Tamara Marina Feiertag, BSc

Generating Domain-Specific Lexicons for Sentiment Analysis

Master's Thesis

to achieve the university degree of

Master of Science

Master's degree programme: Software Engineering and Management

submitted to

Graz University of Technology

Supervisor

Assoc.Prof. Dipl.-Ing. Dr.techn. Denis Helic

Institute of Interactive Systems and Data Science
Head: Univ.-Prof. Dipl.-Inf. Dr. Stefanie Lindstaedt

Graz, July 2019

This document is set in Palatino, compiled with pdfLaTeX2e and `Biber`.

The LaTeX template from Karl Voit is based on KOMA script and can be found online: https://github.com/novoid/LaTeX-KOMA-template

Tamara Marina Feiertag, BSc

Generierung Domänenspezifischer Lexika für Sentiment-Analyse

Masterarbeit

zur Erlangung des Akademischen Grades

Diplom Ingenieur

Masterstudium: Software Engineering and Management

an der

Technischen Universität Graz

Betreuer

Assoc.Prof. Dipl.-Ing. Dr.techn. Denis Helic

Institut für Informationssysteme und Computer Medien
Leitung: Univ.-Prof. Dipl.-Inf. Dr. Stefanie Lindstaedt

Graz, Juli 2019

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis.

Date

Signature

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Dissertation identisch.

Datum

Unterschrift

Abstract

Social media has introduced novel possibilities to study human behavior. Due to online communities, large amounts of texts composed by individuals became easily accessible. Researchers recognized the rich potential of gaining knowledge from these data sources, and various new research fields emerged. One of them is sentiment analysis, a subfield of natural language processing concerned with detecting emotions and opinions in spoken and written language. By analyzing social media texts for sentiments, people's attitudes, positive or negative, toward any topic can be discovered. This intelligence is extremely valuable for diverse kinds of decisions, such as in marketing or politics.

A significant resource for sentiment analysis are sentiment lexicons, which are specialized dictionaries that assign a polarity value (positive, negative, or neutral) to each word. Multiple sentiment analysis approaches exploit this sentiment information to reveal opinions in texts. At present, most lexicons either only mirror correct sentiments for a single domain or consist of overly general words. Because the latter comes at the cost of accuracy, domain-specific sentiment dictionaries are needed. However, creating such lexicons has several challenges, like the scarcity of ground truth data. Consequently, high-quality sentiment word banks are limited to a few domains.

This master's thesis focuses on generating domain-specific sentiment lexicons automatically. A system is developed to build a lexicon for any domain. It is based on word embeddings and a label spreading technique to learn sentiments of words and phrases. Experiments are conducted to assess the effectiveness of the method. In particular, the performances of different sentiment dictionaries is compared in simple sentiment analysis tasks. It is found that the generated lexicons outperform other baseline approaches and some well-established sentiment dictionaries.

Kurzfassung

Durch soziale Medien entstanden neuartige Möglichkeiten zur Untersuchung des menschlichen Verhaltens. Aufgrund von Online-Communities wurden große Mengen an Texten, verfasst von den unterschiedlichsten Personen, leichter zugänglich. Forscher erkannten das große Potenzial, aus diesen Daten Wissen zu gewinnen und es entstanden neue Forschungsfelder. Eine davon ist die Sentiment-Analyse, ein Teilgebiet von *Natural Language Processing*, das sich mit der Erkennung von Emotionen und Meinungen in gesprochener und geschriebener natürlicher Sprache befasst. Mittels der Analyse von Social-Media-Texten nach Gefühlen können positive als auch negative Einstellungen der Menschen zu jeglichen Themen ermittelt werden. Dieses Wissen ist äußerst wertvoll für die verschiedensten Arten von Entscheidungen, beispielsweise im Marketing oder in der Politik.

Eine wichtige Ressource für die Sentiment-Analyse sind Sentiment-Lexika, eine spezielle Art von Wörterbüchern, die jedem Wort eine Polarität (positiv, negativ oder neutral) zuweisen. Mehrere Ansätze zur Sentiment-Analyse nutzen diese Informationen, um Meinungen in Texten zu erkennen. Zurzeit spiegeln die meisten Lexika nur richtige Gefühle für eine einzelne Domäne wider oder sie bestehen aus übermäßig allgemeinen Wörtern. Da sich Letzteres negativ auf die Genauigkeit auswirkt, werden domänenspezifische Sentiment-Lexika bevorzugt. Die Erstellung derartiger Lexika ist jedoch mit mehreren Herausforderungen verbunden, wie beispielsweise der generellen Knappheit von *Ground Truth* Daten. Folglich sind hochwertige Polaritätswörterbücher auf wenige Domänen beschränkt.

Diese Masterarbeit beschäftigt sich mit der automatischen Generierung domänenspezifischer Sentiment-Lexika. Es wurde ein System entwickelt, mithilfe dessen Lexika für mehrere Domänen erstellt werden können. Die Methode lernt, basierend auf *Word Embeddings* und einer *Label Spreading* Technik, Polaritäten von Wörtern und Phrasen. Um die Effektivität der Methode zu bewerten, werden Experimente durchgeführt, in denen verschiedene Sentiment-Lexika bei der Sentiment-Analyse verwendet und miteinander verglichen werden. Die Resultate zeigen, dass die generierten Sentiment-Lexika grundlegende Ansätze und bereits etablierte Lexika übertreffen.

Contents

List of Figures

List of Tables

1 Introduction

Sentiment analysis is concerned with automatically analyzing natural language text for opinions. An opinion can either express negative or positive emotions. To take a case in point, the sentence "I love this movie" displays positive sentiment toward a movie while "I do not like dogs" infers a negative opinion about dogs. The objective of sentiment analysis is to identify the positivity of large amounts of texts to gain knowledge about people's views on various subjects. To do so, multiple techniques to perform sentiment analysis exist. These can be categorized into unsupervised, supervised, semi-supervised, and domain adaption methods.

The field of unsupervised sentiment analysis mainly consists of so-called lexicon-based opinion detection. Lexicon-based approaches utilize sentiment lexicons for the analysis. A sentiment lexicon is a dictionary that assigns a positivity (e.g., positive, negative, or neutral) to every word. With this information opinions in texts can be extracted, for example, by counting the number of sentiment-bearing words in a sentence or document. Besides unsupervised methods, sentiment lexicons can also be effectively integrated into other techniques like supervised machine learning. Furthermore, lexicons are a prominent resource for generating ground truth data.

Sentiment lexicons can be created in two different ways. The first method requires manual labor to create a dictionary of words, phrases, or other text tokens and annotating each entry with a sentiment value. The second approach encompasses the automatic creation of a lexicon. In general, both methods have their disadvantages. While the manual creation is very resource-intensive, methods to automatically create lexicons suffer from various challenges. For example, sentiments of words usually are domain sensitive. This means words can express opposing emotions in different contexts. Therefore, lexicons that were constructed based on a specific domain are less accurate for other domains. However, lexicons that are too general tend to have lower accuracy as well. Also, most techniques depend on ground truth data, which is often infeasible to obtain.

The focus of this master's thesis is to generate domain-specific sentiment lexicons automatically. The aim is to overcome some limitations of existing approaches to build a high-quality lexicon for any domain quickly.

1.1 Motivation

It is part of human nature to strive for knowledge about others' opinions. We want to find out what other people think to form our own opinion and to make decisions. Before the appearance of the internet, we would ask a rather small pool of people for their points of view on various subjects. For example, when we were thinking about buying a new car, we would ask friends for advice. Restaurant owners personally interviewed their customers to get some feedback about served dishes and their staff. However, with the widespread use of the internet, a digital social platform flourished. Lots of people started to share their experiences by putting them online. Nowadays, we cannot only refer to people we know for opinions and recommendations but also to a vast community on the web. Within seconds, we can find out what others think about a movie, restaurant, political party, TV, car, or any other topic. Such access to numerous ideas is fascinating to individuals. Likewise, the benefits of analyzing people's attitudes become more and more important to companies and political parties.

Businesses always monitored customer experiences with their products or services to be able to make better decisions. Similarly, political parties seek knowledge about what the public thinks about new policies, and so on. With the emergence of social media, a new medium to track such opinions has become available. The size of social media, however, has some controversial aspects. On the one hand, organizations can gain insights on the broader public opinion like never before, making it a powerful tool. On the other hand, the extensive amount of new information created every day makes finding relevant parts hard and overwhelming.

Because reading through tons of texts is exhausting and often impossible, researchers started to look for ways to automatically extract and aggregate opinions. This is when the research field of sentiment analysis (or opinion mining) emerged. Today, machines can reveal opinions, positive or negative, on an immense scale. Knowing sentiments on social media about products, services, or topics provides valuable information. Consequently, organizations can adapt their strategies and plan for the future way more efficiently.

An essential part of sentiment analysis are lexicons. Having a high-quality sentiment lexicon at one's disposal enables accurate identification of attitudes and emotions within texts. Nonetheless, it is well-known that optimal lexicons are very difficult to obtain.

1.2 Problem Definition

Most would intuitively argue that the adjective “unpredictable” expresses negative sentiment. This assumption is indeed true in the context of vehicles like in “unpredictable steering”, but it would be misleading to say that an “unpredictable plot” of a movie is not positive. The example shows that the semantic orientation of words is sensitive to the domain. As a result, most sentiment lexicons fail to match polarities of words for multiple topics correctly. In addition, it has been demonstrated that lexicons only including unambiguous words (i.e., words that can only have one type of sentiment) are too general to perform well.

As a high-quality sentiment lexicon must be of domain specificity, the challenge to build an individual lexicon for each domain arises. While creating such dictionaries by hand is an infeasible effort, most automatic methods require large amounts of ground truth data. Ground truth (or gold standard) is text corpora where every piece of text is labeled with a polarity score. Because acquiring a labeled dataset for various domains is difficult as well, approaches that build upon a gold standard cannot be used broadly.

In short, to efficiently obtain sentiment lexicons, the generation method should be automatic and work without the necessity for ground truth data. In addition, the sentiments within each lexicon should be sensitive to the domain at hand.

1.3 Structure of the Work

The content of this thesis is divided into six chapters. Chapter 2 explores background and work related to sentiment analysis, ground truth and sentiment lexicon generation. Furthermore, word representation learning is discussed. In chapter 3, the developed method to automatically generate a sentiment dictionary is presented. In particular, an overview of the system architecture, concepts relevant to the approach, and details of the technical implementation are given. The experiments conducted to determine the effectiveness of the developed technique are examined in chapter 4. The findings are discussed in chapter 5. Finally, a summary of this work and thoughts on further improvements are given in chapter 6.

2 Background and Related Work

In this chapter, background information and findings about literature of related research areas are presented. The first sections describe the field of sentiment analysis and its background. Then, subjectivity analysis and the relation to sentiment analysis is introduced. The next section compromises challenges of sentiment and subjectivity analysis. It is followed by an elaboration of existing methods to perform sentiment analysis. Because ground truth is a significant factor, various ways to generate it are outlined afterward. Finally, sentiment lexicon generation that is forming a significant part of current research projects, and the closely correlated topic of word representation learning are discussed.

2.1 Natural Language Processing and Machine Learning

Natural Language Processing (NLP) is a subfield of *Artificial Intelligence* that focuses on the ability of computers to understand and process human (i.e., natural) language (Jurafsky, 2000). While humans can intuitively understand natural language, computers do not. Human language is a very complex and diverse concept, which makes it hard for machines to grasp its meaning. NLP is not only about analyzing words, but it is also about understanding context. There are two main techniques for NLP; rule-based methods and machine learning. While rule-based approaches define language by a broad set of manually created rules (e.g., grammar rules), machine learning tries to learn the rules automatically.

Machine learning uses algorithms and statistics to build and improve a model about some data (Bishop, 2006; Jurafsky, 2000). NLP and machine learning are the basis for several current research fields. Among others, these are language translation, text summarization, speech recognition, speech to text conversion and vice versa, topic modeling, and sentiment analysis. This master's thesis focuses on the latter.

2.2 Defining Sentiment and Sentiment Analysis

Sentiment Analysis is a field of research based on Natural Language Processing (NLP), linguistics, and machine learning (Liu, 2012). Research in this field started around the year 2000, among the pioneers were Das and Chen (2001), Morinaga, Yamanishi, Tateishi, and Fukushima (2002), Pang, Lee, and Vaithyanathan (2002), Tong (2001), Turney (2002), Janyce M. Wiebe, Bruce, and O'Hara (1999). Although some literature differentiates between *Sentiment Analysis* and *Opinion Mining*, most recent work uses the two terms interchangeably. Another term that refers to the same field of research is *Polarity Analysis*.

In general, sentiment analysis aims to analyze a text for the sentiment it is bearing automatically. Documents, sentences, and phrases can express positive or negative opinions. The goal is to determine which sentiment it is. Commonly, sentiment is classified as positive, negative (or neutral). Sometimes instead of three- or two-class classification, sentiment is defined with a score (i.e., strength), e.g., within a range from -1 to 1. Sentiment is called *valence*, *polarity*, or *degree of positivity* as well (Pang & Lee, 2008).

In literature and research, sentiment is also explained by *semantic orientation*. According to Hatzivassiloglou and Wiebe (2000), opinion words have a positive or negative semantic orientation based on the *state* they express. In specific, positive orientation corresponds to *desirable states* like "beautiful", while negative orientation is inferred from *undesirable states* like "ugly".

2.3 Sentiment Analysis Applications

Elaborating sentiment of large text corpora has many applications in market research, politics, stock markets, social sciences, and more (Feldman, 2013; Nakov, Ritter, Rosenthal, Sebastiani, & Stoyanov, 2016; Patodkar & I.R, 2016). According to Feldman (2013), classifying reviews of products and services is the most widespread application of sentiment analysis. In particular, automatically determining the overall opinion of people about a product, company, service, or brand is widely implemented. Manufacturing companies can find out what people think about their current products, such as what features they like or dislike, or if they are likely to buy another product. A popular example in politics is the tracking of public opinion about a candidate or political party. Such analysis can be performed by evaluating sentiments of tweets on Twitter. Among others, the gain for political parties is the knowledge about whether or not people support their current program.

2.4 Sentiment Levels

Liu (2012) divides current work on sentiment analysis in three main levels. First, document level analysis tries to classify the polarity of a whole document. A document could, for example, be a product review holding someone's opinion about a product. The goal of document level sentiment classification would be to reveal the sentiment of the opinion. It must be assumed that the whole document (i.e., review) only holds the opinion about one product, not multiple ones.

Second, sentence level analysis focuses on sentiment expressed within a single sentence (Feldman, 2013). Documents can contain many sentences, and the sentences could bear varying sentiments. The assumption for sentence level analysis is that there is only one opinion within each sentence. Some approaches distinguish between objective and subjective sentences before analyzing subjective ones for positive or negative sentiments (see section 2.5).

Third, entity and aspect level analysis, also called feature-based opinion mining by Hu and Liu (2004), looks more closely at the object the opinion refers to. It tries to evaluate the target of a sentiment. For example, a restaurant review could include a comment about the service. Aspect level sentiment analysis concentrates on linking the sentiment to the service (i.e., the target), instead of linking it to the restaurant itself. Performing analysis on this level results in very fine-grained and detailed sentiment information. While document and sentence level classification are already very difficult, entity level analysis is even harder to solve.

2.5 Subjectivity Analysis

A topic that goes hand in with sentiment analysis is subjectivity analysis. Subjectivity analysis aims at distinguishing subjective text from objective one (J. Wiebe, 2000; J. Wiebe & Riloff, 2005). Subjective text is opinion-, emotion-, or evaluation-oriented, while objective text represents facts. In the field of natural language processing, subjectivity classification should detect whether a document, sentence, phrase, or word is subjective or not. For example, J. Wiebe (2000) defines the sentence "At several different layers, it's a fascinating tale" to be a subjective sentence and "Bell Industries Inc. increased its quarterly to 10 cents from 7 cents a share" to be an objective sentence.

Janyce M. Wiebe et al. (1999) and Bruce and Wiebe (1999) found that the presence of adjectives in text correlates strongly with its subjectivity. Hatzivassiloglou and Wiebe (2000) look more closely at adjectives, and divide them into *dynamic adjectives,* *semantically oriented adjectives*, and *gradable adjectives*. They found that using either one of those sets for subjectivity analysis provides better results than using a complete set of all adjectives. In addition to adjectives, other word types like adverbs, nouns, and verbs are good indicators for sentiment and subjectivity (Pang et al., 2002). Among other features, Riloff, Wiebe, and Wilson (2003) exploited the use of so-called *subjective nouns* for subjectivity classification. Subjective nouns correspond to words implicitly bearing some emotion or opinion. Examples are "concern", "hope", or "support".

2.6 Challenges

Sentiment and subjectivity analysis face several challenges that add additional difficulty to the problems. In the following, the most relevant issues for this master's thesis are outlined. These are the challenges posed by unsatisfactory inter-annotator agreement, domain dependency of sentiment, multiple meanings of words, and implicit sentiments. As it would go out of the scope of this work, further obstacles like irony and sarcasm will not be discussed.

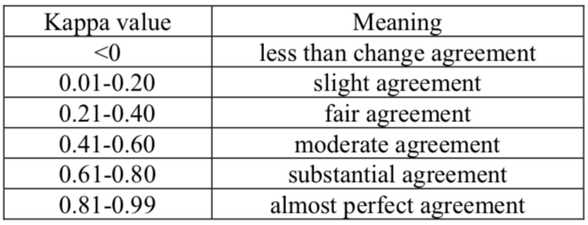
2.6.1 Inter-annotator Agreement

The inter-annotator agreement is a measure used to evaluate how often annotator interpretations match. In particular, it measures the amount of agreement between two or more humans about the polarity of text. Hence, if individuals are given the same pieces of text, how often do they agree about the texts' sentiments (e.g., positive, negative, or neutral).

Wilson, Wiebe, and Hoffmann (2005) assessed the human agreement about the polarity of subjective expressions. The study was depicted with two humans. The results showed that the annotators agreed in 82% of the cases, while the resulting Kappa value was 0.72. (see Table 2.1). However, 18% of the subjective expressions were classified as unclear by one or both annotators.

Saif, Fernandez, He, and Alani (2013) evaluated manually labeled datasets for sentiment analysis on document-level. In specific, the datasets consisted of tweets from Twitter[1] annotated by humans. One of the datasets was labeled by three graduate students who individually assigned a polarity class to each tweet. The resulting dataset was used to measure the inter-annotator agreement. A Krippendorff

[1] https://twitter.com



Kappa value	Meaning
<0	less than change agreement
0.01-0.20	slight agreement
0.21-0.40	fair agreement
0.41-0.60	moderate agreement
0.61-0.80	substantial agreement
0.81-0.99	almost perfect agreement

Table 2.1: Interpretation of Kappa values (Ku, Lo, & Chen, 2007)

alpha[2] (α) value of 0.765 (0 corresponding to perfect disagreement and 1 perfect agreement) was reached. Nevertheless, Shelley and Krippendorff (1984) state that an α of 0.8 and above can be seen as reliable data, while an α between 0.667 and 0.8 should only be used to draw *tentative conclusions*.

Another dataset evaluated by Saif et al. (2013) was established in a study conducted by Diakopoulos and Shamma (2010). Diakopoulos and Shamma (2010) reported an inter-annotator agreement of 0.655, expressed by the average Pearson correlation[3] of aggregated annotations by non-experts. In comparison, the inter-annotator agreement of three expert raters was 0.744 in terms of the average Pearson correlation.

