Genre and Domain Dependencies in Sentiment Analysis

Von der Fakultät für Mathematik und Informatik
der Universität Leipzig
angenommene

DISSERTATION

zur Erlangung des akademischen Grades

DOCTOR rerum naturalium
(Dr. rer. nat.)

im Fachgebiet

Informatik

vorgelegt

von Dipl. Inf. Robert Remus

geboren am 3. Juni 1984 in Erfurt

Die Annahme der Disseration haben empfohlen:

1. Prof. Dr. Alexander Mehler, Goethe-Universität Frankfurt am Main
2. Prof. Dr. Gerhard Heyer, Universität Leipzig

Die Verleihung des akademischen Grades erfolgt
mit Bestehen der Verteidigung am 23. April 2015
mit dem Gesamtprädikat *magna cum laude*.

dedicated to the variety of language

ABSTRACT

Genre and domain influence an author's style of writing and therefore a text's characteristics. Natural language processing is prone to such variations in *textual characteristics*: it is said to be genre and domain dependent.

This thesis investigates *genre and domain dependencies in sentiment analysis*. Its goal is to support the development of robust sentiment analysis approaches that work well and in a predictable manner under different conditions, i. e. for different genres and domains.

Initially, we show that a prototypical approach to sentiment analysis—viz. a supervised machine learning model based on word n-gram features—performs differently on gold standards that originate from differing genres and domains, but performs similarly on gold standards that originate from resembling genres and domains. We show that these gold standards differ in certain textual characteristics, viz. their *domain complexity*. We find a strong linear relation between our approach's accuracy on a particular gold standard and its domain complexity, which we then use to *estimate our approach's accuracy*.

Subsequently, we use certain textual characteristics—viz. *domain complexity*, *domain similarity*, and *readability*—in a variety of applications. Domain complexity and domain similarity measures are used to determine parameter settings in two tasks. Domain complexity guides us in *model selection* for in-domain polarity classification, viz. in decisions regarding word n-gram model order and word n-gram feature selection. Domain complexity and domain similarity guide us in *domain adaptation*. We propose a novel domain adaptation scheme and apply it to cross-domain polarity classification in semi- and unsupervised domain adaptation scenarios. Readability is used for *feature engineering*. We propose to adopt readability gradings, readability indicators as well as word and syntax distributions as features for subjectivity classification.

Moreover, we generalize a framework for *modeling and representing negation* in machine learning-based sentiment analysis. This framework is applied to in-domain and cross-domain polarity classification. We investigate the relation

between implicit and explicit negation modeling, the influence of negation scope detection methods, and the efficiency of the framework in different domains.

Finally, we carry out a *case study* in which we transfer the core methods of our thesis—viz. domain complexity-based accuracy estimation, domain complexity-based model selection, and negation modeling—to a gold standard that originates from a genre and domain hitherto not used in this thesis.

ACKNOWLEDGMENTS

I would like to sincerely thank everyone who supported this thesis in any way. I especially thank my wonderful friends and dear family for their outstanding encouragement.

I am deeply indebted to my supervisor Prof. Dr. Gerhard Heyer, who provided invaluable guidance and advice that went beyond just research. I would like to thank the whole natural language processing group at the University of Leipzig. I am particularly grateful for the many fruitful discussions I had with Ingmar Schuster, Andreas Niekler and Christoph Teichmann. Dr. Stefan Bordag, Dr. Christian Hänig and Dr. Mathias Bank shaped my understanding of natural language processing and sentiment analysis, and inspired part of my work. Thanks go also to Prof. Dr. Uwe Quasthoff, who always provided helpful feedback, extraordinary examples and a different perspective. Special thanks go to Melanie Kunoth for proofreading an earlier draft of this thesis, and to Renate Schildt.

I gratefully acknowledge the financial support I received from Sächsischen Staatsministerium für Wissenschaft und Kunst via a Sächsisches Landesstipendium.

Judith, without you and your love I never would have finished this thesis.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

CRF	Conditional Random Field
CO	cut off
CV	cross validation
DA	domain adaptation
DRI	Devereux Readability Index
DT	Decision Tree
EA	EasyAdapt
EL	Easy Listening Formula
FI	Fog Index
FKS	Flesh-Kincaid Score
FS	feature selection
IG	Information Gain
IS	Instance Selection
JS	Jensen-Shannon
KL	Kullback-Leibler
LR	linear regression
LogReg	logistic regression
MAP	maximum a posteriori
ML	Machine Learning
MRSE	mean residual standard error
NB	Naïve Bayes
NLP	Natural Language Processing
NP	noun phrase
NREI	New Reading Ease Index
NM	negation modeling
NS	negation scope
NSD	negation scope detection
POS	Part of Speech
PTBTS	Penn Treebank Tag Set
RBF	radial basis function
SA	Sentiment Analysis
SCL	Structural Correspondence Learning
SFA	Spectral Feature Alignment
SMS	Short Message Service
STTS	Stuttgart Tübingen Tag Set
SVM	Support Vector Machine
VP	verb phrase
WSD	Word Sense Disambiguation

LIST OF DATASETS

DSRC	Darmstadt Service Review Corpus
MDSD v2.0	Multi-Domain Sentiment Dataset v2.0
MPQA v2.0	Multi-Perspective Question Answering Corpus v2.0
PD v2.0	Polarity Dataset v2.0
RND	Ratingz Network Dataset
SD v1.0	Subjectivity Dataset v1.0
SE-2007-T14D	SemEval-2007 Task 14 Dataset
SE-2013-T2BD	SemEval-2013 Task 2B Dataset
SPD v1.0	Sentence Polarity Dataset v1.0
T-MDSD	Twitter Multi-Domain Sentiment Dataset

LIST OF NOTATIONS

$p(a)$	Probability of a
$p(a \mid b)$	Conditional probablity of a given b
$\mathbf{x}$	Vector $\mathbf{x}$
Δ	Difference

1 INTRODUCTION

> The advantage of the emotions is that they lead us astray,
> and the advantage of science is that it is not emotional.
> — *Oscar Wilde,*
> *The Picture of Dorian Gray*

With the advent of the world wide web our information gathering behavior drastically changed. Instead of asking our friends and family, we query search engines to fulfill our information needs. Moreover, people actively participate in the creation and distribution of new content, e. g. texts, pictures, and videos. In social media, e. g. FaceBook[1] and Twitter[2], people not only share their daily life, but they also express their views on basically everything—ranging from politics to cultural events to their latest purchases—and discuss the views of others on the same issues. The sheer amount of such "opinionated" content requires technology that enables opinion-aware applications. This technology is subsumed under the term *Sentiment Analysis (SA)*.

Sentiment analysis

1.1 MOTIVATION

SA has manifold applications in human-computer interaction (e. g. Hudlicka, 2003), intelligence, market research (e. g. Qiu et al., 2010), and end-user products (e. g. Yu and Hatzivassiloglou, 2003; Seki et al., 2005). In its manifold applications SA faces a wide variety of challenges, many of which are inherently bound to the data to be analyzed: natural language text.

Because just like Natural Language Processing (NLP) in general, SA—an NLP task—is dependent on what we subsume under *contextual parameters*. Such contextual parameters are e. g. the point of time at which something is expressed—the zeitgeist—or the author's social, cultural, and educational background (see Lahiri et al., 2011) against which something is expressed. Contextual parameters may influ-

Contextual parameters

[1] http://www.facebook.com
[2] http://twitter.com

ence a text's characteristics, e. g. its vocabulary or its structure, which in turn may influence the level of accuracy that certain NLP—and SA—techniques reach. While many contextual parameters, e. g. the point of time, certainly do influence an author's style of writing and therefore a text's characteristics, their influence is rather subtle[3]. Not subtle is the influence of two other contextual parameters: the genre and the domain in which something is expressed. Their influence on a text's characteristics poses a central challenge to NLP in general: many NLP techniques are said to be genre and domain dependent[4].

Such genre and domain dependencies are particularly pronounced in SA: Sentiment—polarity and subjectivity—is expressed both via vocabulary and structure (see Wiebe et al., 2004), both of which are influenced by genre and domain. Furthermore, sentiment is expressed differently in different genres[5] and different domains[6]. These considerations lead to the *core hypothesis* of our thesis:

Core hypothesis

> "SA is genre and domain dependent".

Consequently, our thesis focuses on a special case of particularly pronounced genre and domain dependencies in NLP: *genre and domain dependencies in SA*.

[3] According to Cook and Stevenson (2012) a word's connotation may change over time: they study amelioration and pejoration. Furthermore, according to Gulordava and Baroni (2011), out of 10,000 randomly sampled words from Google n-grams corpus, 1.6% undergo a semantic change when comparing the 1960s to the 1990s.

[4] Sekine (1997) showed that syntax parsing is domain dependent; Escudero et al. (2000) showed that Word Sense Disambiguation (WSD) is domain dependent. Note that both Sekine (1997) and Escudero et al. (2000)—actually most NLP research—uses domain in a much broader sense than we do that often also comprises the genre.

[5] Imagine for example how an opinion regarding a political decision is expressed in a tweet—which is limited to 140 characters—vs. how it is expressed in a newspaper commentary. While the former underlies no formal editing process and therefore is expressed rather spontaneously and direct, the latter usually underlies a rigorous editing process and therefore is expressed "with thought".

[6] Turney (2002) was among the very first to notice domain dependence in the semantic orientation of adjectives like "unpredictable": when "unpredictable" describes a movie plot, it is positively connotated, while it is negatively connotated when it describes the steering of a car.

1.2 OUTLINE

Our thesis is structured as follows. In this introduction, we yet outline our goals (see Section 1.3), clarify our terminology (see Section 1.4), and describe challenges in SA apart from genre and domain dependencies (see Section 1.5).

Subsequently, Part I of our thesis provides all necessary background information: In Chapter 2 we describe the gold standard datasets we use throughout this thesis and how we preprocess natural language text. In Chapter 3 we describe measures to capture certain language characteristics: genre and domain specifics. In Chapter 4 we describe Machine Learning (ML) algorithms used in this thesis as well as methods on how to evaluate ML models.

Part II forms the core of our thesis: In Chapter 5 we show that SA is genre and domain dependent. In Chapter 6 we leverage genre and domain characteristics for model selection and feature engineering in different SA subtasks. In Chapter 7 we describe how to model and to represent negation in ML-based SA.

Part III applies the knowledge we gained in Part II: In Chapter 8 we carry out a case study on previously unseen data. In Chapter 9 we summarize our findings and conclude.

1.3 GOALS

To (i) add to the understanding of similarities and dissimilarities between genres and domains and to (ii) support the development of robust SA approaches that work well and in a predictable manner under different conditions—i. e. for different genres and domains—are the main goals of our thesis. Therefore, we work within the paradigm of *language engineering*[7]. We like to sharpen the consciousness that "off-the-shelf" SA[8]—just like NLP—approaches are prone to genre and domain specifics. Therefore, we ex-

Language engineering

[7] "Language Engineering is the discipline or act of engineering software systems that perform tasks involving processing human language. Both the construction process and its outputs are measurable and predictable." (see Cunningham, 1999)

[8] "Off-the-shelf" SA can be found in e. g. R text mining module (`http://cran.r-project.org/web/packages/tm/index.html`), Python Natural Language Toolkit (`http://www.nltk.org/`), and Java LingPipe (`http://alias-i.com/lingpipe/`).

pect them to perform differently for data that originates from differing genres and domains. Throughout our thesis, our desiderata to reach these goals are:

- We focus on a supervised, ML-based SA approach that is purely data-driven and does not incorporate external knowledge, e. g. prior polarity dictionaries like SentiWordNet (see Esuli and Sebastiani, 2006). This allows us to apply the same SA approach to all kinds of genres and domains without the need for adapting the external knowledge or the SA approach itself.
- We prefer well studied, widely used ML techniques. This allows us to focus on the understanding of the relation between genre and domain characteristics and our SA approach rather than on ML technique specifics.
- We introduce as little free parameters in our experiments as possible. This avoids that parameter tuning masks the effects that genre and domain characteristics have on our SA approach.

1.4 TERMINOLOGY

In this section we clarify our terminology related to SA (see Section 1.4.1) as well as our notions of genre and domain (see Section 1.4.2).

