma: return	10.	0.355 -	Lemma: recommend

Looking at the `RESTAURANT#GENERAL` category, we find that some concepts are really missing from the ontology, namely the idea of someone liking a restaurant. This is often expressed as wanting to go back to that place, recommending it to others, or that it is worth a visit or worth the money. The top 10 features for this category are listed in Table 8.6 below to illustrate this.

At a first glance, the word "wrong" looks a bit out of place here, but at closer inspection of the data, it seems this word is often used in the phrase "you can't go wrong ...", which is indeed a positive remark about a restaurant in general.

For sentiment classification, a clear feature analysis is not very feasible, since the features are not binary, but are weighted according to distance and sentiment dictionary values, when applicable. Furthermore, the SVM model is trained on three sentiment values, which means that internally, a 1-vs-all strategy is performed, so the weights would be a combination of three trained models and not so descriptive anymore.

8.6 Conclusion

In this chapter we presented a method for aspect-based sentiment analysis that utilizes domain ontology information, both for aspect detection and for sentiment analysis of these aspects. The performance of the SVM classifier is improved with this additional knowledge, although the improvement for aspect detection is more pronounced.

For aspect detection, it can indeed be concluded that fewer training data are needed, because performance drops much less when fewer training data are available

compared to the same method without ontology features. This is not the case for the sentiment analysis method applied, where due to the fact that all methods use external resources in the form of sentiment dictionaries, the added value of the ontology remains limited. However, the ontology-enhanced method keeps outperforming the basic methods on the sentiment analysis task with about the same difference, even at smaller amounts of available training data.

When interpreting the internal weights assigned by the SVM, we see that for aspect detection the ontology features are in most cases the most informative. Only when the aspect categories themselves are not clearly defined (e.g., `RESTAURANT#MISC`), or when that category is not described in the ontology (e.g., `RESTAURANT#GENERAL`), do we see the SVM using mostly word-based features.

This leads us to conclude that ontology concepts are useful, especially for aspect detection, but also for sentiment analysis, and that ontology-enhanced aspect-based sentiment analysis is a promising direction for future research.

In terms of future work, we would suggest expanding the ontology by including more domain-specific sentiment expressions. Given the fact that there are more than three times as many target concepts in the current ontology than sentiment expressions, focusing as much on sentiment expressions could lead to a more pronounced increase in performance on the sentiment analysis task as well. Furthermore, this process could be automated, scraping the information from the Web, where this type of data is readily available. Linking the ontology to others in the Linked Open Data cloud is also a direction that could be further explored.

While we propose methods for both aspect detection and sentiment analysis, there is still a subtask that is not yet covered, which is determining the target of an aspect within the sentence. Even though we use the target location information for the sentiment analysis task, we currently do not determine this information, predicting only the aspect category label. To create a complete method that can be deployed on real-life data, this missing link will need to be dealt with.

Chapter 9

Semi-Automatic Ontology Creation for Aspect-Based Sentiment Analysis*

With so much opinionated, but unstructured, data available on the Web, sentiment analysis has become popular with both companies and researchers. Aspect-based sentiment analysis goes one step further by relating the expressed sentiment in a text to the topic, or aspect, the sentiment is expressed on. This enables a detailed analysis of the sentiment expressed in, for example, reviews of products or services. In this chapter we propose a knowledge-driven approach to aspect sentiment analysis that complements traditional machine learning methods. By utilizing common domain knowledge, as encoded in an ontology, we improve the sentiment analysis of a given aspect. The domain knowledge is extracted from unlabeled reviews in a semi-automatic fashion and used to determine which words are expressing sentiment on the given aspect as well as to disambiguate sentiment carrying words or phrases. The proposed method has a highly competitive performance of over 0.8 F_1 on the SemEval-2016 data, significantly outperforming the considered baselines. Using the semi-automatically created ontology lowers accuracy by only 0.03 points.

*This chapter is based on "Kim Schouten and Flavius Frasincar. Ontology-Driven Sentiment Analysis of Product and Service Aspects. Acepted in *Extended Semantic Web Conference (ESWC 2018)*. Springer, 2018.

With so much opinionated, but unstructured, data available on the Web, sentiment analysis has become popular with both companies and researchers. Its goal is to extract the sentiment of content creators, such as the writers of consumer reviews, and to aggregate this information into easy to digest overviews, infographics, or dashboards. Depending on the specific scenario, sentiment can be modeled as a set of emotions, or, more commonly, as a point on a polarity scale ranging from positive to negative. Polarity can be binary with just the positive and negative value, or it can be modeled as a 5-star score, or even as a real number within a given interval (e.g., between -1.0 and 1.0).

Since reviews often go into detail about certain characteristics of the entity under review, it is useful to go one step further and perform aspect-based sentiment analysis. Here, instead of computing a sentiment score for the whole review, or even per sentence, the goal is to locate the different characteristics, or aspects, the reviewer writes about, and then compute a sentiment score for each of the mentioned aspects. This yields more in-depth results, as people are often not positive (or negative) about every aspect of the product or service they bought.

In general, aspect-based sentiment analysis methods can be classified as knowledge-based or as machine learning based (Schouten and Frasincar, 2016). This is of course not a perfect classification, as machine learning methods often incorporate information from dictionaries, such as sentiment lexicons. Nevertheless, it is a useful distinction as machine learning methods require a sufficient amount of training data to perform well, while knowledge-based methods do not. In the pursuit of high performance, machine learning classifiers have become very popular at the expense of knowledge-based systems. However, (Schouten et al., 2017b) shows that both have their use and that machine learning classifiers and knowledge-based systems can in fact be complementary. To alleviate the need for manual construction of knowledge bases, we have designed an algorithm to create domain ontologies from unlabeled reviews in a semi-automatic fashion. The domain knowledge, coupled with some logical axioms is used to decide what sentiment to assign in which situation. In addition, a bag-of-words model, with additional features, such as a sentiment value of the sentence, complements this knowledge-driven method. The bag-of-words model is implemented using a standard machine learning classifier, which in our case is a Support Vector Machine. With the focus being solely on the sentiment analysis of

aspects, the aspect detection phase is not considered in this work, and hence, the aspect annotations in the data are used as a starting point.

For this research, the widely used set of restaurant reviews from SemEval-2016 (Pontiki et al., 2016) is employed. The SemEval-2016 data consists of a training set of 350 reviews with in total 2506 sentiment-labeled aspects and a test set of 90 reviews with in total 859 sentiment-labeled aspects. More data statistics can be found in (Pontiki et al., 2016; Schouten et al., 2017b).

An excerpt of the raw data is given in Fig. 9.1. The provided annotations already split the dataset into reviews and sentences, and each sentence can be labeled with zero or more opinions, which is an aspect together with the expressed sentiment related to that aspect. The task of aspect sentiment classification is to give the sentiment value for each aspect, where the aspects are already provided. Thus, all annotations, like the ones given in Fig. 9.1, are provided, except the values of the `polarity` fields. The accuracy of the classifier is simply the number of correct classifications over the total number of aspects to be classified.

This work is structured as follows. A selection of related work is discussed in Sect. 9.1. In Sect. 9.2, the employed domain ontology is explained, as well as the rules to predict a sentiment value for a given aspect. It also contains a short overview of the used bag-of-words model. The base models and the hybrid combinations are evaluated in Sect. 9.3, and in Sect. 9.4, conclusions are given and directions for future research are provided.

9.1 Related Work

The related work is split into three sections. The first part discusses aspect-based sentiment analysis methods that employ ontologies, while the second presents methods for (semi-)automatic ontology construction.

9.1.1 Ontology-Enhanced Aspect-based Sentiment Analysis

A short overview of the field of affective computing, which encompasses sentiment analysis, is presented in (Cambria, 2016). The author argues that hybrid methods, combining the intuitive nature and explanatory power of knowledge-driven ap-

```
<sentence id="1032695:1">
<text>Everything is always cooked to perfection, the service is
    excellent, the decor cool and understated.</text>
<Opinions>
<Opinion target="NULL" category="FOOD#QUALITY" polarity="positive"
    from="0" to="0"/>
<Opinion target="service" category="SERVICE#GENERAL" polarity="
    positive" from="47" to="54"/>
<Opinion target="decor" category="AMBIENCE#GENERAL" polarity="
    positive" from="73" to="78"/>
</Opinions>
</sentence>
```

Figure 9.1: A snippet from the used dataset showing an annotated sentence from a restaurant review.

proaches and the high performance of statistical methods, are the most promising way to improve the effectiveness of affective algorithms. This forms the research hypothesis of this work as well. Similar to our work, (Cambria and Hussain, 2015) combines statistical methods with a set of linguistic patterns based on the SenticNet (Cambria et al., 2014) knowledge base. Each sentence is processed in order to find the potential knowledge base concepts expressed in it. The discovered concepts are linked to the SenticNet knowledge repository, which enables the inference of the sentiment value associated to the sentence. If there are no concepts expressed in this sentence or if the found concepts are not in the knowledge base, then a deep learning, bag-of-words method is employed to determine the sentiment for that sentence. Note that this is a sentence-level approach and not an aspect-based approach, like we consider here. Our work has a similar setup in that it first tries to use the knowledge-driven approach to make a prediction, using the statistical method as a backup when the knowledge-base is insufficient.

A multi-domain approach to sentence-level sentiment analysis is presented in (Dragoni et al., 2014). While sentiment is assigned to sentences instead of aspects, the sentences can come from different domains, so the proposed method needs to disambiguate sentiment words based on the domain the sentence is from. This is similar to our approach where sentiment words are disambiguated based on the aspect they are about. Differently from (Dragoni et al., 2014), our ontology does not feature a strict separation of semantic information and sentiment information, as we

believe that the sentiment value is sometimes dependent on the semantic information (i.e., context-dependent sentiment). Furthermore, (Dragoni et al., 2014) uses fuzzy membership functions to describe the relations between concepts, sentiment, and domains, and while this gives more modeling flexibility, it makes it harder to reason over the knowledge graph, which is one of the things we want to explore in this work.

In (Xia et al., 2015), a method is presented that predicts the sentiment value for sentiment-bearing words based on the context they are in. For this task, a Bayesian model is created that uses the words surrounding a sentiment-bearing word, including the words that denote the aspect, to predict the actual sentiment value of the word given the context. Similar to our approach, it uses a two-stage setup, where a backup method is used when the first method cannot make a decision. In this case, if the Bayesian model cannot make a decision about the sentiment value of the word, the previous opinion in the text is checked and if there is a conjunction between the two (i.e., no contrasting or negation words), it will assign the same sentiment value to the current word. Different from (Xia et al., 2015), we focus on aspect-based sentiment analysis and make use of a domain ontology.

9.1.2 Ontology Construction

A method to semi-automatically construct an ontology for sentiment analysis is presented in (Coden et al., 2014). This approach, called the Semantic Asset Management Workbench (SAMW), is a recursive method for developing and refining an ontology that starts with a seed set of terms and then grows the ontology using a corpus of text documents. The set of documents is searched for each of the terms in the ontology, and whenever a term is found, the pattern describing the textual context in which this term is found is extracted. The collected patterns are used to find additional terms to potentially add to the ontology. Using confidence, support, and prevalence scores, the top terms are selected and presented to the user, who has to approve which terms are to be stored in the ontology. The whole process is then repeated, using the updated list of terms. Compared to (Coden et al., 2014), the focus of this work is put more on finding sentiment words and their polarity. Furthermore, in this work, the constructed ontology is used to perform aspect-based sentiment anal-

ysis, demonstrating the usefulness and quality of the obtained ontology, while also comparing it against using a manually constructed ontology.

In (Thakor and Sasi, 2015), an ontology-based method for sentiment analysis of Twitter messages is presented, with the goal of finding negative content. To this end, an ontology is populated in a semi-automatic fashion. A large set of tweets, related to the domain of choice, is extracted and parsed using a dependency parser. The nouns are then considered to be the concepts in the ontology while the verbs are designated as the object properties. Actually adding the concepts and properties to the ontology is however still manually done. A major difference with the research presented in this work is the fact that adjectives are used as signals for sentiment and thus the ontology plays an active role in the sentiment computation, while (Thakor and Sasi, 2015) uses the SentiStrength (Thelwall et al., 2012) tool to determine the sentiment of each tweet. Furthermore, in this work, the found concepts are added to the ontology in a semi-automatic fashion.

A method to learn fuzzy product ontologies is presented in (Lau et al., 2014). The learned ontologies are meant to be used for aspect-based sentiment analysis, so special attention is given to the relations between products, product aspects, and sentiment. An LDA-based model (Blei et al., 2003) is used to extract both implicit and explicit product aspects from a set of product descriptions. The found product aspects are then located in a set of rated reviews, and within a word window, the adjectives and adverbs are collected. Each of these is paired with the aspect to form an aspect-sentiment pair, and each pair is assigned the sentiment given to the whole review. The idea is that aspect-sentiment pairs that regularly occur in negative reviews are negative, and similarly, aspect-sentiment pairs that mostly appear in positive reviews are positive. Note that in this stage, negation within the word window is taken into account by flipping the sentiment of words that appear after a negation word. Experiments showed that this method outperformed Text-to-Onto, an existing ontology builder (Maedche and Staab, 2001). While this work does not aim at creating a fuzzy ontology, it employs a method very similar to (Lau et al., 2014) to determine the sentiment of aspect-sentiment pairs. However, the ontology building process in this work still includes a human validation step, to safeguard the quality of the generated ontology.

9.2 Method

All review sentences are preprocessed using the Stanford CoreNLP package (Manning et al., 2014), performing basic operations such as tokenization, part-of-speech tagging, lemmatization, syntactic analysis, as well as sentiment analysis. The latter is an already trained neural network that assigns a numeric sentiment score to each syntactic constituent in a parse tree (Socher et al., 2013).

For the machine learning backup method, we opted for a Support Vector Machine (SVM) with a radial basis function kernel, given that SVMs have proven to be very effective for text classification problems (Mullen and Collier, 2004). Preliminary experiments have shown the radial basis kernel to be the best performing. Since the `polarity` field can have three sentiment values, a multi-class SVM is trained that is able to classify an aspect into one of three sentiment values: positive, neutral, or negative. For this work, the Weka (Frank et al., 2016) implementation of the multiclass SVM (Keerthi et al., 2001) is utilized, which internally performs 1-vs-1 pairwise classifications.

9.2.1 Ontology Design

For the ontology, the aim is to limit the number of asserted facts, preventing data duplication, and to use the reasoner to infer the sentiment of a given expression. The ontology consists of three main classes: *AspectMention*, *SentimentMention*, and *SentimentValue*. The latter simply has *Positive* and *Negative* as its subclasses, and the setup is such that if a certain concept is positive, it is a subclass of *Positive* and if it expresses a negative sentiment, that concept is modeled as a subclass of *Negative*. The *AspectMention* class models the mentions of aspects and *SentimentMention* models the expressions of sentiment. A schematic overview of the ontology is shown in Fig. 9.2.

The *SentimentMention*s can be divided into three types. The first group is formed by type-1 *SentimentMention*s, which always denote a positive (negative) sentiment, regardless of which aspect they are about. In Fig. 9.2, these are denoted with hexagons. These subclasses of *SentimentMention* are also a subclass of the sentiment class they express. Hence, *Good* is a subclass of both *SentimentMention* and

Positive. Type-2 *SentimentMention*s are those expressions that are exclusively used for a certain category of aspects, meaning that the presence of the aspect category can be inferred from the occurrence of the *SentimentMention*. In Fig. 9.2, these classes are denoted with rounded squares. For instance, *Delicious* is a subclass of *SentimentMention*, but also of both *SustenanceMention* and *Positive*, where *SustenanceMention* encompasses concepts related to food and drinks. This means that if we want to predict the sentiment value of an aspect in the service category, we will ignore the word "delicious" if it is encountered, because it cannot possibly be about the current aspect. The third type (type-3) of *SentimentMention*s contains context-dependent sentiment expressions, and this group is shown as an ellipse in Fig. 9.2. Here, the inferred sentiment depends on the aspect category. For instance, *Small* when combined with *Price* is a subclass of *Positive*, while when it is combined with *Portion* it is a subclass of *Negative*. Some of the words in this group are not ambiguous per se, but are simply not indicative of any particular aspect category while at the same time not being generally applicable. An example is the concept *Fresh*, which is always positive, but can only be combined with certain aspects: it matches well with subclasses of *SustenanceMention* (e.g., "fresh ingredients") and *AmbienceMention* (e.g., "a fresh decor"), but not with subclasses of, e.g., *PriceMention* or *LocationMention*.

When a type-1 `SentimentMention` is encountered, its sentiment value is used for the classification of all aspects within scope (i.e., the sentence). While the scope of the complete sentence can be considered too broad, as generic sentiment words usually apply to just one aspect, not all of them, in preliminary experiments, it was shown that limiting the scope to a word window or to steps over the grammatical graph is sub-optimal. A type-2 `SentimentMention` is only used for the classification of aspects that belong to the implied aspect category. For type-3 `SentimentMention`s, a new class is created that is a subclass of both the property class and the aspect class. If the ontology provides any information on that combination, its sentiment value can be inferred. Otherwise, the ontology does not provide any sentiment information for that combination of aspect and property.

The ontology is lexicalized by attaching annotations of type *lex* to each concept. A concept can have multiple lexicalizations, and since this ontology is designed to work within a single domain, there are not many ambiguous words that would point

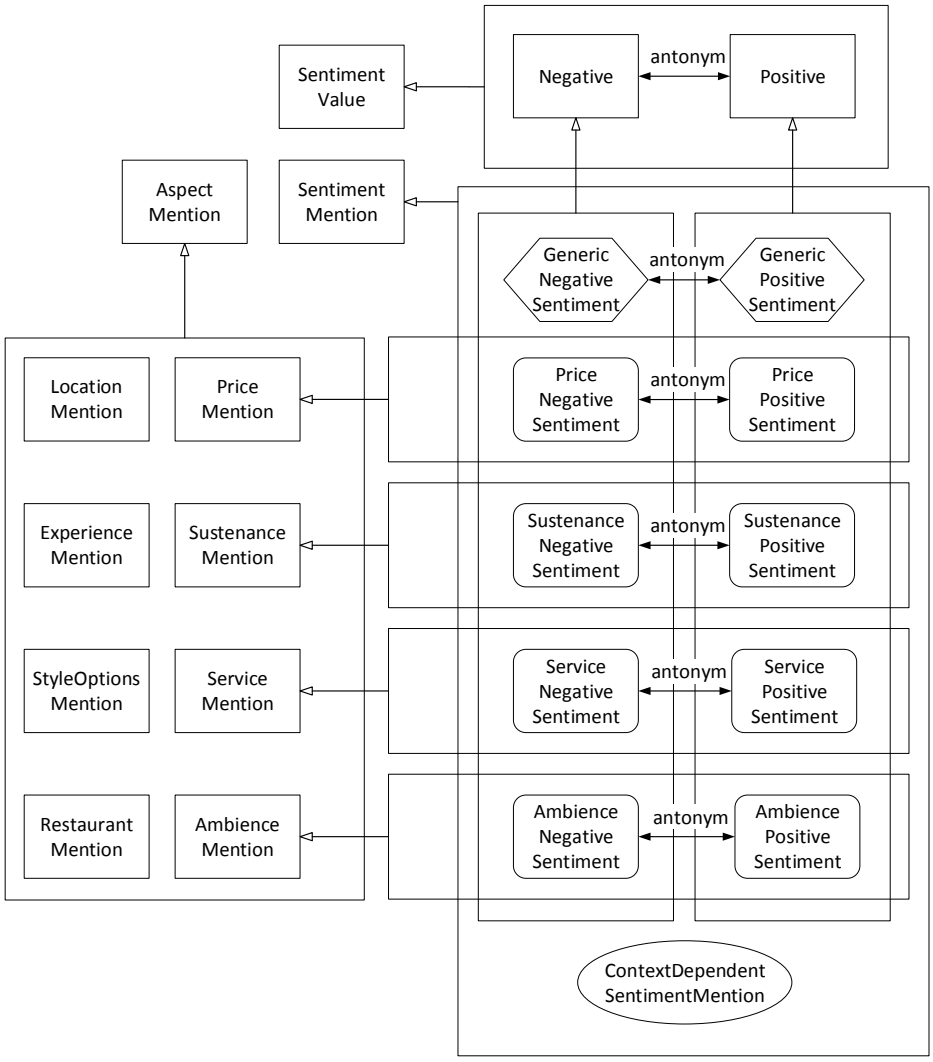

Figure 9.2: A schematic overview of the main ontology classes

to more than one concept in the ontology. Furthermore, some concepts have one or more *aspect* properties, which link a concept to one of the aspect categories in the data annotations. This means that such a concept, and all of its subclasses fit within that aspect category. For instance, the `Ambience` concept has an *aspect* property with the value "AMBIENCE#GENERAL". Last, concepts that are a subclass of `SentimentValue` have an *antonym* property that links that concept to its antonym (e.g., `Positive` has antonym `Negative`). This is used when found ontology concepts are subject to negation.