These results show, not even for humans it is trivial to agree about the sentiment of texts. It can be observed that sentiment analysis is difficult. When machines learn from data that has been annotated by humans, it is likely to output similar inconsistencies.

Aroyo and Welty (2015) take on a different perspective about inter-annotator agreement overall. They argue that there is not always only one correct interpretation for text. In specific, when persons interpret sentences differently, all of them could be right. Rather than interpreting annotator disagreement as a problem that needs to be eliminated, Aroyo and Welty (2015) show that valuable information can be gained by looking more closely at why disagreement happens. It was found that annotator disagreement hints sentence ambiguity. Finding such sentences (texts) can be helpful in many cases.

2.6.2 Domain Dependency

Sentiment analysis is usually very dependent on the domain (Liu, 2012). The nature of the language used can differ significantly. Pang and Lee (2008) explain the domain dependency with differences in vocabulary, but also show more indirect cases. For

[2] Krippendorff, 2011.

[3] Snow, O'Connor, Jurafsky, and Ng, 2008.

example, the polarity of the sentence "go read the book" can have various interpretations. In a book review domain, the sentence would imply positive sentiment toward the book. Whereas in a movie review domain, it would express a negative opinion toward the movie.

In his study, Turney (2002) observed that the adjective "unpredictable" can have positive and negative sentiment depending on the context. An "unpredictable plot" of a movie would be a positive opinion about the movie, while "unpredictable steering" of a car implies a negative sentiment.

As a result, in machine learning when training a sentiment classifier on data from one domain, it is likely that the classification will perform poorly in other domains (Read, 2005). However, instead of building a generalized classifier, Owsley, Sood, and Hammond (2006) demonstrate that domain specificity is a major factor for better classification accuracy.

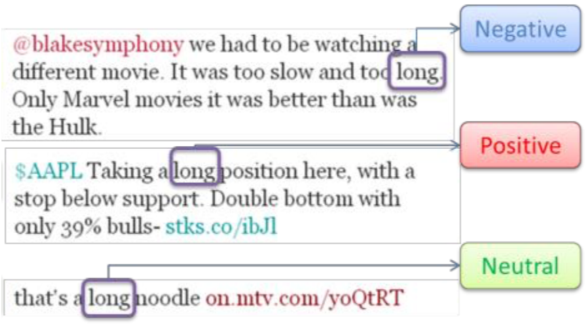
2.6.3 Multiple Meanings

Liu (2012) argues that words can have numerous meanings (i.e., word senses), especially in different contexts. As an illustration, they use the word "suck". While in the sentence "this camera sucks" it expresses negative sentiment, "this vacuum cleaner really sucks" implies a positive sentiment. Tai and Kao (2013) illustrated various senses of the word "long" (see Figure 2.1).

According to Chklovski and Mihalcea (2003), there are around 20,000 words that have more than one meaning. Ambiguity provides an additional challenge to sentiment analysis. Although it is often easy for humans to infer the correct sense from context, it is a challenge to machines (Chklovski & Mihalcea, 2003). That is why, a number of researches investigated the field of Word Sense Disambiguation (WSD), which aims at automatically finding a word's sense matching the current context.

2.6.4 Implicit Sentiments

Sentiment cannot only be expressed explicitly through opinion words but also implicitly (Fang & Zhan, 2015; Pang et al., 2002). That is, neutral words alone can communicate sentiment. For example, the sentence "item as described" of a product review reveals a positive opinion without including any sentiment specific words. In a movie review, the sentence "How could anyone sit through this movie?", obviously infers a negative opinion for humans. However, detecting implicit sentiments is one of the hardest challenges for machines.



@blakesymphony we had to be watching a different movie. It was too slow and too long. Only Marvel movies it was better than was the Hulk.

$AAPL Taking a long position here, with a stop below support. Double bottom with only 39% bulls- stks.co/ibJl

that's a long noodle on.mtv.com/yoQtRT

Figure 2.1: The examples by Tai and Kao (2013) demonstrate different meanings of the word "long". In the first tweet, "long" stands for the movie length and implies a negative emotion. In the second tweet, it corresponds to the "long" buying position of investors, and is positive. In the last one, it is a neutral description about the noodle. (Tai & Kao, 2013)

2.7 Sentiment Analysis Methods

Since the year 2000, many researchers investigated the problem of sentiment analysis. Thus, numerous approaches have emerged. In their survey, Pang and Lee (2008) categorize them into unsupervised, supervised, semi-supervised, and domain adaptation approaches. There also exist hybrid methods combining two or more of them. The sections below describe each of the categories in more detail.

2.7.1 Unsupervised Approaches

There are a number of different techniques to perform unsupervised sentiment analysis. A widely used method is based on the use of a so-called sentiment lexicon. In this work, sentiment analysis methods that are based on lexicons are referred to as lexicon-based approaches like proposed by Taboada, Brooke, Tofiloski, Voll, and Stede (2011). Lexicon-based sentiment analysis makes use of words that are assigned specific polarities in order to determine sentiments for text.

Sentiment Lexicons

A sentiment lexicon is a dictionary that assigns a polarity to all the words (and phrases) it includes (Gilbert, 2014). The polarity can be presented as a score within a range of numbers. For example, the range could go from the number ten corresponding to very positive sentiment, to the number minus ten for very negative sentiment. Alternatively, the polarity is simply denoted as either positive, negative, or neutral.

Lexicon	Values	Entries
MPQA	[−1, 0, 1]	7,192
GI	[-1, 1]	3,629
SentiWordNet	−1.0 → 1.0	147,700
LIWC	[−1, 0, 1]	2,322
ANEW	1 → 9	1,034
VADER	−4 → 4	7,502

Table 2.2: Summary of well-known sentiment and subjectivity lexicons. It includes an overview of whether continuous or binary scores are assigned to words, and how many words are present in the lexicons (based on Gilbert, 2014; Reagan, Danforth, Tivnan, Williams, & Dodds, 2017).

Moreover, some lexicons only distinguish between subjective and objective words or phrases without assigning a degree of positivity. To provide an overview, some well-known and extensively studied lexicons are listed in Table 2.2.

The Multi-Perspective Question Answering (MPQA) is a subjectivity lexicon by J. Wiebe, Wilson, and Cardie (2005). SentiWordNet, ANEW, and VADER are sentiment lexicons that express polarity with continuous scores. SentiWordNet (Baccianella, Esuli, & Sebastiani, 2010) is an extension of WordNet (Fellbaum, 1998) with sentiment values relating to *positivity*, *negativity*, and *objectivity (neutrality)*. Affective Norms for English Words (ANEW) (Bradley & Lang, 1999) is a sentiment lexicon with valence scores from 1 (very negative) to 9 (very positive) with a neutral midpoint at 5 (neutral). VADER (Gilbert, 2014) is a lexicon specifically tailed for Twitter sentiment classification. The Harvard General Inquirer (GI) (Stone, Dunphy, & Smith, 1966) and LIWC (Pennebaker, Francis, & Booth, 2001) categorize words into binary classes.

Many dictionaries do not include informal words often used on social media platforms like Twitter (Peng & Park, 2011). That is why these dictionaries perform poorly for such domains. To overcome this issue, researchers started to build dictionaries specifically tailored to micro-blogging domains. For example, the new ANEW dictionary by Nielsen (2011). Feldman (2013) stated that for almost all sentiment analysis approaches, sentiment lexicons play the most important role.

The creation of sentiment lexicons is discussed in section 2.10.

Lexicon-based Sentiment Analysis

In order to obtain a sentiment score for a document or sentence, the polarity of individual words and phrases is combined to a single score. The main idea is to either count the number of positive and negative words or to check the presence of sentiment-bearing words, to infer an aggregated polarity score. Hu and Liu (2004) and Ding, Liu, and Yu (2008) calculate sentiment scores on sentence level based on the polarity of opinion words in the sentence. Opinion words are words expressing a subjective opinion or sentiment. In the work of Hu and Liu (2004), opinion words are all adjectives present in the corpora. Their algorithm extracts the adjectives from a sentence and looks up the adjective's polarity in a self-constructed lexicon. If the positive words outweigh the negative ones in a sentence, the sentence is tagged as positive, and vice versa. If a sentence includes an equal amount of positive and negative words, *effective opinions* are averaged and used to assign the sentence polarity (Hu & Liu, 2004).

In addition to adjective sentiments, Kim and Hovy (2004) induce sentiment polarity using verb and noun polarities. The used lexicon includes both a negative and a positive score for each word. Taking into account the context of a word, the right score, positive or negative, was used to evaluate the word's sentiment within a sentence. Negation words like *not* were used to reverse sentiment scores. It was found that the presence of a negative sentiment word better described the polarity of a sentence than the exact score.

The study by Annett and Kondrak (2008) shows that a lexicon has to have just the right number of entries, not too much nor too less, in order to perform well. Otherwise, documents are likely to be under or over analyzed. Annett and Kondrak (2008) also concluded that achieving an accuracy higher than 65% with a classic lexicon-based method is hard. According to Reagan, Danforth, Tivnan, Williams, and Dodds (2017), lexicon-based approaches have three major disadvantages. First, the authors argue that they are only suitable for texts longer than one sentence. Second, it has been shown that machine learning techniques can outperform lexicon-based ones on specific corpora. Third, the words in the lexicon might have a misleading polarity for the corpus at hand if their sentiment has been evaluated based on the usage in a different domain.

More sophisticated lexicon-based techniques make use of linguistics patterns. For example, Ding et al. (2008) integrated external information to evaluate the polarity of opinion words, phrases, and other language constructs. The usage of these constructs within the text corpora was exploited to infer domain-specific sentiment scores. In specific, conjunction rules, synonym and antonym rules, and negations provided the basis for the algorithm. The polarity constructs were evaluated on a review dataset yielding F-scores above 0.9.

Turney (2002), as well, classified reviews, hence performing document level analysis, based on the polarity of phrases. Groups of words that often appear together are extracted from documents and assigned polarities. The semantic orientation of the phrases within a document is averaged to obtain the sentiment score of the document.

2.7.2 Supervised Approaches

Supervised machine learning has become popular with opinion mining. In contrast to lexicon-based approaches, machine learning approaches make use of human-labeled text. Liu (2012) distinguishes between sentiment classification and sentiment regression. While the goal of classification is to assign one category out of a fixed set of categories, regression assigns a score within a range of values. The categories are usually the classes positive, negative, and neutral. However, most research omits the neutral class. Before going into more detail, the concept of features is described.

Features

According to Liu (2012), finding effective features is crucial to supervised sentiment classification. Feature vectors are representations of text that can be used as inputs for machine learning algorithms. The process of selecting appropriate features is often referred to as *feature engineering* in literature (Scott & Matwin, 1999). A complete discussion of possible features for text classification would go out of the scope of this work. To provide a basic idea of the most commonly used features in sentiment analysis, a selection adapted from Liu (2012) and Pang and Lee (2008) is outlined as follows.

- **Term Frequencies or Term Presence:** In order to represent a text document as a vector, the text is split into its individual terms (i.e., unigrams). Each entry in the feature vector corresponds to one term. In the case of frequencies, the entry is the number of occurrences of that term in the text. In case of presence, the value is 1 if the term appeared in the text, 0 if not. When term positions are not taken into account, the *bag-of-words* model is exploited. In other words, the *bag-of-words* representation does not preserve the order of terms from the original text. A prominent example of using word presence for polarity classification are Pang et al. (2002). To overcome the information loss of term order, some works use position features in addition to word features. For instance, Kim and Hovy (2006) exploit the nature of document summarization. That is, important sentences are often positioned at the beginning and end of paragraphs.
- **N-grams:** Word n-grams are "sequences of n consecutive words extracted from text", (Majumder, Mitra, & Chaudhuri, 2002). In other words, an n-gram is a contiguous series of terms in a text. A 1-gram is called *unigram* and consists of

a single word. An n-gram of two words is called *bigram*, and so on. Different length n-grams can be used as entities in feature vectors. Bigrams and trigrams are a popular choice when it comes to integrating phrases. However, Pang et al. (2002) argue that unigrams work better with sentiment classification than higher order n-grams. Dave, Lawrence, and Pennock (2003) challenge this finding by showing that bigrams and trigrams increase accuracy in product review polarity classification.

- **Part of Speech:** According to Macmillan-Education (2018), part of speech (POS) is "one of the main grammatical groups that a particular word belongs to according to the way it is used in a sentence, for example, noun, verb, adjective, or adverb". In the natural language processing domain, such groups are called *POS tags*. A broad tag set could consist of noun, verb, adjective, adverb, pronoun, numeral, preposition, and conjunction, like in the work of Petrov, Das, and McDonald (2011). A finer-grained tag set was created by Marcus, Marcinkiewicz, and Santorini (1993), and is used in various polarity analysis approaches. Among others, Hatzivassiloglou and McKeown (1997) showed that adjectives can effectively indicate sentiments. Studies of Pang et al. (2002) have indicated that nouns and verbs are good indicators as well. That is why POS tagging is often exploited to extract adjectives, adverbs, verbs, and nouns from text in order to use them as features.
- **Sentiment Shifters:** According to Liu (2012), sentiment (or valence) shifters are words that change the orientation of opinion words. The most studied sentiment shifters are negation words like the expression "not". As an illustration, "I like apples" and "I don't like apples" have opposing semantic orientation. Negations can be indirectly addressed using second-order features. That is, first creating feature vectors from terms without considering negations, then changing the feature vectors to be negation-aware (Pang & Lee, 2008). There also exist practices that directly include negation into the initial features. For example, (Pang et al., 2002) added the prefix "NOT_" to feature entries of negated terms. Shifters have to be handled with care because they do not always imply opposite sentiment.

Researches have introduced multiple other ways of feature engineering for polarity analysis. Recently, word embedding (i.e., word vectors) in vector space were elaborated. They will be explained in section 2.11.

Sentiment Classification

Pang et al. (2002) were one of the first performing sentiment classification. The aim was to classify movie reviews as either positive or negative. Well-known text classification algorithms were evaluated to determine the effectiveness of machine learning for sentiment classification. The three algorithms are Naive Bayes, maximum entropy classification, and support vector machines (SVMs), as these were already successfully deployed for topic-based classification (McCallum, Nigam, et al., 1998). It was observed that opinion can be expressed in a very subtle manner. For example, the text "How could anyone sit through this movie?", (Pang et al., 2002) reveals a negative sentiment toward the movie without the explicit use of opinion words. Therefore, it was assumed that machine learning methods would require more *understanding* than in other use cases. To train the classifier labeled movie reviews from IMDB[4] were used. Various combinations of input features were tested. The concept of features was previously introduced in section 2.7.2. In general, the features consisted of unigrams and bigrams.

Additionally, when a negation term preceded a word, a negation tag got attached to the word. It was shown that SVMs using only unigrams as input features performed best. Feature vectors containing presence information rather than frequency counts of unigrams resulted in higher accuracy. This finding differs from topic-based classification because frequent keywords tend to emphasize topics (McCallum, Nigam, et al., 1998).

Sentiment Regression

In the field of machine learning, classification is about predicting a discrete class label (positive, negative, or neutral), while regression predicts a continuous quantity (Michalski, Carbonell, & Mitchell, 2013). Sentiment strength or degrees of positivity is a natural way to express polarity. Consequently, formulating the task as a regression problem rather than a classification one appears to be suitable for sentiment analysis. To take a case in point, the problem of predicting one-to-five star ratings for reviews was tackled using both regression and classification methods by several researchers. Pang and Lee (2005) studied both methods to predict rating scores from movie reviews. The advantage over classification is that regression can capture the relationship between close labels. For example, a one-star rating is similar to two stars, and four stars are similar to five.

It is worth highlighting that in contrast to unsupervised approaches, supervised ones require a large amount of labeled training data (i.e., ground truth). That is, sentences (or documents) with sentiment scores. However, acquiring such quantities of data can be difficult (Zhu, 2005).

[4] https://www.imdb.com

2.7.3 Semi-Supervised and Active Learning Approaches

Semi-supervised learning utilizes unlabeled data in combination with labeled data (Liu, 2012). The shortage of gold standard labeled data for sentiment analysis and the availability of large sets of unlabeled data promotes the use of semi-supervised techniques. They try to overcome the lack by integrating unannotated texts. Hajmohammadi, Ibrahim, Selamat, and Fujita (2015) exploited semi-supervised self-training to classify unlabeled data examples. To do so effectively, the existing annotated data used to train the initial classifier needs to be of high quality. The newly labeled examples are added to the training set, and the classifier is retrained. In addition, active learning was employed. Active learning selects *most informative* examples and gives them to a human for labeling. These are later on included in the training set as well. Goldberg and Zhu (2006) implemented semi-supervised learning for the prediction of star ratings. The authors used a graph-based approach, where reviews were placed on the nodes and links presented the similarity between reviews. The task of assigning polarity to unannotated nodes from similar annotated ones was addressed as an optimization problem over the graph.

2.7.4 Domain Adaptation

As a large amount of labeled data from a specific domain is seldom available, domain adaptation utilizes labeled data from one domain (i.e., original domain, or *source domain*) for sentiment classification in another domain (i.e., *target domain*). However, as discussed in subsection 2.6.2, the performance of a sentiment classifier heavily depends on the domain its training data came from. When the same classifier is used for a different domain, the results can be very dissatisfying. This is due to the reason that ways and words to express opinions depend on the context. According to Liu (2012), there are two main methods used for domain adaptation.

For the first method, a small amount of labeled data from the target domain is necessary. Aue and Gamon (2005) compared four algorithms to train a sentiment classifier for a domain where little annotated data is available. The best performing algorithm makes use of a small amount of labeled, and a big amount of unlabeled data from the target domain. Incorporating annotated data from different domains nor combining multiple classifiers returned better results.

For the second method, only unlabeled data from the target domain is needed. Most of these methods transform a classifier trained on the source domain in order to be suitable for the target domain. Blitzer, McDonald, and Pereira (2006) introduced so-called *structural correspondence learning (SCL)*. SCL identifies features that correlate in the source and target domain. A Part of Speech (POS) tagger is trained using labeled data from the source domain. Then, unlabeled data from both domains is used to define common features. Finally, the classifier is trained with the combined features from the source domain. It was found that the classifier performs well for the target domain. Blitzer, Dredze, and Pereira (2007) studied domain adaptation specific to sentiment classification with an improved version of structural correspondence learning. Additionally, a measure for domain similarity was elaborated. This measure describes how well a source domain is suited to adapt a classifier to the target domain.

2.8 Ground Truth

Ground truth, or *Gold Standard* "exist in order to train, test, and evaluate algorithms that do empirical analysis", (Aroyo & Welty, 2015). In other words, ground truth is the basis for creating and evaluating machine learning methods. For sentiment analysis, that is corpora annotated with polarity information. For example, a dataset of sentences labeled with positive, negative, or neutral tags that can be used for sentence level classification. The more closely annotated data mirrors the real world, the more accurately the trained classifier can reflect it in the end. Often, human annotated corpora is assumed to be the best gold standard. Nevertheless, Aroyo and Welty (2015) argue that good ground truth does not necessarily imply high inter-annotator agreement (see subsection 2.6.1).

2.9 Ground Truth Generation

In order to collect ground truth for sentiment analysis, various methods have been established in past studies. In the following sections, an outline of the most common ways to create ground truth is given.

2.9.1 Human Labeling

Human labeling means people manually annotate the data (Aroyo & Welty, 2015). Every entry in a dataset is labeled with a polarity tag (e.g., positive, negative, or neutral), or a continuous score by a person. People who label a dataset can be either experts or non-experts. Usually experts are trained *subject matter experts* (SMEs). Often, SMEs are not available, or a large enough number is not feasible.