1.4.1 *Sentiment Analysis*

According to Pang and Lee (2008)'s comprehensive study, Nasukawa and Yi (2003)'s article was the first to use the term SA. It says:

> "The essential issue in sentiment analysis is to identify how sentiments are expressed in [natural language] texts and whether the expressions indicate positive (favorable) or negative (unfavorable) opinions toward the subject."

Nasukawa and Yi (2003) go on and expatiate that SA involves several subtasks, i. e. identifying sentiment expressions, determining polarity and strength of these expressions as well as their relation to the subject.

Sentiments

Sentiments themselves express our *feelings* and *emotions*. From Damasio (2004)'s neurobiological perspective

Emotions

> "emotions are bioregulatory reactions that aim at promoting, directly or indirectly, the sort of physiological states that secure not just survival but survival in the range that we, conscious and thinking creatures, identify with well-being."

whereas

Feelings

> "feelings are the mental representation of the physiologic changes that occur during an emotion."

In a broader sense Damasio (2004) defines feeling as the "perception of an emotional state (...)". Consequently, SA aims for identifying what we typically describe with terms like FEAR, ANGER, SADNESS, and JOY—mental representations of our emotions, i. e. our feelings.

SA subtasks

To achieve this aim, several *SA subtasks* need to fulfilled, all of which are in concordance with Nasukawa and Yi (2003)'s initial thoughts: identification and classification of subjectivity, of polarity and of emotions, as well as extraction of opinion expressions, opinion holders, and opinion targets. We now define these terms.

Subjectivity

Subjectivity classification

Subjectivity encompasses "aspects of language used to express opinions, evaluations, and speculations" (see Wiebe et al., 2004). *Subjectivity classification* of e. g. a document or a sentence tries to discriminate between two categories: subjective and objective. An utterance as in Example (1) clearly bears an opinion, i. e. it is subjective, whereas an utterance as in Example (2) clearly does not, i. e. it is objective[9].

(1) In the end, though, it is only mildly amusing when it could have been so much more.

(2) The movie takes place in mexico, 2002.

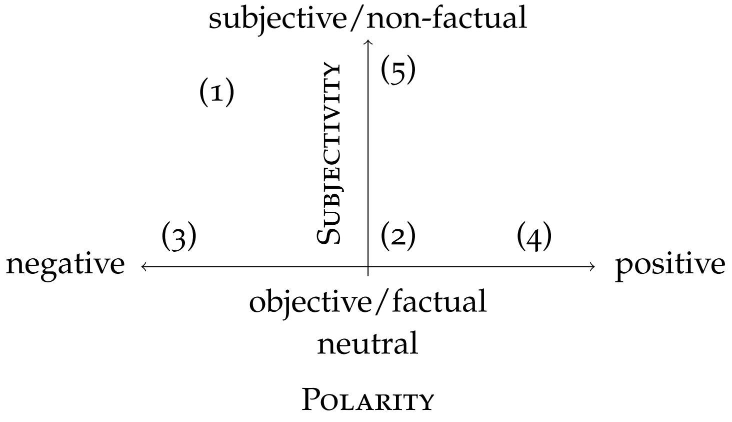
Polarity

Polarity classification

Polarity or semantic orientation of e. g. a word or a phrase refers to its evaluative characteristics (see Hatzivassiloglou and McKeown, 1997), i. e. whether it has positive or negative associations; or, put in Nasukawa and Yi (2003)'s words, whether it is favorable or unfavorable. Words with positive polarity are e. g. "famous" or "remarkable". Words with negative polarity are e. g. "contagious" or "ignorant" (see Hatzivassiloglou and McKeown, 1997). *Polarity classi-*

[9] Example (1) and (2) are adapted from Pang and Lee (2004)'s SD v1.0. Examples (3), (4), (5), (6) and (7) are adapted from Wiebe et al. (2003)'s MPQA v2.0 corpus. Example (8) is taken from Toprak et al. (2010)'s DSRC (see Chapter 2).

Figure 1.: The relation between subjectivity and polarity. Examples (1), (2), (3), (4), and (5) are plotted for illustrative purposes only.



fication tries to discriminate between either two or three categories: positive and negative, or positive, negative, and neutral, respectively. When polarity is assumed to be orthogonal to subjectivity (see Figure 1), it is crucial to distinguish between *factual* and *non-factual polarity* (see Balahur and Steinberger, 2009; Balahur et al., 2010). On the one hand, objective utterances may bear polarity. Both Example (3) and (4) report facts:

Factual vs. non-factual polarity

(3) As a result, at least four Palestinians were reportedly killed and more than 30 wounded.

(4) Last Wednesday the Merval index posted a sharp 5.94 percent rise when the stock exchange reopened.

But the fact that four Palestinians were killed and 30 others were wounded is certainly negative news; the fact that the Merval index rose by almost 6% is positive news. Both utterances bear factual polarity. In contrast, Example (1) is subjective and negative, i. e. it bears non-factual polarity. On the other hand, subjective utterances must not necessarily bear polarity. Example (5) is subjective, but neither positive or negative—it is neutral. Figure 1 illustrates this.

(5) I think that the antiglobalists do not realize what has already been done.

Opinion expression

An *opinion expression*—as its name suggests—expresses someone's opinion towards something. In Example (6) "worst mistake" is an opinion expression. The extraction of opinion expressions has been studied extensively (e. g. Breck et al., 2007; Johansson and Moschitti, 2013).

Opinion holder

An *opinion holder* is the source of an opinion (see Wiegand and Klakow, 2010). In Example (6) "former Minister Roque Fernandez" is the holder of the opinion that it was "the worst mistake in the history of the Argentine economy". In Example (7) "The U.S. commanders" are the holders of the opinion that the prisoners are "unlawful combatants".

(6) According to former Minister Roque Fernandez, it was "the worst mistake in the history of the Argentine economy".

(7) The U.S. commanders consider the prisoners to be "unlawful combatants" as opposed to prisoners of war, a distinction that allows the U.S. to provide different treatment under international law.

Opinion holder extraction

Opinion holder extraction is an information extraction task that tries to identify the opinion holder(s) in text (e. g. Choi et al., 2005, 2006; Kim and Hovy, 2005, 2006; Wiegand and Klakow, 2010).

Opinion target

An *opinion target* is what the opinion is about. Example (8) contains two opinion targets: "features" that are not "earth-shattering" and "eCircles" that provides "a great place to keep in touch".

(8) While none of the features are earth-shattering, eCircles does provide a great place to keep in touch.

Opinion target extraction

Opinion target extraction is an information extraction task that tries to identify the opinion target(s) in text (e. g. Kim and Hovy, 2006; Zhuang et al., 2006; Bloom et al., 2007; Kessler and Nicolov, 2009; Jakob and Gurevych, 2010).

1.4.2 *Genre and Domain*

Language variety

Unfortunately, there is neither a clear notion of nor a clear distinction between the terms and concepts of genre and domain as well as text type, style, register etc. in research of (computational) linguistics (see Lee, 2001a) on language variety. A *language variety* "refer[s] to any system of linguistic expression whose use is governed by situational variables" (see Crystal, 2008, p. 509). We resort to the terms of genre and domain and understand them as follows:

Genre

A *genre* is an identifiable text category (see Crystal, 2008, p. 210) based on external, non-linguistic criteria such as intended audience, purpose, and activity type (see Lee, 2001a) as well as textual structure, form of argumentation, and level of formality (see Crystal, 2008, p. 210). Genres are assigned based on their use, rather than their form. They are recognized as legitimate groupings of texts within a speech community (see Swales, 1990). Genre examples are news articles, letters, novels, recipes, fora posts, weblog posts etc. A genre may have sub-genres, e. g. specific types of news articles or novels. It is noteworthy that these genres "can have (...) different levels of generality" (see Lee, 2001a): some genres are very broad, in that texts from these genres vary considerably, while other genres are rather narrow, in that texts from these genres do not vary much.

Text Type

Style

Register

Field

In contrast to genres, *text types* are based on internal, linguistic criteria (see Biber, 1988). Registers, or sublanguages, and styles, are also to be distinguished from genres. *Style* usually refers to the individual use of language (see Lee, 2001a). A certain *register* refers to a variety of language used in a certain social situation, e. g. scientific or informal English (see Crystal, 2008, p. 409). Registers may further be classified according to their *field*, i. e. their subject-matter (see Crystal, 2008, p. 409).

Domain

Similarly, a *domain* is a genre attribute that describes the subject area that an instantiation of a certain genre deals with (see Steen, 1999; Lee, 2001a). E. g., a text from the genre NEWSPAPER ARTICLE may be belong to the domain FINANCE. Other domains may be art, science, religion, politics, sports, economy, technology etc. A domain may have sub-domains, e. g. IMPRESSIONISM is a sub-domain of ART, and TABLE TENNIS is a sub-domain of SPORTS. Note that this notion of domain deviates from Crystal (2008, p. 155)'s notion, who considers a domain as an "area of experience". This deviation is rooted in the usual use of the term domain in the NLP and SA research communities.

Medium

Channel

In this thesis we focus on writing, which is the secondary *medium* as opposed to speech, which is the primary medium (see Crystal, 2008, p. 300). Note that medium is different from *channel*, "which refers to the physical means whereby a (spoken or written) message is transmitted, such as a wire, air, light (...)." (see Crystal, 2008, p. 300).

1.5 CHALLENGES

In this section we describe challenges in the research field of SA apart from genre and domain dependencies, some of which are currently addressed in the literature but have not been solved satisfactorily yet, some of which remain open to date. We subdivide these challenges into those inherent to natural language (see Section 1.5.1) and those inherent to technology used to process natural language (see Section 1.5.2).

1.5.1 *Language-inherent*

Language-inherent challenges in SA are, among others:

IRONY AND SARCASM. *Irony* is an intricate rhetorical device that expresses "one's meaning by using language that normally signifies the opposite, typically for humorous or emphatic effect."[10]. Example (9) is an ironic expression:

(9) Don't go overboard with the gratitude.[1]

Sarcasm uses "irony to mock or convey contempt"[11]. Example (10) is a sarcastic reference to a book, Example (11) is a sarcastic description of an e-book reader, both taken from Tsur et al. (2010):

(10) [I] love the cover.

(11) Great idea, now try again with a real product development team.

Identification of irony (e. g. Reyes and Rosso, 2012, 2013) and sarcasm (e. g. Tsur et al., 2010; Lukin and Walker, 2013) is one of the most apparent challenges in SA, and—genre and domain dependencies aside—also one of the biggest: irony and sarcasm are inherently ambiguous (see Tsur et al., 2010) and highly variable in form (see Lukin and Walker, 2013). Being

[10] http://oxforddictionaries.com/definition/english/irony (accessed June 28, 2013)

[11] http://oxforddictionaries.com/definition/english/sarcasm (accessed June 28, 2013)

able to identify irony and sarcasm means being able to discover that there is a discrepancy between what is said and what is meant. This may be the difference between classifying Example (10) as having positive or negative polarity.

HUMOR. Closely related to irony and sarcasm is *humor*, "the quality of being amusing or comic, especially as expressed in literature or speech"[12]. Example (12) is a humorous one-liner taken from Mihalcea and Strapparava (2005):

(12) Take my advice; I don't use it anyway.

The recognition of humor in writing (e. g. Mihalcea and Strapparava, 2005) is similarly challenging as the identification of irony and sarcasm: humor may be subtle and its understanding requires world knowledge. Although we are far from fully understanding humor, separating humorous from non-humorous text can be done on a high level of accuracy (see Mihalcea and Pulman, 2007).

POINT OF VIEW. The interpretation of SA results depends on one's *point of view* (see Scholz and Conrad, 2012) or one's *perspective*. What may be favorable to one party, might be unfavorable to another and vice versa. Example (13) is of positive polarity from SPD's point of view, but of negative polarity from any opposing party's point of view.

(13) The SPD man Klaus Wowereit is elected governing major of Berlin.

The identification of an entity's viewpoint requires contextual information that is often not available, rendering it impossible to automatically decide whether something is favorable or unfavorable to a certain party.