For this research, a domain ontology is manually constructed using the OntoClean methodology Guarino and Welty (2002), and represented in OWL. To demonstrate the usefulness of ontologies, a choice is made for a relatively small, but focused ontology. Hence, it contains about 365 concepts, predominantly *AspectMention*s, but also including 53 type-1 *SentimentMention*s, 38 type-2 *SentimentMention*s, and 15 type-3 *SentimentMention*s. The maximum depth of the class hierarchy, not counting *owl:Thing* at the top, is 7.

9.2.2 Semi-Automatic Ontology Creation

Given the ontology design, as presented in the previous section and tailored for the sentiment classification, this section will elaborate on the ontology creation algorithm. Creating the ontology consists of three major steps. The first step is the creation of a skeleton ontology, which will form the basis for the final ontology. The next step consists of the selection and filtering of the terms that are to be added to the ontology. The last step is assigning a suitable parent class (i.e., superclass) to each new concept.

For the semi-automatic ontology builder, additional external data is required. First, a set of domain related reviews is needed to learn the terms to add to the ontology. For the restaurant domain, data from the Yelp Dataset Challenge (Yelp, 2017) is used and in particular the subset of reviews that is about restaurants. The second type of external data that is required is a set of contrasting documents, so the ontology learner can distinguish between real product aspects and general high-frequency words. For the contrasting documents, a corpus of non-copyrighted literature from Project Gutenberg (2017) is exploited.

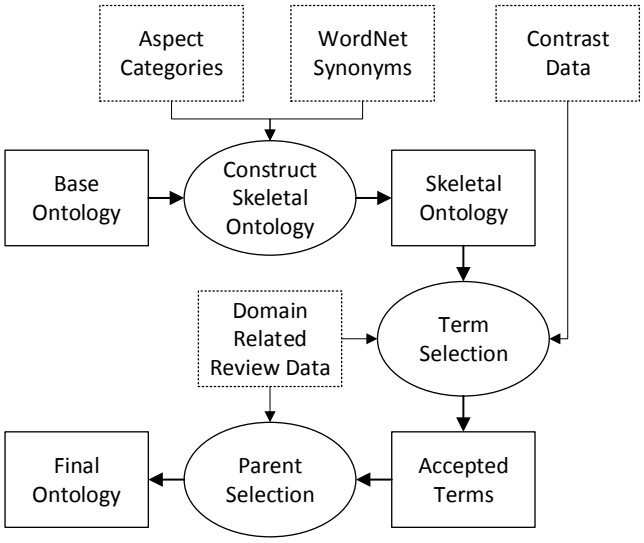

Figure 9.3: Flowchart of the semi-automatic ontology builder framework

The skeleton ontology extends the base ontology, which contains the domain independent top concepts, by adding the top level domain dependent concepts. For this, the aspect categories, as given in the SemEval data, are used. These categories are of the form ENTITY#ATTRIBUTE, where ENTITY is a general domain feature and ATTRIBUTE is a specific aspect of that entity. For example, the aspect category FOOD#QUALITY is about the general entity FOOD, with a specific focus on the QUALITY of the FOOD. Note that several attributes occur with different entities. Furthermore, each entity has the GENERAL attribute that can be used to categorize expressions about the entity as a whole. With this information, the base ontology is expanded according to the following rules:

(1) Each individual entity gets its own *AspectMention* subclass *ENTITYMention*, where ENTITY is equal to the name of the entity. This class is further related to the name of the entity through the *lex* property. Last, all aspect categories of the particular entity are related to the class via the *aspect* property.

(2) Next, each attribute, except for GENERAL, also gets its own *AspectMention* subclass. The name of this subclass is *ATTRIBUTEMention*, where ATTRIBUTE is the name of the attribute. This class also has the *lex* property linking it to the name of the attribute. In the case that the attribute name contains the character '&' or '_', the two parts of the name are split and both are separately added as lexicalizations, as well as the combination. Thus, for example, the *AspectMention* subclass related to the aspect STYLE&OPTIONS will have the lexicalizations 'style&options', 'style', and 'options'. The *ATTRIBUTEMention* class also has the *aspect* property relating it to all aspect categories containing that particular attribute.

(3) Finally, all of the above subclasses of *AspectMention* get corresponding *SentimentMention* subclasses, except for the *AspectMention* subclass representing the domain as a whole, e.g., *RestaurantMention*. Each class gets both a negative sentiment mention subclass as well as a positive sentiment mention subclass. Thus the class *AmbienceMention* gets the subclasses *AmbienceNegativeSentimentMention* and *AmbiencePositiveSentimentMention* which are respectively subclasses of *Negative* and *Positive*.

The last step in developing the skeleton ontology is the addition of synonyms. We attempt to enhance the lexicalizations of each of the subclasses of *AspectMention*. Given the lexicalizations found using the rules described earlier, we use WordNet (Fellbaum, 1998) to suggest a maximum of 20 synonyms for each lexicalization to the user. Synonyms accepted by the user are added via the *lex* property to the *AspectMention* subclass, which is the last step in completing the skeleton ontology.

Term Selection

Given a skeleton ontology specific to the domain under consideration, the next stage is to help the user further populate the ontology using a corpus of domain specific texts. The data to be used for the domain specific texts consists of external domain related user rated reviews (Yelp, 2017).

Similarly to (Coden et al., 2014) we first extract the nouns and adjectives within the corpus of domain related texts. These two word types are chosen as they most naturally translate to aspect mentions and sentiment mentions. Once the potential ontology concepts have been retrieved from the reviews, we calculate several measures

for each term. The first is domain pertinence, a measure of the domain relevance of a term (Park et al., 2002):

$$DP_D(t) = \frac{freq(t, D)}{max_j(freq(t, C_j))}, \tag{9.1}$$

where t is the term, D is the domain corpus, $freq(t, D)$ is the number of times term t occurs within the corpus D, C is the contrastive domain corpus, and C_j is a document within the contrastive domain corpus. For the contrastive domain corpus our implementation makes use of literature with expired copyright (Project Gutenberg, 2017).

The next measure that we determine for each term is domain consensus, where the idea is that there should be consensus on the considered term among the documents of a domain corpus (Navigli and Velardi, 2002). Thus a term should appear regularly across documents and not only in a few select documents. Domain consensus is calculated as

$$DC_D(t) = -\sum_{d \in D} norm\_freq(t, d) \cdot log(norm\_freq(t, d)), \tag{9.2}$$

where t is the term, D is the domain corpus, and $norm\_freq(t, d)$ is the normalized frequency of term t in document d. The normalized frequency is defined as follows:

$$norm\_freq(t, d) = \frac{freq(t, d)}{max_{d \in D}(freq(t, d))}, \tag{9.3}$$

where t is the term, d is the document, D is the domain corpus, and $freq(t, D)$ is the number of times term t occurs within the corpus D.

Given the domain pertinence and domain consensus of a term, an overall score can be computed, where both parts are normalized to allow for a relative comparison De Knijff et al. (2013).

$$term\_score(t, D) = \alpha \cdot \frac{DP_D(t)}{max_{t \in D}(DP_D(t))} + \beta \cdot \frac{DC_D(t)}{max_{t \in D}(DC_D(t))}, \tag{9.4}$$

where t is the term, and D is the domain corpus. The parameters α and β represent the weight given to domain pertinence and domain consensus, respectively, and both take values between 0 and 1. Using a grid search, both are optimized simultaneously, with a step size of 0.1. To determine the optimal combination we select the pair that maximizes the number of terms accepted by the user as relevant domain ontology concepts during a partial run of the framework implementation. This pair is $\alpha = 0.4$ and $\beta = 0.1$.

Once the term scores have been calculated they are ranked on the basis of this score and only the top 10% of the nouns and top 10% of the adjectives are suggested to the user in order to keep the number of suggestions, and their quality, at an adequate level. The user then selects the terms to be added to the ontology.

Parent Selection

The last stage in our semi-automatic ontology builder framework is the addition of the previously selected terms to the ontology. In order to add a term to the ontology as a concept, we first need to determine the correct parent class, and for this we use the subsumption method (Sanderson and Croft, 1999). We choose to use the subsumption method because we are creating rather shallow ontologies and the subsumption method has been shown to work well for shallow ontologies (De Knijff et al., 2013).

The subsumption method is based on the co-occurrence of words and works on the principle that a parent concept is present in most, if not all, documents in which the child concept appears (Sanderson and Croft, 1999). A term p is a potential parent of term x when

$$P(p|x) \geq t, P(x|p) < 1, \tag{9.5}$$

where t is a threshold between 0 and 1. The above conditions say that the parent term should occur in at least a proportion t of the documents in which child term x appears, whilst the child term x should not appear in at least one document in which the parent term p occurs. The value of threshold t is determined by considering the acceptance ratio and the total number of acceptances found during a run of our semi-automatic ontology builder. The above conditions (Eq. 9.5) however result in

multiple potential parent suggestions, and thus we give each parent term p a score:

$$parent\_score(p, x) = P(p|x). \tag{9.6}$$

For each accepted term an initial list of plausible parent classes is compiled. For nouns this list contains all the subclasses of *AspectMention* that are not also subclasses of *SentimentMention*, as some sentiment words that directly imply an aspect are subclasses of *AspectMention* as well. For adjectives this list contains all the subclasses of *SentimentMention*. Next, we filter out all the parent classes that do not meet the conditions of the subsumption method.

For adjectives we further consider two additional steps. First, a sentiment score is computed for each adjective as the pseudo-expected value word score introduced by Cesarano et al. (2004). It is defined for an adjective t as:

$$sentiment\_score(t) = \frac{\sum_{d \in D}(rating(d) \cdot \frac{n(t,d)}{\sum_{w \in adj(D)} n(w,d)})}{\sum_{d \in D} rating(d)}, \tag{9.7}$$

where:

- D is the domain corpus (data set of external domain related user rated reviews);
- $rating(d)$ denotes the star rating of review d converted to a score between 0 and 1;
- $n(t, d)$ indicates the number of occurrences of adjective t in review d; and
- $adj(D)$ refers to the set of adjectives occurring in the domain corpus.

To ensure that positive adjectives are suggested to belong to a subclass of the *Positive* concept, the parent score of all subclasses of *Positive* are doubled. The same is done for subclasses of *Negative* when a negative adjective needs to be assigned to a parent class. In this way, concepts of the right sentiment value will be at the top of the list of suggestions the human curator has to review. Thus, for example, in the case of the adjective 'delicious' the score of the potential parent *FoodPositiveSentimentMention*, among others, gets doubled.

Besides the sentiment value, it is also informative to look at the nouns that the adjective is describing. In the second step, this information is leveraged to create better parent suggestions for *SentimentMention*s. From the large set of external, domain related reviews, all adjectives are extracted, as well as the nouns that they are grammatically related to. The goal is to find dependencies relating an adjective to a noun that is a known lexicalization of an *AspectMention* subclass. Thus, for each adjective we keep track of which words they were related to via the adjectival modifier and nominal subject dependencies. If a lexicalization of a potential parent class is contained within the set of grammatically related nouns, the score for this parent class is doubled. For example, if the review data set contains the sentence "The food is delicious.", then for the adjective 'delicious' the scores of the potential parents *FoodPositiveSentimentMention* and *FoodNegativeSentimentMention* are doubled, as they are both subclasses of *FoodMention*, which has the lexicalization 'food' and subclasses of *SentimentMention*.

Once the scores of the potential parents of a term have been established, they can be suggested to the user. For nouns, the potential parents are ordered by score and shown to the user in descending order. The user then either accepts or rejects the parent relations. Once a parent class has been found for a particular term, the remaining potential parents are no longer suggested. The suggestion of parent classes for adjectives works similarly to the case of nouns, the only difference being that for adjectives the framework first proposes the *GenericNegativeSentimentMention* and *GenericPositiveSentimentMention* classes. The reason for this is that the subsumption method works on the basis of word frequencies, however, the generic sentiment classes do not have lexicalizations and therefore they would never be suggested to the user. Yet, these two classes are extremely important for sentiment analysis and should therefore not be disregarded. Adjectives which are determined to be negative (positive) by the sentiment score are suggested the negative (positive) generic sentiment class first, and if this parent is rejected by the user then the positive (negative) generic sentiment class is suggested, which is covering the case where the sentiment of the adjective has been incorrectly determined. Using the sentiment scores in this way increases the overall acceptance ratio of the framework.

Once a term has been accepted and its parent concept has been found, a concept corresponding to the term is automatically added to the ontology as a subclass of

the accepted parent. The new concept is further related via the *lex* property to the accepted term. The ontology concepts representing terms do not get *aspect* properties as it is sufficient for their ancestor classes to be related to the correct aspect category. In Fig. 9.4 we present a snippet of an ontology built semi-automatically by the framework for the restaurant domain. The ontology snippet in the figure shows some of the concepts added to the ontology under the *ServiceMention* class. Note, however, that the figure only shows two examples of the *aspect* and *lex* properties and that in reality there are more properties related to the visible classes.

9.2.3 Sentiment Computation

An overview of the sentiment computation method is shown in Alg. 9.1, outlining the three cases for type-1, type-2, and type-3 sentiment expressions, respectively. The input for sentiment prediction is an ontology, an aspect, and whether or not a bag-of-words model is used as a backup method in case the ontology does not specify a single sentiment value for this aspect. The PREDICTSENTIMENT method starts by retrieving all the words that are linked to the ontology with a URI and that are in the sentence containing the aspect. It also checks whether the current word is negated or not. For this we look for the existence of a *neg* relation in the dependency graph, or the existence of a negation word in a window of three words preceding the current word (Hogenboom et al., 2011).

In the next step, the type of the concept is retrieved from the ontology and, depending on its type, the algorithm will execute one of three cases. As mentioned before, if the concept is a type-2 sentiment expression, then its inferred aspect category has to match with the current aspect, otherwise it is ignored. For example, when encountering the word "delicious", it leads to the concept `Delicious` because of the lexical property, which is a subclass of `SustenancePositiveProperty`.

1. $\texttt{Delicious} \equiv \exists lex.\{\text{“delicious”}\}$
2. $\texttt{Delicious} \sqsubseteq \texttt{SustenancePositiveProperty}$
3. $\texttt{SustenancePositiveProperty} \sqsubseteq \texttt{Sustenance} \sqcap \texttt{Positive}$

Furthermore, the `Sustenance` concept is linked to several aspect categories that exist in the annotated dataset by means of an *aspect* property.

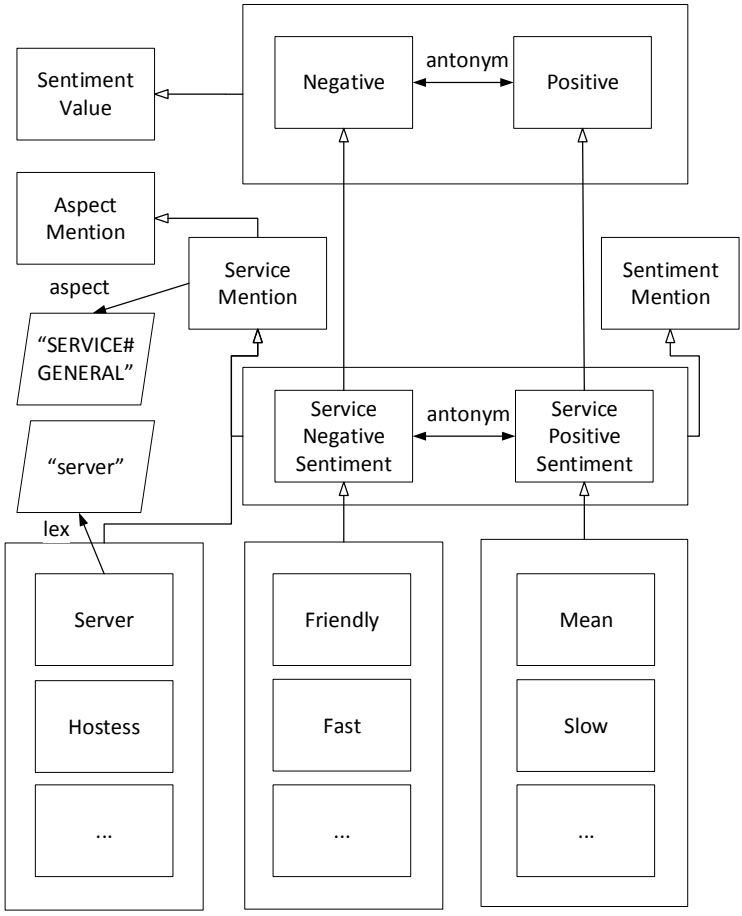

Figure 9.4: Semi-automatically constructed restaurant domain ontology snippet

Algorithm 9.1: Pseudocode for computing aspect sentiment

```
input  : Ontology o
input  : Aspect a
input  : boolean useBOW
output : SentimentValue
Set<String> foundURIs = ⊥;
Set<Word> words = getWordsWithURI(getSentence(a));
foreach Word w ∈ words do
    boolean negated = isNegated(w);
    String URI = getURI(o, w);
    if isType1(o, URI) then
        foundURIs = foundURIs ∪ getSuperclasses(o, URI, negated);
    end
    if isType2(o, URI) ∧ categoryMatches(a,URI) then
        foundURIs = foundURIs ∪ getSuperclasses(o, URI, negated);
    end
    if isType3(o, URI) then
        foreach String relURI ∈ getRelatedAspectMentions(w) do
            String newURI = addSubclass(o, URI, relURI);
            foundURIs = foundURIs ∪ getSuperclasses(o, newURI, negated);
        end
    end
end
boolean foundPositive = (PositiveSentimentURI ∈ foundURIs);
boolean foundNegative = (NegativeSentimentURI ∈ foundURIs);
if foundPositive ∧ ¬ foundNegative then
    return Positive;
else if ¬ foundPositive ∧ foundNegative then
    return Negative;
else if useBOW then
    return getBOWPrediction(a);
else
    return getMajorityClass() // this is Positive in our data sets;
end
```

4. `Sustenance` ≡ ∃*aspect*.{“FOOD#QUALITY”}

5. `Sustenance` ≡ ∃*aspect*.{“FOOD#STYLE_OPTIONS”}

Hence, when the current aspect for which we want to compute the sentiment is annotated with either one of those two categories, the word “delicious” is considered to be of positive sentiment. For aspects with a different category, the same word is considered to be neutral, as it cannot possibly describe those aspects.

If the current `SentimentMention` is generic (type-1) or matching aspect-specific (type-2), then all superclasses are added to the set of $foundURIs$. If the current

concept is a type-3, or context-dependent, `SentimentMention`, we need to check if it is grammatically related to a lexicalization of an `AspectMention` and whether the combination of those two triggers a class axiom or not. To that end, we create a subclass with both the `SentimentMention` and the `AspectMention` as its direct superclasses, and add all (inferred) superclasses to the set of $foundURIs$. If there is a class axiom covering this combination, then the set of all inferred superclasses of this new subclass will include either `Positive` or `Negative`. When the current word was determined to be negated, the GETSUPERCLASSES method will add the antonym of each superclass instead, provided the ontology has an antonym for that class.

A good example of a Type-3 `SentimentMention` is `Small`, for which the ontology contains two sentiment-defining class axioms in the ontology, as well as a property that links the concept to the lexical representation "small".