Therefore, ways to overcome this limitation arose. One of them is crowdsourcing, and it has become a common medium to gather human annotated data. The idea behind crowdsourcing is having a large number of individuals who label a lot of data entries. Because these are non-experts, implementing measures to validate the annotations are necessary. Popular platforms for crowd labeling are Amazon Mechanical Turk[5] (mTurk) and Figure Eight[6] (previously known as CrowdFlower).

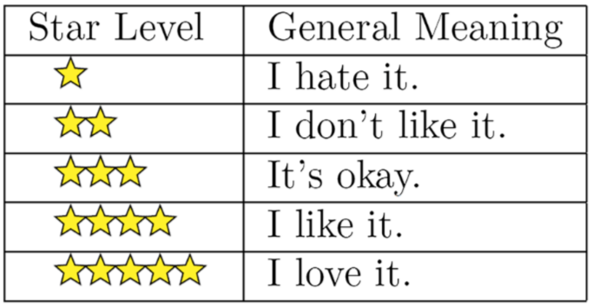
Amazon mTurk is an online crowdsourcing marketplace for work tasks (Nowak & Rüger, 2010). *Human Intelligence Tasks* (HITs) can be created, and workers at MTurk, so-called *turkers* can do them for a predefined reward. In case of sentiment analysis, a HIT could consist of a text that should be annotated with a polarity by the turkers. Figure Eight is a platform similar to Amazon mTurk.

For quality control, Amazon mTurk provides the possibility to ask for a proof of qualification from the turkers (Nowak & Rüger, 2010). Most turkers are not trained for the tasks they are performing. Therefore, their competence can be elaborated using a qualification test. This is supposed to filter out low quality workers. In addition, Amazon provides an approval rating of every turker. The approval rating is the ratio between the accepted HITs of a turker compared to the total HITs executed (Akkaya, Conrad, Wiebe, & Mihalcea, 2010). Other methods to assure quality are, for example, *repeated labeling* or *hidden tests*. Usually, multiple annotators are asked to label the same text in order to collect more than one label per text. With this data, agreement measures between annotators can be calculated and used to validate the labels. To illustrate how quality control has been implemented in recent studies, the procedures of Nakov et al. (2016) and Gilbert (2014) are outlined as follows.

Nakov et al. (2016) collected annotations with Amazon's Mechanical Turk and CrowdFlower for the *Sentiment Analysis in Twitter* task. For quality assurance, turkers were requested to have an approval rate higher than 95% with at least 50 approved HITs. Repeated labeling was performed, because every HIT needed to be executed by five turkers. According to Nakov et al. (2016), CrowdFlower provides better quality control than Amazon mTurk. That is why, CrowdFlower was used to annotate the test dataset. At CrowdFlower it is possible to add hidden tests, and that is what they did. In past years, to consolidate the annotations the label of the majority voting (> 50%) was selected. In 2016, a more complex method was introduced to consolidate the five-scale ratings. If at least three annotators agreed, the rating that was agreed on was used. Otherwise, the average of the five ratings was calculated and mapped to the closest integer value.

[5] https://www.mturk.com

[6] https://www.figure-eight.com



Star Level	General Meaning
★	I hate it.
★★	I don't like it.
★★★	It's okay.
★★★★	I like it.
★★★★★	I love it.

Table 2.3: Interpretation of Amazon's rating system (Fang & Zhan, 2015)

Gilbert (2014) used Amazon's Mechanical Turk as well. Instead of labeling documents (tweets) various lexical features were annotated. Each lexical feature was rated by ten turkers on a scale from -4 (extremely negative) to 4 (extremely positive). In order to obtain high quality labels, four steps were taken. First, the annotators were checked for good reading comprehension skills. Second, they had to perform an online training on sentiment rating. Third, hidden tests in the form of lexical features that had been pre-labeled with a sentiment score were included in the tasks. Finally, turkers delivering very high quality ratings were rewarded.

As human labeling is very expensive from a resource and time perspective, researchers started to look for automated approaches to generate ground truth data.

2.9.2 Inference from Ratings

One technique to annotate unlabeled datasets automatically, without human labeling efforts, is to infer labels from ratings. The method can be used for any type of text that has some kind of rating attached to it. This could, for example, be user reviews that include a star rating. Fang and Zhan (2015) and Blitzer et al. (2007) used this approach to generate ground truth for sentiment analysis of product reviews. The dataset of Fang and Zhan (2015) consists of 5.1 million product reviews from Amazon[7], and it is assumed that star ratings generally correspond to the meanings pointed out in Table 2.3. Reviews with a 4 or 5 star rating were tagged as positive, and reviews having a 1 or 2 star rating were tagged as negative. Reviews with a 3 star rating were labeled with the neutral tag. In contrast, while Blitzer et al. (2007) implemented the same approach to create polarity labels for Amazon product reviews, they discarded 3 star ratings because of ambiguity assumptions.

[7] https://www.amazon.com

Another work that uses ratings to generate labels is the sentiment classification study of Pang et al. (2002). To tackle the problem of missing ground truth data for training machine learning algorithms, Pang et al. (2002) labeled IMDB[8] movie reviews based on ratings. Star or numerical ratings were extracted automatically from the reviews in the dataset, and each review got assigned a positive, negative, or neutral tag depending on the rating. The same approach was used for one of their follow-up studies in 2004 (Pang & Lee, 2004).

Advantages of using ratings to label sentiment text are that it is a reliable and easy way to generate ground truth data. However, ratings are often not available, making it infeasible to use for most corpora outside of the review domain.

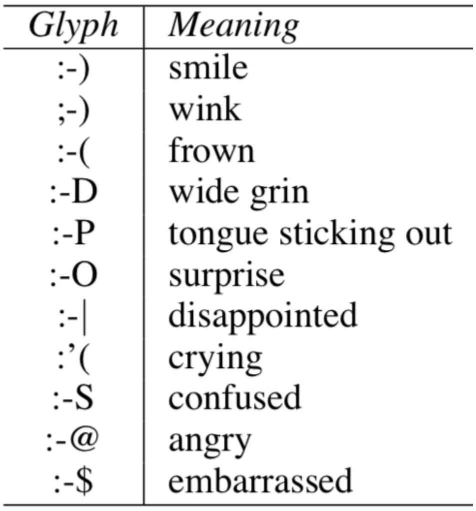
2.9.3 Inference from Emoticons and Hashtags

Instead of using ratings to generate ground truth for sentiment analysis, techniques exist that use emoticons (and hashtags). These overcome the limitation of the necessity to have explicit rating information for every text. The pioneer who first used emoticons to automatically label texts was Read (2005). Emoticons (or smileys) are "visual clues that are associated with emotional states in an attempt to state the emotion that their text represents", (Read, 2005). In other words, non-textual entities in a text that express sentiments are known as emoticons (or smileys). Figure 2.2 shows some existing emoticons and their meanings. Studies exist on how to build a sentiment lexicon of emoticons (or emojis), like the one by Kralj Novak, Smailović, Sluban, and Mozetič (2015).

Read (2005) collected articles from Usenet newsgroups that included at least one emoticon. Then, all articles containing a *smile* emoticon were tagged as positive, and articles containing a *frown* emoticon as negative. The gained dataset constituted the ground truth data for further sentiment classification.

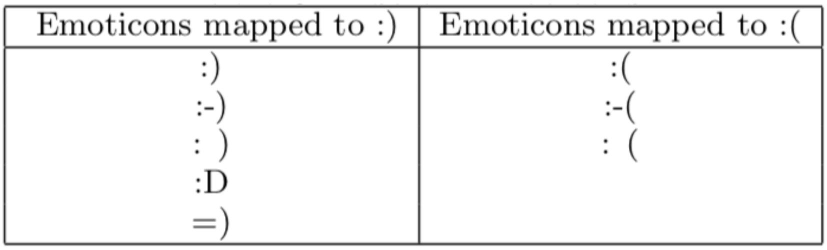
Especially for the Twitter domain, such approaches have been successfully implemented. For example, Go, Bhayani, and Huang (2009) make use of emoticons as noisy labels for tweets. Positive tweets were retrieved from the Twitter API using the ":)" query, and negative ones were retrieved using the ":(" as the query. The Twitter API would return tweets containing any positive (or negative) smiley (see Figure 2.3). The collected tweets were used as training data for the sentiment classifier. While an accuracy of 81% was achieved for binary classification, the authors were not able to gain good accuracy for three class (incl. neutral) classification.

[8] https://www.imdb.com



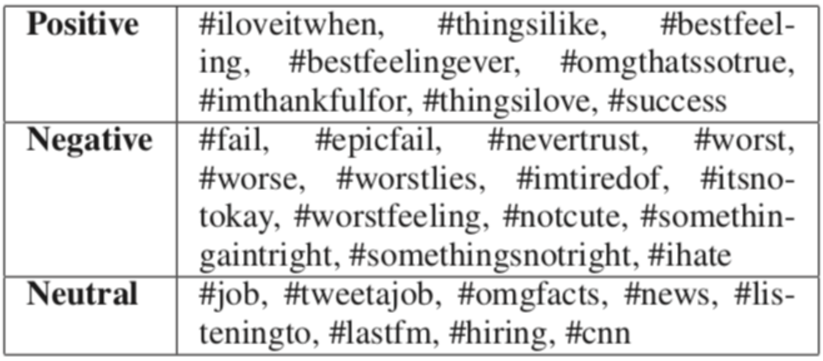
Glyph	*Meaning*
:-)	smile
;-)	wink
:-(	frown
:-D	wide grin
:-P	tongue sticking out
:-O	surprise
:-|	disappointed
:'(	crying
:-S	confused
:-@	angry
:-$	embarrassed

Figure 2.2: Examples of emoticons used in Usenet newsgroups (based on Read, 2005)



Emoticons mapped to :)	Emoticons mapped to :(
:)	:(
:-)	:-(
:)	: (
:D	
=)	

Figure 2.3: Emoticon mapping performed by the Twitter API (Go, Bhayani, & Huang, 2009)



Positive	#iloveitwhen, #thingsilike, #bestfeeling, #bestfeelingever, #omgthatssotrue, #imthankfulfor, #thingsilove, #success
Negative	#fail, #epicfail, #nevertrust, #worst, #worse, #worstlies, #imtiredof, #itsnotokay, #worstfeeling, #notcute, #somethingaintright, #somethingsnotright, #ihate
Neutral	#job, #tweetajob, #omgfacts, #news, #listeningto, #lastfm, #hiring, #cnn

Figure 2.4: Examples of positive, negative, and neutral hashtags (Kouloumpis, Wilson, & Moore, 2011)

A more recent study by Patodkar and I.R (2016) makes use of the method proposed by Read (2005) as well. Patodkar and I.R (2016) perform sentiment analysis and opinion mining in the micro-blogging domain using Twitter. In contrast to Go et al. (2009), the goal was to achieve good results for three class sentiment classification. In order to establish ground truth to build classifiers, emoticons were used to attach a sentiment to every tweet of the training data. In particular, positive and negative tweets were collected using queries of happy and sad emoticons. It was assumed that tweets from newspapers and magazines are objective. That is why, to gather objective data, tweets from multiple newspapers were queried. As a side note, the authors found that specific POS tags of a text can indicate whether the text bears some sentiment or not.

Beside emoticons, the use of hashtags for tweet annotation has been exploited. Davidov, Tsur, and Rappoport (2010) tried to predict hashtags and smileys for tweets in order to perform sentiment identification. In the study, sentiment hashtags and smileys were used as annotations for tweets. Sentiment hashtags are user-defined hashtags that bear some kind of sentiment. Fifty of such common hashtags were manually selected. In addition, often used and unambiguous smileys were elaborated and included. Hence, hashtags and smileys were utilized as sentiment labels.

Kouloumpis, Wilson, and Moore (2011) used hashtags for three class sentiment classification. To begin with, the authors retrieved the most common hashtags and manually labeled positive, negative, and neutral hashtags they thought to be most useful. The resulting hashtags are shown in Figure 2.4. For the purpose of generating ground truth from an existing tweets dataset, tweets containing one of these hashtags were selected and tagged with the hashtag's polarity class.

Guthier, Ho, and Saddik (2017) introduced a more sophisticated approach to detect the polarities of hashtags. The method is based on Wang, Wei, Liu, Zhou, and Zhang (2011) and includes multiple steps. First, known emoji sentiments are exploited to map sentiments to hashtags that co-occur with these emojis. Then, a graph of hashtags is built that connects hashtags that often appear together in tweets. Finally, a hashtag's polarity is calculated from its neighboring hashtags' sentiments. The resulting hashtag sentiment dictionary is used to annotate tweets that become the ground truth data.

Ground truth generation methods based on emoticons (and hashtags) have been shown to produce satisfactory data in domains where emoticons (and hashtags) are common. In addition, these are language-independent, and thus, provide a tool for sentiment classification in any language (Guthier et al., 2017). Nevertheless, the necessity of having emoticons (or hashtags) available in the text make these approaches unsuitable for various domains that do not have this characteristic.

2.9.4 Lexicon-based Ground Truth Generation

Lexicon-based techniques rest on word lists that consist of terms labeled with polarities. For every text in the dataset, the presence or frequency of polarity words is evaluated. This information is used to annotate the text with a sentiment value. To generate ground truth for sentence level classification, Fang and Zhan (2015) exploited a sentiment lexicon. After splitting every sentence into separate tokens, the number of negative and positive tokens was calculated. The sentence was labeled as positive if there were more positive than negative tokens in the sentence, and vice versa.

Methods based on sentiment lexicons do require neither ratings nor emoticons nor sentiment hashtags. Therefore, lexicon-based techniques can be useful in a wider range of domains. However, the performance of such techniques heavily depends on the used sentiment lexicon (Reagan et al., 2017). Usually, words in lexicons have been annotated with a polarity specific to one-word sense. When a word has a different sense in the evaluated context, the lexicon's polarity of that word might be wrong. This means lexicon-based approaches perform worse if the lexicon's domain does not match the dataset's domain.

2.10 Sentiment Lexicon Generation

A sentiment lexicon can be inducted in three main ways; human labeling, dictionary-based, and corpus-based (Feldman, 2013). The first method implies that humans are hand-labeling the lexicon. The second approach includes a dictionary and expands a list of seed words with known sentiments with more sentiment words. The third method uses a text corpora from a single domain to gather sentiment information for words (or phrases). Another technique that closely correlates to the corpus-based approach exhibits words in a vector space to determine their sentiments. In the following, these methods will be described in more detail.

2.10.1 Human Labeling

Similar to human labeling for gathering ground truth data like described in subsection 2.9.1, manual annotation can be performed for sentiment lexicon generation. However, Feldman (2013) indicates that hand-labeling a sentiment lexicon is infeasible, as every domain requires its version, and efforts are too high. This is why we will not go into detail about human labeling here.

2.10.2 Dictionary-based Lexicon Generation

Dictionary-based approaches depend on the information that can be extracted from a thesaurus (or dictionary) in order to assign sentiment labels to words. Most methods build around the concept of spreading polarities from a small number of seed words to other words. Seed words are terms that have known, unambiguous polarities that should be the same in every domain. For example, "happy" should always imply a positive sentiment. Often, sentiments are propagated exploiting the principle of synonyms and antonyms. In this case, it is assumed that a word's synonyms bear the same or similar sentiment, while antonyms have opposing sentiments (Kim & Hovy, 2004). Multiple researchers employed this method.

Kim and Hovy (2004) classified words as positive or negative, starting from two lists of seed words. The authors created one list of positive and another list of negative seed words. Then, the lists were grown using synonym and antonym relationships from WordNet[9] (Miller, 1995). In specific, synonyms of positive words and antonyms of negative words were added to the positive list, and vice versa for the negative list. The process was repeated until the lists included around 12.000 words in total. To counter word ambiguity, a measure for sentiment strength was developed.

Another study that deployed a similar technique is Hu and Liu (2004). Hu and Liu (2004) used WordNet's synonym and antonym structure to identify adjective polarities. Starting from a seed list of positive and negative adjectives, the list was grown similar to Kim and Hovy (2004). This way, the authors were able to label nearly all adjectives of their corpora. Using the adjective polarities, sentences were

[9] https://wordnet.princeton.edu/

labeled with a sentiment value. In the end, it was argued that using WordNet to create a sentiment lexicon for lexicon-based sentence annotation is highly effective.

Adreevskaia and Bergler (2006) introduced a *clean-up* phase to the approach. In specific, after every round of adding synonyms and antonyms to the lists, a clean-up was conducted. During the phase words that had been labeled both negative and positive were filtered out. The authors found that the quality of the resulting sentiment lexicon depended heavily on the seed list. It was highlighted that ambiguous adjectives in the seed list should be avoided.

Label Propagation using Graphs

Kamps, Marx, Mokken, De Rijke, et al. (2004) proposed to assign sentiments to words using a graph model. The authors built a graph based on synonym relations extracted from WordNet. Words represented the nodes and were connected if they had a synonym relationship. It was assumed that similar words were linked. The distance (i.e., the shortest path) revealed the similarity between words. The polarity was calculated using the shortest path from the nodes "good" and "bad". Parts of the resulting graph from the perspective of the adjective "good" are shown in Figure 2.5. It was found that although "good" and "bad" should have opposing sentiments, they were close in the graph. This is why, for every word the distance to both words has to be taken into account in order to imply a valid polarity.

The lexicons acquired using a thesaurus and expansion or propagation techniques are usually domain independent. Consequently, they fail to reflect the detailed polarities of words in specific domains (Feldman, 2013). In contrast, corpus-based approaches indirectly incorporate domain peculiarities, making semantic orientations more precise.