Irony and sarcasm, humor, and viewpoints are language-inherent challenges that are specific to SA. Other language-inherent challenges that are non-specific to SA but present general challenges for many NLP techniques are, among others:

[12] `http://oxforddictionaries.com/definition/english/humour` (accessed June 28, 2013)

AMBIGUITY. Natural language is highly *ambiguous*, i. e. the same utterance may have different meanings, depending on who expressed the utterance, to whom the utterance was expressed, where and when the utterance was expressed etc. Ambiguities may occur on different language levels, e. g. on sense-level or on sentence-level. Resolving such ambiguities, e. g. in WSD (e. g. Brown et al., 1991; Yarowsky, 1995), is a hard NLP task on its own.

EXPRESSIVE POWER OF HUMAN LANGUAGE. The human capability to compose and comprehend a virtually infinite number of syntactically and semantically valid sentences (see Hauser et al., 2002; Pullum and Scholz, 2010) leads to a vast number of ways to express sentiment. This impedes a compact representation of language—sentiment expressions—and hinders generalization.

CREATIVE LANGUAGE USE. Humans are *creative* in their language use. E. g., they emphasize that they are very hungry by writing that they are "huuungry" or they derivate the verb "to google" from searching the web via Google's search engine[13]. Again, this impedes a compact representation of language and hinders generalization.

NOISE. Closely related to creative language use and the expressive power of human language is the presence of *noise*, especially in texts without editorial control, e. g. in posts to social media, webfora, or blogs. Noise includes orthographic flaws, creative language use, uncommon abbreviations etc. Again, this impedes a compact representation of language and hinders generalization.

MULTILINGUALITY. Languages may differ in all their linguistic characteristics, e. g. in the alphabet they use, in their morphology, in their syntax etc. Generally, it is not straightforward to transfer an SA approach from one language to another (e. g. Mihalcea et al., 2007; Balahur and Turchi, 2012).

COMMON SENSE. *World knowledge* and *common sense reasoning* (e. g. Cambria et al., 2012a,b) are often neces-

[13] `http://www.google.com`

sary to understand an ongoing discourse or universally accepted facts, e. g. that a beer is best served cold, while a pizza is best served hot. Although SA would certainly benefit from world knowledge and common sense reasoning, both are non-trivial to implement.

1.5.2 *Technology-inherent*

Technology-inherent challenges for SA and NLP in general are, among others:

SUPERVISION. ML-based NLP techniques often rely on extensive *supervision* (see Section 4.1.1) in the form of manually annotated and thus costly training data. SA methods that are unsupervised (e. g. Lin and He, 2009; Wang and Liu, 2011; Scheible and Schütze, 2012) do not reach the quality level of supervised and semi-supervised methods yet.

ERROR PROPAGATION. High-level NLP techniques such as SA often rely on low(er)-level NLP techniques, e. g. tokenization, sentence segmentation, WSD, and Part of Speech (POS) tagging, many of which pose challenging research questions on their own. Errors made in earlier stages of an NLP workflow *propagate* up to later stages such as SA.

Part I

BACKGROUND

2 DATASETS

> It is a capital mistake to theorize before one has data.
> Insensibly one begins to twist facts to suit theories,
> instead of theories to suit facts.
> — *Arthur Conan Doyle,*
> *A Scandal in Bohemia*

In this chapter we introduce the gold standard datasets that are used in this thesis (see Section 2.1). Furthermore, we describe how we preprocess them (see Section 2.2).

Gold standard

Corpus

Generally, a *gold standard*[1] denotes the at that time best available tool to compare different measures (see Claassen, 2005). We consider a dataset—a *corpus*[2]—that is manually labeled as a gold standard. Labeling the same data automatically allows us to compare automatically and manually determined labels to assess the quality of the method that automatically labeled the data.

2.1 GOLD STANDARDS

Table 1 provides an overview of all gold standards that we use in this thesis and lists the genres they originate from and the domains they contain. We now describe these gold standards in detail.

2.1.1 *Darmstadt Service Review Corpus*

Toprak et al. (2010)'s Darmstadt Service Review Corpus (DSRC)[3] is a gold standard for sentence- and expression-level SA. DSRC contains 474 *reviews* of 2 domains: online universities (240 reviews) and online services (234 reviews). The reviews consist of 8,877 sentences and 152,300 word tokens (see Toprak et al., 2010). Among others, on sentence

[1] Gold standard is originally a historical term from economics.

[2] A corpus is a "collection of linguistic data, either written texts or a transcription of recorded speech (. . .)" (see Crystal, 2008, p. 117).

[3] http://www.ukp.tu-darmstadt.de/data/sentiment-analysis/darmstadt-service-review-corpus/

Table 1.: Overview of gold standards.

GOLD STANDARD	GENRE(S)	DOMAIN(S)
DSRC	reviews	universities, web services
MDSD v2.0	reviews	various
MPQA v2.0	news articles	various
PD v2.0	reviews	movies
RND	reviews	various
SD v1.0	reviews, plot summaries	movies
SE-2007-T14D	news headlines	various
SE-2013-T2BD	tweets, SMS texts	various
SPD v1.0	reviews	movies
T-MDSD	tweets	various

level topic relevancy, polarity, and subjectivity are annotated. Among others, on expression level opinion-bearing terms, opinion holders, and opinion targets are annotated.

2.1.2 *Multi-Domain Sentiment Dataset v2.0*

Blitzer et al. (2007)'s Multi-Domain Sentiment Dataset v2.0 (MDSD v2.0)[4] is a gold standard for in- and cross-domain document-level polarity classification. It contains star-rated Amazon[5] *product reviews* of various domains, out of which we chose 10 domains: apparel, books, dvds, electronics, health & personal care, kitchen & housewares, music, sports & outdoors, toys & games, and videos. Those are exactly the domains for which a pre-selected, balanced amount of 1,000 positive and 1,000 negative product reviews is available. Additionally, MDSD v2.0 contains large amounts of unlabeled reviews per domain. Blitzer et al. (2007) consider reviews with more than 3 stars as positive, and less than 3 stars as negative—they omit 3-star reviews. Table 2 provides an overview of MDSD v2.0's labeled data.

[4] http://www.cs.jhu.edu/~mdredze/datasets/sentiment/

[5] http://www.amazon.com

Table 2.: Overview of MDSD v2.0.

DOMAIN	TYPES	TOKENS
apparel	10,163	133,746
books	28,513	359,872
dvd	30,282	392,313
electronics	15458	232,362
health	13,087	186,339
kitchen	13,021	196,339
music	25,281	292,614
sports	14,290	215,087
toys	14,090	204,049
video	25,376	326,501

2.1.3 *Multi-Perspective Question Answering Corpus v2.0*

Wiebe et al. (2005)'s Multi-Perspective Question Answering Corpus v2.0 (MPQA v2.0)[6] is a gold standard for a variety of SA subtasks, e. g. opinion holder extraction. It contains 10,657 sentences from 535 *news articles* from 187 different news sources.

From these news articles Riloff et al. (2006) extracted 5,380 subjective and 4,352 objective sentences according to a definition to be found in the MPQA v2.0 manual: "A sentence was considered subjective if 1 OR 2:

1. The sentence contains a "direct-subjective" annotation WITH attribute intensity NOT IN ['low', 'neutral'] AND NOT WITH attribute insubstantial.

2. The sentence contains a "expressive-subjectivity" annotation WITH attribute intensity NOT IN ['low']."

This subset of MPQA v2.0 contains 19,407 word types and 261,137 word tokens and functions as a gold standard for sentence-level subjectivity classification.

[6] http://www.cs.pitt.edu/mpqa/

2.1.4 *Polarity Dataset v2.0*

Pang and Lee (2004)'s Polarity Dataset v2.0 (PD v2.0)[7] is a gold standard for document-level polarity classification. It contains 2,000 *movie reviews* from IMDb[8] that consist of 47,219 word types and 1,523,410 word tokens. All reviews were automatically labeled for their polarity; there are 1,000 positive and 1,000 negative reviews. PD v2.0 is fully lowercased, i. e. it contains no capitalization.

2.1.5 *Ratingz Network Dataset*

The Ratingz Network[9] is an online platform, where people can review and rate places and services in the United States of America. Websites that are affiliated with The Ratingz Network, e. g. `clubratingz.com` and `vetratingz.com`, share a common structure. We partially crawled reviews from 9 The Ratingz Network websites on May 23rd, 2013 and compiled them into a gold standard—as from now called Ratingz Network Dataset (RND)—for in- and cross-domain document-level polarity classification.

RND contains *reviews* of 9 domains: summer camps, preschools & childcare, nightclubs, medical doctors, lawyers & attorneys, radio shows, real estate agents, restaurants, and veterinarians. All reviews are star-rated for several aspects. E. g., RESTAURANTS are rated for their food, their ambience, and their service. LAWYERS & ATTORNEYS are rated for their knowledge, their communication, their tenacity, their work quality, and their value. From these aspects we derive an overall rating by averaging their ratings. Reviews with an average rating of 3.5 stars or higher are considered positive. Reviews with 2.5 stars or lower are considered negative. Reviews with more than 2.5 stars but less than 3.5 stars are omitted. For each domain 2,000 positive and 2,000 negative reviews are available. Table 3 provides an overview of RND.

[7] `http://www.cs.cornell.edu/people/pabo/movie-review-data/`

[8] `http://www.imdb.com`

[9] `http://www.ratingz.net/`

Table 3.: Overview of RND.

DOMAIN	TYPES	TOKENS
camps	17,330	296,022
childcare	15,326	303,635
clubs	15,762	185,126
doctors	16,558	298,167
lawyer	147,08	247,683
radio	17,712	200,226
real estate	14,231	262,527
restaurant	22,761	336,251
vet	17,941	379,700

2.1.6 *SemEval-2007 Task 14 Dataset*

Strapparava and Mihalcea (2007)'s SemEval-2007 Task 14 Dataset (SE-2007-T14D)[10] is a gold standard for sentence-level polarity classification. It contains 1,250 *news headlines* from e. g. The New York Times[11], CNN[12], and BBC[13] that consist of 3,852 word types and 8,796 word tokens. All news headlines are annotated for their polarity intensity. Polarity intensity varies within $[-100, 100]$, where Strapparava and Mihalcea (2007) consider a value within $[-100, 0)$ as negative, $(0, 100]$ as positive and 0 as neutral or no polarity. Accordingly, there are 561 positive, 674 negative, and 15 neutral news headlines.

Undersampling

When we discard the neutral headlines and balance the positive and negative headlines via *undersampling*[14] the majority class, i. e. the negative headlines, we are left with 1,122 headlines (561 positive and 561 negative).

[10] http://www.cse.unt.edu/~rada/affectivetext/

[11] http://www.nytimes.com/

[12] http://www.cnn.com

[13] http://www.bbc.com

[14] Undersampling removes randomly chosen instances from the majority class in order to create a balanced dataset (see Akbani et al., 2004).

2.1.7 *SemEval-2013 Task 2B Dataset*

SemEval-2013 Task 2B Dataset (SE-2013-T2BD)[15] used in Wilson et al. (2013)'s SemEval-2013 shared task on SA in Twitter is a gold standard for document-level polarity classification. It contains *tweets* from various domains—e. g. regarding certain persons (Gaddafi, Steve Jobs etc.), products (Kindle, Android phones etc.), and events (Japan earthquake, NHL playoffs etc.). All tweets are annotated for their polarity. Table 4 provides an overview of SE-2013-T2BD split by training, development, and test data as provided by the shared task organizers after we removed duplicate tweets.

Table 4.: Overview of SE-2013-T2BD. "Pos" denotes positive, "Neg" negative, and "Neu" neutral tweets.

SPLIT	POS	NEG	NEU	ALL	TYPES	TOKENS
Training	3,263	1,278	4,132	8,673	29,453	192,854
Development	384	197	472	1,053	5,936	24,132
Test	1,572	601	1,640	3,813	16,325	88,206
All	5,219	2,076	6,244	13,539	41,092	305,192

Additionally, SE-2013-T2BD contains 2,094 *Short Message Service (SMS) texts* (492 positive, 394 negative, 1,208 neutral) from various domains that were used for testing out-of-genre and out-of-domain performance in SemEval-2013 Task 2B. The SMS texts consist of 4,631 word types and 37,549 word tokens.