1. `Small` $\equiv \exists lex.\{$"small"$\}$
2. `Small` $\sqcap$ `Price` $\sqsubseteq$ `Positive`
3. `Small` $\sqcap$ `Serving` $\sqsubseteq$ `Negative`

Furthermore, `Portion` $\sqsubseteq$ `Serving` and we assume the review text contains a phrase like "small portions", "portions are small", or something similar. First, the words "small" and "portions" are linked to their respective ontology concepts by means of the `lex` attribute. Then, since, `Small` is neither a generic type-1 `SentimentMention`, nor an aspect-specific type-2 `SentimentMention`, it is paired with related words in the sentence to see if there are any class axioms to take advantage of. In this case, *small* is directly related in the dependency graph to *portions*, so a new class is created called `SmallPortion`, that is a direct subclass of `Small` and `Portion`:

4. `SmallPortion` $\sqsubseteq$ `Small` $\sqcap$ `Portion`

This triggers the class axiom defined earlier, leading to

5. `SmallPortion` $\sqsubseteq$ `Negative`

Hence, `Negative` is added to the list of found classes.

The last step is to check whether the previous inferences have resulted in finding `Positive` or `Negative`. If we find one but not the other, the aspect is determined

to have the found sentiment value. If either no sentiment value is found, or both sentiment values are found, the ontology does not give a definitive answer. In that case, if we opt to use a bag-of-words backup model, then it is used here. If bag-of-words is not used, we default to predicting `Positive` as that is the majority class.

9.2.4 Bag-of-words model

The bag-of-words model is used both as a baseline, and as a backup model in case the ontology cannot decide which sentiment to assign. For the most part, it is a classical bag-of-words model with binary features for each lemma in the review that contains the current aspect. In preliminary experiments, this gave better results than using the lemmas from the sentence only. We hypothesize that this might be due to the fact that with more words, it is easier to get the overall sentiment of the review correctly, while for sentences, being a lot smaller, this would be harder. Given that the majority of the aspects follow the overall sentiment of the review, the effect of having more words to work with is larger than the effect of missing out on those aspects with a sentiment value different from the overall review. Furthermore, there is a set of dummy features to encode the aspect category as well as a numerical feature denoting the sentiment of the sentence. This sentiment score is computed by a sentiment component (Socher et al., 2013) in the Stanford CoreNLP package and falls roughly in the range of [-1,1]. The model is trained as a multi-class Support Vector Machine that is able to predict positive, negative, and neutral. These last two features are aspect-specific and sentence-specific, so the model is technically not bound to predict the same sentiment for all aspects within the same review. The feature vector is illustrated in Fig. 9.5.

9.2.5 Bag-of-words model with ontology features

Besides the rule-based ontology method using the bag-of-words model as a backup, it also makes sense to use the bag-of-words model as the leading model and add ontology information in the form of additional features. Hence, we add two binary features to the bag-of-words model, one to denote that the presence of the `Positive` concept and one to denote the presence of the `Negative` concept (see Fig. 9.5). Furthermore, to keep it in line with the rule-based ontology method, when both `Positive` and

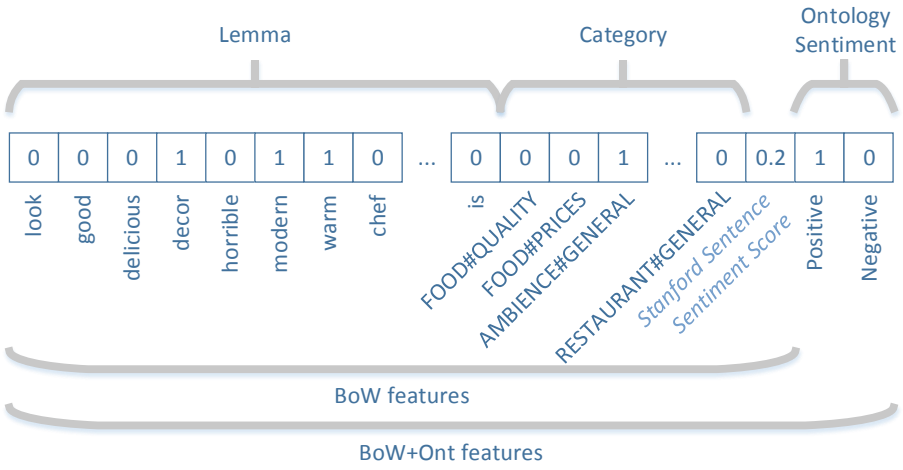

Figure 9.5: Feature vector example for BoW+Ont model

`Negative` are present, this is regarded as having no information so both features will be zero.

9.3 Evaluation

The evaluation consists of two parts, where first the ontology creation method is evaluated and after that the quality of the ontology is measured by using it to perform aspect-level sentiment analysis.

9.3.1 Ontology Creation

Once the term score parameters α and β have been optimized, we determine the value of the subsumption threshold t. To select a value for t we consider two measures: the overall acceptance ratio and the number of acceptances. These are equal to, respectively, the fraction and number of suggested terms and parent relations accepted by the user. Fig. 9.6 shows the effects of the subsumption threshold t on

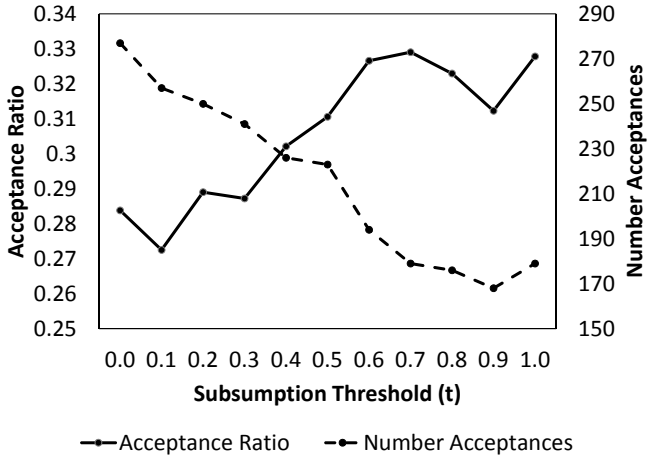

Figure 9.6: The effects of the subsumption threshold on the acceptance ratio and the number of acceptances

the overall acceptance ratio and the number of acceptances over the full run of the ontology builder. The framework was run on the restaurant domain with a total of 2001 external restaurant reviews. The figure shows that there is a payoff between the acceptance ratio and the number of accepted terms. The higher the value of t the more likely it is that a suggestion by the framework is accepted, however similarly the created ontology will be of a smaller size due to the fall in the number of user acceptances. Since neither measure on its own gives conclusive results we implement a subsumption threshold of $t = 0.2$, a value at which the acceptance ratio is not too low and the number of acceptances is relatively high. Ultimately, we want to create an ontology that can be used for sentiment analysis and therefore, a larger ontology is desired.

Our last consideration was the number of external reviews to be used during the ontology build. When increasing the number of external reviews, there are two aspects to consider. The primary justification behind increasing the number of external reviews would be to increase the number of lexicalizations within the resulting ontology. However, increasing the number of external reviews comes at the price of

Table 9.1: Comparison of all methods on the SemEval-2016 restaurant review data

method	out-of-sample F_1	cross-validation F_1	cross-validation st.dev
OntM	0.753	0.737	0.0670
OntSA	0.728	0.690	0.0665
BoW	0.796	0.776	0.0535
OntM+BoW	0.838	0.834	0.0412
OntSA+BoW	0.818	0.805	0.0456
BoW+OntM	0.801	0.785	0.0535
BoW+OntSA	0.733	0.781	0.0507

an increased ontology construction time. In Fig. 9.7, the payoff between the number of external reviews and construction time is shown. It is noticeable that, while both the construction time and the number of lexicalizations increase with the number of external reviews, the growth rate of both decrease. For the number of ontology lexicalizations, the growth rate is likely to decrease due to the fact that at a certain point the number of domain related terms will reach a cap. The probability of finding a new domain pertinent term decreases when the number of processed reviews increases. This in turn influences the run time of the ontology builder. The less domain related words are found, the less additional words need to be processed by the framework and appraised by the user. For our goals, we would like to maximize the number of lexicalizations which would suggest a higher number of external reviews. We decided on a total of 5001 external reviews, which had a reasonable construction time of 40 minutes. Considering that increasing the number of lexicalizations from 4001 to 5001 reviews only resulted in an additional three lexicalizations, it is clear that increasing the number of lexicalizations even further would not have a clear additional benefit. Please note however that construction time is of course dependent on the available hardware. Using machines more powerful than the laptop with an Intel Core i5 processor at 1.40GHz with 4GB of RAM that has been used for this work, will likely result in decreased construction times.

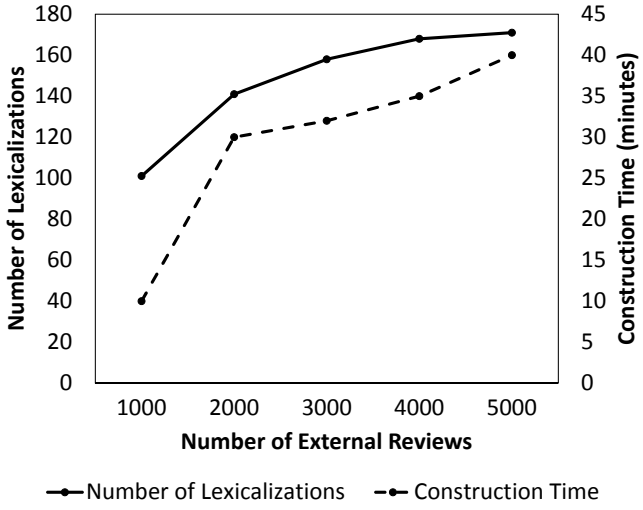

Figure 9.7: The effects of the number of external reviews on the number of lexicalizations and construction time

9.3.2 Ontology Performance with Aspect-Based Sentiment Analysis

To evaluate the performance of the proposed method, both with the manually and semi-automatically created ontology, and the baselines, all methods are trained on the training data and tested on the official test data. To determine the required (hyper)parameters, such as C and *gamma* for the SVM, about 20% of the training data is reserved as a validation set. After the optimal values have been found, the model is trained using those settings on the whole training data. This evaluation is performed using the Restaurant data set from the SemEval ABSA task and the results are shown in Table 9.1. The different variants that are evaluated are the pure ontology method using either the manual ontology (OntM) or the semi-automatic one (OntSA), the bag-of-words method which has all features except the ontology related ones (BoW), the ontology method with bag-of-words as the backup method using either the manual (OntM+BoW) or the semi-automatic (OntSA+BoW) ontology,

and the bag-of-words model with additional ontology features that are extracted using the manual (BoW+OntM) or semi-automatic (BoW+OntSA) ontology.

From the results, we can conclude that the ontology method on its own is not sufficient. This is most likely caused by the fact that in about 50% of the cases, the ontology does not provide the necessary information. In those cases, there are either no lexicalizations present or there are signals for both positive and negative contained within the context of an aspect. As the ontology method will predict the majority class (i.e., positive in this case) in these instances, its performance is low. However, as evidenced by the increased performance for both hybrid methods, the ontology method is able to provide information that is complementary to the information contained in the bag-of-words. Furthermore, while the best performing method is the ontology method with the manual ontology and the bag-of-words backup method, the second-best method is the same method but with the semi-automatic ontology. While it is inferior to the manual ontology, it is still able to add sufficient information to the machine learning classifier to be of use.

Because the ontology-based method so clearly distinguishes between different choices based on whether the positive and/or negative class is detected in the sentence, an overview is given in Table 9.2 of the performance of the ontology-based method (without BoW backup) as well as the bag-of-words model (without Ont features), split out per scenario. This table shows that for all sentences where only the `Positive` class is found, the ontology method achieves an F_1 of 0.931, while the bag-of-words model only gets 0.876. For sentences where only the `Negative` class is found, the ontology method also outperforms the bag-of-words model. However, for sentences where either both `Positive` and `Negative` are present, or neither, the ontology method defaults to predicting the majority class, so in those cases the bag-of-words model performs better.

Interestingly, the aspects for which both a positive and a negative sentiment is detected in the sentence are harder for the bag-of-words model to predict the sentiment for than sentences where the ontology did not find any sentiment expressions. The fact that the bag-of-words model does relatively well on the latter suggests omissions in the ontology. Clearly, the bag-of-words model is able to find some clues as to what sentiment to predict, even though these are not present in the ontology.

Table 9.2: The complementary performance of the ontology and bag-of-words methods.

	size	OntM F_1	BoW F_1
Found only `Positive`	55.3%	**0.931**	0.876
Found only `Negative`	8.4%	**0.732**	62.0%
Found both	4.1%	0.629	**0.686**
Found none	33.8%	0.507	**0.797**

9.4 Conclusion

In this chapter, an ontology-based method for aspect sentiment analysis is presented. It utilizes domain information, encoded in an ontology, to find cues for positive and negative sentiment. When such cues are either not found, or when both positive and negative cues are present, a bag-of-words model is used as a backup method. The ontology-based method and the bag-of-words model are shown to complement each other, resulting in a hybrid method that outperforms both. Since the ontology-based model does not need any training data, the performance of the hybrid method also depends less on having sufficient training data, and this effect was illustrated empirically as well.

As future work, we would like to validate our results on a different domain then the currently employed one. Furthermore, we would like to expand the ontology current ontology learning process to produce type-3 context-dependent sentiment axioms

Chapter 10

Conclusions and Outlook

In this dissertation, we have investigated the problem of aspect-based sentiment analysis. In particular, we have analyzed the use of intra-textual discriminants, such as co-occurrence data, and discourse structure as well as extra-textual discriminants that originate from external knowledge bases including domain ontologies. A variety of techniques were used, ranging from Bag-of-Words models to rule-based approaches exploiting discourse structures or domain ontologies. Our most important findings are discussed in the next section. This chapter ends with an informed outlook on the field of aspect-based sentiment analysis in which we put our research into perspective and provide suggestions for possible future research.

10.1 Conclusions

After carefully reviewing the body of literature and organizing the various approaches in a taxonomy, we can conclude that the field of aspect-based sentiment analysis has transcended its early stage. Initially, there was a great lack of standardized testing, with most of the work not sharing data sets or even evaluation metrics. However, we can conclude that after three successful iterations of the Aspect-Based Sentiment Analysis shared task at the SemEval 2014, 2015 and 2016 workshops, many of the more recent works are utilizing the presented data sets and evaluation metrics. This is something we have also aimed to do throughout this dissertation. The second major conclusion after the literature review is that the future of aspect-based sentiment analysis can be found in more semantically-oriented approaches, which naturally integrate common sense knowledge, general world knowledge, and domain knowledge. This dissertation actively contributes to such a future.

Before developing more semantic, but also more complex, models for aspect-based sentiment analysis, we first investigated the use of co-occurrence data for detecting aspects. Co-occurrence data may be one of the most basic discriminants to be used for text analysis, and while performance is not up to par with more advanced machine learning methods, we show that a relatively good performance can be obtained using simple techniques.

To be informed about which kinds of discriminants are useful for aspect-based sentiment analysis, a study is performed where various intra-textual discriminants are compared based on their Information Gain with respect to the sentiment analysis subtask. From this study we conclude that only some of the discriminants are actually informative, and that most of them can be filtered out. In this way, a massive performance boost is attainable, with a training time that is only 20% of that needed when using all discriminants and is only 2.9% less accurate.

Given the fact that many sentences contain multiple aspects which can have different sentiment values, it is important to determine which parts of the sentence are relevant for which aspect. To that end, we explored the use of discourse information to determine the optimal scope within a sentence to determine the sentiment of a given aspect. This scope is subsequently used to determine the sentiment of aspects using a simple dictionary-based method. Using this method for sentiment analysis, we show that having the scope determined by the discourse structure is better than using the whole sentence in which as aspect is embedded.

Even though information regarding the discourse structure of text is high-level information, it is still only a source of intra-textual discriminants. Therefore, following the outlook presented in the literature survey, we move towards a knowledge-driven approach by implementing an ontology-based method for aspect-based sentiment analysis. We show that using external domain knowledge significantly improves aspect detection and is also beneficial for sentiment analysis. Furthermore, logic axioms are used to determine the sentiment for aspects with context-dependent sentiment words, such as ‘dry’, where the sentiment value of these descriptors depends on the aspect they are describing (e.g., ‘dry meat’ vs. ‘dry wine’, with the first usually being a negative expression and the last a neutral one).

External domain knowledge is shown to limit the need for labeled training data, but since ontologies are generally created by hand, there is potentially still a lot of

manual labor involved. To that end, we also investigate the use of a semi-automatic method for ontology creation, where the human input is not so much distilling facts out of texts into a knowledge base, but rather curating the facts extracted by the proposed algorithm. This helps to save a tremendous amount of time and makes ontology-based methods for aspect-based sentiment analysis also feasible for domains where not much training data is available.

The research question proposed for this work at the beginning of this dissertation is:

> **Which intra-textual and extra-textual discriminants are beneficial for performing aspect-based sentiment analysis?**

Looking back at our results, we can state that the occurrences of certain words, as in the Bag-of-Words model, are strong discriminants for aspect-based sentiment analysis and should generally be included in any predictive algorithm. However, we also saw that many words are non-informative. Additionally, we investigated the use of a variety of features, ranging from syntactic to more semantic ones and saw that, while they do contain useful information, there are even more non-discriminative features among those, making it harder to effectively use them in practice.

With discourse information we took a different approach, using that information to select the relevant piece of text for our sentiment computation. This also allows us to limit the number of non-informative discriminants and can be seen as an alternative to selecting discriminants based on a measure such as Information Gain. While the notion of using the discourse structure is semantically more appealing than using a global selection method, as the latter does not solve the issue of having multiple aspects with a different sentiment value in the same textual context, it is also more challenging. Errors in the discourse parser, as well as errors in the selection mechanism, heavily influence the final result.

In terms of extra-textual discriminants we look at using the Bag-of-Concepts approach, where we investigate which concepts from the domain ontology are present in the text. These signals are especially useful for aspect detection. For sentiment analysis, we not only look for concepts that describe aspects, but also concepts that describe sentiment expressions. These can be combined such that the sentiment concept can be inferred from the combination of aspect concept and sentiment ex-

pression concept. This results in a strong signal for sentiment, which is measured by having a sentiment analysis method that solely relies on this discriminant. We show that this works well when there are concepts from the used domain ontology present in the text. However, this highly depends on the domain coverage of the ontology. Therefore, both the manually designed ontology and the semi-automatically created ontology still benefit from being combined with a Bag-of-Words model.

10.2 Outlook

One of the attractive characteristics of research is how, while answering one research question, more questions arise in the process. This also happened in the course of performing the research for this dissertation. While there are many questions still unanswered, we would like to touch upon a couple of major ones that we think are potential steps forward for the field of aspect-based sentiment analysis.

One of the major things that are missing, and that we started to address with our research on using discourse information for sentiment analysis, is a robust method to determine which parts of the text are relevant when computing the sentiment value of a given aspect. This problem is conceptually similar to determining the scope of a negation and can be approached similarly. Examples include the use of a word window around each aspect, or a number of steps over the grammatical dependency graph starting at the aspect, or using discourse units. During our research we found that to determine the sentiment value of an aspect, there is no fixed scope (e.g. 5 words around the aspect) that will always, or even in most cases, give all the necessary information. Scopes vary wildly, ranging from cases where there is just a single word that is needed, to needing the whole sentence or even a previous sentence. Even using discourse analysis to find the scope is not a full solution, as we still found cases where we needed more discourse units, or less, or even half a discourse unit.

The previous remark leads to our next point, which is the use of global expressions for sentiment. While the vast majority of the aspects are given rather explicit sentiment expressions (typically with a small scope), the sentiment of some aspects is described using sometimes colorful expressions that might span the full sentence or even multiple sentences. The fact that these expressions consist of multiple words,

none of which may bear particular strong sentiment, makes them difficult to detect. This also points out one of the weak points of the ontology-driven approach we propose in our previous chapter, which is the fact that while the concepts from the domain ontology are clear and descriptive, aligning them with the raw text is not always straightforward. Some concepts have just a single word that is indicative of the concept's presence, but as mentioned before, there are many concepts where this is not the case and where the combination of words that denote a concept are grammatically linked rather than being next to each other in the text. In these cases, ontology concepts should be labeled, not with a lexical expression, but with an expression describing the local context in the grammatical graph.