2.10.3 Corpus-based Lexicon Generation

Corpus-based approaches rely on linguistic heuristics and the structure of texts to determine the sentiment of words (or phrases). These methods are based on a large corpus of texts (sentences or documents). Hatzivassiloglou and McKeown (1997) were the first to propose corpus-based sentiment detection for adjectives. The authors exploited basic principles of conjunction rules. In particular, conjunctions should reveal the semantic similarity of the words linked by the conjunction. For example, "and" usually connects words (or adjectives) of the same sentiment, while "but" separates words (or adjectives) of differing sentiments. A linear regression model was employed to the adjectives and conjunctions, which resulted in a graph. The

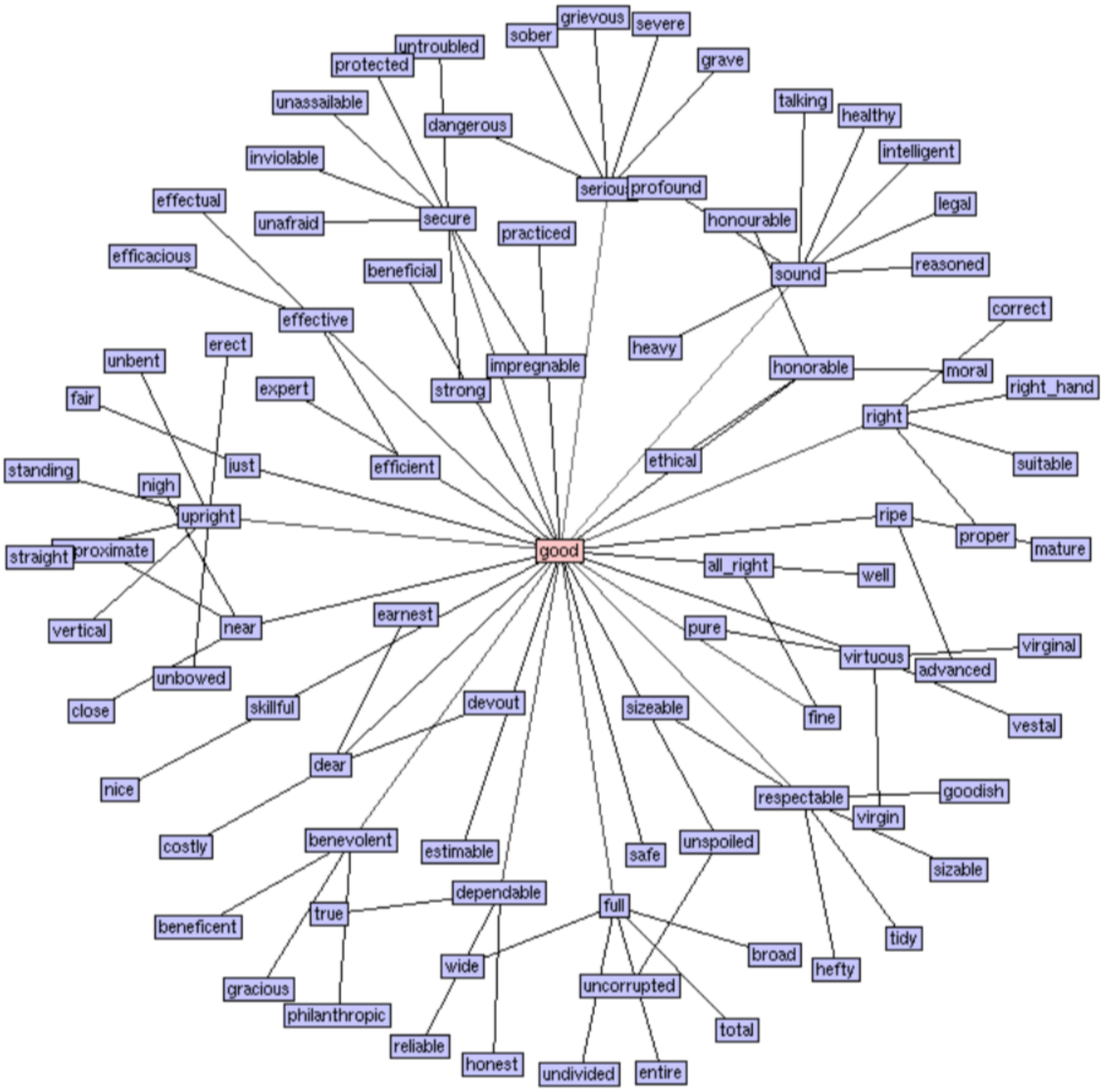


Figure 2.5: Shortest path neighbors of "good" based on WordNet synonyms (Kamps, Marx, Mokken, De Rijke, et al., 2004)

nodes would correspond to the adjectives, and the links would denote the sameness or difference in polarity. Then, the adjectives were separated into a positive and negative subset after applying a clustering algorithm. They were able to achieve accuracies of up to 92% for adjective sentiment classification.

To determine polarities of phrases, Turney (2002) created a *Pointwise Mutual Information and Information Retrieval* (PMI-IR) algorithm. The similarity of phrases was determined by a mutual information measure. The mutual information of two phrases is based on their co-occurrence in the corpus. In specific, the similarity of a phrase to the word "excellent" (positive sentiment) was subtracted from the similarity of the phrase to the word "poor" (negative sentiment). The resulting value described the semantic score of the phrase, in short, its tendency to co-occur with "excellent" or "poor".

Because corpus-based methods build on the underlying corpus, domain-dependent sentiment information is gained (Hatzivassiloglou & McKeown, 1997). That is, the automatically generated sentiment lexicon mirrors correct sentiments for the corpus' domain. This is an advantage over pure dictionary-based methods.

Peng and Park (2011) argue that a combination of information from both a dictionary and the corpus outperforms using either one of them individually. This argument has been validated by Tai and Kao (2013) who combined dictionary- and corpus-based methods to generate a domain-specific sentiment lexicon.

2.10.4 Lexicon Domain Adaptation

Domain adaptation methods come in handy when lots of labeled data is available for one domain (i.e., source), but none or little exists for the target domain. It aims to transfer sentiment information from one domain to another. The algorithm proposed by Li, Pan, Jin, Yang, and Zhu (2012), implemented a bootstrapping technique to extend a small list of seed words in the target domain using labeled data from the source domain. The basis is formed by commonality opinion words that exist and have the same sentiment in both domains. Additionally, sentiment words representative for the target topic are extracted exploiting topic sentiment words from the source domain. Finally, bootstrapping is used to propagate polarities and generate a sentiment lexicon for the target domain (Li et al., 2012).

2.11 Word Representation Learning

In the context of this master's thesis, word representation learning corresponds to the practice of encoding words (or phrases) as vectors. Models that embed words in a continuous vector space are called vector space models[10] (VSM). The distance or angles between vectors in the vector space represent their similarity (Maas et al., 2011). As for word vectors, these measures correspond to the similarity of words. It is assumed that words appearing in the same context (e.g., sentence, document, or word window) have common semantics or meanings (Miller & Charles, 1991). This means when word vectors are learned from corpora, the vectors encode word relations within the corpora, e.g., semantic *term-document* information. In other words, semantically similar words are placed nearby each other in the vector space (Maas et al., 2011).

There are two main practices making use of the semantics principle; *count models* and *predictive models* (Baroni, Dinu, & Kruszewski, 2014). On the one hand, count-based methods create word vectors from context statistics. Specifically, words co-occurring with a word (i.e., neighbors) are counted and used to initialize the word's representation. Normally, various transformations are performed on the vectors after that. Probably the best-known count model is Latent Semantic Analysis[11] (LSA). On the other hand, predictive methods aim at predicting a word from its context (i.e., its neighbors). This is why the learned word embeddings are also called *context-predicting vectors*. Mikolov, Yih, and Zweig (2013) showed that neural network approaches generally outperform LSA in capturing syntactic and semantic regularities.

The study by Collobert et al. (2011, Aug), based on Collobert and Weston (2008), predicted distributed representations of words. The authors use a multiplayer neural network depicted in Figure 2.6. Two types of scopes have been elaborated; sentence and window. Because the window context is more relevant for this master's thesis, the focus is set on the window approach. The window of a word is an ngram. That is, words to the left and right of the current word. The objective of the network is to output a higher score for the correct ngram than for a false one. For this purpose, the ranking criterion, as follows, is minimized.

$$\theta \mapsto \sum_{x \epsilon X} \sum_{w \epsilon D} max\{0, 1 - f_\theta(x) + f_\theta(x^{(w)})\}, \tag{2.1}$$

[10] Salton, Wong, and Yang, 1975.

[11] Landauer, Foltz, and Laham, 1998.

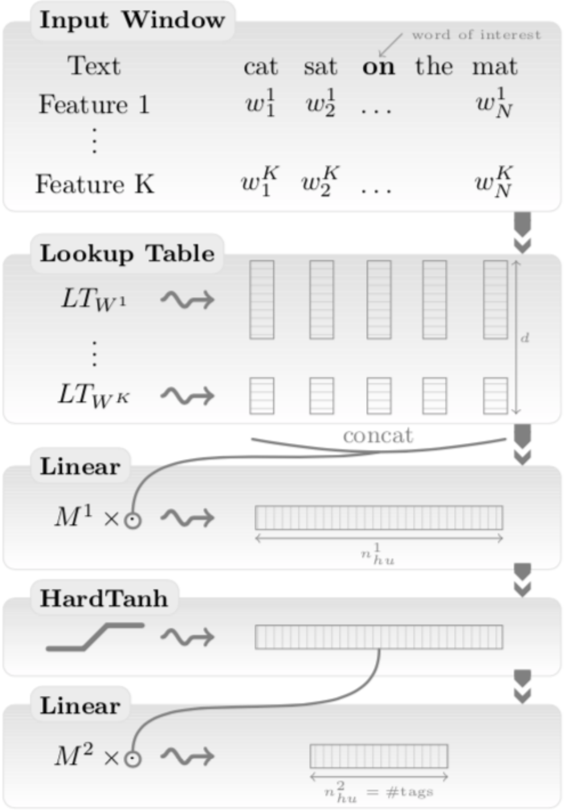


Figure 2.6: The C&W model is based on Bengio, Ducharme, Vincent, and Jauvin (2003, Feb). Before inputted in the neural network, little feature engineering on the input words is performed. In the first layer, words are mapped into real-valued feature vectors using a lookup table. The feature vectors in the lookup table are trained using backpropagation. The second layer incorporates the word's context (i.e., window). The next layers perform standard neural network calculations and transformations (Collobert et al., 2011, Aug).

where X is a set of all possible text windows, D is the dictionary, x is a text window, $x^{(w)}$ is the text window where the central word is replaced by a random other word, $f_\theta(x)$ is the score for a text window. When the objective is minimized, the model learns to assign higher scores to correct windows than to false windows. The results showed that word embeddings successfully learn syntactic and semantic meanings. For example, country names were close in the vector space, as were the gaming consoles "xbox" and "playstation" (Collobert et al., 2011, Aug).

Another important study on word embeddings by Mikolov, Chen, Corrado, and Dean (2013) elaborated continuous representations of words with neural networks. The models provide major advantages over previous studies. For example, efficiency is higher because there is no need for an expensive hidden layer. Additionally, a wider word context can be integrated, making the embeddings more accurate. The presented schemes of language modeling are called Continuous Bag-of-Words (CBOW) and Skip-gram model and are described in more detail in subsection 3.4.1.

All in all, Mikolov, Chen, et al. (2013) were able to compute word vectors of high quality with simple models. In addition, the authors claim that it is feasible to train the models with very large corpora and a huge vocabulary. The computational efficiency and quality make it a prominent choice in current research. The follow-up study (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013) extended the previous Skip-Gram model of word representation learning for phrase vectors. This means the learned vectors can embed idiomatic phrases, e.g., Air Asia. Phrase vectors consist of words that frequently appear with each other, and seldom separately. This causes "this is" not to be detected as a phrase, while "New York Times" is (Mikolov, Sutskever, et al., 2013).

2.11.1 Incorporating Polarity

Most vector space models do only encode the syntactic and semantic context of words and do not integrate polarity information (Maas et al., 2011; Tang, Wei, Yang, et al., 2014). In particular, although words like "wonderful" and "amazing" are probably close in vector space, there is no hint about them being positive. Also, opinion words of opposing sentiments are likely to be placed nearby each other. For example, "good" and "bad" could be close in vector space, because they share similar syntactic regularities in the corpora. Thus, either existing word representation learning algorithms have to be adapted, or new ones need to be created in order to incorporate sentiments effectively.

Maas et al. (2011) adapted the probabilistic topic model of Latent Dirichlet Allocation[12] (LDA) to embed words in vector space. Sentiment information was explicitly incorporated in the model as an objective. To learn the model, ratings were mapped to sentiments for the documents. For comparison, Latent Semantic Analysis[13] (LSA), a well-known count-based vector space model, was applied as well. In order to evaluate the performance, the learned word vectors were directly used as inputs for sentiment classification.

A recent study by Tang, Wei, Yang, et al. (2014) addressed the problem of performing sentiment classification specific to Twitter from word embeddings. To tackle the task, three neural networks were trained to learn so-called *sentiment-specific word embedding* (SSWE). The basis is formed by a tweets dataset consisting of tweets

[12] Blei, Ng, and Jordan, 2003, Jan.

[13] Landauer et al., 1998.

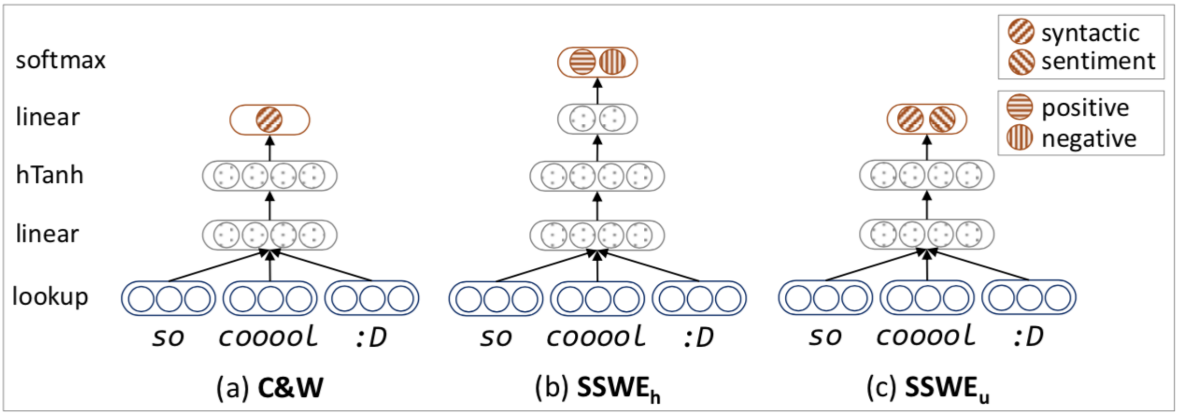


Figure 2.7: The first model visualizes the C&W network by Collobert et al. (2011, Aug). It consists of the lookup, linear, hTanh, and another linear layer (a). The second model shows the SSWE$_h$ network that adds a *softmax* layer on top. This model's output is a tuple of two values, i.e. [1,0] or [0,1] for positive and negative polarity (b). The third model is the final SSWE$_u$ combining both syntactic and sentiment information in the output. Instead of the C&W linear layer, it outputs a two-dimensional vector, one dimension for syntactic and the other for polarity information (c). All three models take ngrams as input (Tang, Wei, Yang, et al., 2014).

labeled as positive or negative by emoticons (for how this dataset is created refer to subsection 2.9.3). The authors argue that although emoticons are noisy, they are effective enough to learn SSWEs. The implemented learning algorithm is extended from the C&W model by Collobert et al. (2011, Aug). The first neural network is called SSWE$_h$. It incorporates polarity by predicting the sentiment distribution from the input. The input is a ngram, i.e. context window of words in a sentence (see Figure 2.7). The output is either [1,0] for positive ngrams or [0,1] for negative ngrams. Because SSWE$_h$ does not allow any decimal values (e.g. [0.8,0.2]), SSWE$_r$ was added. Rather than a strict *softmax* output layer, the second neural network, SSWE$_r$ has a ranking objective function. The third neural network, SSWE$_u$ model combines the principles of C&W, SSWE$_h$, and SSWE$_r$ models. The SSWE$_u$ model predicts a two-dimensional vector for each ngram, one dimension corresponding to the syntactic context and the other representing the sentiment. Finally, the performance for sentiment classification of all three networks was compared. SSWE$_u$ yielded the best results.

2.11.2 Lexicon Generation from Word Representations

Word embeddings with sentiment information can be exploited to construct a sentiment lexicon (Tang, Wei, Qin, Zhou, & Liu, 2014; Tang, Wei, Yang, et al., 2014).

Recent studies by Tang, Wei, Qin, et al. (2014) and Tang, Wei, Yang, et al. (2014) addressed the problem of creating a sentiment lexicon specific to Twitter from word embeddings. Tang, Wei, Yang, et al. (2014) focused on learning *sentiment-specific word embedding* (SSWE) to be used as features for sentiment classification, but also evaluated their suitability for lexicon generation. Specifically, the similarity of close words in the vector space was measured, resulting in an accuracy metric (Tang, Wei, Yang, et al., 2014),

$$Accuracy = \frac{\sum_{i=1}^{L} \sum_{j=1}^{N} f(w_i, n_{ij})}{L \times N} \tag{2.2}$$

where L is a count for the words in the sentiment lexicon, N is the number of neighbors taken into account, w_i is the *i-th* word in the lexicon, n_{ij} is the *j-th* neighbor of w_i, and $f(w_i, n_{ij})$ is

$$f(w,n) = \begin{cases} 1, & \text{if } w \text{ and } n \text{ have the same polarity} \\ 0, & \text{otherwise} \end{cases} \tag{2.3}$$

The more consistent the polarity among words that are near to each other is, the higher is the accuracy (Tang, Wei, Yang, et al., 2014). The performance was compared to other word embedding techniques, and their representations were the most accurate ones. The results show that both accurate syntactic and polarity information are important to achieve semantically correct embeddings including sentiments.

A more recent study by Tang, Wei, Qin, et al. (2014) focuses on building a phrase sentiment lexicon specifically for Twitter from sentiment word embeddings. To learn the *sentiment-specific word embeddings* (SSPE), a combination of Mikolov's skip-gram model, a dictionary, and an integration of polarity values was deployed. Additionally, two different sets of training data (i.e., gold standard) were used. The first set were tweets labeled with polarities by emoticons. The second set was a small set of words from the previously learned embeddings annotated with sentiments (i.e., seed words). The learning approach consists of two parts. First, the representation of words and phrases in vector space is learned with the annotated tweets dataset. Second, the Urban Dictionary was exploited to extend the list of seed sentiment words. The results of both parts are combined to train the phrase-level sentiment classifier. The output of the classifier is a "low-dimensional, dense, and real-valued vector" that maps the sentiment and syntactic context of phrases into a vector space.

More specifically, the skip-gram phrase embedding algorithm by Mikolov, Sutskever, et al. (2013) was enriched by integrating sentiment information in the loss function of a neural network (Tang, Wei, Qin, et al., 2014). A comparison of the models is depicted in Figure 2.8. To put it shortly, the skip-gram model tries to predict neighboring words of w_i from its word (or phrase) embedding e_i. In addition to neighboring words, the adapted model tries to predict the polarity of the sentence containing w_i. For this purpose, the training objective is to learn a sentence representation that predicts the polarity for the sentence s_j. In particular, the objective maximizes the average log probability for the sentence sentiment,

$$\frac{1}{S}\sum_{j=1}^{S} logp(pol_j|se_j), \tag{2.4}$$

where S is the number of sentences in the training data, se_j is the vector representation of s_j, and pol_j is the polarity of s_j (Tang, Wei, Qin, et al., 2014). The syntactic and polarity objectives are combined linearly with

$$\alpha\frac{1}{T}\sum_{i=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} logp(w_{i+j}|e_i) + (1-\alpha)\frac{1}{S}\sum_{j=1}^{S} logp(pol_j|se_j), \tag{2.5}$$

where the first term corresponds to the syntactic phrase embedding learning; T is the number of phrases in the training data, c are the number of neighbors to the left and right (i.e., the context) of the center phrase w_i, e_i is the center phrase's embedding, and α gives tuneable weights to the two parts. The second term matches the aforementioned sentiment objective. This results in mapping phrases with similar sentiments and syntactic meanings nearby each other in vector space (Tang, Wei, Qin, et al., 2014).

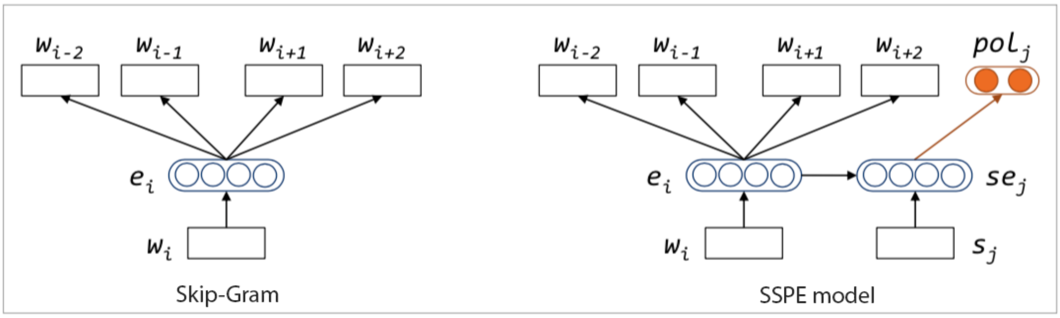


Figure 2.8: Comparison of the skip-gram model and the model for SSPE (Tang, Wei, Qin, Zhou, & Liu, 2014).