2.1.8 *Sentence Polarity Dataset v1.0*

Sentence Polarity Dataset v1.0 (SPD v1.0)[7] by Pang and Lee (2005) is a gold standard for sentence-level polarity classification. It contains 10,662 text snippets—roughly one sentence per snippet—from Rotten Tomatoes[16] *movie reviews* that consist of 20,530 word types and 229,750 word tokens. All text snippets were automatically labeled for their polarity. There are 5,331 positive and 5,331 negative text snippets. SPD v1.0 is fully lowercased, i. e. it contains no capitalization.

[15] http://www.cs.york.ac.uk/semeval-2013/task2/

[16] http://www.rottentomatoes.com/

2.1.9 *Subjectivity Dataset v1.0*

Pang and Lee (2004)'s Subjectivity Dataset v1.0 (SD v1.0)[7] is a gold standard for sentence-level subjectivity classification. It contains 10,000 text snippets that were automatically labeled for their subjectivity (5,000 subjective text snippets from Rotten Tomatoes[16] movie reviews and 5,000 objective text snippets from IMDb[8] plot summaries). In total, SD v1.0 consists of 23,918 word types and 240,571 word tokens. SD v1.0 is fully lowercased, i. e. it contains no capitalization.

2.1.10 *Twitter Multi-Domain Sentiment Dataset*

Twitter Multi-Domain Sentiment Dataset (T-MDSD) is a gold standard for document-level polarity classification that contains 4 of the domains that are also found in MDSD v2.0: apparel, electronics, health & personal care, and kitchen & housewares. But T-MDSD originates from a different genre—Twitter *tweets*.

We compiled T-MDSD as follows: We extracted named entities from the aforementioned 4 domains of MDSD v2.0 using OpenNLP's[17] named entity recognition. We manually filtered out non-named entities and named entities that are not associated with the specific domain. For each named entity, i. e. for each keyword (see Table 40) we crawled Twitter using the following query:

```
<keyword> :( OR :)
```

Queries that yielded tweets that are not associated with the specific domain were filtered out manually. Per domain we merged the tweets and sampled a balanced amount of 1,000 tweets that contain the emoticon `:(` and 1,000 tweets that contain the emoticon `:)` but not both `:(` and `:)`.

`:(` and `:)` function as a proxy (see Chapter 4.1.1): we assume that all tweets containing `:(` or similar emoticons may be considered as of negative polarity, while all tweets containing `:(` or similar emoticons (see Table 5) may be considered as of positive polarity (see Go et al., 2009). Finally, we removed these proxies from the tweets but saved their labels.

[17] http://opennlp.apache.org

Table 5.: Emoticons similar to :(and :) (see Go et al., 2009).

EMOTICON	SIMILAR EMOTICONS
:(	:(, :-(, : (
:)	:), :-), :), :D, =)

2.2 PREPROCESSING

Before algorithms for high-level NLP such as SA can be applied natural language text needs to be preprocessed. Our preprocessing comprises tokenization, sentence segmentation, POS tagging and syntactic parsing.

2.2.1 *Tokenization*

Tokenization divides text into tokens, e. g. words, numbers, and punctuation marks (see Manning and Schütze, 1999, p. 124). We use OpenNLP's[17] maximum entropy tokenizer for any tokenization. Tokenizing Example 14

(14) Don't buy these shows for running!

using this tokenizer yields the tokens "Do", "n't", "these", "shoes", "for", "running", and "!".

2.2.2 *Sentence Segmentation*

Sentence segmentation divides text into sentences. We use OpenNLP's[17] sentence detection tool for sentence segmentation.

2.2.3 *Part of Speech Tagging*

Tag set

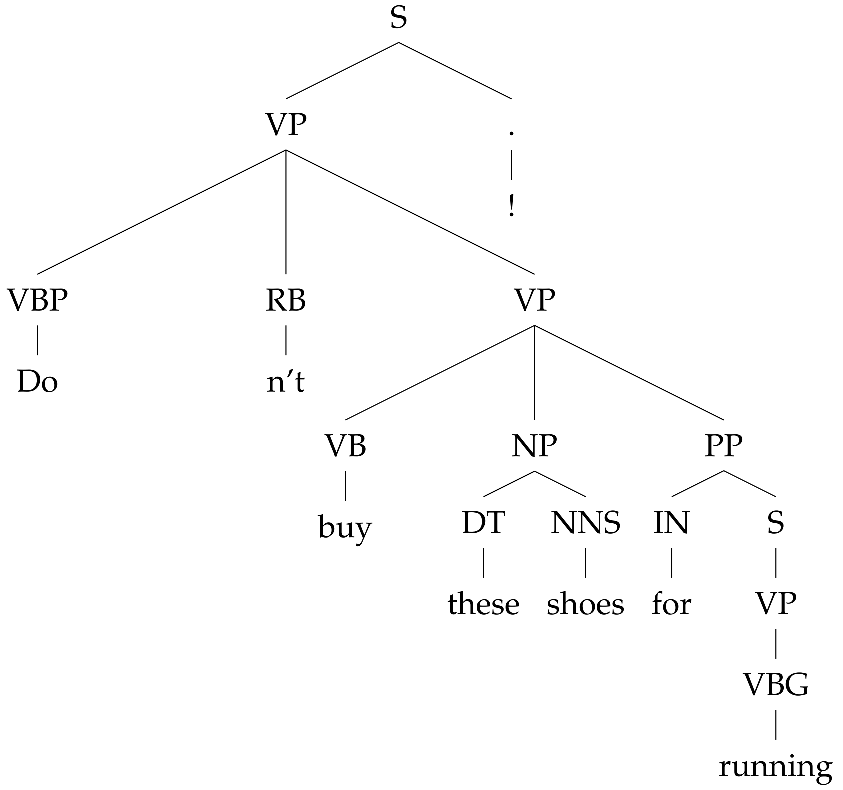
POS tagging assigns a POS tag from a *tag set* to each token in a text. Common tag sets are e. g. the Penn Treebank Tag Set (PTBTS)[18] and the Stuttgart Tübingen Tag Set (STTS)[19]. Throughout this thesis, we use Stanford Log-linear Part-of-Speech Tagger[20] (see Toutanova et al., 2003) for POS tag-

[18] http://www.cst.dk/mulinco/filer/PennTreebankTS.html

[19] http://www.ims.uni-stuttgart.de/forschung/ressourcen/lexika/TagSets/stts-table.html

[20] http://nlp.stanford.edu/software/tagger.shtml

Figure 2.: A phrase structure tree of "Don't buy these shows for running!".



ging. POS tagging Example 14 using this POS tagger and the PTBTS yields

(15) Do|VBP n't|RB buy|VB these|DT shoes|NNS for|IN running|VBG !|.

2.2.4 *Syntactic Parsing*

Phrase structure tree

Syntactic parsing reconstructs the so called *phrase structure tree* that gives rise to a particular sentence (see Manning and Schütze, 1999, p. 107). We use Klein and Manning (2003)'s Stanford Parser[21] for syntactic parsing. Parsing Example 14 using this parser yields the phrase structure—or constituency—parse tree shown in Figure 2.

Typed dependency

Stanford parser not only outputs syntax parse trees, but also *typed dependencies* (see de Marneffe et al., 2006), i. e. certain grammatical relations, e. g. nominal subject or temporal modifier. The (collapsed) typed dependencies of Example 14 are

[21] `http://nlp.stanford.edu/software/lex-parser.shtml`

```
aux(buy, Do)
neg(buy, n't)
det(shoes, these)
dobj(buy, shoes)
prepc_for(buy, running)
```

where—according to the parser's output—"Do" is an auxiliary of "buy", "n't" is a negation modifier of "buy", "these" is a determiner of "shoes", "shoes" is a direct object of "buy", and "buy" is a prepositional modifier of "running".

3 TEXTUAL CHARACTERISTICS

> Darüber hinaus muss eine erste prinzipielle Vorstellung über die *Art des Werkstoffs* bestehen (. . .). Eine Vorstellung über die Gestalt genügt oft nicht, sondern erst die Festlegung *prinzipieller Werkstoffeigenschaften* ermöglicht eine zutreffende Aussage über den *Wirkzusammenhang*. Nur die Gemeinsamkeit von physikalischem Effekt sowie geometrischen und stofflichen Merkmalen (. . .) lässt das Prinzip der Lösung sichtbar werden.
>
> — *Pahl and Beitz (1986, p. 30)*

Textual characteristics

Textual characteristics, e. g. word frequencies, word distributions, and word transition probabilities, quantify language use in text via language statistics (see Bank et al., 2012). Consequently, textual characteristics quantify similarities and dissimilarities between texts, either by comparing their language use directly or indirectly by comparing "fingerprints" of their language use.

This chapter describes textual characteristics for direct comparison of language use—domain similarity (see Section 3.1)—and textual characteristics for indirect comparison of language use: domain complexity (see Section 3.2) and readability (see Section 3.3).

3.1 DOMAIN SIMILARITY

An instance of a specific genre and a specific domain is represented by a collection of documents—a corpus (see Chapter 2). Therefore, for our purposes, *domain similarity* is identical to corpus similarity and domain complexity is identical to corpus complexity.

Measures for corpus similarity and corpus complexity "consider only raw word-counts" instead of e. g. "lemmas, or word senses, or syntactic constituents" to "be as theory-neutral as possible" (see Kilgarriff and Rose, 1998). We then rewrite—without discontinuity—word frequencies as word probabilities (see Halliday, 1991, p. 82). Thus, a corpus may be seen as a probability distribution over words, which can be encoded as a vector. We now present several domain similarity approximations: information-theoretic measures (see Section 3.1.1) and geometrically-motivated measures (see Section 3.1.2).

3.1.1 *Information-theoretic Measures*

Several information-theoretic measures were proposed to approximate domain similarity (see Van Asch and Daelemans, 2010; Plank and van Noord, 2011). Note that all measures presented below actually measure *divergence*, i. e. dissimilarity instead of similarity. A divergence function $div \rightarrow [0,1]$ can always be converted into a similarity function $sim \rightarrow [0,1]$ by $sim := 1 - div$. Proposed measures are, among others:

Divergence

KULLBACK-LEIBLER DIVERGENCE. Kullback-Leibler (KL) divergence (see Kullback and Leibler, 1951) is defined as in Equation (3.1)

$$D_{KL}(Q||R) = \sum_{w \in W} Q(w) \log \frac{Q(w)}{R(w)} \tag{3.1}$$

where Q and R are probability distributions over a finite set W, e. g. words. KL divergence is not necessarily symmetric and it is undefined if $\exists w' \in W : Q(w') > 0$ but $R(w') = 0$. The larger $D_{KL}(Q||R)$, the more Q and R diverge, i. e. the less similar they are.

JENSEN-SHANNON DIVERGENCE. Jensen-Shannon (JS) divergence (see Lin, 1991) is based on KL divergence and is defined as in Equation (3.2)

$$D_{JS}(Q||R) = \frac{1}{2}\left[D_{KL}(Q||M) + D_{KL}(R||M)\right] \tag{3.2}$$

where $M = \frac{1}{2}(Q + R)$ is the average distribution of Q and R and $0 \leqslant D_{JS}(Q||R) \leqslant 1$. In contrast to KL divergence, JS divergence is symmetric as well as well-defined for noncontinuous probability distributions. The larger $D_{KL}(Q||R)$, the more Q and R diverge, i. e. the less similar they are.

RENYI DIVERGENCE. Renyi divergence (see Rényi, 1961) is defined as in Equation (3.3)

$$D_{\text{Renyi}}(Q||R;\alpha) = \frac{1}{\alpha-1}\log_2\left(\sum_{w\in W}\frac{Q(w)^\alpha}{R(w)^{\alpha-1}}\right) = \frac{1}{\alpha-1}\log_2\left(\sum_{w\in W} Q(w)^\alpha \cdot R(w)^{1-\alpha}\right) \tag{3.3}$$

where $\alpha \in [0,1)$ is a free parameter. $D_{\text{Renyi}}(Q||R;\alpha)$ approaches $D_{\text{KL}}(Q||R)$ for $\alpha \to 1$.

SKEW DIVERGENCE. Skew divergence (see Lee, 2001b) is also based on the KL divergence and defined as in Equation (3.4)

$$D_{\text{Skew}}(Q||R;\alpha) = D_{\text{KL}}(R||M) \tag{3.4}$$

where $\alpha \in [0,1)$ is a free parameter and $M = \alpha Q + (1-\alpha)R$ is a "mixed" distribution of Q and R.