While we analyze the relation between machine learning and knowledge-driven approaches, we did not look into combining that with discourse structure information. In order to focus on the knowledge-driven nature of the method, we used the whole sentence as the scope. While this gives a clear comparison with a basic Bag-of-Words approach, it is very interesting to see how this would combine with using discourse structure information to determine the scope.

We believe that there is a place for semantic approaches to aspect-based sentiment analysis, next to the power play of deep neural networks, as their intuitive approach to analyzing natural language is easy to explain and can provide intuitive and accurate results. We look forward to a future where the natural strengths of both will be utilized to create explainable and high-performing algorithms for sentiment analysis.

Bibliography

M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Technical report, Google, 2015. URL `http://tensorflow.org/`. `download.tensorflow.org/paper/whitepaper2015.pdf`.

T. Abeel, Y. Van de Peer, and Y. Saeys. Java-ML: A Machine Learning Library. *Journal of Machine Learning Research*, 10:931–934, 2009.

M. Adjei, S. Noble, and C. Noble. The Influence of C2C Communications in Online Brand Communities on Customer Purchase Behavior. *Journal of the Academy of Marketing Science*, 38(5):634–653, 2009.

C. C. Aggarwal and C. Zhai. *Mining Text Data*. Springer Science & Business Media, 2012.

Alias-i. Lingpipe 4.1.2, 2017. [WWW document] `http://alias-i.com/lingpipe` (accessed 19th July 2017).

Apache Jena. Apache Jena, 2017. [WWW document] `https://jena.apache.org/` (accessed 19th July 2017).

Apache OpenNLP. OpenNLP: A Java-based NLP Toolkit, 2017. [WWW document] `opennlp.apache.org` (accessed 19th July 2017).

N. Asher, F. Benamara, and Y. Y. Mathieu. Appraisal of Opinion Expressions in Discourse. *Lingvisticæ Investigationes*, 32(2):279–292, 2009.

S. Baccianella, A. Esuli, and F. Sebastiani. Multi-facet Rating of Product Reviews. In *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval (ECIR 2009)*, pages 461–472. Springer, 2009.

S. Baccianella, A. Esuli, and F. Sebastiani. SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC 2010)*, volume 10, pages 2200–2204, 2010.

S. Bagchi, G. Biswas, and K. Kawamura. Task Planning Under Uncertainty Using a Spreading Activation Network. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 30(6):639–650, 2000.

M. E. Basiri, A. R. Naghsh-Nilchi, and N. Ghasem-Aghaee. Sentiment Prediction Based on Dempster-Shafer Theory of Evidence. *Mathematical Problems in Engineering*, 2014.

S. Bethard, M. Carpuat, D. Cer, D. Jurgens, P. Nakov, and T. Zesch, editors. *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. Association for Computational Linguistics, 2016.

B. Bickart and R. M. Schindler. Internet Forums as Influential Sources of Consumer Information. *Journal of Interactive Marketing*, 15(3):31–40, 2001.

C. Biemann and A. Mehler. *Text Mining: From Ontology Learning to Automated Text Processing Applications*. Springer, 2014.

S. Bird, E. Loper, and E. Klein. *Natural Language Processing with Python*. OâĂŹReilly Media Inc., 2009.

S. Blair-Goldensohn, T. Neylon, K. Hannan, G. A. Reis, R. Mcdonald, and J. Reynar. Building a Sentiment Summarizer for Local Service Reviews. In *Proceedings of WWW 2008 Workshop on NLP in the Information Explosion Era (NLPIX 2008)*. ACM, 2008.

D. M. Blei and P. J. Moreno. Topic Segmentation with an Aspect Hidden Markov Model. In *Proceedings of the 24th Annual International ACM SIGIR Conference on*

Research and Development in Information Retrieval (SIGIR 2001), pages 343–348. ACM, 2001.

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

J. Bloch. *Effective Java, 2nd Edition*. Addison-Wesley Professional, 2008.

Bloomberg. Finding novel ways to trade on sentiment data, 2017. [WWW document] `https://www.bloomberg.com/professional/blog/finding-novel-ways-trade-sentiment-data/` (accessed 30th August 2017).

P. F. Bone. Word-of-mouth Effects on Short-term and Long-term Product Judgments. *Journal of Business Research*, 32(3):213 – 223, 1995.

F. Boon, L. de Graaf, D. van Ham, H. Kuilboer, and Z. Wang. Semantics-Driven Aspect Based Sentiment Analysis, 2016. [WWW document] `www.kimschouten.com/papers/unpublished/seminar2016_BAQM_ABSASentiment.pdf` (accessed 19th July 2017).

S. Brody and N. Elhadad. An Unsupervised Aspect-Sentiment Model for Online Reviews. In *Proceedings of the Conference of the North American Chapter of the Association of Computational Linguistics: Human Language Technologies 2010 (HLT-NAACL 2010)*, pages 804–812. Association for Computational Linguistics, 2010.

T. Brychcín, M. Konkol, and J. Steinberger. UWB: Machine Learning Approach to Aspect-Based Sentiment Analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 817–822. Association for Computational Linguistics and Dublin City University, 2014.

C. R. C. Brun, D. N. Popa. XRCE: Hybrid Classification for Aspect-based Sentiment Analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 838–842. Association for Computational Linguistics and Dublin City University, 2014.

E. Cambria. Affective Computing and Sentiment Analysis. *IEEE Intelligent Systems*, 31(2):102–107, 2016.

E. Cambria and A. Hussain. *Sentic Computing: A Common-Sense-Based Framework for Concept-Level Sentiment Analysis*. Springer, 2015.

E. Cambria, B. Schuller, Y. Xia, and C. Havasi. New Avenues in Opinion Mining and Sentiment Analysis. *IEEE Intelligent Systems*, 28(2):15–21, 2013.

E. Cambria, D. Olsher, and D. Rajagopal. SenticNet 3: A Common and Common-Sense Knowledge Base for Cognition-driven Sentiment Analysis. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI 2014)*, pages 1515–1521. AAAI, 2014.

G. Carenini, R. T. Ng, and A. Pauls. Multi-Document Summarization of Evaluative Text. In *Proceedings of the 11th Conference of the European Chapter of the ACL (EACL 2006)*, pages 305–312. Association for Computational Linguistics, 2006.

G. Castellucci, S. Filice, D. Croce, and R. Basili. UNITOR: Aspect Based Sentiment Analysis with Structured Learning. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 761–767. Association for Computational Linguistics and Dublin City University, 2014.

C. Cesarano, B. Dorr, A. Picariello, D. Reforgiato, A. Sagoff, and V. Subrahmanian. OASYS: An Opinion Analysis System. In *AAAI Spring Symposium on Computational Approaches to Analyzing Weblogs (CAAW 2006)*, pages 21–26, 2004.

S. W. Chan and M. W. Chong. Sentiment Analysis in Financial Texts. *Decision Support Systems*, 94:53 – 64, 2017. ISSN 0167-9236.

C.-C. Chang and C.-J. Lin. LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Y. Chen and J. Xie. Online Consumer Review: Word-of-Mouth as a New Element of Marketing Communication Mix. *Management Science*, 54(3):477–491, 2008.

Y. Choi and C. Cardie. Learning with Compositional Semantics as Structural Inference for Subsentential Sentiment Analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2008)*, pages 793–801, 2008.

K. W. Church and P. Hanks. Word Association Norms, Mutual Information, and Lexicography. *Computational Linguistics*, 16(1):22–29, 1990.

A. Coden, D. Gruhl, N. Lewis, P. N. Mendes, M. Nagarajan, C. Ramakrishnan, and S. Welch. Semantic Lexicon Expansion for Concept-Based Aspect-Aware Sentiment Analysis. In *Proceedings of the 1st Semantic Web Evaluation Challenge (SemWebEval 2014)*, volume 475, pages 34–40. Springer, 2014.

Collins English Dictionary. Collins English Dictionary âĂŞ Complete and Unabridged, June 2015. URL `http://www.thefreedictionary.com`. "Opinion".

K. Cortis, A. Freitas, T. Dauert, M. Huerlimann, M. Zarrouk, S. Handschuh, and B. Davis. SemEval-2017 Task 5: Fine-Grained Sentiment Analysis on Financial Microblogs and News. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 510–526. Association for Computational Linguistics, 2017.

K. Crammer and Y. Singer. Pranking with Ranking. In *Advances in Neural Information Processing Systems 14 (NIKS 2001)*, pages 641–647. MIT Press, 2001.

F. Crestani. Application of Spreading Activation Techniques in Information Retrieval. *Artificial Intelligence Review*, 11(6):453–582, 1997.

H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, N. Aswani, I. Roberts, G. Gorrell, A. Funk, A. Roberts, D. Damljanovic, T. Heitz, M. A. Greenwood, H. Saggion, J. Petrak, Y. Li, and W. Peters. *Text Processing with GATE (Version 6)*. GATE, 2011.

J. De Knijff, F. Frasincar, and F. Hogenboom. Domain Taxonomy Learning from Text: The Subsumption Method versus Hierarchical Clustering. *Data & Knowledge Engineering*, 83:54–69, 2013.

M.-C. de Marneffe and C. D. Manning. *Stanford Typed Dependencies Manual*, September 2008.

O. de Rooij, A. Vishneuski, and M. de Rijke. xTAS: Text Analysis in a Timely Manner. In *Proceedings of the 12th Dutch-Belgian Information Retrieval Workshop (DIR 2012)*, pages 89–90. ACM, 2012.

L. R. Dice. Measures of the Amount of Ecologic Association Between Species. *Ecology*, 26(3):297–302, 1945.

N. Dosoula, R. Griep, R. den Ridder, R. Slangen, **K. Schouten**, and F. Frasincar. Detection of Multiple Implicit Features per Sentence in Consumer Review Data. In G. Arnicans, V. Arnicane, J. Borzovs, and L. Niedrite, editors, *Proceedings of the 12th International Baltic Conference, (DB&IS 2016)*, pages 289–303, Cham, 2016. Springer International Publishing.

M. Dragoni, A. Tettamanzi, and C. da Costa Pereira. A Fuzzy System for Concept-Level Sentiment Analysis. In *Semantic Web Evaluation Challenge*, pages 21–27. Springer, 2014.

D. A. duVerle and H. Prendinger. A Novel Discourse Parser Based on Support Vector Machine Classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2 (ACL 2009)*, pages 665–673. Association for Computational Linguistics, 2009.

A. Esuli and F. Sebastiani. SentiWordNet: A publicly available lexical resource for opinion mining. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC 2006)*, volume 6, pages 417–422. European Language Resources Association (ELRA), 2006.

W. Fan, L. Wallace, S. Rich, and Z. Zhang. Tapping the Power of Text Mining. *Communications of the ACM*, 49(9):76–82, Sept. 2006. ISSN 0001-0782.

M. Farhadloo, R. A. Patterson, and E. Rolland. Modeling Customer Satisfaction from Unstructured Data using a BayesianÂăApproach. *Decision Support Systems*, 90:1 – 11, 2016. ISSN 0167-9236.

F. Fdez-Riverola, D. Glez-PeÃśa, H. LÃşpez-FernÃąndez, M. Reboiro-Jato, and J. MÃĺndez. A JAVA Application Framework for Scientific Software Development. *Software: Practice and Experience*, 42(8):1015–1036, 2012.

R. Feldman. Techniques and Applications for Sentiment Analysis. *Communications of the ACM*, 56(4):82–89, 2013.

C. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

V. W. Feng and G. Hirst. Text-level Discourse Parsing with Rich Linguistic Features. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, pages 60–68. Association for Computational Linguistics, 2012.

E. Frank, M. A. Hall, and I. H. Witten. *The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques", Fourth Edition*. Morgan Kaufmann, 2016.

E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional, 1994.

M. Ganapathibhotla and B. Liu. Mining Opinions in Comparative Sentences. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, pages 241–248. Association for Computational Linguistics, 2008.

G. Ganu, N. Elhadad, and A. Marian. Beyond the Stars: Improving Ratig Predictions using Review Content. In *Proceedings of the 12th International Workshop on the Web and Databases (WebDB 2009)*, 2009.

A. Garcia-Pablos, M. Cuadros, S. Gaines, and G. Rigau. V3: Unsupervised Genration of Domain Aspect Terms for Aspect Based Sentiment Analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 833–837, Dublin, Ireland, 2014.

R. E. Goldsmith and D. Horowitz. Measuring Motivations for Online Opinion Seeking. *Journal of Interactive Advertising*, 6(2):3–14, 2006.

N. Guarino and C. Welty. Evaluating Ontological Decisions with OntoClean. *Communications of the ACM*, 45(2):61–65, 2002.

E. Haddi, X. Liu, and Y. Shi. The role of text pre-processing in sentiment analysis. *Procedia Computer Science*, 17:26 – 32, 2013.

Z. Hai, K. Chang, and J. Kim. Implicit Feature Identification via Co-occurrence Association Rule Mining. In *Proceedings of the 12th International Conference on*

Computational Linguistics and Intelligent Text Processing (CICLing 2011), pages 393–404. Springer, 2011.

Z. Hai, G. Cong, K. Chang, W. Liu, and P. Cheng. Coarse-to-fine Review Selection via Supervised Joint Aspect and Sentiment Model. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2014)*, pages 617–626. ACM, 2014.

Z. S. Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954.

V. Hatzivassiloglou and K. R. McKeown. Predicting the Semantic Orientation of Adjectives. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics (ACL 1997)*, pages 174–181. Morgan Kaufman Publishers / Association for Computational Linguistics, 1997.

B. Heerschop, F. Goossen, A. Hogenboom, F. Frasincar, U. Kaymak, and F. de Jong. Polarity Analysis of Texts using Discourse Structure. In *Proceedings of the 20th ACM Conference on Information and Knowledge Management (CIKM 2011)*, pages 1061–1070. ACM, 2011.

H. Hernault, H. Prendinger, D. A. duVerle, M. Ishizuka, et al. HILDA: a Discourse Parser using Support Vector Machine Classification. *Dialogue and Discourse*, 1(3): 1–33, 2010.

T. Hofmann. Learning the Similarity of Documents: An Information-Geometric Approach to Document Retrieval and Categorization. In *Advances in Neural Information Processing Systems (NIPS 2000*, pages 914–920. MIT Press, 2000.

A. Hogenboom, P. van Iterson, B. Heerschop, F. Frasincar, and U. Kaymak. Determining Negation Scope and Strength in Sentiment Analysis. In *2011 IEEE International Conference on Systems, Man, and Cybernetics (SMC 2011)*, pages 2589–2594. IEEE, 2011.

A. Hogenboom, F. Frasincar, F. de Jong, and U. Kaymak. Using Rhetorical Structure in Sentiment Analysis. *Communications of the ACM*, 58(7):69–77, 2015.

R. Hoogervorst, E. Essink, W. Jansen, M. van den Helder, **K. Schouten**, F. Frasincar, and M. Taboada. Aspect-Based Sentiment Analysis on the Web Using Rhetorical Structure Theory. In *Proceedings of the 16th International Conference on Web Engineering (ICWE 2016)*, volume 9671 of *Lecture Notes in Computer Science*, pages 317–334, 2016.

M. Hu and B. Liu. Mining Opinion Features in Customer Reviews. In *Proceedings of the 19th National Conference on Artifical Intelligence (AAAI 2004)*, pages 755–760. AAAI, 2004a.

M. Hu and B. Liu. Mining and Summarizing Customer Reviews. In *Proceedings of 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2004)*, pages 168–177. ACM, 2004b.

R. A. Hummel and S. W. Zucker. On the Foundations of Relaxation Labeling Processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(3):267–287, 1983.

N. Jakob and I. Gurevych. Extracting Opinion Targets in a Single- and Cross-Domain Setting with Conditional Random Fields. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, pages 1035–1045. Association for Computational Linguistics, 2010.

K. Järvelin and J. Kekäläinen. Cumulated Gain-based Evaluation of IR Techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002.

A. Jensen and N. Boss. Textual Similarity: Comparing Texts in Order to Discover How Closely They Discuss the Same Topics. Bsc. thesis, Technical University of Denmark, 2008. `http://etd.dtu.dk/thesis/220969/bac08_15.pdf`.

L. Jia, C. Yu, and W. Meng. The Effect of Negation on Sentiment Analysis and Retrieval Effectiveness. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM 2009)*, pages 1827–1830. ACM, 2009.

O. Jin, N. N. Liu, K. Zhao, Y. Yu, and Q. Yang. Transferring Topical Knowledge from Auxiliary Long Texts for Short Text Clustering. In *Proceedings of the*

20th ACM International Conference on Information and Knowledge Management (CIKM 2011), pages 775–784. ACM, 2011.

W. Jin, H. H. Ho, and R. K. Srihari. OpinionMiner: a Novel Machine Learning System for Web Opinion Mining and Extraction. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2009)*, pages 1195–1204. ACM, 2009.

N. Jindal and B. Liu. Identifying Comparative Sentences in Text Documents. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2006)*, pages 244–251. ACM, 2006.

N. Jindal and B. Liu. Opinion Spam and Analysis. In *Proceedings of the International Conference on Web Search and Web Data Mining (WSDM 2008)*, pages 219–230. ACM, 2008.

JLanguageTool. `wiki.languagetool.org/java-api`, 2016.

Y. Jo and A. H. Oh. Aspect and Sentiment Unification Model for Online Review Analysis. In *Proceedings of the Forth International Conference on Web Search and Web Data Mining (WSDM 2011)*, pages 815–824. ACM, 2011.

H. Kamp and U. Reyle. *From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*, volume 42. Springer Science & Business Media, 1993.

A. Katifori, C. Vassilakis, and A. Dix. Ontologies and the Brain: Using Spreading Activation through Ontologies to Support Personal Interaction. *Cognitive Systems Research*, 11(1):25–41, 2010.

S. Keerthi, S. Shevade, C. Bhattacharyya, and K. Murthy. Improvements to Platt's SMO Algorithm for SVM Classifier Design. *Neural Computation*, 13(3):637–649, 2001.

J. S. Kessler and N. Nicolov. Targeting Sentiment Expressions through Supervised Ranking of Linguistic Configurations. In *Proceedings of the 3rd International AAAI*

Conference on Weblogs and Social Media (ICWSM 2009), pages 90–97. AAAI, 2009.

A. Kilgariff and M. Palmer, editors. *Proceedings of the Pilot SensEval*. Association for Computational Linguistics, September 1998.

H. D. Kim and C. Zhai. Generating Comparative Summaries of Contradictory Opinions in Text. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM 2009)*, pages 385–394. ACM, 2009.

S.-M. Kim and E. Hovy. Determining the Sentiment of Opinions. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*. Association for Computational Linguistics, 2004.

Y. Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014*, pages 1746–1751. The Association for Computer Linguistics, 2014.

S. Kiritchenko, X. Zhu, C. Cherry, and S. M. Mohammad. NRC-Cananda-2014: Detecting Aspects and Sentiment in Customer Reviews. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 437–442. Association for Computational Linguistics and Dublin City University, 2014.

N. Kobayashi, R. Iida, K. Inui, and Y. Matsumoto. Opinion mining on the web by extracting subject-aspect-evaluation relations. In *Proceedings of the AAAI Spring Symposium 2006: Computational Approaches to Analyzing Weblogs (AAAI-SS 2006)*, pages 86–91. AAAI, 2006.

H. Krekel. execnet: Distributed Python Deployment and Communication, 2017. [WWW document] `http://codespeak.net/execnet/` (accessed 19th July 2017).

L.-W. Ku, Y.-T. Liang, and H.-H. Chen. Opinion Extraction, Summarization and Tracking in News and Blog Corpora. In *Proceedings of the AAAI Spring Symposium 2006: Computational Approaches to Analyzing Weblogs (AAAI-SS 2006)*, pages 100–107. AAAI, 2006.

S. Kullback and R. A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

B. S. Kumar and V. Ravi. A Survey of the Applications of Text Mining in Financial Domain. *Knowledge-Based Systems*, 114(C):128–147, Dec. 2016. ISSN 0950-7051.

J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*, pages 282–289. Morgan Kaufmann Publishers Inc., 2001.

H. Lakkaraju, C. Bhattacharyya, I. Bhattacharya, and S. Merugu. Exploiting Coherence for the Simultaneous Discovery of Latent Facets and Associated Sentiments. In *SIAM International Conference on Data Mining 2011 (SDM 2011)*, pages 498–509. SIAM, 2011.