The second part of the learning approach was the training of the phrase-level sentiment classifier (Tang, Wei, Qin, et al., 2014). For this purpose, training data had to be collected. To begin with, the most frequent unambiguous 500 words (or phrases) from the previously learned SSPEs were hand-annotated with polarities (positive, negative, or neutral). Then, Urban Dictionary[14] was used to retrieve similar words for each of the seed words. The assumption was that similar words would hold similar sentiments. To expand the seed list, first the k-nearest neighbor (k-NN) algorithm was implemented to build a classifier on the seed words. After the classifier was established, the similar words were inputted, and their polarities were elaborated from the output. This resulted in a dataset of words and phrases annotated with sentiments. Finally, the labeled training data and the learned phrase embeddings were used to train the sentiment classifier with

$$y(w) = softmax(\theta e_j + b), \tag{2.6}$$

where θ and b are the parameters, e_j is the phrase embedding of w_i and $y(w)$ is the sentiment distribution of w_i. When the output is greater than 0.5 the word (or phrase) is labeled positive (negative) in the sentiment lexicon. Tang, Wei, Qin, et al. (2014).evaluated the created lexicon called TS-Lex, and found that it outperforms other sentiment lexicons.

Nevertheless, a closer look reveals some shortcomings of current approaches. First, emoticons, which were used for establishing ground truth, have been shown to be noisy. That is, because they often do not share the same sentiment as the text (Go et al., 2009). Second, learning of the sentiment word (or phrase) vectors required labeled data, and it cannot be assumed enough exists for every domain.

2.12 Summary

To sum up, sentiment analysis is a research field of natural language processing and machine learning concerned with determining the degree of positivity in a text. In specific, given a piece of text, the aim is to analyze whether it includes a positive or negative opinion. Because sentiment can be expressed in various ways and is very subjective to human interpretation, many challenges arise. Additionally, words and phrases used to indicate opinions vary between domains.

In machine learning, methods to perform sentiment analysis can be divided into supervised, unsupervised, semi-supervised, and domain adaptation approaches. Most methods need labeled data, especially supervised ones build upon large amounts of annotated data from the target domain. This data is also called ground truth (or gold standard). As acquiring hand-labeled data is very resource intensive,

[14] https://www.urbandictionary.com

a lot of researchers investigated the problem of automatically generating ground truth. Among others, ratings, emoticons, hashtags, and sentiment lexicons have been exploited.

Sentiment lexicons are one of the most important factors for sentiment analysis. Often, they are applied directly for sentiment classification, or to create a gold standard. In previous research, such lexicons have been established manually, with the use of word relations in dictionaries, and through various techniques using text corpora. One promising way to generate sentiment lexicons of high quality is from word (or phrase) vectors calculated from the target corpora.

The literature review shows that there is still room for improvement in sentiment lexicon generation. Especially, the problem of establishing high-quality domain-specific sentiment lexicons without annotated ground truth data available seems to be unexplored. To our current knowledge, TS-Lex Tang, Wei, Qin, et al. (2014) is the only study that yielded a high-quality lexicon from word (phrase) embeddings, but it required labeled tweets.

3 Method

This chapter explains the system architecture and the individual components of the developed sentiment lexicon generation approach. First, the architecture and its motivational factors are outlined. Then, text preprocessing in general and its major parts, namely text cleaning, tokenization, and phrase detection, are discussed. Next, the representation learning of word and phrase vectors is presented. Afterward, a description of the sentiment learning technique combining these representations and sets of seed words is given. Lastly, the generation of the sentiment lexicon is demonstrated.

3.1 System Architecture

Considering findings of recent studies in sentiment analysis, the system architecture is directed at overcoming some of their limitations. In specific, the reasoning behind this sentiment dictionary generation approach arises from three main aspects. First, there is a lack of high-quality lexicons for various domains. As elaborated in subsection 2.6.2, the sentiment of words heavily depends on the domain at hand. A lot of sentiment lexicon generation approaches build upon the unique characteristics of one domain. As a result, these lexicons suffer from domain limitation, because the words' polarities in the lexicon only correctly match the sentiment of one domain. Our goal is to propose a method that makes it possible to create a sentiment lexicon suitable for any specific domain.

Second, most techniques require ground truth data to learn, but such data is hard to obtain like described in section 2.9. On the one hand, human labeling is very resource intensive, which makes it infeasible to annotate large datasets in ground truth quality. Automated approaches, on the other hand, usually rely on domain specificities. That is, these methods exploit features that are limited to specific domains. For example, some techniques use rating information of reviews to generate a labeled dataset of reviews to be utilized as ground truth. Others create a ground truth dataset by labeling tweets by emoticons they contain. Such methods cannot be used without explicit ratings (or emoticons), making them inappropriate for other domains. To overcome this limitation, the developed approach is intended to work without labeled data.

Third, corpus-based sentiment learning techniques that use word representations have been shown to outperform previous methods (see subsection 2.11.2). Corpus-based techniques operate on the structure of the text to gain information. Word representations that were learned from text data implicitly contain relations between words within a given corpus. By effectively exploiting these findings, it is hoped to learn polarities of words from their similarities.

All in all, the aim is to define a sentiment learning system based on word embeddings that allow generating sentiment lexicons for multiple domains without having ground truth data available. The system architecture defines the proposed approach to create a domain-specific sentiment lexicon, as visualized in Figure 3.1. The method is based on four main components, namely, text preprocessing, representation learning, sentiment learning, and lexicon generation. To start with, before text data can be processed, it needs to be acquired. The obtained corpora is then preprocessed into tokens like words and phrases. During representation learning these tokens are transformed to vectors that inherently reflect their characteristics. Based on the embeddings in vector space and a set of seed words, sentiment labels are learned for each word and phrase. Finally, a sentiment lexicon is generated from the computed positive, negative, and neutral labels.

3.2 Data Acquisition

The pillar of the architecture is to use text corpora from a specific domain. Consequently, the final lexicon will mirror correct word sentiments for that domain. When the lexicon is supposed to be used on Twitter data, the input corpus should consist of tweets. As lexicons that are too general can decrease sentiment classification performance, the corpora should not contain texts from multiple sources. Ultimately, it is crucial to obtain a large amount of text. In general, the more data can be acquired, the better.

3.3 Text Preprocessing

When the data has been collected, the texts need to be preprocessed. That is, the raw text is transformed into a machine-readable format. Sentences are cleaned and split into single word tokens and phrases. In the process of cleaning, punctuation,

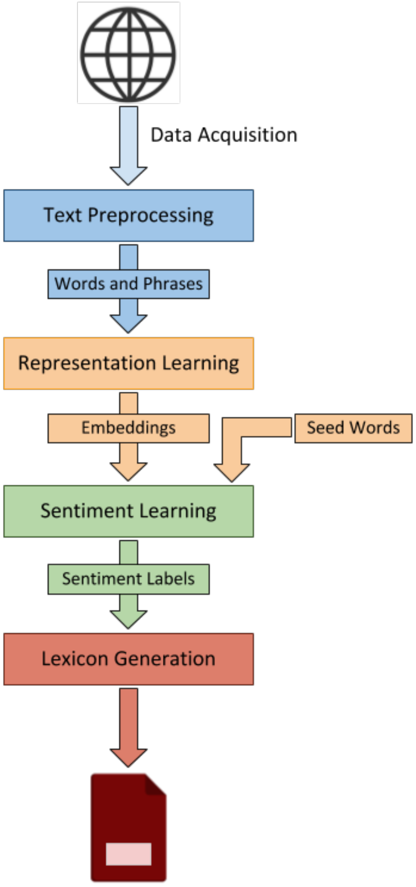


Figure 3.1: System architecture for generating a domain-specific sentiment lexicon.

and stop words (i.e., words that are less relevant to the result) are removed. The aim of preprocessing is to reduce the size of the data for more efficient processing afterward.

Nevertheless, it is crucial to achieve a balance between the reduction of data and the retention of valuable content. Throwing away relevant parts of the data can lead to poor final results. This component outputs tokens like words and phrases of the corpora.

3.3.1 Text Cleaning and Tokenization

Text corpora usually consist of sentences formulated by humans. During text cleaning, the text data is transformed into a format that machines can work with. It includes stripping unnecessary and unwanted parts of the data in order to reduce the data size. However, when data is cleaned extensively, it often goes with sacrificing information that could have been useful. Therefore, in the course of this work, minimal text cleaning was performed. In specific, punctuation symbols such as "., !, ?" were removed. Stopwords were only removed after phrase detection. Stopwords are words that are used very frequently and seldom bare a relevant meaning to the problem at hand. Examples are "the, is, of". A predetermined list of English stopwords offered by the Natural Language Toolkit[1] (NLTK) python library, plus the tokens "rt" and "via", was utilized. The latter is standing for *retweet* and "via" links at the citation of the original author in tweets.

Tokenization refers to splitting sentences into individual words and other types of tokens, for example, emoticons. A regular expressions approach, depicted in Listing 3.1, was chosen to retain smileys such as :), hashtags (terms starting with #), mentions (terms starting with @), URLs, ampersands, emoticons like the heart shape <3, numbers, words connected by a hyphen (-), and HTML tags. The regex was adapted from Marco Bonzanini[2].

Neither lemmatization nor stemming was performed.

3.3.2 Phrase Detection

In order to include common phrases (or collocations) in the sentiment learning process, phrase detection was implemented. Collocations are multi-word expressions

[1] Bird, Klein, and Loper, 2009.

[2] http://marcobonzanini.com/

```
emoticons = r"""
(?:
[:=;] # eyes
[oO\-']?  # optional nose
[D\)\]\(\]/\\OpP]+  # mouth
)|(?:\&lt\;3) # heart <3 emoticon
"""

token_regex = [
emoticons,
r'<[^>]+>',  # HTML tags
r'(?:@[\w_]+)',  # @-mentions
r"(?:\#+[\w_]+[\w\'_\-]*[\w_]+)",  # hash-tags
r'http[s]?://(?:[a-z]|[0-9]|[$-_@.&amp;+]|[!*\(\),]|(?:%[0-9a
   -f][0-9a-f]))+',  # URLs
r'(?:\&[a-zA-Z]+;)',  # &amp;
r'(?:(?:\d+,?)+(?:\.?\d+)?)',  # numbers
r"(?:[a-z][a-z'\-_]+[a-z])",  # words with - and '
r'(?:[\w_]+)',  # other words
r'(?:\S)'  # anything else
]
```

Listing 3.1: Regular expression to detect tokens

- words that often occur together. In natural language processing, phrases are referred to as n-grams. If a phrase consists of two words, it is called "bigram". If a phrase is made up of three words, it is called "trigram", and so on.

The phrase detector is directed at finding bigrams and trigrams occurring at least 100 times in the text corpus. To do so, Gensim's[3] *phrases* model was employed. Since stopwords are not counted, the detected phrases can consist of more than three-word tokens. For example, word sequences like "need a hug", "going to rain" and "cannot wait to see" are valid phrases.

After phrase detection, individual word tokens that form a collocation are replaced with the corresponding phrase. At this point, all sentences are split into word and phrase tokens keeping their original order in the corpus.

3.4 Representation Learning

Representation learning refers to the determination of vector representations of words based on their context, as outlined in section 2.11. Hence, the representation learning component of the system is responsible for calculating vectors of words and phrases extracted during text preprocessing. Semantically similar words (or phrases) will be embedded closer to each other in the vector space than dissimilar ones. The similarity of words is interpreted in terms of common contexts. When words appear together in a sentence, document, or within a particular word window, they are assumed to share a context. A word window of fixed size c refers to c words left and c words right of the center word. Also, words that do not occur within the same context, but are used in an irreplaceable manner are considered similar. The algorithm to create these word embeddings is described in more detail in subsection 3.4.1.

Because the input to the representation learning component comes from a specific domain, the resulting word embeddings reflect the relationships of words and phrases within that data. In particular, when the data was acquired from a certain domain, the vector space represents semantics and meanings of words and phrases specific to the domain. For example, word representations learned from movie review data would precisely embed relations of words relevant to the movie review domain. However, these embeddings would probably not mirror correct word similarities and characteristics for the book review domain.

[3] Řehůřek and Sojka, 2010.

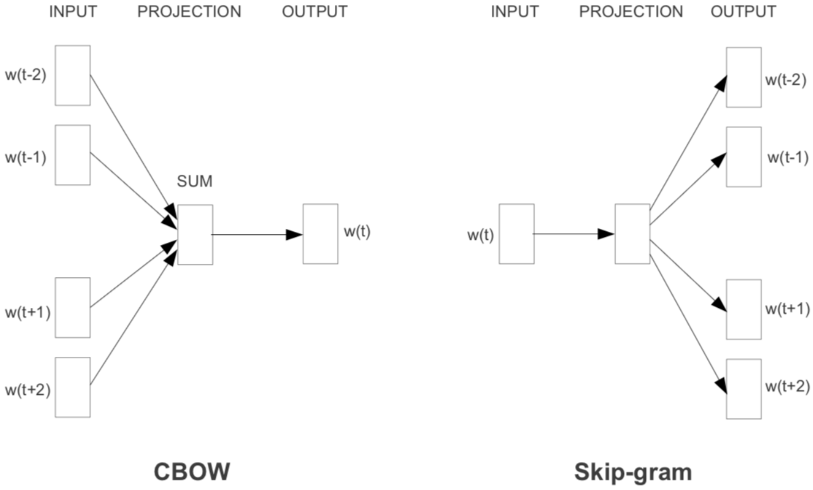


Figure 3.2: The CBOW and Skip-gram architectures: CBOW predicts a word from its context words, while skip-gram predicts the neighbors of a word (Mikolov, Chen, Corrado, & Dean, 2013)

3.4.1 Concept of Word Embeddings

The concept of word embeddings originates from NLP. As already mentioned in section 2.11, a word embedding is a word's continuous vector representation (Mikolov, Chen, et al., 2013; Mikolov, Sutskever, et al., 2013). It is constructed based on the context of a word. Such kinds of representations have been shown to capture semantic and syntactic similarities within the vocabulary very well. Similar words are usually close in vector space. Since around 2013, multiple techniques to create embeddings have been proposed.

Word2vec is a model developed by Mikolov, Chen, et al. (2013) to efficiently learn high dimensional word vectors. It includes two architectures: the Continuous Bag-of-Words (CBOW) and the Skip-gram (SG) model (see Figure 3.2). The CBOW model tries to predict a word from c words before and c words after it. These words are called "context" or "neighbors". The order of words is not taken into account, making it a bag-of-words approach. The objective of the CBOW model is to maximize the conditional probability of a word from given context words.

$$W_{t,c} = \{w_{t-c}, ..., w_{t-1}, w_{t+1}, ..., w_{t+c}\}, \tag{3.1}$$

$$\max p(w_t | W_{t,c}), \tag{3.2}$$

where w_t is the center word, c is the window size, and $p(w_t|W_{t,c})$ is the conditional probability of the word w_t given the context words $W_{t,c}$ (Rong, 2014).

The CBOW model maximizes the average log probabilities of a center word given its context words.

$$\frac{1}{T}\sum_{t=1}^{T} logp(w_t|W_{t,c}), \tag{3.3}$$

where T is the number of words (or phrases) in the training data (Mikolov, Sutskever, et al., 2013).

The Skip-gram (SG) model works the other way around (Mikolov, Sutskever, et al., 2013; Rong, 2014). It tries to predict the context words from the central word. In other words, it aims at finding a word's embedding from the embeddings of neighbors. Thus, the model's objective is to maximize the conditional probability of the context words given the central word,

$$\max p(W_{t,c}|w_t). \tag{3.4}$$

More specifically, it tries to maximize the average log probabilities of the neighbors for a center word.

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} logp(w_{t+j}|w_t), \tag{3.5}$$

where T is the number of words (or phrases) in the training data, c is the window size, w_t is the center word, and $p(w_{t+j}|w_t)$ is the conditional probability of the context word w_{t+j} given the center word. $p(w_{t+j}|w_t)$ is defined as the *hierarchical softmax* that calculates the probability of the neighbor word w_{t+j} given the center word w_t. It is assumed that the context words are disjoint. As for illustration, the traditional *softmax* is outlined

$$p(w_s|w_t) = \frac{exp(v\prime_{w_s}^{\top} v_{w_t})}{\sum_{w=1}^{W} exp(v\prime_{w}^{\top} v_{w_t})}, \tag{3.6}$$

where W is the size of the vocabulary, $v\prime_w$ is the output word representation of w, and v_w is the input word representation of w in a neural network. The *softmax* calculation was approximated by the *hierarchical softmax*, because the latter is more efficient (Mikolov, Sutskever, et al., 2013).

Hierarchical Softmax

The *hierarchical softmax*, a more efficient way to calculate the *softmax*, was introduced by Morin and Bengio (2005). It is based on binary trees, which creates a hierarchical output layer for the neural network. In specific, the layer is composed of a binary tree with the words of the vocabulary as its leaves. The nodes contain the relative probabilities of the child nodes. Therefore, the balanced tree has a depth of $log_2(W)$. In order to calculate the hierarchical softmax of a word, the path from the root to the leaf is followed. As a consequence, instead of having to calculate over W nodes, the probability can be evaluated looking at $log_2(W)$ nodes only. Therefore, it provides a big gain in efficiency over traditional softmax. Mikolov, Sutskever, et al. (2013) use a Huffman tree (Huffman, 1952) for the hierarchical softmax layer.

Negative Sampling

Negative Sampling is an alternative to the hierarchical softmax (Mikolov, Sutskever, et al., 2013). It is also used to improve speed compared to the traditional softmax. Because the softmax operation needs to update every word in the vocabulary while calculating the probability of a center word and its context words, the model is very slow. In contrast, negative sampling reduces the number of words looked at besides the context words. Negative sampling considers the center word, its neighbors, and a small set of words outside the context for each prediction. The problem is formulated as a binary classification task that tries to predict whether a word is a true neighbor (within the context) or a false neighbor (not a context word) of the center word. The remain of the vocabulary can be ignored, and the updates only affect the true and false context words.

3.4.2 Learning Word and Phrase Representations

As described above, embeddings are elaborated based on the context of words (or phrases). That is, how useful the word is to predict other words in a specified context, or how useful context words are in predicting a center word. The first method is called Skip-gram (SG), while the second one is called Continuous Bag-of-Words (CBOW). The word embeddings are "by-products" of training the model.

The SG and CBOW models developed by Mikolov, Sutskever, et al. (2013) to train word and phrase embeddings are available as an open source toolkit called *word2vec*[4].

[4] code.google.com/p/word2vec

Gensim's *word2vec* implementation is a ported version of Google's toolkit that adds additional functionality and performance improvements (Řehůřek & Sojka, 2010).

In this work, the word and phrase representations are learned using Gensim's *word2vec* algorithms. Since the goal was to find out whether or not the idea behind the system architecture works, the emphasis was predominantly set on speed. Therefore, various parameters were chosen to reduce running times. Others were set according to suggestions in literature or by the algorithm's authors. The following list summarizes these parameter choices.