3.1.2 *Geometrically-motivated Measures*

Several geometrically-motivated measures were proposed to approximate domain similarity. Proposed measures are, among others:

COSINE SIMILARITY. Cosine similarity $\cos(\mathbf{q},\mathbf{r})$ measures the cosine of the angle between 2 vectors and is defined as in Equation (3.5)

$$\cos(\mathbf{q},\mathbf{r}) = \frac{\mathbf{q}\cdot\mathbf{r}}{||\mathbf{q}||\cdot||\mathbf{r}||} = \frac{\sum_{i=1}^{n} q_i \times r_i}{\sqrt{\sum_{i=1}^{n} q_i^2} \times \sqrt{\sum_{i=1}^{n} r_i^2}} \tag{3.5}$$

where $\mathbf{q},\mathbf{r} \in \mathbb{R}^n$ are n-dimensional vectors. The larger $\cos(\mathbf{q},\mathbf{r})$, the more similar $\mathbf{q}$ and $\mathbf{r}$.

EUCLIDEAN DISTANCE. Euclidean distance $\text{euc}(\mathbf{q},\mathbf{r})$ is defined as in Equation (3.6).

$$\text{euc}(\mathbf{q},\mathbf{r}) = \sqrt{\sum_{i=1}^{n}(q_i - r_i)^2} \tag{3.6}$$

The smaller $\text{euc}(\mathbf{q},\mathbf{r})$, the more similar $\mathbf{q}$ and $\mathbf{r}$.

TAXICAB GEOMETRY. Taxicab geometry (see Krause, 1987), L_1 distance or Manhattan distance $\text{tax}(\mathbf{q}, \mathbf{r})$ is defined as in Equation (3.7).

$$\text{tax}(\mathbf{q}, \mathbf{r}) = \sum_{i=1}^{n} |q_i - r_i| \tag{3.7}$$

The smaller $\text{tax}(\mathbf{q}, \mathbf{r})$, the more similar $\mathbf{q}$ and $\mathbf{r}$.

3.2 DOMAIN COMPLEXITY

Ponomareva and Thelwall (2012a) introduce *domain complexity* as a measure that "reflects the difficulty of [a] classification task for a given data set." We will now present several domain complexity approximations that were proposed.

3.2.1 *Ponomareva and Thelwall (2012a)*

Ponomareva and Thelwall (2012a) suggest several functions to approximate domain complexity:

PERCENTAGE OF RARE WORDS. Percentage of rare words is defined as in Equation (3.8)

$$\text{PRW} = \frac{|\{w \in W \mid c(w) < 3\}|}{|W|} \tag{3.8}$$

where W is the vocabulary, vocabulary size $|W|$ equals the number of *types*, i. e. the number of different words in a text sample, and $c(w)$ is the number of occurrences of w in a text sample. Bank et al. (2012) suggested a language statistic for corpus comparison that is very similar to percentage of rare words: vocabulary dispersion.

Types

WORD RICHNESS. Word richness is defined just as the ordinary *type/token ratio* TTR in Equation (3.9)

Type/token ratio

$$\text{TTR} = \frac{|W|}{\sum_{w \in W} c(w)} \tag{3.9}$$

where $\sum_{w \in W} c(w)$ equals the number of *tokens*, i. e. the total number of words in a text sample (see Crystal, 2008, p. 498). Bank et al. (2012) suggested a language statistic for corpus comparison that is very similar to type/token-ratio: relative vocabulary size.

Tokens

RELATIVE ENTROPY. Relative entropy is defined in Equation (3.10)

$$H_{\text{rel}} = \frac{H}{H_{\text{max}}} \tag{3.10}$$

where H as in Equation (3.11)

$$H = -\sum_{w \in W} p(w) \log_2 p(w) \tag{3.11}$$

is the *entropy* of W's distribution and H_{max} as in Equation (3.12)

Entropy

$$\begin{aligned} H_{\text{max}} &= -\sum_{w \in W} \frac{1}{|W|} \log_2 \frac{1}{|W|} \\ &= \log_2 |W| \end{aligned} \tag{3.12}$$

is the *maximum entropy* of W's distribution, i. e. its entropy if W was distributed uniformly. Bank et al. (2012) suggested to use entropy as a language statistic for corpus comparison.

Maximum entropy

3.2.2 *Remus (2012)*

Remus (2012) proposed *corpus homogeneity* (see Kilgarriff and Rose, 1998; Kilgarriff, 2001) as another approximation of domain complexity. Corpus homogeneity, corpus self-similarity or simply homogeneity uses repeated random subsampling validation and is estimated as shown in Pseudocode 1.

Corpus homogeneity

Pseudocode 1: Corpus homogeneity.

```
for i = 1, ..., k {
    shuffle corpus c
    split c into 2 equally-sized subcorpora $c_1$, $c_2$
    selfsimilarity $s_i$:= sim($c_1$, $c_2$)
}
homogeneity $Hom$ := average($s_1$,...,$s_k$)
```

sim(c_1, c_2) is any similarity function (see Section 3.1.1 and Section 3.1.2). If the corpus is document-based the documents are shuffled, if the corpus is sentence-based the sentences are shuffled etc. For $k \to \infty$ the estimate approaches the "actual" corpus homogeneity. In our later experiments we set k to 10 and use JS divergence (see Section 3.1.1) as our similarity function.

Remus (2012)'s primary motivation to use corpus homogeneity as an approximation of domain complexity instead of Ponomareva and Thelwall (2012a)'s and Bank et al. (2012)'s suggestions is Kilgarriff (2001), who puts it this way:

> "Ideally, the same measure can be used for similarity and homogeneity, as then Corpus 1/Corpus 2 distances will be directly comparable with heterogeneity for Corpus 1 and Corpus 2."

3.2.3 *Ho and Basu (2002)*

Ho and Basu (2002) pointed out that in general a classification problem can be difficult for 3 reasons:

1. Its classes are ambiguous.
2. Its decision boundary is complex.
3. Its sample size is too small.

Classes may be "ambiguous either intrinsically or due to inadequate feature measurements." (see Ho and Basu, 2002) Complex decision boundaries or complex subclass structures may obviate a compact description of the decision boundary or subclass structure. Small sample sizes and sparsity may obviate constraints on the generalizations because of the curse of dimensionality. All domain complexity approximations introduced in Section 3.2.1 and Section 3.2.2 measure classification difficulty according to Ho and Basu (2002)'s 3rd category.

Class boundary complexity

In contrast to the domain complexity approximations introduced in Section 3.2.1 and Section 3.2.2, Ho and Basu (2002) proposed several "measures that characterize the difficulty of a classification problem, focusing on geometrical complexity of the class boundary", i. e. measures that fall into their 2nd category. Such descriptors of *class boundary complexity* can be divided in several categories: measures of overlap of individual feature values, e. g. maximum Fisher's discriminant ratio, measures of separability of classes, e. g. the fraction of points on the class boundary, and measures of geometry, topology, and density of manifolds, e. g. the average number of points per dimension. Most of these measures rely on labeled data and are unmeasurable in an unsupervised manner. In a "real-world"

Table 6.: Overview of subcorpora of eng_news_2010.

SENTENCES	TYPES	TOKENS
100	1,055	2,066
300	2,484	6,115
1,000	5,929	20,560
3,000	12,580	61,944
10,000	26,589	206,619
30,000	51,079	618,920
100,000	102,825	2,062,683
300,000	193,383	6,196,353
1,000,000	389,418	20,648,187

scenario only unsupervised measures are utile (see Chapter 5 and Chapter 6). Hence, we do not consider Ho and Basu (2002)'s measures further in our thesis.

3.2.4 *Properties*

We hypothesize that the domain complexity approximations as described in Section 3.2.1 and Section 3.2.2 have an important property: they are *sample size dependent*, similar to what was shown in Remus and Bank (2012) for Bank et al. (2012)'s textual characteristics.

Sample size dependence

To verify our hypothesis, we investigate how domain complexity approximations behave when applied to corpora of increasing size. We use an English-language corpus provided by the *Wortschatz project*[1] (see Quasthoff et al., 2006) that contains newspaper articles: eng_news_2010. Out of eng_news_2010 we construct subcorpora C_{size} containing $size \in \{10^2, 3 \cdot 10^2, 10^3, 3 \cdot 10^3, 10^4, 3 \cdot 10^4, 10^5, 3 \cdot 10^5, 10^6\}$ sentences such that

Wortschatz project

$$\forall l < m : C_l \subset C_m$$

i. e. any smaller corpus is always a real subset of any larger corpus. Table 6 provides an overview of the eng_news_2010 subcorpora we constructed.

Measuring domain complexity of these subcorpora leads to the results shown in Figure 3. We note: the larger the

[1] http://wortschatz.uni-leipzig.de/

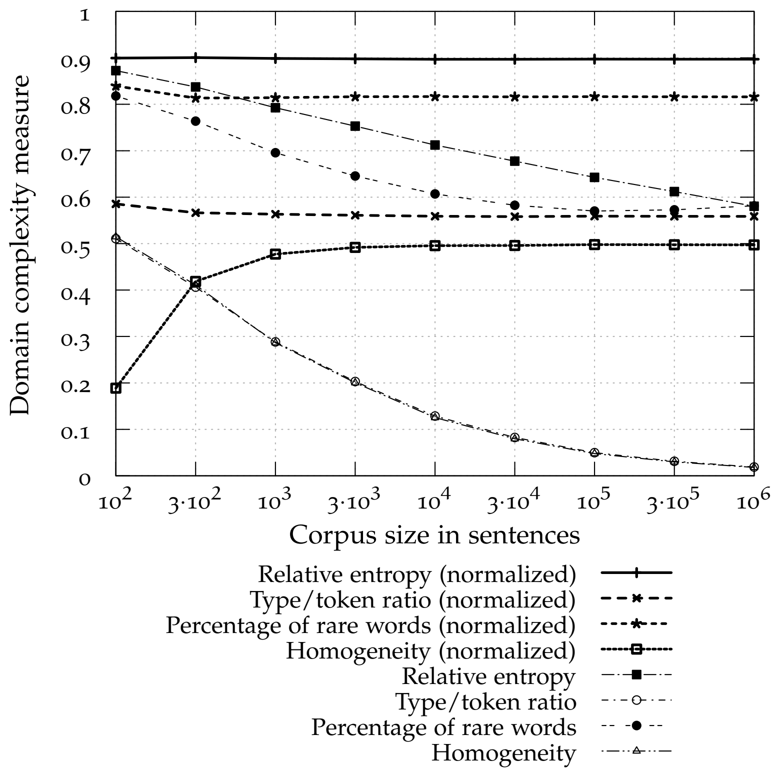


Figure 3.: Behavior of domain complexity measures with and without sample size normalization for `eng_news_2010` subcorpora of increasing size.

corpus

- the smaller its relative entropy,
- the smaller its type/token ratio,
- the smaller its percentage of rare words and
- the smaller its homogeneity.

Percentage of rare words stabilizes at around 10,000 sentences. Although this is not generally the case, type/token ratio and homogeneity strongly correlate ($r > 0.999$).

We conclude that domain complexity approximations are indeed sample size dependent and must not be used to compare corpora that differ greatly in size. To ensure comparability across different-sized corpora, we normalize domain complexity with respect to sample size.

Sample Size Normalization

We compute percentage of rare words, type/token ratio, and relative entropy on fixed length subsamples rather than on the full sample as shown in Pseudocode 2.

Pseudocode 2: Sample size-normalized domain complexity.

```
for $i$ = 1, ..., $k$ {
    subsample $s_i$ := extract random word window of size
        1000 from full sample
    measurement $m_i$ := domain complexity($s_i$)
}
normalized domain complexity := average($m_1$,...,$m_k$)
```

Using a sufficient number of iterations k—10,000 in our case—we obtain a stable approximation of the expected domain complexity value, which is normalized with respect to sample size.

To compute sample size-normalized homogeneity, we proceed as shown in Pseudocode 1. But instead of shuffling the corpus and splitting it into 2 equally-sized subcorpora, we randomly extract 2 fixed length subsamples s_i^1, s_i^2 analog to Pseudocode 2 with the constraint that s_i^1, s_i^2 must not overlap. We then measure $\mathtt{sim}(s_i^1, s_i^2)$. In deviation from corpus homogeneity as described in Section 3.2.2 we here set k to 10,000 instead of 10.