A. Lascarides, N. Asher, and J. Oberlander. Inferring Discourse Relations in Context. In *Proceedings of the 30th Annual Meeting on Association for Computational Linguistics (ACL 1992)*, pages 1–8. Association for Computational Linguistics, 1992.

R. Y. Lau, C. Li, and S. S. Liao. Social Analytics: Learning Fuzzy Product Ontologies for Aspect-Oriented Sentiment Analysis. *Decision Support Systems*, 65(C):80–94, 2014.

A. Lazaridou, I. Titov, and C. Sporleder. A Bayesian Model for Joint Unsupervised Induction of Sentiment, Aspect and Discourse Representations. In *Proceedings of the 51th Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, pages 1630–1639. Association for Computational Linguistics, 2013.

M. Lesk. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of the Fifth Annual International Conference on Systems Documentation (SIGDOC 1986)*, pages 24–26. ACM, 1986.

F. Li, C. Han, M. Huang, X. Zhu, Y.-J. Xia, S. Zhang, and H. Yu. Structure-Aware Review Mining and Summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, pages 653–661. Association for Computational Linguistics, 2010.

Z. Li, M. Zhang, S. Ma, B. Zhou, and Y. Sun. Automatic Extraction for Product Feature Words from Comments on the Web. In *Proceedings of the 5th Asia Information Retrieval Symposium on Information Retrieval Technology (AIRS 2009)*, pages 112–123. Springer, 2009.

B. Liu. *Sentiment Analysis and Opinion Mining*, volume 16 of *Synthesis Lectures on Human Language Technologies*. Morgan & Claypool, 2012.

B. Liu, M. Hu, and J. Cheng. Opinion Observer: Analyzing and Comparing Opinions on the Web. In *Proceedings of the 14th International Conference on World Wide Web (WWW 2005)*, pages 342–351. ACM, 2005.

C.-L. Liu, W.-H. Hsaio, C.-H. Lee, G.-C. Lu, and E. Jou. Movie Rating and Review Summarization in Mobile Environment. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 42(3):397–407, 2012.

C. Long, J. Zhang, and X. Zhut. A Review Selection Approach for Accurate Feature Rating Estimation. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, pages 766–774. Association for Computational Linguistics, 2010.

A. Lourenço, R. Carreira, S. Carneiro, P. Maia, D. Glez-Peña, F. Fdez-Riverola, E. C. Ferreira, I. Rocha, and M. Rocha. @Note: A Workbench for Biomedical Text Mining. *Journal of Biomedical Informatics*, 42(4):710 – 720, 2009.

B. Lu, M. Ott, C. Cardie, and B. K. Tsou. Multi-Aspect Sentiment Analysis with Topic Models. In *Proceedings of the 2011 IEEE 11th International Conference on Data Mining Workshops (ICDMW 2011)*, pages 81–88. IEEE, 2011.

D. Lyubimov and A. Palumbo. *Apache Mahout: Beyond MapReduce*. CreateSpace Independent Publishing Platform, 2016.

A. Maedche and S. Staab. Ontology Learning for the Semantic Web. *IEEE Intelligent Systems*, 16(2):72–79, 2001.

W. C. Mann and S. A. Thompson. Rhetorical Structure Theory: Toward a Functional Theory of Text Organization. *Text*, 8(3):243–281, 1988.

C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60. Association for Computational Linguistics, 2014.

D. Marcheggiani, O. Täckström, A. Esuli, and F. Sebastiani. Hierarchical Multi-label Conditional Random Fields for Aspect-Oriented Opinion Mining. In *Proceedings of the 36th European Conference on Information Retrieval (ECIR 2014)*, pages 273–285. Springer, 2014.

D. Marcu. The Rhetorical Parsing of Natural Language Texts. In *Proceedings of the Eighth Conference on European Chapter of the Association for Computational Linguistics (EACL 1997)*, pages 96–103. Association for Computational Linguistics, 1997.

A. K. McCallum. *MALLET: A Machine Learning for Language Toolkit.* `http://mallet.cs.umass.edu`, 2002.

Q. Mei, X. Ling, M. Wondra, H. Su, and C. Zhai. Topic Sentiment Mixture: Modeling Facets and Opinions in Weblogs. In *Proceedings of the 16th International Conference on World Wide Web (WWW 2007)*, pages 171–180. ACM, 2007.

X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen, D. Xin, R. Xin, M. J. Franklin, R. Zadeh, M. Zaharia, and A. Talwalkar. MLib: Machine Learning in Apache Spark. *Journal of Machine Learning Research*, 17(1):1235–1241, 2016.

R. Mihalcea. SemCor-3, 2016. [WWW document] `http://web.eecs.umich.edu/~mihalcea/downloads.html#semcor` (accessed 19th July 2017).

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

T. M. Mitchell. *Machine Learning.* McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997. ISBN 0070428077, 9780070428072.

S. Moghaddam and M. Ester. Opinion Digger: an Unsupervised Opinion Miner from Unstructured Product Reviews. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM 2010)*, pages 1825–1828. ACM, 2010.

S. Moghaddam and M. Ester. ILDA: Interdependent LDA Model for Learning Latent Aspects and their Ratings from Online Product Reviews. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2011)*, pages 665–674. ACM, 2011.

S. Moghaddam and M. Ester. On the Design of LDA Models for Aspect-Based Opinion Mining. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM 2012)*, pages 803–812. ACM, 2012.

S. Moghaddam and M. Ester. The FLDA Model for Aspect-based Opinion Mining: Addressing the Cold Start Problem. In *Proceedings of the 22nd International Conference on World Wide Web (WWW 2013)*, pages 909–918. ACM, 2013a.

S. Moghaddam and M. Ester. Tutorial at WWW 2013: 'Opinion Mining in Online Reviews: Recent Trends', 2013b. [WWW document] `http://www.cs.sfu.ca/~ester/papers/WWW2013.Tutorial.Final.pdf` (accessed 19th July 2017).

K. Moilanen and S. Pulman. Sentiment Composition. In *Proceedings of Recent Advances in Natural Language Processing 2007 (RANLP 2007)*, pages 378–382, 2007.

A. Mukherjee and B. Liu. Modeling Review Comments. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, pages 320–329. Association for Computational Linguistics, 2012.

T. Mullen and N. Collier. Sentiment Analysis using Support Vector Machines with Diverse Information Sources. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, volume 4, pages 412–418. ACL, 2004.

R. Narayanan, B. Liu, and A. Choudhary. Sentiment Analysis of Conditional Sentences. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2009)*, pages 180–189. Association for Computational Linguistics, 2009.

T. Nasukawa and J. Yi. Sentiment Analysis: Capturing Favorability Using Natural Language Processing. In *Proceedings of the 2nd International Conference on Knowledge Capture (K-CAP 2003)*, pages 70–77. ACM, 2003.

R. Navigli and P. Velardi. Semantic Interpretation of Terminological Strings. In *Proceedings of 6th International Conference Terminology and Knowledge Engineering (TKE 2002)*, pages 95–100, 2002.

O. Netzer, R. Feldman, J. Goldenberg, and M. Fresko. Mine Your Own Business: Market-Structure Surveillance through Text Mining. *Marketing Science*, 31(3): 521–543, 2012.

T. H. Nguyen, K. Shirai, and J. Velcin. Sentiment analysis on social media for stock movement prediction. *Expert Systems with Applications*, 42(24):9603 – 9611, 2015.

J. Nickolls, I. Buck, M. Garland, and K. Skadron. Scalable Parallel Programming with CUDA. *Queue*, 6(2):40–53, 2008. ISSN 1542-7730.

J. Nivre, J. Hall, J. Nilsson, A. Chanev, G. Eryigit, S. Kübler, S. Marinov, and E. Marsi. MaltParser: A Language-independent System for Data-driven Dependency Parsing. *Natural Language Engineering*, 13(2):95âĂŞ135, 2007.

B. Pang and L. Lee. Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2008.

B. Pang, L. Lee, and S. Vaithyanathan. Thumbs Up?: Sentiment Classification Using Machine Learning Techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 79–86. Association for Computational Linguistics, 2002.

Y. Park, R. J. Byrd, and B. K. Boguraev. Automatic Glossary Extraction: Beyond Terminology Identification. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002)*, pages 1–7. ACL, 2002.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

I. Peñalver-Martinez, F. Garcia-Sanchez, R. Valencia-Garcia, M. Ángel Rodríguez-García, V. Moreno, A. Fraga, and J. L. Sánchez-Cervantes. Feature-Based Opinion Mining through Ontologies. *Expert Systems with Applications*, 41(13):5995 – 6008, 2014.

L. Polanyi and A. Zaenen. *Contextual Valence Shifters*, pages 1–10. Springer, 2006.

M. Pontiki, D. Galanis, J. Pavlopoulos, H. Papageorgiou, I. Androutsopoulos, and S. Manandhar. Semeval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35. Association for Computational Linguistics and Dublin City University, 2014.

M. Pontiki, D. Galanis, H. Papageorgiou, S. Manandhar, and I. Androutsopoulos. SemEval-2015 Task 12: Aspect Based Sentiment Analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495. Association for Computational Linguistics, 2015.

M. Pontiki, D. Galanis, H. Papageorgiou, I. Androutsopoulos, S. Manandhar, M. AL-Smadi, M. Al-Ayyoub, Y. Zhao, B. Qin, O. De Clercq, V. Hoste, M. Apidianaki, X. Tannier, N. Loukachevitch, E. Kotelnikov, N. Bel, S. M. Jiménez-Zafra, and G. Eryiğit. SemEval-2016 Task 5: Aspect Based Sentiment Analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, San Diego, California, June 2016. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/S16-1002`.

A.-M. Popescu and O. Etzioni. Extracting Product Features and Opinions from Reviews. In *Proceedings of the Conference on Human Language Technology and Conference on Empirical Methods in Natural Language Processing 2005 (HLT/EMNLP 2005)*, pages 339–346. Association for Computational Linguistics, 2005.

Project Gutenberg. Project Gutenberg, 2017. [WWW document] `http://www.gutenberg.org/wiki/Main_Page` (accessed 19th July 2017).

PubMed. PubMed Central, 2017. [WWW document] `https://www.ncbi.nlm.nih.gov/pmc/` (accessed 19th July 2017).

G. Qiu, B. Liu, J. Bu, and C. Chen. Expanding Domain Sentiment Lexicon Through Double Propagation. In *Proceedings of the 21st International Jont Conference on Artifical Intelligence (IJCAI 2009)*, pages 1199–1204. Morgan Kaufmann Publishers Inc., 2009.

G. Qiu, B. Liu, J. Bu, and C. Chen. Opinion Word Expansion and Target Extraction through Double Propagation. *Computational Linguistics*, 37(1):9–27, 2011.

S. Raju, P. Pingali, and V. Varma. An Unsupervised Approach to Product Attribute Extraction. In *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval (ECIR 2009)*, pages 796–800. Springer, 2009.

D. Reforgiate Recupero, V. Presutti, S. Consoli, A. Gangemi, and N. A.G. Sentilo: Frame-Based Sentiment Analysis. *Cognitive Computing*, 7(2):211–225, 2015.

S. E. Robertson, S. Walker, S. Jones, et al. Okapi at TREC-3. In *Proceedings of the Third Text REtrieval Conference (TREC-3)*, pages 109–126, 1995.

J. Saias. Sentiue: Target and aspect based sentiment analysis in semeval-2015 task 12. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 767–771. Association for Computational Linguistics, 2015.

G. Salton. Mathematics and information retrieval. *Journal of Documentation*, 35(1): 1–29, 1979.

G. Salton and M. McGill. *Introduction to Modern Information Retrieval*. McGraw Hill, 1983.

M. Sanderson and B. Croft. Deriving Concept Hierarchies from Text. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1999)*, pages 206–213. ACM, 1999.

C. Sauper and R. Barzilay. Automatic Aggregation by Joint Modeling of Aspects and Values. *Journal of Artificial Intelligence Research*, 46(1):89–127, 2013. ISSN 1076-9757.

C. Scaffidi, K. Bierhoff, E. Chang, M. Felker, H. Ng, and C. Jin. Red Opal: Product-Feature Scoring from Reviews. In *Proceedings of the 8th ACM Conference on Electronic Commerce (EC 2007)*, pages 182–191. ACM, 2007.

K. Schouten and F. Frasincar. Finding implicit features in consumer reviews for sentiment analysis. In *Proceedings of the 14th International Conference on Web Engineering (ICWE 2014)*, pages 130–144, 2014.

K. Schouten and F. Frasincar. Survey on Aspect-Level Sentiment Analysis. *IEEE Transactions on Knowledge and Data Engineering*, 28(3):813–830, 2016.

K. Schouten, F. Frasincar, and F. de Jong. COMMIT-P1WP3: a Co-occurrence Based Approach to Aspect-level Sentiment Analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 203–207. Association for Computational Linguistics and Dublin City University, 2014.

K. Schouten, F. Frasincar, and F. de Jong. COMMIT at SemEval-2017 Task 5: Ontology-based Method for Sentiment Analysis of Financial Headlines. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval 2017)*, pages 883–887. Association for Computational Linguistics, 2017a.

K. Schouten, F. Frasincar, and F. de Jong. Ontology-Enhanced Aspect-Based Sentiment Analysis. In *Proceedings of the 17th International Conference on Web Engineering (ICWE 2017)*, pages 302–320. Springer International Publishing, 2017b.

K. Schouten, O. van der Weijde, F. Frasincar, and R. Dekker. Supervised and unsupervised aspect category detection for sentiment analysis with co-cccurrence data. *IEEE Transactions on Cybernetics*, TBD(TBD):TBD, 2017c. ISSN 2168-2267. doi: 10.1109/TCYB.2017.2688801.

R. P. Schumaker, Y. Zhang, C.-N. Huang, and H. Chen. Evaluating Sentiment in Financial News Articles. *Decision Support Systems*, 53(3):458 – 464, 2012. ISSN 0167-9236.

R. P. Schumaker, A. T. Jarmoszko, and C. S. Labedz. Predicting wins and spread in the premier league using a sentiment analysis of twitter. *Decision Support Systems*, 88:76 – 84, 2016. ISSN 0167-9236.

S. Sen and D. Lerman. Why are you telling me this? An Examination into Negative Consumer Reviews on the Web. *Journal of Interactive Marketing*, 21(4):76–94, 2007.

G. Shafer. *A Mathematical Theory of Evidence*, volume 1. Princeton University Press, 1976.

D. Smith, S. Menon, and K. Sivakumar. Online Peer and Editorial Recommendations, Trust and Choice in Virtual Markets. *Journal of Interactive Marketing*, 73(5):90–102, 2005.

R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. P. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods on Natural Language Processing (EMNLP 2013)*, pages 1631–1642. Association for Computational Linguistics, 2013.

R. Soricut and D. Marcu. Sentence Level Discourse Parsing Using Syntactic and Lexical Information. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1 (NAACL 2003)*, pages 149–156. Association for Computational Linguistics, 2003.

P. R. Spence, K. A. Lachlan, X. Lin, and M. del Greco. Variability in twitter content across the stages of a natural disaster: Implications for crisis communication. *Communication Quarterly*, 63(2):171–186, 2015.

Q. Su, K. Xiang, H. Wang, B. Sun, and S. Yu. Using pointwise mutual information to identify implicit features in customer reviews. In Y. Matsumoto, R. Sproat, K.-F. Wong, and M. Zhang, editors, *Computer Processing of Oriental Languages. Beyond the Orient: The Research Challenges Ahead*, volume 4285 of *Lecture Notes in Computer Science*, pages 22–30. Springer Berlin Heidelberg, 2006.

Q. Su, X. Xu, H. Guo, Z. Guo, X. Zang, B. Swen, and Z. Su. Hidden Sentiment Association in Chinese Web Opinion Mining. In *Proceedings of the 17th Conference on World Wide Web (WWW 2008)*, pages 959–968. ACM, 2008.

M. Surdeanu, T. Hicks, and M. A. Valenzuela-Escárcega. Two Practical Rhetorical Structure Theory Parsers. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL HLT 2015)*, pages 1–5. Association for Computational Linguistics, 2015.

M. Taboada, K. Voll, and J. Brooke. Extracting Sentiment as a Function of Discourse Structure and Topicality. Technical Report TR 2008-20, Simon Fraser University School of Computing Science, 2008. URL `ftp://fas.sfu.ca/pub/cs/TR/2008/CMPT2008-20.pdf`.

K. S. Tai, R. Socher, and C. D. Manning. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015*, pages 1556–1566. Association for Computational Linguistics, 2015.

H. Tang, S. Tan, and X. Cheng. A Survey on Sentiment Detection of Reviews. *Expert Systems with Applications*, 36(7):10760–10773, 2009a.

Y. Tang, Y.-Q. Zhang, N. V. Chawla, and S. Krasser. SVMs Modeling for Highly Imbalanced Classification. *IEEE Transactions on: Systems, Man, and Cybernetics, Part B: Cybernetics*, 39(1):281–288, 2009b.

R. N. Taylor, N. Medvidović, and E. M. Dashofy. *Software Architecture: Foundations, Theory, and Practice*. John Wiley and Sons, 2009.

P. Thakor and S. Sasi. Ontology-based Sentiment Analysis Process for Social Media Content. In *Proceedings of the INNS Conference on Big Data 2015*, pages 199 – 207. Springer, 2015.

M. Thelwall, K. Buckley, and G. Paltoglou. Sentiment Strength Detection for the Social Web. *Journal of the American Society for Information Science and Technology*, 63(1):163–173, 2012.

I. Titov and R. McDonald. A Joint Model of Text and Aspect Ratings for Sentiment Summarization. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (HLT 2008)*, pages 308–316. Association for Computational Linguistics, 2008a.

I. Titov and R. McDonald. Modeling Online Reviews with Multi-Grain Topic Models. In *Proceedings of the 17th International Conference on World Wide Web (WWW 2008)*, pages 111–120. ACM, 2008b.

Z. Toh and J. Su. Nlangp: Supervised machine learning system for aspect category classification and opinion target extraction. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 496–501. Association for Computational Linguistics, 2015.

C. Toprak, N. Jakob, and I. Gurevych. Sentence and Expression Level Annotation of Opinions in User-Generated Discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 575–584. Association for Computational Linguistics, 2010.

M. Trusov, R. Bucklin, and K. Pauwels. Effects of Word-of-mouth Versus Traditional Marketing: Findings from an Internet Social Networking Site. *Journal of Marketing*, 73(5):90–102, 2009.

M. Tsytsarau and T. Palpanas. Survey on Mining Subjective Data on the web. *Data Mining and Knowledge Discovery*, 24(3):478–514, 2012.

P. D. Turney. Thumbs Up or Thumbs Down?: Semantic Orientation Applied to Unsupervised Classification of Reviews. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL 2002)*, pages 417–424. Association for Computational Linguistics, 2002.

R. Usbeck, M. Röder, A.-C. Ngonga Ngomo, C. Baron, A. Both, M. Brümmer, D. Ceccarelli, M. Cornolti, D. Cherix, B. Eickmann, P. Ferragina, C. Lemke,

A. Moro, R. Navigli, F. Piccinno, G. Rizzo, H. Sack, R. Speck, R. Troncy, J. Waitelonis, and L. Wesemann. GERBIL: General Entity Annotation Benchmark Framework. In *Proceedings of the 24th Conference on World Wide Web (WWW 2015)*, pages 1133–1143. International World Wide Web Conferences Steering Committee, 2015.

E. van Kleef, H. C. van Trijp, and P. Luning. Consumer Research in the Early Stages of New Product Development: a Critical Review of Methods and Techniques. *Food Quality and Preference*, 16(3):181–201, 2005.

B. C. Wallace. Computational Irony: A Survey and New Perspectives. *Artificial Intelligence Review*, pages 1–17, 2013. ISSN 0269-2821.

H. Wang, Y. Lu, and C. Zhai. Latent Aspect Rating Analysis on Review Text Data: a Rating Regression Approach. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge discovery and Data Mining (KDD 2010)*, pages 783–792. ACM, 2010.

H. Wang, Y. Lu, and C. Zhai. Latent Aspect Rating Analysis without Aspect Keyword Supervision. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2011)*, pages 618–626. ACM, 2011.