- The minimum frequency of words and phrases that should be considered is set to 100. Words and phrases appearing less often are thrown away.
- The maximum number of training epochs was kept at 30 epochs.
- The used architecture is CBOW. The decision to employ the CBOW training algorithm was made, because it is faster than SG. According to the original authors, the latter tends to produce better results for infrequent words.
- The default sub-sampling (see section 3.4.2) factor of 0.001 is applied. While 1.0 would sample words in relation to their frequency, 0.0 samples words independent of how often they occur.

In addition, hyper-parameter search was performed in order to find optimal settings for the remaining parameters listed as follows.

- The dimensionality of the vector to represent each word (or phrase).
- The maximum distance between a word and its neighboring words.
- The model's training algorithm, namely hierarchical softmax or negative sampling.
- When negative sampling is used, the number of "false neighbors" (noise samples).

The results can be found in section 4.2.

Sub-sampling of Frequent Words

Mikolov, Sutskever, et al. (2013) argue that there are accuracy and speed improvements when the most frequent words in a corpus are down-sampled. It is based on the assumption that the co-occurrence of rare words reveals more information than a rare word in combination with a frequent word like "the". For example, it is more significant to detect the words "Paris" and "France" occurring together than the words "the" and "Paris". Neither the vector representations of rare words nor the of high-frequency ones is influenced a lot by down-sampling. However, major efficiency benefits are gained.

The sub-sampling factor is a threshold that defines what common words are down-sampled. The probability that a frequent random word is skipped is

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}}, \tag{3.7}$$

where $f(w_i)$ is the frequency of a word w_i in the corpora and t is the threshold (Mikolov, Sutskever, et al., 2013).

The output of this component are vector representations (i.e., embeddings) of the words and phrases from the previously acquired and preprocessed data.

3.5 Sentiment Learning

The focus of sentiment learning lies in automatically identifying correct sentiments (i.e., polarities) of words and phrases. That is, a label for every word and phrase within the dataset is determined that indicates its degree of positivity. The label can be one of *negative*, *positive*, or *neutral*, or more fine-grained categories. A score usually is a value within a range of numbers where the lower bound corresponds to very negative sentiment, and the upper bound corresponds to very positive sentiment. In this work, polarity is defined over three classes; negative, positive, and neutral.

The basis for learning sentiments in the developed approach is twofold. On the one hand, the method depends heavily on the learned word and phrase embeddings. The relationship between words (and phrases) in the vector space affects how and which sentiments are learned. On the other hand, so-called *seed words* form the starting point of the sentiment learning process. Seed words are a small set of unambiguous words that are already labeled with correct sentiment values. Since both unannotated and annotated data are utilized, the approach is semi-supervised.

In the beginning, the vectors matching the seed words are assigned the corresponding class labels (negative, positive, or neutral). Then, sentiment learning is performed using a spreading technique, described in more detail in subsection 3.5.1. Principally, the sentiments of labeled vectors are spread to neighboring ones. Thus, vectors receive label information from their neighbors. The labels are spread in an iterative way so that sentiments are recalculated multiple times. The process stops when the vectors reach a global state in which labels do no longer change more than a specified threshold. In other words, every word's vector consistently matches the sentiment of its neighborhood. As a result, this component outputs sentiment labels for words and phrases based on their context. Based on these features, label spreading is performed.

3.5.1 Label Spreading

Label Spreading is a semi-supervised machine learning algorithm that learns labels for unlabeled data within a dataset (Zhou, Bousquet, Lal, Weston, & Schölkopf, 2004). In general, labeled data is more challenging to obtain than unlabeled data (Zhu, 2005). While unsupervised machine learning works with unlabeled data, supervised learning requires labeled data. In order to successfully apply many of the existing machine learning techniques, large amounts of data are needed. Semi-supervised techniques are directed at effectively combining labeled and unlabeled data in machine learning to overcome the shortage of annotated data. The intrinsic information of unlabeled data is exploited together with some available ground truth of the labeled data. Thus, the opportunity to produce high-quality results without having large amounts of ground truth data available arises.

Semi-supervised learning is performed by spreading label information from labeled to unlabeled data points. Consequently, all data points should receive a label. In particular, the problem is represented as a graph $G = (V, E)$, where the vertex set V corresponds to the data points and the edges E connect the points. As semi-supervised learning implies, parts of the original data have class labels assigned while the rest does not contain any label information. Like in traditional k-Nearest-Neighbor (kNN), it is assumed that close points are more similar than far away ones. Thus, points that are close to each other likely share the same label. Also, points in the same "manifold structure" or "cluster" are assumed to belong to the same class. Because the algorithm spreads label information from points to their neighbors, it causes unlabeled points to receive label information from similar points. Spreading is performed iteratively until reaching a stable global state.

The algorithm is performed on a dataset $X = \{x_1, x_2, ..., x_n\} \ \epsilon \mathbb{R}^d$ of n data points with dimension d and on a set of c class labels $L = \{1, ..., c\}$. The labeled data points $x_i \epsilon X$ are pre-annotated by a class label $y_i \epsilon L$ while the others have no class label assigned.

The $n \times c$ matrix Z is defined as

$$Z_{ij} = \begin{cases} 1, & \text{if } y_i = j \\ 0, & \text{otherwise} \end{cases} \tag{3.8}$$

F_t is the $n \times c$ label distribution matrix at time step t with $F_{t=0} = Z$.

In the beginning, an affinity matrix that reflects the similarity over the dataset is constructed. By default, the affinity matrix A is defined using the Gaussian Kernel

$$A_{ij} = exp(\frac{-||x_i - x_j||^2}{2\sigma^2}) \text{ and } A_{ii} = 0. \tag{3.9}$$

In other words, the affinity matrix represents the weights of the edges E between the data points V. The edges connecting points within the same cluster should receive higher values while inter-cluster edges should receive lower ones.

The affinity matrix is normalized by calculating the symmetric normalized Laplacian matrix S with

$$S = D^{-1/2} A D^{-1/2}, \tag{3.10}$$

where D is a diagonal matrix with each diagonal value corresponding to the sum of row i in A of the respective vertex i

$$D_{ii} = \sum_{j=1}^{n} A_{ij}, \tag{3.11}$$

and all other positions set to 0.

Next, label information is spread by calculating the label distribution of the next time step.

$$F_{t+1} = \alpha S F_t + (1 - \alpha) Z, \tag{3.12}$$

where α is the so-called clamping factor that allows weights of the ground truth labels to be changed by a degree α. In specific, at each time step data points receive information from their neighbors with $\alpha S F_t$ and keep parts of their initial label information with $(1 - \alpha)Z$. The label distribution is recalculated until it converges. This means, the values do not change more than a previously defined tolerance.

Finally, for each point x_i the class label y_i is obtained by assigning the class it received most information from. That is,

$$y_i = argmax F^*_{ij}, \tag{3.13}$$

where F^* denotes F at the final time step. In the end, all data points were assigned a label that matches their neighborhood and manifold structure.

For the implementation of label spreading, the model by Pedregosa et al. (2011, Oct) is utilized. The KNN kernel was chosen because it requires much less memory and converges faster than the RBF kernel. The algorithm's configuration for this work was decided based on experiments with various hyper-parameter settings. The configurable parameters are listed as follows.

- The clamping factor (i.e., alpha value) defines how much label information is drawn from neighboring vectors as opposed to the vector's initial label information.
- The neighbor count of the KNN kernel function.
- The default tolerance of 0.001 was kept. The tolerance refers to the threshold to which values change until the vector space is considered stable.

The experiments can be found in section 4.2.

Sentiment Seed Words

Sentiment seed words are words (or tokens) that are annotated with a polarity value. These tokens are used by the label spreading algorithm to diffuse initial sentiment information over the vector space. Thus, seed words and the previously learned embeddings form the basis of sentiment learning.

In the course of this work, two different sets of seed words are considered. The first is a list of sentiment seed tokens manually labeled by Tang, Wei, Qin, et al. (2014). It consists of negative, positive, and neutral words and emoticons. The second list reuses Tang et al.'s neutral seeds but replaces the negative and positive lists by a self-constructed list. The negative and positive seed words were manually selected. The corresponding seed words labeled with sentiment values are depicted in Table 3.1 and Table 3.2 respectively.

The effectiveness of the two sets is compared in section 4.2. Based on these results, one set of seeds is selected and used for lexicon generation.

It was found that the results depend more on the chosen seed words than on the exact configuration of the label spreading algorithm itself.

Tang et al.’s Seed Words		
Negative	Neutral	Positive
:(, :-(, sorry, sad, bad, hate, ill, shit, sick, fuck, hard, tired, :’(, damn, cry, ugh, wrong, poor, sucks, fucking, ass, bored, hurt, crying, stupid, pain, dead, bitch, worst, hell, jealous, worry, scared, hungry, sigh, weird, :((, boring, horrible, stuck, died, ugly, worse, annoying, lonely, broken, lazy, unfortunately, scary, terrible, fucked, awful, badly, worried, cried, depressed, disappointed, stress, ruined, regret, angry, trouble, confused, failed, silly, difficult, nervous, painful, mistake, dirty, ruin, ache, loss, weak, chilling, cancer, annoyed, exhausted, loser, fool, rubbish, ughhh, disadvantage, disagree, disgust, dishonest, evil, fraud, frustrated, garbage, harm, horrible, idiot, impolite, insane, jerk, nasty, offend, panic, desperate, hopeless, pity, regret, reject, ridiculous, sarcasm, scummy, shame, negative	i, you, the, to, my, a, me, and, it, for, so, is, in, of, i’m, on, rt, that, be, this, with, your, was, are, at, all, do, get, now, day, know, go, we, if, one, it’s, see, what, when, time, today, how, from, about, its, he, think, come, they, had, her, there, or, night, been, am, here, you’re, tomorrow, some, she, then, an, them, him, that’s, did, people, say, school, who, has, us, life, way, take, our, week, look, were, man, twitter, everyone, his, after, where, phone, year, something, doing, thing, said, talk, having, world, thought, other, everything, such, hair, since, start, these, myself, house, video, person, hours, both, mention, years, summer, send, car, boy, use, job, saturday, picture, weeks, sister, part, hour, photo, season, weather, news, number, voice, email, pizza, throat, skype, laptop, minute, university	:), :d, love, :-), like, good, lol, happy, thanks, haha, <3, wish, thank, best, yeah, great, yes, nice, better, :p, okay, hahaha, ;), fun, amazing, pretty, cute, =), beautiful, awesome, welcome, cool, smile, excited, luck, lmao, sweet, glad, enjoy, wow, perfect, fine, yay, dear, funny, lovely, goodnight, super, favorite, win, lucky, proud, hopefully, bless, congrats, :’), thankyou, ^^, laugh, wonderful, gorgeous, strong, sexy, trust, appreciate, honestly, sadly, gift, ^_^, important, honey, congratulations, cheers, sweetie, interesting, handsome, adorable, fantastic, pleasure, happiness, smart, exciting, brilliant, healthy, celebrate, sweetheart, honest, famous, success, prefer, interested, yummy, ^.^, joy, award, delicious, fabulous, thankful, talent, greatest, excellent, romantic, agreed, useful, triumph, treasure, thoughtful, suitable, sufficient, sincerely, satisfy, reasonable, positive, powerful, intelligent, inspiring, impressive, honor, fortunate, distinguished, courageous, confident, clever, applaud, admire

Table 3.1: Negative, neutral, and positive seed words by Tang, Wei, Qin, Zhou, and Liu (2014)

Self-constructed Seed Words		
Negative	Neutral	Positive
abuse, anger, angry, anguish, apathetic, apathy, awful, bad, badly, bankrupt, bastard, betrayal, bloody, catastrophe, crime, criminal, crisis, cruel, cruelty, damage, damn, dead, deceit, deceitful, deceive, deception, defect, despair, desperate, destructive, dire, disastrous, disgust, dreadful, dumb, evil, fake, fraud, fraudulent, guilt, guilty, hate, hell, horrible, humiliation, idiot, idiotic, jerk, kill, liar, loser, loss, lost, lunatic, mad, madness, miserable, murderous, nasty, obnoxious, outrage, panic, ridiculous, selfish, selfishness, sinful, terrible, terror, trauma, traumatic, treason, ugly, vile, violent, woeful, worry, worsen	i, you, the, to, my, a, me, and, it, for, so, is, in, of, i'm, on, rt, that, be, this, with, your, was, are, at, all, do, get, now, day, know, go, we, if, one, it's, see, what, when, time, today, how, from, about, its, he, think, come, they, had, her, there, or, night, been, am, here, you're, tomorrow, some, she, then, an, them, him, that's, did, people, say, school, who, has, us, life, way, take, our, week, look, were, man, twitter, everyone, his, after, where, phone, year, something, doing, thing, said, talk, having, world, thought, other, everything, such, hair, since, start, these, myself, house, video, person, hours, both, mention, years, summer, send, car, boy, use, job, saturday, picture, weeks, sister, part, hour, photo, season, weather, news, number, voice, email, pizza, throat, skype, laptop, minute, university	admire, adorable, adore, affection, affectionate, amazing, award, beautify, beloved, best, bliss, blissful, brilliant, celebrate, charm, cheery, delight, ecstatic, enthusiastic, excellence, excellent, excited, excitement, fabulous, faithful, fantastic, genial, glad, glamorous, good, goodness, gracious, grand, grateful, great, happiness, happy, heroic, impress, impressive, joy, joyful, love, lovely, loyal, loyalty, luck, lucky, marvel, marvelous, merry, nice, outstanding, overjoyed, paradise, perfect, pleasant, pleased, pleasure, popular, praise, prosperous, rejoice, sparkle, splendid, successful, super, terrific, triumph, triumphant, visionary, wonderful, woo

Table 3.2: Self-constructed negative, neutral, and positive seed words

3.6 Sentiment Lexicon Generation

Lexicon generation refers to the creation of a sentiment dictionary of words (and phrases). Like described in subsection 2.7.1, a sentiment lexicon consists of words and their polarity values. The input to this component are word and phrase embeddings annotated with sentiment labels. In the process of sentiment learning, every word (and phrase) received a probability for each sentiment value (negative, neutral, and positive). The lexicon is created by saving each word (or phrase) as positive sentiment word if the positive class received the highest probability, and accordingly for negative and neutral words.

The resulting lexicon should mirror domain-specific sentiment because the method is intended to build upon data from a particular domain. That is why it can then be used effectively for sentiment analysis in the corpus' domain. In other words, for whichever domain sentiment analysis needs to be performed, using this approach, it should be possible to generate a sentiment lexicon solely using large amounts of text from that domain.

4 Experiments

In this chapter, the experiments with various hyper-parameter configurations are presented, which tune the performance of the system. Moreover, sentiment classification using the generated lexicons is compared to baseline lexicons and two baseline approaches.

4.1 Datasets

In the following, the datasets used in the course of this work's experiments are described. Parts of the text corpora are applied as inputs to the system, while the remaining data is employed to compare the performance of the generated sentiment lexicons to baselines. The experiments are conducted with three datasets from different domains. Therefore, the effectiveness of the approach is assessed for multiple types of text. To the best of our knowledge, all the obtained corpora are publicly available and free to use for research purposes.

Stanford Twitter Sentiment Corpus

Go et al. (2009) introduced the Stanford Twitter Sentiment Corpus[1]. It consists of a training and test dataset, referred to as STS-Training and STS-Test. The training data consists of 1.6 million tweets balanced between positive and negative texts. The data was labeled automatically using emoticons. A tweet was annotated as negative if it includes negative emoticons such as ":(" or ":-(" and annotated as positive if it includes positive emoticons such as ":)", ":-)", or ":D".

Nevertheless, the labels of the training data are removed in this work for two main reasons. First, sentiments of emoticons can diverge from the meaning of the tweet. Second, the developed approach is based on unlabeled data.

[1] http://help.sentiment140.com/for-students

The test data has been labeled manually by the authors (Go et al., 2009). It contains 177 negative, 139 neutral, and 182 positive tweets. In this work, the STS-Test set is used for evaluation.

Amazon Reviews Corpus

The Amazon Reviews Corpus[2] was initiated for Blitzer et al. (2007). The dataset consists of product reviews of various product types. The data comes in multiple formats; the original raw version, split into negative and positive reviews, already preprocessed, and preprocessed with a balance of positive and negative reviews. At Amazon, every review has a 5-scale star rating associated with it. The rating information determined the negative and positive reviews. That is, reviews with a rating value smaller than three were labeled as negative while the ones with ratings bigger than three were labeled as positive. The rest was discarded. In this work, only the original version is utilized. In specific, the texts are extracted from the XML-based review information. The final dataset consists of more than 1.4 million reviews.

Large Movie Review Dataset

The Large Movie Review Dataset v1.0[3], introduced by Maas et al. (2011), consists of 100,000 IMDB[4] reviews. The data is split into 50,000 labeled and 50,000 unlabeled texts. At IMDB every movie review has a rating between 1 and 10. The labels were assigned by setting reviews with ratings $>= 7$ to positive and reviews with ratings $<= 4$ to negative. That is why, no supposedly neutral reviews are included in this part of the data. The number of positive and negative reviews is balanced (25,000 positive and 25,000 negative texts). In contrast, the unlabeled dataset contains an equal amount of reviews with ratings $<= 5$ and ratings > 5.

Dataset	Number of entries
Sentiment140	1,600,000 tweets
Amazon Product Reviews	1,422,328 review texts
Large Movie Review	100,000 review texts

Table 4.1: Summary of characteristics of the datasets used in this work

[2] http://www.cs.jhu.edu/ mdredze/datasets/sentiment/
[3] http://ai.stanford.edu/ amaas/data/sentiment/
[4] https://www.imdb.com

4.2 Hyper-parameter Tuning

In order to find optimal parameters for the models and algorithms presented in chapter 3, a hyper-parameter search was performed. That is, sentiment lexicons with various parameter combinations were created and tested. The quality of the constructed sentiment lexicons was evaluated by comparing against a gold standard.

4.2.1 Gold Standard

The gold standard lexicon utilized is the Twitter-specific, human-annotated SemEval-2015 English Twitter Lexicon[5] (S. Kiritchenko, Zhu, & Mohammad, 2014; Rosenthal et al., 2015). The sentiment dictionary consists of $1,515$ words and phrases as well as common misspellings, hashtags, negated expressions, emoticons, and internet slang with associated sentiment scores. The positivity is expressed using a range from -1 for very negative sentiment to 1 for very positive sentiment. The scores were created using crowdsourcing and best-worst scaling[6]. An excerpt of the lexicon is provided in Table 4.2.

Entry	Sentiment Score
amazing	0.969
#innovation	0.453
(:	0.406
no matter	0.000
can't sing	-0.219
kill	-0.982

Table 4.2: An excerpt of the SemEval-2015 English Twitter Lexicon (S. Kiritchenko, Zhu, & Mohammad, 2014; Rosenthal et al., 2015)

While the constructed sentiment lexicons include binary classes (i.e., positive, negative, or neutral), the sentiment scores of the SemEval-2015 English Twitter Lexicon are continuous. To be able to compare the generated lexicons to the gold standard, the SemEval lexicon's values were converted to binary scores. In particular, to evenly split the scale the lexicon entries with original scores > 0.35 were assigned a positive label, while entries with scores < -0.35 were assigned a negative label. The range in between is assumed to contain neutral words and expressions. After conversion, the SemEval lexicon consisted of 422 positive, 433 negative, and 660 neutral entries.

[5] http://www.saifmohammad.com/WebPages/SCL.html

[6] Svetlana Kiritchenko and Mohammad, 2016.