As we can see in Figure 3, sample size-normalized domain complexity approximations stabilize at around 1,000 to 3,000 sentences and are asymptotically consistent.

3.3 READABILITY

After presenting measures for domain similarity (see Section 3.1) and domain complexity (see Section 3.2), we now present measures for readability. *Readability* is referred to as "the degree to which a given class of people find certain reading matter compelling and, necessarily, comprehensible" (see McLaughlin, 1969).

(16) Do not buy these shoes for running!

(17) Like all shoes, they needed to be 'broken in' and at 9 months, you aren't very capable of that.

Clearly, it is easier to comprehend Example (16) than Example (17). According to Klare (1974)'s survey there are 3 possible solutions to "tell whether a particular piece of writing is likely to be readable to a particular group of readers": A first solution is simply to guess. A second solution are tests, manually built and refined. A third solution are *readability gradings* and *readability indicators*. We will measure readability using such gradings (see Section 3.3.1) and indicators (see Section 3.3.2) because many of them are automatically computable (see Remus, 2011).

Readability has to be distinguished from domain complexity (see Section 3.2). While domain complexity indicates how hard it is for an ML algorithm to "comprehend" textual data, readability indicates how hard it is for a human reader to comprehend textual data.

3.3.1 *Gradings*

According to McLaughlin (1969) a *readability grading* is a formula derived by linear regression (LR) (see Chapter 4.1.5),

> "which best expresses the relationship between (...) a measure of the difficulty experienced by people reading a given text, and a measure of the linguistic characteristics of that text. This formula can then be used to predict reading difficulty from the linguistic characteristics of other texts."

Early Work

There is a large body of readability formulae (see Klare, 1974). We present only readability formulae that are automatically computable and do not depend on lexical resources, e. g. certain word lists. We present only the formulae themselves. Their underlying ideas, their development, and the derivation of their variables and constants is described in the original work as referenced below.

DEVEREUX READABILITY INDEX. Smith (1961)'s Devereux Readability Index (DRI) is calculated as shown in Equation (3.13)

$$R_{\mathrm{DRI}} = 1.56\,wl + 0.19\,sl - 6.49 \tag{3.13}$$

where wl is the average word length in characters and sl is the average sentence length in words. The larger R_{DRI}, the less readable the text.

EASY LISTENING FORMULA. Fang (1966)'s Easy Listening Formula (EL) is calculated as shown in Equation (3.14)

$$R_{\mathrm{EL}} = npsw \tag{3.14}$$

where $npsw$ is the average number of polysyllabic words per sentence, i. e. words with strictly more than one syllable. EL is—as its name suggests—more tailored to "listenability" than to readability. Therefore, the larger R_{EL}, the less "listenable" the text.

FLESH-KINCAID SCORE. Flesh-Kincaid Score (FKS) was introduced in Kincaid et al. (1975) and is calculated as shown in Equation (3.15)

$$R_{\mathrm{FKS}} = 0.39\,sl + 11.8\,nsw - 15.59 \tag{3.15}$$

where nsw is the average number of syllables per words. The larger R_{FKS}, the less readable the text.

FOG INDEX. Fog Index (FI) was introduced in Gunning (1952) and reformulated by Powers et al. (1958). It is calculated as shown in Equation (3.16)

$$R_{\mathrm{FI}} = 3.068 + 0.0877\,sl + 0.0984\,nmsw \tag{3.16}$$

where $nmsw$ is the average number of monosyllabic words per sentence, i. e. words with exactly one syllable. The larger R_{FI}, the less readable the text.

Table 7.: Linguistic characteristics measured by different readability gradings.

	DRI	EL	FI	FKS	FORCAST	NREI	SMOG
wl	✓						
sl	✓		✓			✓	
$nmsw$			✓		✓	✓	
$npsw$		✓					✓
nsw				✓			

FORCAST. Caylor et al. (1973)'s FORCAST is calculated as shown in Equation (3.17).

$$R_{\text{FORCAST}} = 20.41 - 0.11\, nmsw \tag{3.17}$$

The larger R_{FORCAST}, the less readable the text.

NEW READING EASE INDEX. Farr et al. (1951)'s New Reading Ease Index (NREI) is calculated as shown in Equation (3.18).

$$R_{\text{NREI}} = 1.599\, nmsw - 1.015\, sl - 31.517 \tag{3.18}$$

The larger R_{NREI}, the less readable the text.

SMOG GRADING. McLaughlin (1969)'s SMOG grading is calculated as shown in Equation (3.19).

$$R_{\text{SMOG}} = 3 + \sqrt{npsw} \tag{3.19}$$

McLaughlin (1969) argues that $npsw$ in R_{SMOG} simultaneously captures word length and sentence length. The larger R_{SMOG}, the less readable the text.

Different readability gradings measure different linguistic characteristics (see Table 7). Not only do they differ in what they measure, but also in their intended outcome. Whereas some readability formulae aim to determine a school grade, some refer to certain tables for further interpretation. For those reasons the reading difficulty estimated by different readability formulae is not directly comparable. Readability formulae do have in common that higher outcomes generally signalize less readable or less "listenable" text.

Recent Work

Most readability gradings were proposed in the 1950s, '60s, and '70s. Only recently, several new readability gradings were proposed:

Si and Callan (2001) learn a weighted linear combination of a unigram Naïve Bayes (NB) model (see Chapter 4.1.3) and a sentence length distribution model to predict the most likely grade level (kindergarden–2nd, 3rd–5th, and 6th–8th) of science web pages. The weights are estimated using expectation maximization (see Hastie et al., 2009, pp. 272–279): their unigram NB model is weighted by 0.91, their sentence length distribution model is weighted by 0.09.

Collins-Thompson and Callan (2004) learn a unigram NB model (see Chapter 4.1.3) to predict grade levels 1st–12th of web pages. Their model smoothes word frequency data within classes using Simple Good-Turing smoothing (see Gale and Sampson, 1995) and across classes via regression. They also perform task-specific feature selection (FS).

Schwarm and Ostendorf (2005) learn a Support Vector Machine (SVM) model (see Chapter 4.1.2) to predict grade levels (2nd, 3rd, 4th, and 5th) of an educational newspaper. Their model uses several features: sentence length, number of syllables per word, FKS, different out-of-vocabulary scores, parse tree height, number of noun phrases (NPs) and verb phrases (VPs), number of clauses per sentence as well as different perplexity scores of uni-, bi-, and trigram language models.

Heilmann et al. (2007) learn a linear interpolation of a unigram NB model (see Chapter 4.1.3) and a k-nearest neighbor (see Mitchell, 1997, pp. 231–236) model to predict the most likely grade level (1st–12th) of first and second language learner texts. Their model is based on syntactic parsing and relies on grammatical constructions, e. g. verb tenses, voice etc.

3.3.2 *Indicators*

Readability gradings (see Section 3.3.1) usually indicate the grade level that is necessary to comprehend a certain text. Thus, they "obscure" the linguistic characteristics encoded by them. Consequently, their "calculated value does not permit to conclude what exactly has to be changed to improve the readability of the text" (see Oelke et al., 2010).

According to McLaughlin (1969) average sentence length in words and average word length in characters have the greatest predictive power regarding reading difficulty. Linguistic characteristics based on the number of syllables per word are also frequently used in readability gradings (see Section 3.3.1). Several other linguistic characteristics were proposed to function as *indicators* of reading difficulty. In addition to sentence length and word length, Oelke et al. (2010) semi-automatically chose linguistic characteristics as readability indicators that are "semantically understandable": noun/verb ratio, number of nominal forms, vocabulary complexity, and parse tree branching factor. These indicators and 2 others—parse tree depth and word frequency class distribution—are defined as follows:

NOUN/VERB RATIO. The noun/verb ratio is defined as the ratio between the number of verbal forms and the number of nominal forms. POS tags of PTBTS that qualify as nominal forms are `NN`, `NNS`, `NNP`, `NNPS`, `PRP`. POS tags of PTBTS that qualify as verbal forms are `VB`, `VBD`, `VBG`, `VBN` `VBP`, `VBZ`, `MD`. The noun/verb ratio of Figure 2 is then 0.33.

NUMBER OF NOMINAL FORMS. The number of nominal forms is just that. Figure 2 has 1 nominal form—`NNS`: "shoes".

VOCABULARY COMPLEXITY. The vocabulary complexity (see Oelke et al., 2010) is the percentage of words that are not among the 1,000 most frequent words in a large corpus. These words may be considered as *uncommon words*. Figure 2 contains 2 words that are not among the 1,000 most frequent words in the Wortschatz corpus `eng_news_2010` (see Section 3.2.4): "n't", "shoes".[2]

Uncommon words

PARSE TREE BRANCHING FACTOR. The parse tree branching factor is defined as the average number of child nodes that non-leaf nodes have in the parse tree (see Genzel and Charniak, 2003). The parse tree branching factor of Figure 2 is 1.467.

[2] `eng_news_2010` contains newspaper articles. Therefore, it does not contain contractions such as "don't" or "I'm".

PARSE TREE DEPTH. The parse tree depth is defined as the length of the longest path from the root node to a leaf node. The parse tree height of Figure 2 is 8. The parse tree depth is almost linearly dependent on the sentence length (see Genzel and Charniak, 2003).

WORD FREQUENCY CLASS DISTRIBUTION. The frequency class $\mathrm{FC}(w)$ of a word w (see Quasthoff et al., 2012) is computed as shown in Equation (3.20)

$$\mathrm{FC}(w) = \log_2 \frac{\mathrm{freq}(w_{\mathrm{top}})}{\mathrm{freq}(w)} \tag{3.20}$$

where $\mathrm{freq}(w)$ denotes the frequency of w and w_{top} denotes the most frequent word. The word frequency class distribution of a text then indicates what proportions of a text consist of high-, mid-, and low-frequency words.

The word frequency class distribution of Figure 2 is as follows: 1/8 of its words fall into word frequency class 2, 1/8 fall into word frequency class 7, 1/4 fall into word frequency class 8, 1/8 fall into word frequency class 10, 1/8 fall into word frequency class 11 and 1/4 fall into word frequency classes larger than 25.

Several readability indicators implicitly carry information about a text's average sentence length in words, e. g. the number of nominal forms, the number of polysyllabic words etc.

4 MACHINE LEARNING

> Above all, however, the machine has no feelings,
> it feels no fear and no hope, which only disturb,
> it has no wishes with regard to the result, it
> operates according to the pure logic of probability.
> For this reason I assert that the robot perceives
> more accurately than man, it knows more about
> the future, for it calculates it, it neither speculates
> nor dreams, but is controlled by its own findings
> (the feedback) and cannot make mistakes;
> the robot has no need of intuition . . .
> — *Max Frisch,*
> *Homo Faber: A Report, p. 76*

Since the advent of computers, the world wide web, and social media, vast amounts of data are generated every second. ML seeks to make sense out of that data: it extracts "important patterns and trends" (see Hastie et al., 2009, p. xi)—it tries to learn from that data.

This chapter describes ML algorithms for classification and regression (see Section 4.1), a method for scaling nonbinary feature values (see Section 4.2), methods for FS (see Section 4.3), evaluation measures (see Section 4.4), and techniques for model assessment (see Section 4.5) used in this thesis.

4.1 CLASSIFICATION AND REGRESSION ALGORITHMS

In this section we describe ML algorithms for classification and regression used in this thesis. Algorithms that are more frequently used—SVMs (see Section 4.1.2) and LR (see Section 4.1.5)—are discussed in more detail than algorithms that are less frequently used—NB (see Section 4.1.3), C4.5 Decision Trees (DTs) (see Section 4.1.4), and logistic regression (LogReg) (see Section 4.1.6).

Generally, there are two types of ML algorithms for classification: *discriminative* and *generative* classifiers (see Ng and Jordan, 2001):

Discriminative
Generative

DISCRIMINATIVE CLASSIFIERS. Given an input $\mathbf{x}$ and an output y, discriminative classifiers model the conditional probability $p(y \mid \mathbf{x})$, which can be used for classification. Some discriminative classifiers model y given $\mathbf{x}$ directly, without modeling $p(y \mid \mathbf{x})$.