W. Wang, H. Xu, and W. Wan. Implicit Feature Identification via Hybrid Association Rule Mining. *Expert Systems with Applications: an International Journal*, 40(9): 3518–3531, 2013.

Watchmaker. The watchmaker framework, 2017. [WWW document] `http://watchmaker.uncommons.org/` (accessed 19th July 2017).

T. White. *Hadoop: The Definitive Guide*. O'Reilly Media, Inc., 2009.

J. Wiebe, T. Wilson, and C. Cardie. Annotating Expressions of Opinions and Emotions in Language. *Language Resources and Evaluation*, 39(2):165–210, 2005.

T. Wilson, J. Wiebe, and P. Hoffmann. Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis. In *Proceedings of the Conference on Human Language*

Technology and Conference on Empirical Methods in Natural Language Processing 2005 (HLT/EMNLP 2005), pages 347–354. Association for Computational Linguistics, 2005.

I. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2nd edition, 2005.

Z. Wu and M. Palmer. Verbs Semantics and Lexical Selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linquistics*, pages 133–138. Association for Computational Linquistics, 1994.

Y. Xia, E. Cambria, A. Hussain, and H. Zhao. Word Polarity Disambiguation Using Bayesian Model and Opinion-Level Features. *Cognitive Computation*, 7(3):369–380, 2015.

B. Yang and C. Cardie. Context-aware Learning for Sentence-level Sentiment Analysis with Posterior Regularization. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, pages 325–335. Association for Computational Linguistics, 2014.

Yelp. Yelp Dataset Challenge, 2017. Retrieved 11-06-2017, from `https://www.yelp.nl/dataset_challenge`.

J. Yu, Z.-J. Zha, M. Wang, and T.-S. Chua. Aspect Ranking: Identifying Important Product Aspects from Online Consumer Reviews. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL 2011)*, pages 1496–1505. Association for Computational Linguistics, 2011.

G. Yule. *Pragmatics*. Oxford University Press, 1996.

M. Zaharia, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin, A. Ghodsi, J. Gonzalez, S. Shenker, and I. Stoica. Apache Spark: A Unified Engine for Big Data Processing. *Communications of the ACM*, 59(11):56–65, 2016.

T.-J. Zhan and C.-H. Li. Semantic Dependent Word Pairs Generative Model for Fine-Grained Product Feature Mining. In *Proceedings of the 15th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD 2011)*, pages 460–475. Springer, 2011.

L. Zhang, B. Liu, S. H. Lim, and E. O'Brien-Strain. Extracting and Ranking Product Features in Opinion Documents. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, pages 1462–1470. Association for Computational Linguistics, 2010.

Y. Zhang and W. Zhu. Extracting Implicit Features in Online Customer Reviews for Opinion Mining. In *Proceedings of the 22nd International Conference on World Wide Web Companion (WWW 2013 Companion)*, pages 103–104. International World Wide Web Conferences Steering Committee, 2013.

J. Zhao, H. Xu, X. Huang, S. Tan, K. Liu, and Q. Zhang. Overview of Chinese Opinion Analysis Evaluation 2008. In *Proceedings of the 1st Chinese Opinion Analysis Evaluation (COAE 2008)*, pages 1–21, 2008.

W. X. Zhao, J. Jiang, H. Yan, and X. Li. Jointly Modeling Aspects and Opinions with a MaxEnt-LDA Hybrid. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, pages 56–65. Association for Computational Linguistics, 2010a.

Y. Zhao, B. Qin, S. Hu, and T. Liu. Generalizing Syntactic Structures for Product Attribute Candidate Extraction. In *Proceedings of the Conference of the North American Chapter of the Association of Computational Linguistics: Human Language Technologies 2010 (HLT-NAACL 2010)*, pages 377–380. Association for Computational Linguistics, 2010b.

X. Zheng, Z. Lin, X. Wang, K.-J. Lin, and M. Song. Incorporating appraisal expression patterns into topic modeling for aspect and sentiment word identification. *Knowledge-Based Systems*, 61(1):29–47, 2014. ISSN 0950-7051.

Y. Zhou. Fine-grained Sentiment Analysis with Discourse Structure. Master's thesis, Saarland University, Germany, 2013. URL `http://lct-master.org/getfile.php?id=783&n=1&dt=TH&ft=pdf&type=TH`.

J. Zhu, H. Wang, B. K. Tsou, and M. Zhu. Multi-Aspect Opinion Polling from Textual Reviews. In *Proceedings of the 18th ACM Conference on Information and knowledge management (CIKM 2009)*, pages 1799–1802. ACM, 2009.

L. Zhuang, F. Jing, and X.-Y. Zhu. Movie Review Mining and Summarization. In *Proceedings of the 15th ACM International Conference on Information and Knowledge Management (CIKM 2006)*, pages 43–50. ACM, 2006.

C. Zirn, M. Niepert, H. Stuckenschmidth, and M. Strube. Fine-Grained Sentiment Analysis with Structural Features. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP 2011)*, pages 336–344. Association for Computational Linguistics, 2011.

Summary in English

People using the Web are constantly invited to share their opinions and preferences with the rest of the world, which has led to an explosion of opinionated blogs, reviews of products and services, and comments on virtually everything. This type of web-based content is increasingly recognized as a source of data that has added value for multiple application domains.

While the large number of available reviews almost ensures that all relevant parts of the entity under review are properly covered, manually reading each and every review is not feasible. Aspect-based sentiment analysis aims to solve this issue, as it is concerned with the development of algorithms that can automatically extract fine-grained sentiment information from a set of reviews, computing a separate sentiment value for the various aspects of the product or service being reviewed.

This dissertation starts with a careful review of the existing body of literature, organizing the various approaches in a clear taxonomy. In doing so, we discover that many of the older works do not use freely accessible data sets and evaluation metrics are often different. This prevents a clear analysis of the state-of-the art. However, more recently, thanks to a successful series of workshops on aspect-based sentiment analysis, an increasing number of newly developed approaches use standardized evaluation methods and open data sets. We conclude by stating that the future of aspect-based sentiment analysis can be found in more semantically-oriented approaches, which naturally integrate common sense knowledge, general world knowledge, and domain knowledge. This dissertation actively contributes to such a future. Next, we describe an open framework that can be used to build text mining algorithms, which supports a wide variety of libraries and enables a developer to easily use standardized academic evaluation procedures.

Then, a series of chapters deals with the development of a variety of algorithms for the sub-problems of aspect detection and aspect sentiment analysis. We start by looking at co-occurrence information, arguably the most basic piece of information that can be found in labeled textual data. We find that a relatively good perfor-

mance can be obtained using simple techniques. Next, a study is performed to investigate which intra-textual discriminants are useful for aspect-based sentiment analysis, where we analyze various features based on their Information Gain with respect to the sentiment analysis sub-task. We find that only some of the discriminants are actually informative. After that, we investigate the use of discourse information to give more weight to parts of the sentence that are relevant for a given aspect. In this way, we can naturally deal with the fact that a single sentence can have multiple aspects, each with different sentiment values.

Even though information regarding the discourse structure of text is high-level information, it is still only a source of intra-textual discriminants. Therefore, following the outlook presented in the literature survey, we move towards a knowledge-driven approach by implementing an ontology-based method for aspect-based sentiment analysis. We show that using external domain knowledge significantly improves aspect detection and is also beneficial for sentiment analysis. Furthermore, logic axioms are used to determine the sentiment value of context-dependent words, such as 'dry', where the sentiment value depends on the aspect they are describing (e.g., 'dry meat' vs. 'dry wine', with the first usually being a negative expression and the last a neutral one). We also present a semi-automatic method for ontology creation, to speed up the process of encoding external knowledge for sentiment analysis.

Nederlandse Samenvatting (Summary in Dutch)

Op het Web wordt iedereen structureel uitgenodigd om zijn of haar meningen en voorkeuren te delen met de wereld. Dit heeft geleid tot een explosieve groei van blogs, recensies van producten en services en commentaar op vrijwel alles. Dit soort online informatie wordt meer en meer gezien als een waardevolle bron van data die tot nut kan zijn voor verscheidene toepassingen.

Hoewel het grote aantal beschikbare recensies er voor zorgt dat alle relevante onderdelen van het gerecenseerde besproken worden, betekent dit ook dat het handmatig lezen van alle individuele recensies niet langer mogelijk is. *Aspect-based sentiment analysis* beoogt dit probleem op te lossen met behulp van algoritmes die op automatische wijze fijnmazige sentiment informatie uit een set recensies kunnen halen, waarbij afzonderlijke sentiment scores toegewezen worden aan de verschillende aspecten van een product of service die in een recensie genoemd worden.

Deze dissertatie begint met een zorgvuldig overzicht van de huidige literatuur, waarbij de verschillende bestaande technieken ingedeeld worden in een heldere taxonomie. Hierbij ontdekken wij dat veel van de wat oudere werken geen vrij toegankelijke data sets gebruiken en dat evaluatie methoden vaak verschillen. Dit maakt een duidelijke analyse van de *state-of-the-art* onmogelijk. Echter, dankzij een succesvolle serie van *aspect-based sentiment analysis* workshops gebruiken steeds meer recente werken gestandaardizeerde evaluatie methoden en open data sets. We eindigen het literatuur overzicht met de conclusie dat de toekomst van *aspect-based sentiment analysis* gevonden kan worden in meer semantisch geöriënteerde methoden, die op natuurlijke wijze informatie vloeiend uit het gezond verstand, algemene kennis van de wereld en domein-specifieke gegevens kunnen integreren. Deze dissertatie draagt bij aan de ontwikkeling in deze richting van het onderzoeksveld. Hierna volgt een beschrijving van een open software framework dat gebruikt kan worden om *text min-*

ing algoritmes te ontwikkelen. De software ondersteunt een wijde selectie van *libraries* en het geeft een ontwikkelaar eenvoudig toegang tot gestandaardizeerde academische evaluatie procedures.

De daaropvolgende serie hoofdstukken behandelen de ontwikkeling van verschillende algoritmes voor de sub-problemen van aspect detectie en de sentiment classificatie van gevonden aspecten. We beginnen met het kijken naar *co-occurrence* informatie, wat beschouwd kan worden als de meest basale informatie die uit gelabelde tekstuele data gehaald kan worden. We zien dat deze eenvoudige technieken al tot redelijk goede resultaten leiden. Hierna volgt een studie waarin onderzocht wordt welke intra-tekstuele discriminanten nuttig zijn voor *aspect-based sentiment analysis*. Van elke discriminant wordt uitgerekend wat de Information Gain is ten opzichte van de aspect sentiment classificatie sub-taak. Dit levert het inzicht op dat slechts een klein deel van de discriminanten daadwerkelijk informatief is en bijdraagt aan de kwaliteit van de sentiment voorspellingen. Het volgende hoofdstuk onderzoekt het gebruik van discourse informatie om meer gewicht te geven aan delen van een zin die relevant zijn voor een gegeven aspect. Op deze manier kunnen we omgaan met het feit dat een enkele zin meerdere aspecten kan bevatten, elk met een ander sentiment.

Hoewel informatie over de discourse structuur van een tekst van een hoog abstractie niveau is, is het nog steeds slechts een bron van intra-tekstuele discriminanten. Daarom, in navolging van de vooruitblik in de literatuur studie, verplaatsen we de focus naar meer kennisgedreven methodes met het implementeren van een methode waarbij het gebruik van externe informatie, zoals gecodeerd in een ontologie, centraal staat. We laten zien dat het gebruik van externe informatie een significante verbetering oplevert bij het detecteren van aspecten en dat het ook nuttig is bij het toewijzen van sentiment labels aan aspecten. De logische structuur van ontologieën maakt het daarbij ook mogelijk om axiomen te gebruiken om het sentiment te bepalen van woorden die context afhankelijk zijn (bijv. ‘kleine porties’ versus ‘kleine prijzen’, waarbij het eerste doorgaans als negatief wordt gezien en het tweede als positief). We presenteren ook een methode om op semi-automatische wijze een ontologie te creëren wat een flinke snelheidswinst oplevert ten opzichte van het handmatig coderen van externe informatie in een ontologie.

About the Author

Kim Schouten (1986) obtained the M.Sc. degree in Economics and Informatics, specializing in computational economics, from the Erasmus University Rotterdam, the Netherlands. Both bachelor and master thesis subjects were already in the domain of natural language processing, which is one of his main research interests.

This was followed by a Ph.D. candidacy on sentiment analysis at the Erasmus Research Institute of Management (ERIM) and the Econometric Insitutate at the Erasmus School of Economics as part of the Dutch national research program COMMIT/Infiniti. Furthermore, Kim was affiliated to the Dutch Research School for Information and Knowledge Systems (SIKS), Erasmus Center of Business Intelligence (ECBI), and Erasmus Studio.

Kim's research is at the crossroads of natural language processing, text mining, machine learning, artificial intelligence, and the Semantic Web. Over the years, he has published in several prestigious conferences and journals and actively participated in the benchmarking community. Furthermore, Kim has been part of the program committee of a hand full of workshops and conferences but also often reviewed papers for high-quality journals.

During his Ph.D. candidacy, Kim has successfully taught several IT related courses. For most of these years he has coordinated the first-year bachelor course Skills 1: IT at the International Bachelor of Business Administration.

Portfolio

Publications

S. de Kok, L. Punt, R. van den Puttelaar, K. Ranta, **K. Schouten**, and F. Frasincar. Review-level aspect-based sentiment analysis using an ontology. In *Proceeings of the 33rd Symposium on Applied Computing (SAC 2018)*, page to appear. ACM, 2018.

K. Schouten, F. Frasincar, and F. de Jong. COMMIT at SemEval-2017 Task 5: Ontology-based Method for Sentiment Analysis of Financial Headlines. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval 2017)*, pages 883–887. Association for Computational Linguistics, 2017a.

K. Schouten, F. Frasincar, and F. de Jong. Ontology-Enhanced Aspect-Based Sentiment Analysis. In *Proceedings of the 17th International Conference on Web Engineering (ICWE 2017)*, pages 302–320. Springer International Publishing, 2017b.

K. Schouten, O. van der Weijde, F. Frasincar, and R. Dekker. Supervised and unsupervised aspect category detection for sentiment analysis with co-cccurrence data. *IEEE Transactions on Cybernetics*, TBD(TBD):TBD, 2017c. ISSN 2168-2267.

D. de Heij, A. Troyanovsky, C. Yang, M. Z. Scharff, **K. Schouten**, and F. Frasincar. An ontology-enhanced hybrid approach to aspect-based sentiment analysis. In *Proceedings of the 18th International Conference on Web Information Systems Engineering (WISE 2017)*, pages 338–345. Springer, 2017.

K. Schouten and F. Frasincar. Survey on Aspect-Level Sentiment Analysis. *IEEE Transactions on Knowledge and Data Engineering*, 28(3):813–830, 2016b.

R. Hoogervorst, E. Essink, W. Jansen, M. van den Helder, **K. Schouten**, F. Frasincar, and M. Taboada. Aspect-Based Sentiment Analysis on the Web Using Rhetor-

ical Structure Theory. In *Proceedings of the 16th International Conference on Web Engineering (ICWE 2016)*, volume 9671 of *Lecture Notes in Computer Science*, pages 317–334, 2016.

K. Schouten and F. Frasincar. Commit at semeval-2016 task 5: Sentiment analysis with rhetorical structure theory. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval 2016)*, pages 356–360. ACL, 2016a.

K. Schouten, F. Baas, O. Bus, A. Osinga, N. van de Ven, S. van Loenhout, L. Vrolijk, and F. Frasincar. Aspect-based sentiment analysis using lexico-semantic patterns. In *Proceedings of the 17th International Conference on Web Information Systems Engineering (WISE 2016)*, volume 10042 of *Lecture Notes in Computer Science*, pages 35–42. Springer, 2016a.

K. Schouten, F. Frasincar, and R. Dekker. An information gain-driven feature study for aspect-based sentiment analysis. In *Proceedings of the 21st International Conference on Applications of Natural Language to Information Systems (NLDB 2016)*, volume 9612 of *Lecture Notes in Computer Science*, pages 48–59. Springer, 2016b.

K. Schouten and F. Frasincar. Using linguistic graph similarity to search for sentences in news articles. In *Selected Papers from the 12th International Baltic Conference on Databases and Information Systems (DB&IS 2016)*, volume 291 of *Frontiers in Artificial Intelligence and Applications*, pages 255–268. IOS Press, 2016c.

K. Schouten and F. Frasincar. Web news sentence searching using linguistic graph similarity. In *Proceedings of the 12th International Baltic Conference on Databases and Information Systems (DB&IS 2016)*, volume 615 of *Communications in Computer and Information Science*, pages 319–333. Springer, 2016d.

N. Dosoula, R. Griep, R. den Ridder, R. Slangen, **K. Schouten**, and F. Frasincar. Detection of Multiple Implicit Features per Sentence in Consumer Review Data. In G. Arnicans, V. Arnicane, J. Borzovs, and L. Niedrite, editors, *Proceedings of the 12th International Baltic Conference, (DB&IS 2016)*, pages 289–303, 2016a. Springer.

N. Dosoula, R. Griep, R. den Ridder, R. Slangen, R. van Luijk, **K. Schouten**, and F. Frasincar. Sentiment analysis of multiple implicit features per sentence in consumer review data. In *Selected Papers from the 12th International Baltic Conference on Databases and Information Systems (DB&IS 2016)*, volume 291 of *Frontiers in Artificial Intelligence and Applications*, pages 241–254. IOS Press, 2016b.

G. van Rooij, R. Sewnarain, M. Skogholt, T. van der Zaan, F. Frasincar, and **K. Schouten**. A data type-driven property alignment framework for product duplicate detection on the web. In *Proceedings of the 17th International Conference on Web Information Systems Engineering (WISE 2016)*, volume 10042 of *Lecture Notes in Computer Science*, pages 380–395. Springer, 2016.

K. Schouten and F. Frasincar. The benefit of concept-based features for sentiment analysis. In *Semantic Web Evaluation Challenges - Second SemWebEval Challenge at ESWC 2015, Revised Selected Papers*, pages 223–233, 2015.

O. Kovalenko, Y. Mrabet, **K. Schouten**, and S. Sejdovic. Linked data in action: Personalized museum tours on mobile devices. In *ESWC Developers Workshop 2015 (ESWC-DEV 2015)*, pages 14–19. CEUR-WS, 2015.

K. Schouten, N. de Boer, T. Lam, M. van Leeuwen, R. van Luijk, and F. Frasincar. Semantics-driven implicit aspect detection in consumer reviews. In *companion publication to the 24th International Conference on World Wide Web (WWW-Companion 2015)*, pages 109–110. International World Wide Web Conferences Steering Committee, 2015.

K. Schouten and F. Frasincar. Finding implicit features in consumer reviews for sentiment analysis. In *Proceedings of the 14th International Conference on Web Engineering (ICWE 2014)*, pages 130–144, 2014a.

N. de Boer, M. van Leeuwen, R. van Luijk, **K. Schouten**, F. Frasincar, and D. Vandic. Identifying explicit features for sentiment analysis in consumer reviews. In *Proceedings of the 15th International Conference on Web Information Systems Engineering (WISE 2014)*, pages 357–371, 2014.

K. Schouten and F. Frasincar. Implicit feature extraction for sentiment analysis in consumer reviews. In *Proceedings of the 19th International Conference on Application of Natural Language to Information Systems (NLDB 2014)*, volume 8455 of *Lecture Notes in Computer Science*, pages 228–231. Springer, 2014b.

K. Schouten and F. Frasincar. Implicit feature detection for sentiment analysis. In *companion publication to the 23rd International Conference on World Wide Web (WWW-Companion 2014)*, pages 367–368. International World Wide Web Conferences Steering Committee, 2014c.

K. Schouten, F. Frasincar, and F. de Jong. COMMIT-P1WP3: a Co-occurrence Based Approach to Aspect-level Sentiment Analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 203–207. Association for Computational Linguistics and Dublin City University, 2014.

A. Hogenboom, F. Hogenboom, F. Frasincar, **K. Schouten**, and O. van der Meer. Semantics-based information extraction for detecting economic events. *Multimedia Tools and Applications*, 64(1):27–52, 2013.