4.2.2 Experiment Description

With the use of grid search (Montgomery, 2017) all possible combinations of the hyper-parameters for *word2vec* and label spreading, depicted in Table 4.3 and Table 4.4 respectively, were tested. For this purpose, experiments with each set of hyper-parameters for the developed lexicon generation method were conducted.

The dataset utilized for the experiments with these hyper-parameter settings is the Sentiment140 corpus. After the data was acquired, and the tweets were extracted from the whole corpus, preprocessing was performed on the texts. Then, the words and phrases obtained were used as inputs for the representation learning algorithm. At this point, the first hyper-parameter combination for *word2vec* was applied.

Next, the learned word and phrase embeddings together with one set of seed words formed the basis for sentiment learning. Here, the first hyper-parameter combination for *label spreading* was used. Finally, the lexicon generated from the learned sentiments was compared to a gold standard lexicon. As to perform a fair and unbiased evaluation, seed words were removed from the generated lexicon and the gold standard lexicon.

This workflow was repeated for all parameter combinations of *word2vec* and *label spreading* (Table 4.3 and Table 4.4). In addition, both sets of seeds words, like described in section 3.5.1, were tested consecutively.

Hyper-parameter	Possible Values
Vector Dimension	$50, 100, 200$
Window Size	$4, 5, 6$
Training Algorithm	Hierarchical Softmax, Negative Sampling
Noise Samples (only Neg. Sampling)	$5, 10, 20$

Table 4.3: Potential values for *word2vec* hyper-parameters

Hyper-parameter	Possible Values
Alpha	$0.01, 0.2, 0.5$
Number of Neighbors	$5, 7, 10, 15$
Seed Words	Tang et al.'s, Self-constructed

Table 4.4: Potential values for *label spreading* hyper-parameters

4.2.3 Evaluation Measures

After having transformed the gold standard lexicon to contain multi-class scores, the generated sentiment lexicons can be assessed. Thereby, the metrics of *precision*, *recall*, and *F1 score* were assessed as follows

$$\text{Precision} = \frac{tp}{tp + fp}, \tag{4.1}$$

$$\text{Recall} = \frac{tp}{tp + fn}, \tag{4.2}$$

where tp is the number of true positives (i.e., entries in the generated lexicon of a particular class c that match the class of these entries in the gold standard), fp is the number of false positives (i.e., entries in the generated lexicon of class c that do not match the class in the gold standard), and fn is the number of false negatives (i.e., entries in the generated lexicon not of class c that belong to class c in the gold standard). Hence, $tp + fp$ corresponds to the count of all entries of a particular class c in the generated lexicon, while $tp + fn$ is the total number of entries that actually belong to class c in the gold standard.

The *F1 score* is the harmonic mean of the *precision* and the *recall*:

$$\text{F1 Score} = \frac{2 \cdot P \cdot R}{P + R} \tag{4.3}$$

4.2.4 Results

The hyper-parameters presented in Table 4.5 were empirically searched and achieved the best results, specifically a precision of 0.898, a recall of 0.887, and an F1 score of 0.888.

For the *word2vec* embedding learning, training with a vector dimension of 200 obtained the best classification accuracy in this scenario. As argued in subsection 3.4.2, it was decided to use the CBOW architecture. According to the authors, a window size of around 5 is suggested when using CBOW. During testing, this recommendation outperformed both the usages of 4 and 6 neighbors.

While the hierarchical softmax is said to perform better for infrequent words, negative sampling should work well with frequent words (both described in section 3.4.1). During this evaluation, negative sampling outperformed hierarchical softmax. This

Hyper-parameter	Values
word2vec	
Vector Dimension	200
Window Size	5
Training Algorithm	Negative Sampling
Noise Samples	20
label spreading	
Alpha	0.5
Number of Neighbors	10
Seed Words	Self-constructed

Table 4.5: Best performing hyper-parameter configuration obtained by evaluating the generated sentiment lexicon against the gold standard

is why, rather than the hierarchical softmax, negative sampling is used to train the model. For negative sampling, the default exponent of 0.75 was chosen. The exponent parameter influences the shape of the negative sampling distribution. While multiple different counts of negative examples were tested, choosing 20 "false neighbors" returned the best results.

Also, experiments with lowercasing letters during preprocessing were conducted. Two variants of lowercasing were tested; the first variant consisted of lowercasing all letters except for the first one, the second lowercased all letters. In short, it was found that the latter resulted in higher classification accuracy. This is why all letters are lowercased before further processing happens. Nonetheless, words that have a different meaning depending on whether the first letter is capitalized or not get ignored. For example, "bush" and "Bush" will both be handled as "bush".

As for *label spreading*, the word and phrase embeddings are first filtered by terms present in the gold standard lexicon. Concerning hyper-parameter settings, both alpha values of 0.2 and 0.5 produced satisfying results. However, an alpha of 0.5 slightly outperformed the other values. That is, for each word keeping 50% of the initial label information and retrieving the other 50% from neighbors obtained the best results. Also, a neighbor count of 10 stably delivered good accuracy.

Last but not least, using the self-constructed seed word lists worked better than the seed words employed by Tang, Wei, Qin, et al. (2014).

4.3 Sentiment Lexicon Experiments

The goal of this master's thesis is to find a way to automatically generate domain-specific sentiment lexicons without having large amounts of ground truth data available. To gain insights into how well the developed approach performs, the following experiments were conducted. In particular, the technique is examined by evaluating the created lexicons in sentiment classification tasks.

4.3.1 Experiment Description

In order to evaluate the generated lexicons, the performance is compared against already well-established lexicon baselines across three different domains. To do so, a simple count-based sentiment analysis method is applied. In specific, to obtain a text's polarity the number of negative terms is compared to the number of positive terms in the text using the following formula:

$$s(text) = \frac{\sum_{i=0}^{T_n} l(w_{n,i})}{L_n} + \frac{\sum_{i=0}^{T_p} l(w_{p,i})}{L_p}, \tag{4.4}$$

where T_n is the number of negative terms (words, phrases, or special characters) in the text, $w_{n,i}$ is the i-th negative token, T_p is the count of the positive terms of the text, $w_{p,i}$ is the i-th positive token, $l(w)$ is the token's sentiment score in the lexicon, L_n is the number of negative entries in the lexicon, and L_p is the number of positive entries in the lexicon. In this simple setting, neutral words do not have an influence.

4.3.2 Baselines

The generated lexicons are compared against seven well-established sentiment dictionaries, a random, and a constant baseline.

The following lexicon baselines for sentiment classification are considered: General Inquirer[7] (Stone et al., 1966), HL[8] (Hu & Liu, 2004), MPQA[9] (J. Wiebe et al., 2005),

[7] http://www.wjh.harvard.edu/ inquirer/

[8] https://www.cs.uic.edu/ liub/FBS/sentiment-analysis.html#lexicon

[9] http://mpqa.cs.pitt.edu/lexicons/subj_lexicon/

AFINN[10] (Nielsen, 2011), VADER[11] (Gilbert, 2014), SentiWords[12] (Gatti, Guerini, & Turchi, 2016), and TS-Lex[13] (Tang, Wei, Qin, et al., 2014). The lexicons differ in the words (and phrases) they include and in whether sentiment is scored with continuous or multi-class values. The General Inquirer (GI) is a widely used manually constructed sentiment lexicon that has been developed since 1966. The GI has a lack of words that are common in the social media domain. The sentiment dictionary by Hu and Liu (HL) is a lexicon constructed by employing a bootstrapping technique on WordNet's synonym and antonym relations (Fellbaum, 1998). In contrast to GI, it includes some sentiment expressions present in social texts and product reviews. The Multi-Perspective Question Answering (MPQA) is a lexicon, partly manually labeled partly machine-learned, based on world press articles. The AFINN lexicon has been manually labeled by Finn Årup Nielsen for micro-blogging domains. The VADER sentiment word-bank was established by extending general lexicons (among others, GI) with emoticons, sentiment-related acronyms (such as "LOL" or "WTF"), and slang expressions (such as "nah" or "meh"), and manually labeling these terms. SentiWords is based on SentiWordNet (Baccianella et al., 2010) and was constructed combining both manual prior sentiments and automatic formulas. TS-Lex is a lexicon built from sentiment word embeddings and is specifically tailored for Twitter. It is described in detail in subsection 2.11.2.

In order to make the classification performance based on multi-class (positive, negative, or neutral) comparable, where necessary the sentiment dictionaries were transformed to contain hard scores rather than ranges. In particular, the values in the AFINN and VADER lexicons were categorized with thresholds set at -1 and $+1$ for negative, neutral, and positive entries. The SentiWords entries with strict 0.0 values were marked neutral while the negative and positive values lie below and above. The final size of each lexicon and its class splits are reported in Table 4.6.

As reported by Nielsen (2011), lexicons can have biases toward the positive or negative (or neutral) class. For example, the AFINN word list contains around twice as much negative than positive words. Similar ratios can be observed for the HL and MPQA sentiment dictionaries. Therefore, when calculating sentiment scores of texts, the sentiment scores are normalized by the number of positive and negative entries in the lexicon (see Equation 4.4).

[10] http://www2.imm.dtu.dk/pubdb/views/publication_details.php?id=6010
[11] http://comp.social.gatech.edu/papers/
[12] https://hlt.fbk.eu/technologies/sentiwords
[13] http://ir.hit.edu.cn/ dytang/

Lexicon	# Entries	Values	# Positive	# Negative	# Neutral
GI	8,640	[-1, 0, 1]	1,551	1,919	5,170
HL	6,786	[-1, 1]	2,006	4,780	0
MPQA	6,869	[−1, 0, 1]	2,296	4,149	424
AFINN	2,477	[-5,...,0,...,5]	670	1,289	518
VADER	7,502	−4 → 4	2,461	3,088	1,953
SentiWords	147,305	−1.0 → 1.0	17,254	20,764	109,287
TS-Lex	347,626	[-1, 1]	178,775	168,651	0

Table 4.6: Baseline lexicons statistics where # *entries* corresponds to the total number of entries in the lexicon, *values* refers to the sentiment scores, and # *positive* / # *negative* / # *neutral* are the lexicon entries belonging originally or after conversion of continuous to multi-class scores to the positive, negative, or neutral class.

4.3.3 Evaluation Measures

The same measures as for the hyper-parameter search are applied for the lexicon experiments. That is, sentiment analysis using the baselines and the generated sentiment lexicons are assessed in terms of *precision*, *recall*, *F1 score*, *coverage*, and *normalized F1 score*. The first three metrics are described in detail in subsection 4.2.3.

The coverage refers to how many texts of the data could be classified. That is the number of texts that includes at least one word present in the lexicon. The *normalized F1 score* refers to the *F1 score* normalized by the *coverage*.

4.3.4 Results

This section outlines the experiment results for three domains; micro-blogging, movie review, and product review.

Micro-blogging Domain

In the micro-blogging domain, a lexicon created based on the STS-Training dataset is compared against the baselines. As for the random baseline, a sentiment (positive, negative, or neutral) is assigned to every text randomly. The scores are averaged over ten iterations. As for the constant baseline, it was decided to always predict the *positive* label.

The generated lexicon contains 3,199 positive, 2,554 negative, and 5,866 neutral terms. Multi-class (positive, negative, or neutral) sentiment analysis with each of the

Lexicon / Baseline	Precision	Recall	F1 Score	Coverage	F1 Score (norm.)
Random	0.34	0.34	0.34	100%	0.34
Constant	0.13	0.37	0.20	100%	0.20
GI	0.49	0.44	0.44	89%	0.39
HL	0.65	0.70	0.66	51%	0.34
MPQA	0.58	0.56	0.55	63%	0.35
AFINN	0.67	0.65	0.66	51%	0.33
VADER	0.70	0.70	0.68	61%	0.42
SentiWords	0.50	0.47	0.46	97%	0.44
TS-Lex	0.45	0.59	0.50	89%	0.45
Generated Lexicon	0.54	0.55	0.53	99%	0.52

Table 4.7: Weighted precision, recall, F1 score, coverage, and normalized F1 score obtained from performing three-class sentiment classification on the STS-Test dataset (i.e., micro-blogging domain).

lexicons is performed and the results are summarized based on precision, recall, and F1 score measures. In addition, classification coverage is examined.

As Table 4.7 shows, HL, AFINN, and VADER achieved the best F1 scores. However, their coverage is rather low. While their classification accuracy is comparably well, only slightly more than half of the texts could be analyzed. For the remaining texts, sentiment could not be determined at all. In contrast, GI, SentiWords, TS-Lex, and the generated lexicon report a high classification coverage. Regardless of the coverage, the generated lexicon works more effectively than the baseline methods and the SentiWords, TS-Lex, and GI lexicons. By examining the normalized F1 Score, it can be seen that the generated lexicon outperforms all baselines.

Movie Review Domain

In order to evaluate the developed technique for the movie review domain, the IMDB dataset (see section 4.1) was split into a training and a test set. 1,000 reviews equally balanced between positive and negative reviews are used for testing while the rest serves as input for the developed approach. The lexicon created based on the IMDB training set is compared against the baselines. The random baseline assigns a positive or negative sentiment to every review at random. The scores are averaged over ten iterations. The constant baseline always predicts the *positive* label.

The generated lexicon contains 2,015 positive, 3,039 negative, and 6,005 neutral terms. Since the test set does not include neutral texts, two-class (positive and negative) sentiment analysis is performed. Therefore, neutral words from the lexicons

Lexicon / Baseline	Precision	Recall	F1 Score	Coverage	F1 Score (norm.)
Random	0.50	0.50	0.50	100%	0.50
Constant	0.25	0.50	0.33	100%	0.33
GI	0.60	0.56	0.52	99%	0.52
HL	0.69	0.60	0.55	99%	0.55
MPQA	0.66	0.55	0.47	99%	0.47
AFINN	0.64	0.59	0.55	98%	0.54
VADER	0.64	0.58	0.53	99%	0.52
SentiWords	0.63	0.50	0.34	100%	0.34
TS-Lex	0.69	0.68	0.68	100%	0.68
Generated Lexicon	0.63	0.61	0.59	100%	0.59

Table 4.8: Weighted precision, recall, F1 score, coverage, and normalized F1 score obtained from performing two-class sentiment classification on the IMDB test dataset (i.e., movie review domain).

are disregarded. The results are measured based on precision, recall, and F1 score. In addition, the classification coverage and the normalized F1 score are examined.

Table 4.8 shows a high classification coverage for every tested lexicon. All lexicons but the MPQA and SentiWords outperform the random baseline, and the constant baseline is surpassed by every lexicon. While TS-Lex achieves the best F1 score of 0.59, the generated lexicon obtains the second best score of 0.59.

Product Review Domain

For the product review domain, 10% of the Amazon dataset (see section 4.1) are taken away from the original corpus as test set. In specific, 10% of the positive and 10% of the negative reviews of every product type is taken into account. The reviews are a selection so that the number of positive and negative texts is balanced for the test set. The remaining texts serve as the basis to generate the product review specific lexicon.

Utilizing the same parameters as for the previous domains, the representation learning module's output was nearly 73,000 embeddings. Measures were taken to reduce the number of vectors. In particular, the minimum word and phrase counts were increased to 400.

It is compared against the seven baselines. The generated lexicon contains 9,271 positive, 3,670 negative, and 11,557 neutral terms. Since the test set does not include neutral texts, two-class (positive and negative) sentiment analysis is performed.

Therefore, neutral words from the lexicons are disregarded. The results are measured based on precision, recall, and F1 score. In addition, the classification coverage and the normalized F1 scores are examined.

Lexicon / Baseline	Precision	Recall	F1 Score	Coverage	F1 Score (norm.)
Random	0.50	0.50	0.50	100%	0.50
Constant	0.25	0.50	0.33	100%	0.33
GI	0.58	0.56	0.52	94%	0.49
HL	0.67	0.60	0.55	93%	0.51
MPQA	0.62	0.57	0.52	95%	0.50
AFINN	0.66	0.61	0.59	81%	0.48
VADER	0.66	0.61	0.57	91%	0.52
SentiWords	0.63	0.52	0.38	100%	0.38
TS-Lex	0.61	0.59	0.57	100%	0.57
Generated Lexicon	0.63	0.63	0.63	99%	0.62

Table 4.9: Weighted precision, recall, F1 score, coverage, and normalized F1 score obtained from performing two-class sentiment classification on the Amazon test dataset (i.e., product review domain).

From the experimental evaluation results depicted in Table 4.9, the following observations can be obtained. The sentiment analysis with the SentiWords, GI, and the AFINN lexicons performs worse than the random baseline but better than the constant baseline. Concerning the coverage, SentiWords, TS-Lex, the generated lexicon achieve the best results. Like in the movie review domain, the TS-Lex and the generated lexicon perform best regarding the normalized F1 score. Ultimately, the generated lexicon outperforms TS-Lex with an F1 score of 0.62.

5 Discussion

The purpose of this master's thesis was to find a way to create high-quality sentiment lexicons for arbitrary domains without the availability of labeled data. In this chapter, the main findings of the conducted experiments are summed up.

The experiments compromised hyper-parameter tuning and sentiment classification in different domains using the generated lexicons. The hyper-parameters obtained during the parameter search were employed as settings for all further lexicon creations. A sentiment lexicon was generated for three domains: micro-blogging, movie review, and product review. By using these lexicons in simple count-based sentiment classification tasks, the performance was assessed. In specific, the classification accuracy was compared against already well-established lexicons and other baseline approaches.

The results presented in section 4.3 show that sentiment analysis using the generated lexicons works more effectively than the random and constant baseline methods in all of the three domains. In particular, a higher F1 score was obtained on the test data. Concerning the coverage, the generated lexicon could classify at least 99% of the texts in each domain. In contrast, none of the other lexicons was able to classify more than 97% in the micro-blogging domain. While SentiWords achieved a coverage of 97%, the next best lexicon would classify only 89% of the tweets. With the HL and AFINN lexicons only slightly more than half of the tweets could be analyzed at all. Although TS-Lex contains more than twice as much entries as the SentiWords lexicon and 30 times more than the generated lexicon, its coverage is about 10% lower. This might infer that the content of the lexicon (i.e., which words are included) is more important than the size. Because the developed method builds upon text corpora from the classification domain, the corresponding lexicon contains most of the relevant words.

When looking at the normalized F1 scores, which is the F1 score normalized by the coverage, the generated lexicon outperforms all other lexicons in the micro-blogging domain. Even though the HL, MPQA, AFINN, and VADER reach higher F1 scores than the created lexicon, their low coverage causes them to perform worse in the overall rating.

In addition, the lexicon created for the movie review domain achieves the highest score out of the tested lexicons. In contrast, TS-Lex obtains better results than the generated lexicon in the movie review domain. It is suspected that the reason behind this observation is the relatively small size of the input dataset. While the corpus of the micro-blogging domain includes 1.6 million tweets and the one of the product review domain consists of more than 1.4 million reviews, the movie review data only contains $100,000$ texts. Thus, the quality of the trained word and phrase embeddings might not be as high as for the other two domains, and sentiment labels would not be able to spread as precisely. Regardless, the lexicon, generated from the movie review data, obtained the second best normalized F1 score.

Overall, the experimental evaluation shows that the developed approach has the ability to automatically generate competitive sentiment lexicons from unlabeled data.

6 Conclusion

In this master's thesis, an approach that generates domain-specific sentiment lexicons without the need for ground truth data was developed. The system, based on large sets of unlabeled texts, effectively learns polarities of words and phrases by spreading sentiments from seed words to the rest of the vocabulary. With this sentiment information, a lexicon is created. As the method is intended to build upon corpora from a specific domain, the lexicon's entries mirror polarity values sensitive to that domain.