GENERATIVE CLASSIFIERS. Given an input $\mathbf{x}$ and an output y, generative classifiers model $p(\mathbf{x} \mid y)$ and $p(y)$, which can be used to generate likely pairs $(\mathbf{x}, y)$. Using Bayes' theorem (see Section 4.1.3) $p(y \mid \mathbf{x})$ can be computed.

4.1.1 *Levels of Supervision*

Another important methodological issue is the *level of supervision* that ML algorithms need. It is best characterized by the quantity and quality of training data that is available to ML algorithms.

SUPERVISED ALGORITHMS. Supervised ML algorithms require *labeled* training data, i. e. data that consists of inputs and desired outputs: the *labels*. They then acquire the functional relation between input and output. Thereby, they try to generalize from the training data. Given that a sufficiently large amount of training data is available, supervised ML algorithms often achieve highly accurate results. However, labeling training data is usually labor-intensive and costly.

Labels

A variant of supervision is *distant supervision*. Here, training data is labeled by a *proxy*. E. g., a tweet that contains an *emoticon* such as : -) —the proxy—may be automatically labeled as positive, one that contains : - (may be automatically labeled as negative (see Go et al., 2009); a review that is given 5 stars may be automatically labeled as positive, one that is given 1 star may be automatically labeled as negative etc.

Distant supervision

Proxy

Emoticon

SEMI-SUPERVISED ALGORITHMS. Semi-supervised ML algorithms rely on only little labeled training data—the *seeds*—and additional larger amounts of unlabeled training data to acquire the functional relation between input and output.

Seeds

In *bootstrapping* (see Zhu, 2006, p. 11) seeds are used

Bootstrapping

to learn an initial model, which is used to label previously unlabeled training data. In turn, newly labeled and previously labeled training data are used in conjunction to learn a revised model. This process is repeated iteratively until pre-defined criteria for termination are met.

UNSUPERVISED ALGORITHMS. In contrast, unsupervised ML algorithms do not rely on any labeled training data, but rely on usually very large amounts of unlabeled training data. Unsupervised ML algorithms try to discover relevant structures autonomously, e. g. by *clustering*, i. e. ordering elements of some set according to some similarity measure.

Clustering

4.1.2 *Support Vector Machines*

SVMs (see Vapnik, 1995; Cortes and Vapnik, 1995) are a supervised, discriminative ML method based on a principle from statistical learning theory—Structural Risk Minimization (see Vapnik, 1998).

Let $T = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ be our training instances with $\mathbf{x}_i \in \mathbb{R}^n$ an n-dimensional input vector and $y_i \in \{1, -1\}$ a class label indicating whether $(\mathbf{x}_i, y_i)$ is a positive (1) or negative (-1) training instance. Assuming linear separability of T, a *hyperplane* in $\mathbb{R}^n$ linearly separates T into 2 half spaces and is defined by Equation (4.1)

Hyperplane

$$\mathbf{w} \cdot \mathbf{x} + b = 0 \tag{4.1}$$

where $\mathbf{w}$ is a vector and b is a scalar. A hyperplane $(\mathbf{w}, b)$ is equally expressed by $(\lambda\mathbf{w}, \lambda b)$ for all $\lambda \in \mathbb{R}^+$. To scale $(\mathbf{w}, b)$, i. e. to set λ we require that $(\mathbf{w}, b)$ separates T such that the "functional distances"

$$\mathbf{x}_i \cdot \mathbf{w} + b \geqslant 1 \quad \text{when} \quad y_i = 1 \tag{4.2}$$

$$\mathbf{x}_i \cdot \mathbf{w} + b \leqslant -1 \quad \text{when} \quad y_i = -1 \tag{4.3}$$

or, more compactly:

$$\forall i: \quad y_i\,(\mathbf{x}_i \cdot \mathbf{w} + b) \geqslant 1 \tag{4.4}$$

In other words: we place $(\mathbf{w}, b)$ in between the $\mathbf{x}_i$ closest to it. The *optimal hyperplane* (see Vapnik, 1995) then maximizes

Optimal hyperplane

the distance to these closest $\mathbf{x}_i$. The distance d between $\mathbf{x}_i$ and a hyperplane $(\mathbf{w}, b)$ is given by Equation (4.5)

$$d\left((\mathbf{w}, b), \mathbf{x}_i\right) = \frac{y_i\left(\mathbf{x}_i \cdot \mathbf{w} + b\right)}{\|\mathbf{w}\|} \tag{4.5}$$

where $\|\mathbf{w}\| = \sqrt{\mathbf{w}^T\mathbf{w}}$. From Inequality (4.4) and Equation (4.5) follows the Inequality (4.6):

$$d\left((\mathbf{w}, b), \mathbf{x}_i\right) \geqslant \frac{1}{\|\mathbf{w}\|} \tag{4.6}$$

The *margin* $\rho(\mathbf{w}, b)$ is then given by Equation (4.7).

Margin

$$\rho(\mathbf{w}, b) = \min_{\mathbf{x}_i : y_i = 1} d\left((\mathbf{w}, b), \mathbf{x}_i\right) + \min_{\mathbf{x}_i : y_i = -1} d\left((\mathbf{w}, b), \mathbf{x}_i\right) \tag{4.7}$$

It follows from Equation (4.7) and Inequality (4.6) that the optimal hyperplane $(\mathbf{w}_0, b_0)$ has a margin of

$$\rho(\mathbf{w}_0, b_0) = \frac{2}{\|\mathbf{w}_0\|} = \frac{2}{\sqrt{\mathbf{w}_0^T\mathbf{w}_0}} \tag{4.8}$$

As we can see from Equation (4.8), maximizing $\rho(\mathbf{w}, b)$ is accomplished by minimizing $\|\mathbf{w}\|$ subject to the constraint (4.4). Therefore, constructing an optimal hyperplane is a constrained optimization problem, often approached as a quadratic programming problem via Lagrange multipliers (see Cortes and Vapnik, 1995).

Cortes and Vapnik (1995) showed that $\mathbf{w}_0$ can be written as Equation (4.9)

$$\mathbf{w}_0 = \sum_{i=1}^{l} y_i \alpha_i \mathbf{x}_i \tag{4.9}$$

where α is an l-dimensional vector of non-negative Lagrange multipliers determined as described in Cortes and Vapnik (1995). It also holds that

$$\forall i : \qquad \alpha_i\left(y_i\left(\mathbf{w} \cdot \mathbf{x}_i + b\right) - 1\right) = 0 \tag{4.10}$$

i. e. if the functional distance of a particular $\mathbf{x}_i > 1$, then $\alpha_i = 0$: this particular $\mathbf{x}_i$ does not contribute to the linear combination of the optimal hyperplane given in Equation (4.9). $\mathbf{x}_i$ for which $y_i\left(\mathbf{x}_i \cdot \mathbf{w} + b\right) = 1$ are referred to as *support vectors*, hence the ML algorithm's name: SVM. To

Support vectors

determine b, we then solve Equation (4.11)

$$b = -\frac{1}{2}\left(\mathbf{w}\cdot\mathbf{x}_{y_i=1} + \mathbf{w}\cdot\mathbf{x}_{y_i=-1}\right) \tag{4.11}$$

as we know Equation (4.12) and Equation (4.13).

$$\mathbf{x}_i\cdot\mathbf{w} + b = 1 \quad \text{when} \quad y_i = 1 \tag{4.12}$$

$$\mathbf{x}_i\cdot\mathbf{w} + b = -1 \quad \text{when} \quad y_i = -1 \tag{4.13}$$

Function (4.14) then classifies an instance $\mathbf{x}_i$: if $f(\mathbf{x}_i) = 1$, then $\mathbf{x}_i$ is classified as positive, if $f(\mathbf{x}_i) = -1$, then $\mathbf{x}_i$ is classified as negative.

$$f(\mathbf{x}_i) = \operatorname{sign}\left(\mathbf{w}_0\cdot\mathbf{x}_i + b_0\right) \tag{4.14}$$

Substituting $\mathbf{w}_0$ by Equation (4.9) finally results in *classification function* (4.15):

Classification function

$$f(\mathbf{x}_j) = \operatorname{sign}\left(\sum_{i=1}^{l} y_i\alpha_i\left(\mathbf{x}_i\cdot\mathbf{x}_j\right) + b_0\right) \tag{4.15}$$

To construct an optimal hyperplane when T is not linearly separable without classification errors, we introduce $\xi_i \in \mathbb{R}^+, i = 1,\dots,l$. ξ_i allows for misclassification of $\mathbf{x}_i$, i.e. $y_i\left(\mathbf{w}\cdot\mathbf{x}_i + b\right) \geqslant 1 - \xi_i$. Thus, ξ_i introduce a so called *soft margin hyperplane*.

Soft margin hyperplane

In its original formulation C–Support Vector Classification then solves the constrained optimization problem (4.16)

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l}\xi_i \\ \text{subject to} \quad & y_i\left(\mathbf{w}\cdot\mathbf{x}_i + b\right) \geqslant 1 - \xi_i, \\ & \xi_i \geqslant 0, i = 1,\dots,l \end{aligned} \tag{4.16}$$

where $C > 0$ is the *cost factor*, i.e. a regularization parameter. A higher C value emphasizes to classify all training instances correctly, a lower C value corresponds to an optimal hyperplane with a more “flexible” soft margin.

Cost factor C

The Kernel Trick

Another approach to separate nonlinearly separable training data T is the so called *kernel trick*. A function ϕ projects

$\mathbf{x}$ into a higher-dimensional feature space, where linear separability is more likely:

$$\phi : \mathbb{R}^n \mapsto \mathbb{R}^N \qquad (n \ll N) \tag{4.17}$$

An optimal hyperplane is then constructed for $\phi(\mathbf{x})$ instead of $\mathbf{x}$. This changes function (4.9) to function (4.18)

$$\mathbf{w}_0 = \sum_{i=1}^{l} y_i \alpha_i \phi\left(\mathbf{x}_i\right) \tag{4.18}$$

and function (4.15) to function (4.19).

$$f(\mathbf{x}_j) = \text{sign}\left(\sum_{i=1}^{l} y_i \alpha_i \left(\phi\left(\mathbf{x}_i\right) \cdot \phi\left(\mathbf{x}_j\right)\right) + b_0\right) \tag{4.19}$$

Thus, we need to calculate dot-products in $\mathbb{R}^N$. This is not feasible when N is very large or possibly infinite. Therefore, we introduce a *kernel function* $K(\mathbf{u}, \mathbf{v})$ for two instances $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ that fulfills Equation (4.20):

Kernel function

$$K(\mathbf{u}, \mathbf{v}) \equiv \phi\left(\mathbf{u}\right) \cdot \phi\left(\mathbf{v}\right) \tag{4.20}$$

To fulfill Equation (4.20), $K(\mathbf{u}, \mathbf{v})$ has to be a continuous symmetric function and positive definite, i. e. the criteria of *Mercer's theorem* (see Mercer, 1909) have to be met. Then, $K(\mathbf{u}, \mathbf{v})$ implements the mapping (4.17) and function (4.19) becomes function (4.21):

Mercer's theorem

$$f(\mathbf{x}_j) = \text{sign}\left(\sum_{i=1}^{l} y_i \alpha_i K\left(\mathbf{x}_i, \mathbf{x}_j\right) + b_0\right) \tag{4.21}$$

Apart from replacing $\phi\left(\mathbf{u}\right) \cdot \phi\left(\mathbf{v}\right)$ by $K(\mathbf{u}, \mathbf{v})$, the optimization task itself remains unchanged. The most commonly used kernels are the linear kernel, the polynomial kernel, and the Gaussian radial basis function (RBF) kernel:

LINEAR KERNEL. The linear kernel is defined as in Equation (4.22)

$$K\left(\mathbf{x}_i, \mathbf{x}_j\right) = \mathbf{x}_i \cdot \mathbf{x}_j \tag{4.22}$$

i. e. it equals the dot-product of $\mathbf{x}_i$ and $\mathbf{x}_j$.

POLYNOMIAL KERNEL. The polynomial kernel is defined as in Equation (4.23)

$$K\left(\mathbf{x}_i, \mathbf{x}_j\right) = \left(\theta + \mathbf{x}_i \cdot \mathbf{x}_j\right)^d \tag{4.23}$$

where θ is a coefficient and d is the polynomial kernel's degree. When $\theta = 0$ the polynomial kernel is called *homogeneous*, when $\theta > 0$ the polynomial kernel is called *inhomogeneous*.