K. Schouten and F. Frasincar. A linguistic graph-based approach for web news sentence searching. In *Proceedings of the 24th International Conference on Database and Expert Systems Applications (DEXA 2013)*, volume 8056 of *Lecture Notes in Computer Science*, pages 57–64. Springer, 2013b.

K. Schouten and F. Frasincar. A dependency graph isomorphism for news sentence searching. In *Proceedings of the 18th International Conference on Application of Natural Language to Information Systems (NLDB 2013)*, volume 7934 of *Lecture Notes in Computer Science*, pages 384–387. Springer, 2013a.

K. Schouten, R. Aly, and R. Ordelman. Searching and hyperlinking using word importance segment boundaries in mediaeval 2013. In *MediaEval 2013*, 2013.

A. Hogenboom, F. Hogenboom, F. Frasincar, U. Kaymak, O. van der Meer, , and **K. Schouten**. Detecting economic events using a semantics-based pipeline. In *Proceedings of the 22nd International Conference on Database and Expert Systems Applications (DEXA 2011)*, volume 6860 of *Lecture Notes in Computer Science*, pages 440–447. Springer, 2011a.

K. Schouten, P. Ruijgrok, J. Borsje, F. Frasincar, L. Levering, and F. Hogenboom. A semantic web-based approach for personalizing news. In *Proceedings of the 25th Symposium on Applied Computing (SAC 2010)*, pages 854–861. ACM, 2010.

Teaching

Lecturer for **Text Mining and Web Analytics** (DBA0006)
post-graduate program Data & Business Analytics at Erasmus School of Economics
2016-2017, 2017-2018

Course coordinator of **Skills 1: IT** (BAP065)
bachelor International Business Administration at Rotterdam School of Management
2014-2015, 2015-2016, 2016-2017, 2017-2018, 2018-2019

Course co-coordinator of **ICT** (FEB11013/FEB11013X)
bachelor Economics / International Bachelor Economics and Business Economics at Erasmus School of Economics
2016-2017, 2017-2018

Tutorial lecturer for **Introduction to Programming** (FEB21011/FEB21011X)
bachelor Economics / International Bachelor Economics and Business Economics at Erasmus School of Economics
2013-2014, 2014-2015

Tutorial lecturer for **Skills 1: IT** (BAP065)
bachelor International Business Administration at Rotterdam School of Management
2013-2014

PhD Courses

ESWC Summer School on Semantic Web
European Summer School in Information Retrieval

Research Methods and Methodology for IKS
Data Mining
Information Retrieval
Learning and Reasoning
Models of Language Learning
Topics in the Philosophy of Science
English (CPE)
Publishing Strategy
Statistical Methods
Scientific Integrity
Social Networks and Market Competition

Conferences/Workshops/Symposia Attended

15th Extended Semantic Web Conference (ESWC 2018)
14th Dutch-Belgian DataBase Day Workshop (DBDBD 2017)
16th Dutch-Belgian Information Retrieval Workshop (DIR 2017)
18th International Conference on Web Information Systems Engineering (WISE 2017)
55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)
11th International Workshop on Semantic Evaluation (SemEval 2017)
17th International Conference on Web Engineering (ICWE 2017)
Wharton Consumer Analytics Initiative: Purchase, Play, & Upgrade Data - Closing Symposium
4th ICT.Open - Interface for Dutch ICT Research (ICT.Open 2017)

13th Dutch-Belgian DataBase Day Workshop (DBDBD 2016)
17th International Conference on Web Information Systems Engineering (WISE 2016)
12th International Baltic Conference on Databases and Information Systems (DB&IS 2016)
16th International Conference on Web Engineering (ICWE 2016)
21st International Conference on Application of Natural Language to Information Systems (NLDB 2016)

15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2016)
10th International Workshop on Semantic Evaluation (SemEval 2016)

15th International Conference on Web Engineering (ICWE 2015)
12th Extended Semantic Web Conference (ESWC 2015)
ESWC Developers Workshop 2015 (ESWC-DEV 2015)
24th International Conference on World Wide Web (WWW 2015)
2nd ICT.Open - Interface for Dutch ICT Research (ICT.Open 2015)

25th International Conference on Computational Linguistics (COLING 2014)
8th International Workshop on Semantic Evaluation (SemEval 2014)
19th International Conference on Application of Natural Language to Information Systems (NLDB 2014)
14th International Conference on Web Engineering (ICWE 2014)
23rd International Conference on World Wide Web (WWW 2014)

10th Dutch-Belgian DataBase Day Workshop (DBDBD 2013)
24th International Conference on Database and Expert Systems Applications (DEXA 2013)
18th International Conference on Application of Natural Language to Information Systems (NLDB 2013)

Review Activity

Journals*

Computer Speech & Language (2x)
Information Processing and Management (2x)
Information Sciences (2x)

*verified at Publons: `https://publons.com/author/1230098/kim-schouten#profile`

Data & Knowledge Engineering
IEEE Transactions on Knowledge and Data Engineering
ACM Transactions on Intelligent Systems and Technology (2x)
IEEE Transactions on Affective Computing (2x)
International Journal of Medical Informatics
Expert Systems with Applications (2x)
Artificial Intelligence Review (2x)
Mobile Information Systems
IEEE Intelligent Systems

Program Committee member for conferences and workshops

Knowledge and Language Processing Track @ ACM Symposium On Applied Computing (SAC) 2019
NLDB 2017
GraphQ 2017
SemEval 2017 (Task 5, Fine-Grained Sentiment Analysis on Financial Microblogs and News)
GraphQ 2016
KDWEB 2016
SemEval 2016 (Task 5, Aspect-Based Sentiment Analysis)
DIR 2015
SemEval 2015 (Task 12, Aspect-Based Sentiment Analysis) SemEval 2014 (Task 4, Aspect-Based Sentiment Analysis)

The ERIM PhD Series

The ERIM PhD Series contains PhD dissertations in the field of Research in Management defended at Erasmus University Rotterdam and supervised by senior researchers affiliated to the Erasmus Research Institute of Management (ERIM). All dissertations in the ERIM PhD Series are available in full text through the ERIM Electronic Series Portal: `http://repub.eur.nl/pub`. ERIM is the joint research institute of the Rotterdam School of Management (RSM) and the Erasmus School of Economics at the Erasmus University Rotterdam (EUR).

Dissertations in the last five years

Abbink, E.J., *Crew Management in Passenger Rail Transport*, Promotors: Prof. L.G. Kroon & Prof. A.P.M. Wagelmans, EPS-2014-325-LIS, `https://repub.eur.nl/pub/76927`

Acar, O.A., *Crowdsourcing for Innovation: Unpacking Motivational, Knowledge and Relational Mechanisms of Innovative Behavior in Crowdsourcing Platforms*, Promotor: Prof. J.C.M. van den Ende, EPS-2014-321-LIS, `https://repub.eur.nl/pub/76076`

Akemu, O., *Corporate Responses to Social Issues: Essays in Social Entrepreneurship and Corporate Social Responsibility*, Promotors: Prof. G.M. Whiteman & Dr S.P. Kennedy, EPS-2017-392-ORG, `https://repub.eur.nl/pub/95768`

Akin Ates, M., *Purchasing and Supply Management at the Purchase Category Level : Strategy, structure and performance*, Promotors: Prof. J.Y.F. Wynstra & Dr E.M.van Raaij, EPS-2014-300-LIS, `https://repub.eur.nl/pub/50283`

Alexander, L., *People, Politics, and Innovation: A Process Perspective*, Promotors: Prof. H.G. Barkema & Prof. D.L. van Knippenberg, EPS-2014-331-S&E, https://repub.eur.nl/pub/77209

Alexiou, A., *Management of Emerging Technologies and the Learning Organization: Lessons from the Cloud and Serious Games Technology*, Promotors: Prof. S.J. Magala, Prof. M.C. Schippers and Dr I. Oshri, EPS-2016-404-ORG, https://repub.eur.nl/pub/93818

Almeida e Santos Nogueira, R.J. de, *Conditional Density Models Integrating Fuzzy andProbabilistic Representations of Uncertainty*, Promotors: Prof. U. Kaymak & Prof. J.M.C. Sousa, EPS-2014-310-LIS, https://repub.eur.nl/pub/51560

Alserda, G.A.G., *Choices in Pension Management*, Promotors: Prof. S.G. van der Lecq & Dr O.W. Steenbeek, EPS-2017-432-F&A, https://repub.eur.nl/pub/103496

Avci, E., *Surveillance of Complex Auction Markets: a Market Policy Analytics Approach*, Promotors: Prof. W. Ketter, Prof. H.W.G.M. van Heck & Prof. D.W. Bunn, EPS-2018-426-LIS, https://repub.eur.nl/pub/106286

Benschop, N., *Biases in Project Escalation: Names, frames & construal levels*, Promotors: Prof. K.I.M. Rhode, Prof. H.R. Commandeur, Prof. M. Keil & Dr A.L.P. Nuijten, EPS-2015-375-S&E, https://repub.eur.nl/pub/79408

Berg, W.E. van den, *Understanding Salesforce Behavior using Genetic Association Studies*, Promotor: Prof. W.J.M.I. Verbeke, EPS-2014-311-MKT, https://repub.eur.nl/pub/51440

Beusichem, H.C. van, *Firms and Financial Markets: Empirical Studies on the Informational Value of Dividends, Governance and Financial Reporting*, Promotors: Prof. A. de Jong & Dr G. Westerhuis, EPS-2016-378-F&A, https://repub.eur.nl/pub/

93079

Bliek, R. de, *Empirical Studies on the Economic Impact of Trust*, Promotor: Prof. J. Veenman & Prof. Ph.H.B.F. Franses, EPS-2015-324-ORG, https://repub.eur.nl/pub/78159

Boons, M., *Working Together Alone in the Online Crowd: The Effects of Social Motivations and Individual Knowledge Backgrounds on the Participation and Performance of Members of Online Crowdsourcing Platforms*, Promotors: Prof. H.G. Barkema & Dr D.A. Stam, EPS-2014-306-S&E, https://repub.eur.nl/pub/50711

Bouman, P., *Passengers, Crowding and Complexity: Models for Passenger Oriented Public Transport*, Prof. L.G. Kroon, Prof. A. Schöbel & Prof. P.H.M. Vervest, EPS-2017-420-LIS, https://repub.eur.nl/100767

Brazys, J., *Aggregated Marcoeconomic News and Price Discovery*, Promotor: Prof. W.F.C. Verschoor, EPS-2015-351-F&A, https://repub.eur.nl/pub/78243

Bunderen, L. van, *Tug-of-War: Why and when teams get embroiled in power struggles*, Promotors: Prof. D.L. van Knippenberg & Dr. L. Greer, EPS-2018-446-ORG, https://repub.eur.nl/pub/105346

Burg, G.J.J. van den, *Algorithms for Multiclass Classification and Regularized Regression*, Promotors: Prof. P.J.F. Groenen & Dr. A. Alfons, EPS-2018-442-MKT, https://repub.eur.nl/pub/103929

Cancurtaran, P., *Essays on Accelerated Product Development*, Promotors: Prof. F. Langerak & Prof. G.H. van Bruggen, EPS-2014-317-MKT, https://repub.eur.nl/pub/76074

Chammas, G., *Portfolio concentration*, Promotor: Prof. J. Spronk, EPS-2017-410-F&E, https://repub.eur.nl/pub/94975

Cranenburgh, K.C. van, *Money or Ethics: Multinational corporations and religious organisations operating in an era of corporate responsibility*, Prof. L.C.P.M. Meijs, Prof. R.J.M. van Tulder & Dr D. Arenas, EPS-2016-385-ORG, https://repub.eur.nl/pub/93104

Consiglio, I., *Others: Essays on Interpersonal and Consumer Behavior*, Promotor: Prof. S.M.J. van Osselaer, EPS-2016-366-MKT, https://repub.eur.nl/pub/79820

Darnihamedani, P., *Individual Characteristics, Contextual Factors and Entrepreneurial Behavior*, Promotors: Prof. A.R. Thurik & S.J.A. Hessels, EPS-2016-360-S&E, https://repub.eur.nl/pub/93280

Dennerlein, T., *Empowering Leadership and Employees' Achievement Motivations: the Role of Self-Efficacy and Goal Orientations in the Empowering Leadership Process*, Promotors: Prof. D.L. van Knippenberg & Dr J. Dietz, EPS-2017-414-ORG, https://repub.eur.nl/pub/98438

Deng, W., *Social Capital and Diversification of Cooperatives*, Promotor: Prof. G.W.J. Hendrikse, EPS-2015-341-ORG, https://repub.eur.nl/pub/77449

Depecik, B.E., *Revitalizing brands and brand: Essays on Brand and Brand Portfolio Management Strategies*, Promotors: Prof. G.H. van Bruggen, Dr Y.M. van Everdingen and Dr M.B. Ataman, EPS-2016-406-MKT, https://repub.eur.nl/pub/93507

Duijzer, L.E., *Mathematical Optimization in Vaccine Allocation*, Promotors: Prof. R. Dekker & Dr W.L. van Jaarsveld, EPS-2017-430-LIS, https://repub.eur.nl/pub/101487

Duyvesteyn, J.G. *Empirical Studies on Sovereign Fixed Income Markets*, Promotors: Prof. P. Verwijmeren & Prof. M.P.E. Martens, EPS-2015-361-F&A, https://repub.eur.nl/pub/79033

Elemes, A., *Studies on Determinants and Consequences of Financial Reporting Quality*, Promotor: Prof. E. Peek, EPS-2015-354-F&A, https://repub.eur.nl/pub/79037

Ellen, S. ter, *Measurement, Dynamics, and Implications of Heterogeneous Beliefs in Financial Markets*, Promotor: Prof. W.F.C. Verschoor, EPS-2015-343-F&A, https://repub.eur.nl/pub/78191

Erlemann, C., *Gender and Leadership Aspiration: The Impact of the Organizational Environment*, Promotor: Prof. D.L. van Knippenberg, EPS-2016-376-ORG, https://repub.eur.nl/pub/79409

Eskenazi, P.I., *The Accountable Animal*, Promotor: Prof. F.G.H. Hartmann, EPS-2015-355-F&A, https://repub.eur.nl/pub/78300

Evangelidis, I., *Preference Construction under Prominence*, Promotor: Prof. S.M.J. van Osselaer, EPS-2015-340-MKT, https://repub.eur.nl/pub/78202

Faber, N., *Structuring Warehouse Management*, Promotors: Prof. M.B.M. de Koster & Prof. A. Smidts, EPS-2015-336-LIS, https://repub.eur.nl/pub/78603

Feng, Y., *The Effectiveness of Corporate Governance Mechanisms and Leadership Structure: Impacts on strategic change and firm performance*, Promotors: Prof. F.A.J. van den Bosch, Prof. H.W. Volberda & Dr J.S. Sidhu, EPS-2017-389-S&E, https://repub.eur.nl/pub/98470

Fernald, K., *The Waves of Biotechnological Innovation in Medicine: Interfirm Cooperation Effects and a Venture Capital Perspective*, Promotors: Prof. E. Claassen, Prof. H.P.G. Pennings & Prof. H.R. Commandeur, EPS-2015-371-S&E, https://hdl.handle.net/1765/79120

Fisch, C.O., *Patents and trademarks: Motivations, antecedents, and value in industrialized and emerging markets*, Promotors: Prof. J.H. Block, Prof. H.P.G. Pennings

& Prof. A.R. Thurik, EPS-2016-397-S&E, https://repub.eur.nl/pub/94036

Fliers, P.T., *Essays on Financing and Performance: The role of firms, banks and board*, Promotors: Prof. A. de Jong & Prof. P.G.J. Roosenboom, EPS-2016-388-F&A, https://repub.eur.nl/pub/93019

Fourne, S.P., *Managing Organizational Tensions: A Multi-Level Perspective on Exploration, Exploitation and Ambidexterity*, Promotors: Prof. J.J.P. Jansen & Prof. S.J. Magala, EPS-2014-318-S&E, https://repub.eur.nl/pub/76075

Gaast, J.P. van der, *Stochastic Models for Order Picking Systems*, Promotors: Prof. M.B.M de Koster & Prof. I.J.B.F. Adan, EPS-2016-398-LIS, https://repub.eur.nl/pub/93222

Giurge, L., *A Test of Time; A temporal and dynamic approach to power and ethics*, Promotors: Prof. M.H. van Dijke & Prof. D. De Cremer, EPS-2017-412-ORG, https://repub.eur.nl/98451

Glorie, K.M., *Clearing Barter Exchange Markets: Kidney Exchange and Beyond*, Promotors: Prof. A.P.M. Wagelmans & Prof. J.J. van de Klundert, EPS-2014-329-LIS, https://repub.eur.nl/pub/77183

Gobena, L., *Towards Integrating Antecedents of Voluntary Tax Compliance*, Promotors: Prof. M.H. van Dijke & Dr P. Verboon, EPS-2017-436-ORG, https://repub.eur.nl/pub/103276

Groot, W.A., *Assessing Asset Pricing Anomalies*, Promotors: Prof. M.J.C.M. Verbeek & Prof. J.H. van Binsbergen, EPS-2017-437-F&A, https://repub.eur.nl/pub/103490

Harms, J. A., *Essays on the Behavioral Economics of Social Preferences and Bounded Rationality*, Prof. H.R. Commandeur & Dr K.E.H. Maas, EPS-2018-457-S&E, https:

//repub.eur.nl/pub/108831

Hekimoglu, M., *Spare Parts Management of Aging Capital Products*, Promotor: Prof. R. Dekker, EPS-2015-368-LIS, https://repub.eur.nl/pub/79092

Hengelaar, G.A., *The Proactive Incumbent: Holy grail or hidden gem? Investigating whether the Dutch electricity sector can overcome the incumbent's curse and lead the sustainability transition*, Promotors: Prof. R.J. M. van Tulder & Dr K. Dittrich, EPS-2018-438-ORG, https://repub.eur.nl/pub/102953

Hogenboom, A.C., *Sentiment Analysis of Text Guided by Semantics and Structure*, Promotors: Prof. U. Kaymak & Prof. F.M.G. de Jong, EPS-2015-369-LIS, https://repub.eur.nl/pub/79034

Hogenboom, F.P., *Automated Detection of Financial Events in News Text*, Promotors: Prof. U. Kaymak & Prof. F.M.G. de Jong, EPS-2014-326-LIS, https://repub.eur.nl/pub/77237

Hollen, R.M.A., *Exploratory Studies into Strategies to Enhance Innovation-Driven International Competitiveness in a Port Context: Toward Ambidextrous Ports*, Promotors: Prof. F.A.J. Van Den Bosch & Prof. H.W. Volberda, EPS-2015-372-S&E, https://repub.eur.nl/pub/78881

Hout, D.H. van, *Measuring Meaningful Differences: Sensory Testing Based Decision Making in an Industrial Context; Applications of Signal Detection Theory and Thurstonian Modelling*, Promotors: Prof. P.J.F. Groenen & Prof. G.B. Dijksterhuis, EPS-2014-304-MKT, https://repub.eur.nl/pub/50387

Houwelingen, G.G. van, *Something To Rely On*, Promotors: Prof. D. de Cremer & Prof. M.H. van Dijke, EPS-2014-335-ORG, https://repub.eur.nl/pub/77320

Hurk, E. van der, *Passengers, Information, and Disruptions*, Promotors: Prof. L.G. Kroon & Prof. P.H.M. Vervest, EPS-2015-345-LIS, https://repub.eur.nl/pub/

78275

Iseger, P. den, *Fourier and Laplace Transform Inversion with Applications in Finance*, Promotor: Prof. R. Dekker, EPS-2014-322-LIS, https://repub.eur.nl/pub/76954

Jacobs, B.J.D., *Marketing Analytics for High-Dimensional Assortments*, Promotors: Prof. A.C.D. Donkers & Prof. D. Fok, EPS-2017-445-MKT, https://repub.eur.nl/pub/103497

Kahlen, M. T., *Virtual Power Plants of Electric Vehicles in Sustainable Smart Electricity Markets*, Promotors: Prof. W. Ketter & Prof. A. Gupta, EPS-2017-431-LIS, https://repub.eur.nl/pub/100844

Kampen, S. van, *The Cross-sectional and Time-series Dynamics of Corporate Finance: Empirical evidence from financially constrained firms*, Promotors: Prof. L. Norden & Prof. P.G.J. Roosenboom, EPS-2018-440-F&A, https://repub.eur.nl/pub/105245