The developed technique is assessed by an experimental evaluation. Simple count-based sentiment analysis is performed using the lexicon. The performance of the generated lexicon is compared against seven baselines. As to evaluate the domain-specificity, a lexicon is generated and tested for three distinct domains. The results show that lexicons with very high coverage can be generated. Also, the experiments reveal good classification accuracy of the created lexicons in comparison to the baselines. Especially in the micro-blogging and in the product review domain, all baselines were outperformed. Nevertheless, automatically generating very precise sentiment dictionaries for arbitrary domains still is a challenge, and improvements to current approaches are worth to investigate.

6.1 Future Work

In future work, additional lexicons for other domains could be created. Furthermore, to tune the hyper-parameters other gold standards could be utilized. Measures could be taken to find and use the gold standard that best matches each domain. Also, an algorithm to determine optimal seed words could be implemented. Like for the gold standard, different seed words might be used depending on the domain. Moreover, experiments, in order to find a way of not lowercasing letters, could be conducted. At the moment, keeping cases would come at the cost of diminished sentiment accuracy. Consequently, words are lowercased during pre-processing. Nevertheless, valuable information might get lost, which could be prevented by retaining the casing.

The lexicon quality for the product review domain could be improved by using a more homogeneous corpus as input. For example, rather than employing review corpora consisting of mixed product types, an Amazon review dataset could be split into its subdomains such as books and DVDs. Like this, an individual lexicon for every subdomain could be generated.

Moreover, instead of assigning discrete three-class labels to each word (or phrase), continuous scores could be calculated. As the label spreading algorithm outputs a probability for every sentiment class, these probabilities could be utilized to create continuous sentiment scores. Lexicons with continuous labels could lead to sentiment accuracy improvements over discrete values.

Bibliography

Adreevskaia, A., & Bergler, S. (2006). Mining wordnet for a fuzzy sentiment: Sentiment tag extraction from wordnet glosses. In *11th conference of the European chapter of the Association for Computational Linguistics*.

Akkaya, C., Conrad, A., Wiebe, J., & Mihalcea, R. (2010). Amazon mechanical turk for subjectivity word sense disambiguation. In *Proceedings of the NAACL HLT 2010 workshop on creating speech and language data with Amazon's Mechanical Turk* (pp. 195–203). Association for Computational Linguistics.

Annett, M., & Kondrak, G. (2008). A Comparison of Sentiment Analysis Techniques: Polarizing Movie Blogs. *Advances in Artificial Intelligence*, *5032*, 25–35. doi:10.1007/978-3-540-68825-9_3

Aroyo, L., & Welty, C. (2015, March 25). Truth Is a Lie: Crowd Truth and the Seven Myths of Human Annotation. *AI Magazine*, *36*(1), 15. doi:10.1609/aimag.v36i1.2564

Aue, A., & Gamon, M. (2005). Customizing sentiment classifiers to new domains: A case study. In *Proceedings of recent advances in natural language processing (RANLP)* (Vol. 1, *3.1*, pp. 2–1). Citeseer.

Baccianella, S., Esuli, A., & Sebastiani, F. (2010). Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Lrec* (Vol. 10, *2010*, pp. 2200–2204).

Baroni, M., Dinu, G., & Kruszewski, G. (2014). Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 238–247). Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). doi:10.3115/v1/P14-1023

Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, *3*, 1137–1155.

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python: Analyzing text with the natural language toolkit*. " O'Reilly Media, Inc."

Bishop, C. M. (2006). *Pattern recognition and machine learning*. springer.

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

Blitzer, J., Dredze, M., & Pereira, F. (2007). Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th annual meeting of the association of computational linguistics* (pp. 440–447).

Blitzer, J., McDonald, R., & Pereira, F. (2006). Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing - EMNLP '06* (p. 120). The 2006 Conference. doi:10.3115/1610075.1610094

Bradley, M. M., & Lang, P. J. (1999). *Affective norms for English words (ANEW): Instruction manual and affective ratings*. Citeseer.

Bruce, R. F., & Wiebe, J. M. [Janyce M]. (1999). Recognizing subjectivity: A case study in manual tagging. *Natural Language Engineering*, *5*(2), 187–205.

Chklovski, T., & Mihalcea, R. (2003). Exploiting agreement and disagreement of human annotators for word sense disambiguation. In *Proceedings of RANLP 2003*.

Collobert, R., & Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning* (pp. 160–167). ACM. doi:10.1145/1390156.1390177

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

Das, S., & Chen, M. (2001). Yahoo! for Amazon: Extracting market sentiment from stock message boards. In *Proceedings of the Asia Pacific finance association annual conference (APFA)* (Vol. 35, p. 43). Bangkok, Thailand.

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

Davidov, D., Tsur, O., & Rappoport, A. (2010). Enhanced sentiment learning using twitter hashtags and smileys. In *Proceedings of the 23rd international conference on computational linguistics: Posters* (pp. 241–249). Association for Computational Linguistics.

Diakopoulos, N. A., & Shamma, D. A. (2010). Characterizing debate performance via aggregated twitter sentiment. In *Proceedings of the 28th international conference on Human factors in computing systems - CHI '10* (p. 1195). The 28th international conference. doi:10.1145/1753326.1753504

Ding, X., Liu, B., & Yu, P. S. (2008). A holistic lexicon-based approach to opinion mining. In *Proceedings of the international conference on Web search and web data mining - WSDM '08* (p. 231). The international conference. doi:10.1145/1341531.1341561

Fang, X., & Zhan, J. (2015, December). Sentiment analysis using product review data. *Journal of Big Data*, *2*(1). doi:10.1186/s40537-015-0015-2

Feldman, R. (2013, April 1). Techniques and applications for sentiment analysis. *Communications of the ACM*, *56*(4), 82. doi:10.1145/2436256.2436274

Fellbaum, C. (1998). A semantic network of English verbs. *WordNet: An electronic lexical database*, *3*, 153–178.

Gatti, L., Guerini, M., & Turchi, M. (2016, October 1). SentiWords: Deriving a High Precision and High Coverage Lexicon for Sentiment Analysis. *IEEE Transactions on Affective Computing*, *7*(4), 409–421. doi:10.1109/TAFFC.2015.2476456

Gilbert, C. H. E. (2014). Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth International Conference on Weblogs and Social Media (ICWSM-14). Available at (20/04/16) http://comp. social. gatech. edu/papers/icwsm14. vader. hutto. pdf.*

Go, A., Bhayani, R., & Huang, L. (2009). Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, *1*(12).

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

Guthier, B., Ho, K., & Saddik, A. E. (2017, October). Language-independent data set annotation for machine learning-based sentiment analysis. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)* (pp. 2105–2110). 2017 IEEE International Conference on Systems, Man and Cybernetics (SMC). doi:10.1109/SMC.2017.8122930

Hajmohammadi, M. S., Ibrahim, R., Selamat, A., & Fujita, H. (2015, October). Combination of active learning and self-training for cross-lingual sentiment classification with density analysis of unlabelled samples. *Information Sciences*, *317*, 67–77. doi:10.1016/j.ins.2015.04.003

Hatzivassiloglou, V., & McKeown, K. R. (1997). Predicting the semantic orientation of adjectives. In *Proceedings of the 35th annual meeting of the association for computational linguistics and eighth conference of the european chapter of the association for computational linguistics* (pp. 174–181). Association for Computational Linguistics.

Hatzivassiloglou, V., & Wiebe, J. M. [Janyce M.]. (2000). Effects of adjective orientation and gradability on sentence subjectivity. In *Proceedings of the 18th conference on Computational linguistics* - (Vol. 1, pp. 299–305). The 18th conference. doi:10.3115/990820.990864

Hu, M., & Liu, B. (2004). Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 168–177). ACM.

Huffman, D. A. (1952). A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, *40*(9), 1098–1101.

Jurafsky, D. (2000). Speech and language processing: An introduction to natural language processing. *Computational linguistics, and speech recognition*.

Kamps, J., Marx, M., Mokken, R. J., De Rijke, M., et al. (2004). Using WordNet to Measure Semantic Orientations of Adjectives. In *LREC* (Vol. 4, pp. 1115–1118). Citeseer.

Kim, S.-M., & Hovy, E. (2004). Determining the sentiment of opinions. In *Proceedings of the 20th international conference on Computational Linguistics - COLING '04* (1367–es). The 20th international conference. doi:10.3115/1220355.1220555

Kim, S.-M., & Hovy, E. (2006). Automatic identification of pro and con reasons in online reviews. In *Proceedings of the COLING/ACL on Main conference poster sessions* - (pp. 483–490). The COLING/ACL. doi:10.3115/1273073.1273136

Kiritchenko, S. [S.], Zhu, X., & Mohammad, S. M. (2014, August 20). Sentiment Analysis of Short Informal Texts. *Journal of Artificial Intelligence Research*, *50*, 723–762. doi:10.1613/jair.4272

Kiritchenko, S. [Svetlana], & Mohammad, S. M. (2016). Capturing Reliable Fine-Grained Sentiment Associations by Crowdsourcing and Best–Worst Scaling. In *Proceedings of The 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*. San Diego, California.

Kouloumpis, E., Wilson, T., & Moore, J. D. (2011). Twitter sentiment analysis: The good the bad and the omg! *Icwsm*, *11*(538-541), 164.

Kralj Novak, P., Smailović, J., Sluban, B., & Mozetič, I. (2015, December 7). Sentiment of Emojis. *PLOS ONE*, *10*(12), e0144296. doi:10.1371/journal.pone.0144296

Krippendorff, K. (2011). Computing Krippendorff's alpha-reliability.

Ku, L.-W., Lo, Y.-S., & Chen, H.-H. (2007). Test collection selection and gold standard generation for a multiply-annotated opinion corpus. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions - ACL '07* (p. 89). The 45th Annual Meeting of the ACL. doi:10.3115/1557769.1557796

Landauer, T. K., Foltz, P. W., & Laham, D. (1998). An introduction to latent semantic analysis. *Discourse processes*, *25*(2-3), 259–284.

Li, F., Pan, S. J., Jin, O., Yang, Q., & Zhu, X. (2012). Cross-domain co-extraction of sentiment and topic lexicons. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1* (pp. 410–419). Association for Computational Linguistics.

Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, *5*(1), 1–167.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011). Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of* *the association for computational linguistics: Human language technologies-volume 1* (pp. 142–150). Association for Computational Linguistics.

Macmillan-Education. (2018). *Part of speech (noun) definition and synonyms — Macmillan Dictionary*. Retrieved December 8, 2018, from https://www.macmillandictionary.com/dictionary/british/part-of-speech

Majumder, P., Mitra, M., & Chaudhuri, B. (2002). N-gram: A language independent approach to IR and NLP. In *International conference on universal knowledge and language*.

Marcus, M. P., Marcinkiewicz, M. A., & Santorini, B. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational linguistics*, *19*(2), 313–330.

McCallum, A., Nigam, K. et al. (1998). A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization* (Vol. 752, *1*, pp. 41–48). Citeseer.

Michalski, R. S., Carbonell, J. G., & Mitchell, T. M. (2013). *Machine learning: An artificial intelligence approach*. Springer Science & Business Media.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013, January 16). Efficient Estimation of Word Representations in Vector Space. arXiv: 1301.3781 [cs]. Retrieved October 28, 2018, from http://arxiv.org/abs/1301.3781

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013, October 16). Distributed Representations of Words and Phrases and their Compositionality. arXiv: 1310.4546 [cs, stat]. Retrieved November 3, 2018, from http://arxiv.org/abs/1310.4546

Mikolov, T., Yih, W.-t., & Zweig, G. (2013). Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 746–751).

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

Miller, G. A., & Charles, W. G. (1991). Contextual correlates of semantic similarity. *Language and cognitive processes*, *6*(1), 1–28.

Montgomery, D. C. (2017). *Design and analysis of experiments*. John wiley & sons.

Morin, F., & Bengio, Y. (2005). Hierarchical probabilistic neural network language model. In *Aistats* (Vol. 5, pp. 246–252). Citeseer.

Morinaga, S., Yamanishi, K., Tateishi, K., & Fukushima, T. (2002). Mining product reputations on the web. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 341–349). ACM.

Nakov, P., Ritter, A., Rosenthal, S., Sebastiani, F., & Stoyanov, V. (2016). SemEval-2016 task 4: Sentiment analysis in Twitter. In *Proceedings of the 10th international workshop on semantic evaluation (semeval-2016)* (pp. 1–18).

Nielsen, F. Å. (2011, March 15). A new ANEW: Evaluation of a word list for sentiment analysis in microblogs. arXiv: 1103.2903 [cs]. Retrieved October 2, 2018, from http://arxiv.org/abs/1103.2903

Nowak, S., & Rüger, S. (2010). How reliable are annotations via crowdsourcing: A study about inter-annotator agreement for multi-label image annotation. In *Proceedings of the international conference on Multimedia information retrieval - MIR '10* (p. 557). The international conference. doi:10.1145/1743384.1743478

Owsley, S., Sood, S., & Hammond, K. J. (2006). Domain Specific Affective Classification of Documents. In *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs* (pp. 181–183).

Pang, B., & Lee, L. (2004). A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics - ACL '04* (271–es). The 42nd Annual Meeting. doi:10.3115/1218955.1218990

Pang, B., & Lee, L. (2005). Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics - ACL '05* (pp. 115–124). The 43rd Annual Meeting. doi:10.3115/1219840.1219855

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

Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up?: Sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - EMNLP '02* (Vol. 10, pp. 79–86). The ACL-02 conference. doi:10.3115/1118693.1118704

Patodkar, V. N., & I.R, S. (2016, December 30). Twitter as a Corpus for Sentiment Analysis and Opinion Mining. *IJARCCE*, *5*(12), 320–322. doi:10.17148/IJARCCE.2016.51274

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Dubourg, V., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of machine learning research*, *12*, 2825–2830.

Peng, W., & Park, D. H. (2011). Generate adjective sentiment dictionary for social media sentiment analysis using constrained nonnegative matrix factorization. In *Fifth International AAAI Conference on Weblogs and Social Media*.

Pennebaker, J. W., Francis, M. E., & Booth, R. J. (2001). Linguistic inquiry and word count: LIWC 2001. *Mahway: Lawrence Erlbaum Associates*, *71*(2001), 2001.

Petrov, S., Das, D., & McDonald, R. (2011). A universal part-of-speech tagset. *arXiv preprint arXiv:1104.2086*.

Read, J. (2005). Using emoticons to reduce dependency in machine learning techniques for sentiment classification. In *Proceedings of the ACL Student Research*

Workshop on - ACL '05 (p. 43). The ACL Student Research Workshop. doi:10.3115/1628960.1628969

Reagan, A. J., Danforth, C. M., Tivnan, B., Williams, J. R., & Dodds, P. S. (2017, December). Sentiment analysis methods for understanding large-scale texts: A case for using continuum-scored words and word shift graphs. *EPJ Data Science*, *6*(1). doi:10.1140/epjds/s13688-017-0121-9

Řehůřek, R., & Sojka, P. (2010, May). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks* (pp. 45–50). http://is.muni.cz/publication/884893/en. Valletta, Malta. ELRA.

Riloff, E., Wiebe, J., & Wilson, T. (2003). Learning subjective nouns using extraction pattern bootstrapping. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003* - (Vol. 4, pp. 25–32). The seventh conference. doi:10.3115/1119176.1119180

Rong, X. (2014, November 11). Word2vec Parameter Learning Explained. arXiv: 1411.2738 [cs]. Retrieved May 4, 2019, from http://arxiv.org/abs/1411.2738

Rosenthal, S., Nakov, P., Kiritchenko, S., Mohammad, S., Ritter, A., & Stoyanov, V. (2015, June). SemEval-2015 Task 10: Sentiment Analysis in Twitter. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)* (pp. 451–463). Denver, Colorado. Association for Computational Linguistics. Retrieved from http://www.aclweb.org/anthology/S15-2078

Saif, H., Fernandez, M., He, Y., & Alani, H. (2013). Evaluation Datasets for Twitter Sentiment Analysis, 13.

Salton, G., Wong, A., & Yang, C. S. (1975, November 1). A vector space model for automatic indexing. *Communications of the ACM*, *18*(11), 613–620. doi:10.1145/361219.361220

Scott, S., & Matwin, S. (1999). Feature engineering for text classification. In *ICML* (Vol. 99, pp. 379–388). Citeseer.

Shelley, M., & Krippendorff, K. (1984, March). Content Analysis: An Introduction to its Methodology. *Journal of the American Statistical Association*, *79*(385), 240. doi:10.2307/2288384

Snow, R., O'Connor, B., Jurafsky, D., & Ng, A. Y. (2008). Cheap and fast—but is it good?: Evaluating non-expert annotations for natural language tasks. In *Proceedings of the conference on empirical methods in natural language processing* (pp. 254–263). Association for Computational Linguistics.

Stone, P. J., Dunphy, D. C., & Smith, M. S. (1966). The general inquirer: A computer approach to content analysis.

Taboada, M., Brooke, J., Tofiloski, M., Voll, K., & Stede, M. (2011, June). Lexicon-Based Methods for Sentiment Analysis. *Computational Linguistics*, *37*(2), 267–307. doi:10.1162/COLI_a_00049

Tai, Y.-J., & Kao, H.-Y. (2013). Automatic Domain-Specific Sentiment Lexicon Generation with Label Propagation. In *Proceedings of International Conference on Information Integration and Web-based Applications & Services - IIWAS '13* (pp. 53–62). International Conference. doi:10.1145/2539150.2539190

Tang, D., Wei, F., Qin, B., Zhou, M., & Liu, T. (2014). Building large-scale twitter-specific sentiment lexicon: A representation learning approach. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers* (pp. 172–182).

Tang, D., Wei, F., Yang, N., Zhou, M., Liu, T., & Qin, B. (2014). Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 1555–1565). Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). doi:10.3115/v1/P14-1146

Tong, R. M. (2001). An operational system for detecting and tracking opinions in on-line discussion. In *Working Notes of the ACM SIGIR 2001 Workshop on Operational Text Classification* (Vol. 1, *6*).

Turney, P. D. (2002). Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02* (p. 417). The 40th Annual Meeting. doi:10.3115/1073083.1073153

Wang, X., Wei, F., Liu, X., Zhou, M., & Zhang, M. (2011). Topic sentiment analysis in twitter: A graph-based hashtag sentiment classification approach. In *Proceedings of the 20th ACM international conference on Information and knowledge management* (pp. 1031–1040). ACM.

Wiebe, J. (2000). Learning subjective adjectives from corpora. *Aaai/iaai*, *20*(0).

Wiebe, J. M. [Janyce M.], Bruce, R. F., & O'Hara, T. P. (1999). Development and use of a gold-standard data set for subjectivity classifications. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics -* (pp. 246–253). The 37th annual meeting of the Association for Computational Linguistics. doi:10.3115/1034678.1034721

Wiebe, J., & Riloff, E. (2005). Creating subjective and objective sentence classifiers from unannotated texts. In *International conference on intelligent text processing and computational linguistics* (pp. 486–497). Springer.

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

Wilson, T., Wiebe, J., & Hoffmann, P. (2005). Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on human language technology and empirical methods in natural language processing* (pp. 347–354). Association for Computational Linguistics.

Zhou, D., Bousquet, O., Lal, T. N., Weston, J., & Schölkopf, B. (2004). Learning with local and global consistency. In *Advances in neural information processing systems* (pp. 321–328).

Zhu, X. J. (2005). *Semi-supervised learning literature survey*. University of Wisconsin-Madison Department of Computer Sciences.