Karali, E., *Investigating Routines and Dynamic Capabilities for Change and Innovation*, Promotors: Prof. H.W. Volberda, Prof. H.R. Commandeur and Dr J.S. Sidhu, EPS-2018-454-S&E, https://repub.eur.nl/pub/106274

Keko. E, *Essays on Innovation Generation in Incumbent Firms*, Promotors: Prof. S. Stremersch & Dr N.M.A. Camacho, EPS-2017-419-MKT, https://repub.eur.nl/pub/100841

Khanagha, S., *Dynamic Capabilities for Managing Emerging Technologies*, Promotor: Prof. H.W. Volberda, EPS-2014-339-S&E, https://repub.eur.nl/pub/77319

Khattab, J., *Make Minorities Great Again: a contribution to workplace equity by identifying and addressing constraints and privileges*, Prof. D.L. van Knippenberg &

Dr A. Nederveen Pieterse, EPS-2017-421-ORG, https://repub.eur.nl/pub/99311

Kim, T. Y., *Data-driven Warehouse Management in Global Supply Chains*, Promotors: Prof. R. Dekker & Dr C. Heij, EPS-2018-449-LIS, https://repub.eur.nl/pub/109103

Klitsie, E.J., *Strategic Renewal in Institutional Contexts: The paradox of embedded agency*, Promotors: Prof. H.W. Volberda & Dr. S. Ansari, EPS-2018-444-S&E, https://repub.eur.nl/pub/106275

Klooster, E. van 't, *Travel to Learn: the Influence of Cultural Distance on CompetenceDevelopment in Educational Travel*, Promotors: Prof. F.M. Go & Prof. P.J. van Baalen, EPS-2014-312-MKT, https://repub.eur.nl/pub/51462

Koendjbiharie, S.R., *The Information-Based View on Business Network Performance: Revealing the Performance of Interorganizational Networks*, Promotors: Prof. H.W. G.M. van Heck & Prof. P.H.M. Vervest, EPS-2014-315-LIS, https://repub.eur.nl/pub/51751

Koning, M., *The Financial Reporting Environment: The Role of the Media, Regulatorsand Auditors*, Promotors: Prof. G.M.H. Mertens & Prof. P.G.J. Roosenboom, EPS-2014-330-F&A, https://repub.eur.nl/pub/77154

Konter, D.J., *Crossing Borders with HRM: An Inquiry of the Influence of Contextual Differences in the Adoption and Effectiveness of HRM*, Promotors: Prof. J. Paauwe, & Dr L.H. Hoeksema, EPS-2014-305-ORG, https://repub.eur.nl/pub/50388

Korkmaz, E., *Bridging Models and Business: Understanding Heterogeneity in Hidden Drivers of Customer Purchase Behavior*, Promotors: Prof. S.L. van de Velde & Prof. D. Fok, EPS-2014-316-LIS, https://repub.eur.nl/pub/76008

Krämer, R., *A license to mine? Community organizing against multinational corporations*, Promotors: Prof. R.J.M. van Tulder & Prof. G.M. Whiteman, EPS-2016-

383-ORG, https://repub.eur.nl/pub/94072

Kroezen, J.J., *The Renewal of Mature Industries: An Examination of the Revival of the Dutch Beer Brewing Industry*, Promotor: Prof. P.P.M.A.R. Heugens, EPS-2014-333-S&E, https://repub.eur.nl/pub/77042

Kysucky, V., *Access to Finance in a Cros-Country Context*, Promotor: Prof. L. Norden, EPS-2015-350-F&A, https://repub.eur.nl/pub/78225

Lee, C.I.S.G., *Big Data in Management Research: Exploring New Avenues*, Promotors: Prof. S.J. Magala & Dr W.A. Felps, EPS-2016-365-ORG, https://repub.eur.nl/pub/79818

Legault-Tremblay, P.O., *Corporate Governance During Market Transition: Heterogeneous responses to Institution Tensions in China*, Promotor: Prof. B. Krug, EPS-2015-362-ORG, https://repub.eur.nl/pub/78649

Lenoir, A.S., *Are You Talking to Me? Addressing Consumers in a Globalised World*, Promotors: Prof. S. Puntoni & Prof. S.M.J. van Osselaer, EPS-2015-363-MKT, https://repub.eur.nl/pub/79036

Leunissen, J.M., *All Apologies: On the Willingness of Perpetrators to Apologize*, Promotors: Prof. D. de Cremer & Dr M. van Dijke, EPS-2014-301-ORG, https://repub.eur.nl/pub/50318

Li, D., *Supply Chain Contracting for After-sales Service and Product Support*, Promotor: Prof. M.B.M. de Koster, EPS-2015-347-LIS, https://repub.eur.nl/pub/78526

Li, Z., *Irrationality: What, Why and How*, Promotors: Prof. H. Bleichrodt, Prof. P.P. Wakker, & Prof. K.I.M. Rohde, EPS-2014-338-MKT, https://repub.eur.nl/pub/77205

Liu, N., *Behavioral Biases in Interpersonal Contexts*, Supervisors: Prof. A. Baillon & Prof. H. Bleichrodt, EPS-2017-408-MKT, https://repub.eur.nl/pub/95487

Liket, K., *Why 'Doing Good' is not Good Enough: Essays on Social Impact Measurement*, Promotors: Prof. H.R. Commandeur & Dr K.E.H. Maas, EPS-2014-307-STR, https://repub.eur.nl/pub/51130

Lu, Y., *Data-Driven Decision Making in Auction Markets*, Promotors: Prof. H.W. G.M. van Heck & Prof. W. Ketter, EPS-2014-314-LIS, https://repub.eur.nl/pub/51543

Ma, Y., *The Use of Advanced Transportation Monitoring Data for Official Statistics*, Promotors: Prof. L.G. Kroon and Dr J. van Dalen, EPS-2016-391-LIS, https://repub.eur.nl/pub/80174

Maira, E., *Consumers and Producers*, Promotors: Prof. S. Puntoni & Prof. C. Fuchs, EPS-2018-439-MKT, https://repub.eur.nl/pub/104387

Manders, B., *Implementation and Impact of ISO 9001*, Promotor: Prof. K. Blind, EPS-2014-337-LIS, https://repub.eur.nl/pub/77412

Mell, J.N., *Connecting Minds: On The Role of Metaknowledge in Knowledge Coordination*, Promotor: Prof. D.L. van Knippenberg, EPS-2015-359-ORG, https://hdl.handle.net/1765/78951

Meulen,van der, D., *The Distance Dilemma: the effect of flexible working practices on performance in the digital workplace*, Promotors: Prof. H.W.G.M. van Heck & Prof. P.J. van Baalen, EPS-2016-403-LIS, https://repub.eur.nl/pub/94033

Micheli, M.R., *Business Model Innovation: A Journey across Managers' Attention andInter-Organizational Networks*, Promotor: Prof. J.J.P. Jansen, EPS-2015-344-S&E, https://repub.eur.nl/pub/78241

Moniz, A, *Textual Analysis of Intangible Information*, Promotors: Prof. C.B.M. van Riel, Prof. F.M.G de Jong & Dr G.A.J.M. Berens, EPS-2016-393-ORG, https://repub.eur.nl/pub/93001

Mulder, J., *Network design and robust scheduling in liner shipping*, Promotors: Prof. R. Dekker & Dr W.L. van Jaarsveld, EPS-2016-384-LIS, https://repub.eur.nl/pub/80258

Naumovska, I., *Socially Situated Financial Markets: A Neo-Behavioral Perspective onFirms, Investors and Practices*, Promotors: Prof. P.P.M.A.R. Heugens & Prof. A. de Jong, EPS-2014-319-S&E, https://repub.eur.nl/pub/76084

Neerijnen, P., *The Adaptive Organization: the socio-cognitive antecedents of ambidexterity and individual exploration*, Promotors: Prof. J.J.P. Jansen, P.P.M.A.R. Heugens & Dr T.J.M. Mom, EPS-2016-358-S&E, https://repub.eur.nl/pub/93274

Okbay, A., *Essays on Genetics and the Social Sciences*, Promotors: Prof. A.R. Thurik, Prof. Ph.D. Koellinger & Prof. P.J.F. Groenen, EPS-2017-413-S&E, https://repub.eur.nl/pub/95489

Oord, J.A. van, *Essays on Momentum Strategies in Finance*, Promotor: Prof. H.K. van Dijk, EPS-2016-380-F&A, https://repub.eur.nl/pub/80036

Peng, X., *Innovation, Member Sorting, and Evaluation of Agricultural Cooperatives*, Promotor: Prof. G.W.J. Hendriks, EPS-2017-409-ORG, https://repub.eur.nl/pub/94976

Pennings, C.L.P., *Advancements in Demand Forecasting: Methods and Behavior*, Promotors: Prof. L.G. Kroon, Prof. H.W.G.M. van Heck & Dr J. van Dalen, EPS-2016-400-LIS, https://repub.eur.nl/pub/94039

Peters, M., *Machine Learning Algorithms for Smart Electricity Markets*, Promotor: Prof. W. Ketter, EPS-2014-332-LIS, https://repub.eur.nl/pub/77413

Petruchenya, A., *Essays on Cooperatives: Emergence, Retained Earnings, and Market Shares*, Promotors: Prof. G.W.J. Hendriks & Dr Y. Zhang, EPS-2018-447-ORG, https://repub.eur.nl/pub/105243

Plessis, C. du, *Influencers: The Role of Social Influence in Marketing*, Promotors: Prof. S. Puntoni & Prof. S.T.L.R. Sweldens, EPS-2017-425-MKT, https://repub.eur.nl/pub/103265

Pocock, M., *Status Inequalities in Business Exchange Relations in Luxury Markets*, Promotors: Prof. C.B.M. van Riel & Dr G.A.J.M. Berens, EPS-2017-346-ORG, https://repub.eur.nl/pub/98647

Pozharliev, R., *Social Neuromarketing: The role of social context in measuring advertising effectiveness*, Promotors: Prof. W.J.M.I. Verbeke & Prof. J.W. van Strien,EPS-2017-402-MKT, https://repub.eur.nl/pub/95528

Protzner, S., *Mind the gap between demand and supply: A behavioral perspective on demand forecasting*, Promotors: Prof. S.L. van de Velde & Dr L. Rook, EPS-2015-364-LIS, https://repub.eur.nl/pub/79355

Pruijssers, J.K., *An Organizational Perspective on Auditor Conduct*, Promotors: Prof. J. van Oosterhout & Prof. P.P.M.A.R. Heugens, EPS-2015-342-S&E, https://repub.eur.nl/pub/78192

Riessen, B. van, *Optimal Transportation Plans and Portfolios for Synchromodal Container Networks*, Promotors: Prof. R. Dekker & Prof. R.R. Negenborn, EPS-2018-448-LIS, https://repub.eur.nl/pub/105248

Rietdijk, W.J.R., *The Use of Cognitive Factors for Explaining Entrepreneurship*, Promotors: Prof. A.R. Thurik & Prof. I.H.A. Franken, EPS-2015-356-S&E, https:

//repub.eur.nl/pub/79817

Rietveld, N., *Essays on the Intersection of Economics and Biology*, Promotors: Prof. A.R. Thurik, Prof. Ph.D. Koellinger, Prof. P.J.F. Groenen & Prof. A. Hofman, EPS-2014-320-S&E, https://repub.eur.nl/pub/76907

Rösch, D., *Market Efficiency and Liquidity*, Promotor: Prof. M.A. van Dijk, EPS-2015-353-F&A, https://repub.eur.nl/pub/79121

Roza, L., *Employee Engagement in Corporate Social Responsibility: A collection of essays*, Promotor: Prof. L.C.P.M. Meijs, EPS-2016-396-ORG, https://repub.eur.nl/pub/93254

Schie, R. J. G. van, *Planning for Retirement: Save More or Retire Later?*, Promotors: Prof. B. G. C. Dellaert & Prof. A.C.D. Donkers, EOS-2017-415-MKT, https://repub.eur.nl/pub/100846

Schoonees, P., *Methods for Modelling Response Styles*, Promotor: Prof. P.J.F. Groenen, EPS-2015-348-MKT, https://repub.eur.nl/pub/79327

Schouten, M.E., *The Ups and Downs of Hierarchy: the causes and consequences of hierarchy struggles and positional loss*, Promotors: Prof. D.L. van Knippenberg & Dr L.L. Greer, EPS-2016-386-ORG, https://repub.eur.nl/pub/80059

Smit, J., *Unlocking Business Model Innovation: A look through the keyhole at the inner workings of Business Model Innovation*, Promotor: Prof. H.G. Barkema, EPS-2016-399-S&E, https://repub.eur.nl/pub/93211

Sousa, M.J.C. de, *Servant Leadership to the Test: New Perspectives and Insights*, Promotors: Prof. D.L. van Knippenberg & Dr D. van Dierendonck, EPS-2014-313-ORG, https://repub.eur.nl/pub/51537

Staadt, J.L., *Leading Public Housing Organisation in a Problematic Situation: A CriticalSoft Systems Methodology Approach*, Promotor: Prof. S.J. Magala, EPS-2014-308-ORG, https://repub.eur.nl/pub/50712

Straeter, L.M., *Interpersonal Consumer Decision Making*, Promotors: Prof. S.M.J. van Osselaer & Dr I.E. de Hooge, EPS-2017-423-MKT, https://repub.eur.nl/pub/100819

Subaşi, B., *Demographic Dissimilarity, Information Access and Individual Performance*, Promotors: Prof. D.L. van Knippenberg & Dr W.P. van Ginkel, EPS-2017-422-ORG, https://repub.eur.nl/pub/103495

Szatmari, B., *We are (all) the champions: The effect of status in the implementation of innovations*, Promotors: Prof. J.C.M & Dr D. Deichmann, EPS-2016-401-LIS, https://repub.eur.nl/pub/94633

Tuijl, E. van, *Upgrading across Organisational and Geographical Configurations*, Promotor: Prof. L. van den Berg, EPS-2015-349-S&E, https://repub.eur.nl/pub/78224

Tuncdogan, A., *Decision Making and Behavioral Strategy: The Role of Regulatory Focusin Corporate Innovation Processes*, Promotors: Prof. F.A.J. van den Bosch,Prof. H.W. Volberda, & Prof. T.J.M. Mom, EPS-2014-334-S&E, https://repub.eur.nl/pub/76978

Uijl, S. den, *The Emergence of De-facto Standards*, Promotor: Prof. K. Blind, EPS-2014-328-LIS, https://repub.eur.nl/pub/77382

Valogianni, K., *Sustainable Electric Vehicle Management using Coordinated Machine Learning*, Promotors: Prof. H.W.G.M. van Heck & Prof. W. Ketter, EPS-2016-387-LIS, https://repub.eur.nl/pub/93018

Vandic, D., *Intelligent Information Systems for Web Product Search*, Promotors: Prof. U. Kaymak & Dr Frasincar, EPS-2017-405-LIS, https://repub.eur.nl/pub/95490

Veelenturf, L.P., *Disruption Management in Passenger Railways: Models for Timetable, Rolling Stock and Crew Rescheduling*, Promotor: Prof. L.G. Kroon, EPS-2014-327-LIS, https://repub.eur.nl/pub/77155

Verbeek, R.W.M., *Essays on Empirical Asset Pricing*, Promotors: Prof. M.A. van Dijk & Dr M. Szymanowska, EPS-2017-441-F&A, https://repub.eur.nl/pub/102977

Vermeer, W., *Propagation in Networks: The impact of information processing at the actor level on system-wide propagation dynamics*, Promotor: Prof. P.H.M.Vervest, EPS-2015-373-LIS, https://repub.eur.nl/pub/79325

Versluis, I., *Prevention of the Portion Size Effect*, Promotors: Prof. Ph.H.B.F. Franses & Dr E.K. Papies, EPS-2016-382-MKT, https://repub.eur.nl/pub/79880

Vishwanathan, P., *Governing for Stakeholders: How Organizations May Create or Destroy Value for their Stakeholders*, Promotors: Prof. J. van Oosterhout & Prof. L.C.P.M. Meijs, EPS-2016-377-ORG, https://repub.eur.nl/pub/93016

Vlaming, R. de., *Linear Mixed Models in Statistical Genetics*, Prof. A.R. Thurik, Prof. P.J.F. Groenen & Prof. Ph.D. Koellinger, EPS-2017-416-S&E, https://repub.eur.nl/pub/100428

Vries, H. de, *Evidence-Based Optimization in Humanitarian Logistics*, Promotors: Prof. A.P.M. Wagelmans & Prof. J.J. van de Klundert, EPS-2017-435-LIS, https://repub.eur.nl/pub/102771

Vries, J. de, *Behavioral Operations in Logistics*, Promotors: Prof. M.B.M de Koster & Prof. D.A. Stam, EPS-2015-374-LIS, https://repub.eur.nl/pub/79705

Wagenaar, J.C., *Practice Oriented Algorithmic Disruption Management in Passenger Railways*, Prof. L.G. Kroon & Prof. A.P.M. Wagelmans, EPS-2016-390-LIS, https://repub.eur.nl/pub/93177

Wang, P., *Innovations, status, and networks*, Promotors: Prof. J.J.P. Jansen & Dr V.J.A. van de Vrande, EPS-2016-381-S&E, https://repub.eur.nl/pub/93176

Wang, R., *Corporate Environmentalism in China*, Promotors: Prof. P.P.M.A.R Heugens & Dr F. Wijen, EPS-2017-417-S&E, https://repub.eur.nl/pub/99987

Wang, T., *Essays in Banking and Corporate Finance*, Promotors: Prof. L. Norden & Prof. P.G.J. Roosenboom, EPS-2015-352-F&A, https://repub.eur.nl/pub/78301

Wasesa, M., *Agent-based inter-organizational systems in advanced logistics operations*, Promotors: Prof. H.W.G.M van Heck, Prof. R.A. Zuidwijk & Dr A. W. Stam, EPS-2017-LIS-424, https://repub.eur.nl/pub/100527

Weenen, T.C., *On the Origin and Development of the Medical Nutrition Industry*, Promotors: Prof. H.R. Commandeur & Prof. H.J.H.M. Claassen, EPS-2014-309-S&E, https://repub.eur.nl/pub/51134

Wessels, C., *Flexible Working Practices: How Employees Can Reap the Benefits for Engagement and Performance*, Promotors: Prof. H.W.G.M. van Heck, Prof. P.J. van Baalen & Prof. M.C. Schippers, EPS-2017-418-LIS, https://repub.eur.nl/99312

Witte, C.T., *Bloody Business: Multinational investment in an increasingly conflict-afflicted world*, Promotors: Prof. H.P.G. Pennings, Prof. H.R. Commandeur & Dr M.J. Burger, EPS-2018-443-S&E, https://repub.eur.nl/pub/104027

Yang, S., *Information Aggregation Efficiency of Prediction Markets*, Promotor: Prof. H.W.G.M. van Heck, EPS-2014-323-LIS, https://repub.eur.nl/pub/77184

Yuan, Y., *The Emergence of Team Creativity: a social network perspective*, Promotors: Prof. D. L. van Knippenberg & Dr D. A. Stam, EPS-2017-434-ORG, https://repub.eur.nl/pub/100847

Ypsilantis, P., *The Design, Planning and Execution of Sustainable Intermodal Port-hinterland Transport Networks*, Promotors: Prof. R.A. Zuidwijk & Prof. L.G. Kroon, EPS-2016-395-LIS, https://repub.eur.nl/pub/94375

Yuferova, D., *Price Discovery, Liquidity Provision, and Low-Latency Trading*, Promotors: Prof. M.A. van Dijk & Dr D.G.J. Bongaerts, EPS-2016-379-F&A, https://repub.eur.nl/pub/93017

Zhang, Q., *Financing and Regulatory Frictions in Mergers and Acquisitions*, Promotors: Prof. P.G.J. Roosenboom & Prof. A. de Jong, EPS-2018-428-F&A, https://repub.eur.nl/pub/103871

Zuber, F.B., *Looking at the Others: Studies on (un)ethical behavior and social relationships in organizations*, Promotor: Prof. S.P. Kaptein, EPS-2016-394-ORG, https://repub.eur.nl/pub/94388