Homogeneous

Inhomogeneous

GAUSSIAN RBF KERNEL. The Gaussian RBF kernel is defined as in Equation (4.24)

$$K\left(\mathbf{x}_i, \mathbf{x}_j\right) = \exp\left(\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2\right) \tag{4.24}$$

where typically $\gamma = -\frac{1}{\sigma^2}$ and σ is a free parameter.

Due to their flexible nature, specific kernels may not only be applied to vectorial, but also to non-vectorial data, e.g. strings (e.g. Lodhi et al., 2002) or parse trees (e.g. Wiegand and Klakow, 2010).

Asymmetric Cost Factors

This section and the following section briefly describe 2 extensions of SVMs that we use in our thesis: asymmetric cost factors and multi-class SVMs.

To deal with imbalanced numbers of positive and negative training instances, i.e. *class imbalance*, Morik et al. (1999) introduce *asymmetric cost factors* C_+ and C_- to allow for penalizing false positives and false negatives (see Section 4.4) differently. Instead of constrained optimization problem (4.16) they then minimize constrained optimization problem (4.25):

Class imbalance

$$\min_{w,b,\xi} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C_+ \sum_{i:y_i=1} \xi_i + C_- \sum_{i:y_i=-1} \xi_i \tag{4.25}$$

y_i equals 1 if i is a positive example, y_i equals -1 if i is a negative instance. Morik et al. (1999) choose C_+, C_- such that

$$\frac{C_+}{C_-} = \frac{|\{\mathbf{x}_i \mid y_i = -1\}|}{|\{\mathbf{x}_i \mid y_i = 1\}|} \tag{4.26}$$

Multi-class SVMs

Standard SVMs are only able to discriminate between positive and negative instances, i.e. standard SVMs solve binary classification problems where $y_i \in \{-1, 1\}$. k-class classification problems where $y_i \in \{1, \dots, k\}$ and $k > 2$ are either solved directly via k*-class SVMs* or decomposed into several binary classifications problems and then solved via *one-against-the-rest SVMs* or *one-against-one SVMs*:

K-CLASS SVMS Weston and Watkins (1998) generalize the constrained optimization problem (4.16) to k-class classification problems as shown in constrained optimization problem (4.27):

(4.27)

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2}\sum_{m=1}^{k} \|\mathbf{w_m}\|^2 + C\sum_{i=1}^{l}\sum_{m\neq y_i} \xi_i^m \\ \text{subject to} \quad & (\mathbf{w}_{y_i}\cdot \mathbf{x}_i) + b_{y_i} \geqslant (\mathbf{w}_m \cdot \mathbf{x}_i) + b_m + 2 - \xi_i^m \\ & \xi_i^m \geqslant 0, i = 1,\dots,l \qquad m \in \{1,\dots,k\}\setminus y_i \end{aligned}$$

They generalize the decision function (4.14) to k-class classification as shown in decision function (4.28). Decision function (4.28) chooses the class with the largest decision value for an instance $\mathbf{x}_j$:

$$f(\mathbf{x}_j) = \arg\max_k \big(\mathbf{w}_i \cdot \mathbf{x}_j + b_i\big) \qquad i = 1,\dots,k \tag{4.28}$$

For details on the solution of the constrained optimization problem see Weston and Watkins (1998).

ONE-AGAINST-THE-REST SVMS One-against-the-rest or one-against-all SVMs (e. g. Schölkopf et al., 1995) construct k binary decision functions as shown in (4.29):

$$\begin{aligned} f_1(\mathbf{x}_j) &= \big(\mathbf{w}_1 \cdot \mathbf{x}_j + b_1\big) \\ &\vdots \\ f_k(\mathbf{x}_j) &= \big(\mathbf{w}_k \cdot \mathbf{x}_j + b_k\big) \end{aligned} \tag{4.29}$$

The i-th decision function is learned using the instances of class i as positive training instances and the instances of the other $k-1$ classes as negative training instances. Function (4.28) then predicts the class of instance $\mathbf{x}_j$.

ONE-AGAINST-ONE SVMS One-against-one or 1-against-1 SVMs (e. g. Kreßel, 1999) construct $k(k+1)/2$ binary decision functions, one for each class pair (i, j). The (i, j)-th decision function is learned using the instances of class i as positive training instances and the instances of class j as negative training instances.

Max wins strategy

The *Max wins strategy* is then (often) applied to predict the class of $\mathbf{x}_l$ (see Hsu and Lin, 2002): If the (i, j)-th decision function's result is 1 class i receives a vote, if it is -1 class j receives a vote. The class with the most votes is predicted for instance $\mathbf{x}_l$.

4.1.3 *Naïve Bayes Classifier*

The NB classifier is a supervised, generative ML method. Let $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ be our training instances with $\mathbf{x}_i \in \mathbb{R}^n$ an n-dimensional input vector and $y_i \in Y$ a class label from a finite set (see Mitchell, 1997, p. 177). Given a new instance $\mathbf{x}_j$, we want to compute $p(y_k \mid \mathbf{x}_j)$ to decide for the most likely—i. e. *maximum a posteriori (MAP)*—class y_{map}:

Maximum a posteriori

$$y_{map} \text{ if } p(y_{map} \mid \mathbf{x}_j) > p(y_k \mid \mathbf{x}_j) \text{ for } y_{map} \neq y_k \tag{4.30}$$

As we usually do not know $p(y_k \mid \mathbf{x}_j)$, we apply *Bayes' theorem* as shown in Equation (4.31)

Bayes' theorem

$$p(y_k \mid \mathbf{x}_j) = \frac{p(\mathbf{x}_j \mid y_k) \cdot p(y_k)}{p(\mathbf{x}_j)} \tag{4.31}$$

where $p(y_k)$ can be thought of as *prior probability*, which is updated by the *evidence* $\frac{p(\mathbf{x}_j \mid y_k)}{p(\mathbf{x}_j)}$, which then yields the *posterior probability* $p(y_k \mid \mathbf{x}_j)$ (see Manning and Schütze, 1999, p. 236).

Prior probability

Evidence

Posterior probability

If we are not interested in the posterior probability but simply want to predict the most likely class y', we can drop the denominator $p(\mathbf{x}_j)$ as it is independent of y_k as shown in Equation (4.32) and work with logarithms.

$$\begin{aligned} y_{map} &= \arg\max_{y_k} p(y_k \mid \mathbf{x}_j) \\ &= \arg\max_{y_k} \frac{p(\mathbf{x}_j \mid y_k) \cdot p(y_k)}{p(\mathbf{x}_j)} \\ &= \arg\max_{y_k} p(\mathbf{x}_j \mid y_k) \cdot p(y_k) \\ &= \arg\max_{y_k} \left[\log p(\mathbf{x}_j \mid y_k) + \log p(y_k)\right] \end{aligned} \tag{4.32}$$

The *NB assumption* states

NB assumption

$$p(\mathbf{x}_j \mid y_k) = \prod_n p(\mathbf{x}_{j,n} \mid y_k) \tag{4.33}$$

i. e. all attributes are independent of each other. Combining Equation (4.32) and Equation (4.33) then yields the *NB decision rule* shown in Equation (4.34):

NB decision rule

$$y_{map} = \arg\max_{y_k} \left[\sum_n \log p(\mathbf{x}_{j,n} \mid y_k) + \log p(y_k)\right] \tag{4.34}$$

The probabilities $p(\mathbf{x}_{j,n} \mid y_k)$ and $p(y_k)$ are approximated via relative frequencies in the training data as shown in Equation (4.35) and Equation (4.36), respectively:

$$p(\mathbf{x}_{j,n} \mid y_k) = \frac{c(\mathbf{x}_{j,n}, y_k)}{\sum_m c(\mathbf{x}_{j,m}, y_k)} \tag{4.35}$$

$$p(y_k) = \frac{c(y_k)}{\sum_i c(y_i)} \tag{4.36}$$

$c(\mathbf{x}_{j,n}, y_k)$ is the number of times $\mathbf{x}_{j,n}$ appears for y_k. $c(y_k)$ is the number of training instances of class k. Both Equation (4.35) and Equation (4.36) are *maximum likelihood* estimates (see Manning and Schütze, 1999, p. 237).

Maximum likelihood

4.1.4 *Decision Trees*

DT induction is a supervised, discriminative ML method that usually approximates discrete-valued target functions. DT induction is robust to noise in the training data (see Mitchell, 1997, p. 52).

In a DT each *node* tests some attribute of some instance. For each possible value of this attribute a *branch* descends from that node. If a node does not branch, i. e. has no *subtree*, it is called a *leaf*. Each leaf provides a classification label. A DT classifies an instance by starting at its *root* node, testing the attribute of the root node, and following the corresponding branch. This is repeated for the node that has been reached. Reaching a leaf, the instance is classified accordingly (see Mitchell, 1997, pp. 52–53). A DT can always be represented as a disjunction of conjunctions of constraints on the attribute values of the instances (see Mitchell, 1997, p. 53)—in other words, as a set of *if-then rules*.

Node
Branch
Subtree
Leaf
Root
If-then rules

While there are numerous algorithms for DT induction, we present one of the most widely used: C4.5. C4.5 (see Quinlan, 1993) is based on ID3 (see Quinlan, 1986), which we present first.

C4.5
ID3

ID3

ID3 induces DTs top-down as follows (see Mitchell, 1997, pp. 52–56): ID3 greedily[1] searches the space of all possible DTs. Starting at the root node, ID3 decides which attribute classifies the remaining training instances best. For each resulting branch it does the same recursively. ID3 never reconsiders previous choices. To decide which attribute classifies the remaining training instances best, ID3 uses an Information Gain (IG) (see Section 4.3.1) criterion. IG with respect to an attribute A is defined as in Equation (4.37)

$$IG_A(T) = H(T) - \sum_{v \in V(A)} \frac{|T_v|}{|T|} H(T_v) \tag{4.37}$$

where $H(T)$ is the entropy (see Chapter 3.2.1) of training data T, $V(A)$ are the values of A, and $T_v \subset T$ is a subset of T in which A has value v.

For each decision only training instances are included which have the attribute values associated with the path to the current node. Furthermore, attributes that have been used higher up in the DT are excluded from using them again. Thus, each attribute can be used only once on each path. The DT is grown until either all remaining training instances at the current node have the same classification label—T_v has zero entropy—or each attribute has been used on this path. This node then becomes a leaf associated with the classification label of the majority of the remaining training instances.

C4.5

There are several issues with ID3, e. g. handling continuous attributes and avoiding to overfit the training data, which are—among others—addressed in extensions to ID3 that result in C4.5.

AVOIDING OVERFITTING Generally, there are 2 ways to avoid overfitting in DT induction: either stop growing the DT before it perfectly fits the training data or allow the DT to perfectly fit the training data but then post-prune it. *Pruning* removes a subtree of a node, thereby makes it a

Pruning

[1] For each decision, a *greedy* algorithm "makes a locally optimal choice in the hope that this choice will lead to a globally optimal solution." (see Cormen et al., 2009, p. 414)

leaf and assigns to it the classification label most common among its remaining training instances (see Mitchell, 1997, p. 70). Pruning is an iterative process that chooses nodes whose removal increases accuracy most, e. g. over a held-out validation set. Pruning stops when it harms accuracy.

C4.5 performs rule post-pruning: After DT induction C4.5 converts the DT into an equivalent set of if-then rules, in which one rule corresponds to one path in the DT. Each rule is then pruned by removing preconditions—if-parts—which increase its estimated accuracy. C4.5 pessimistically estimates its accuracy on the training set instead of using a held-out validation set (see Mitchell, 1997, pp. 71–72). The rules are then sorted by their estimated accuracy and considered as a sequence for classification.

HANDLING CONTINUOUS ATTRIBUTES Continuous attribute values are addressed by dynamically introducing new boolean attributes A_c that are true if $A < c$ and false otherwise (see Mitchell, 1997, p. 72). The threshold c is chosen to maximize IG. As IG is biased towards attributes with many values, C4.5 additionally penalizes such attributes.

4.1.5 *Linear Regression*

Response

Predictor

LR is a supervised ML method for predicting a real-valued output or *response* y based on *predictors*—an input vector $\mathbf{x}^T = (x_1, x_2, \dots, x_n)$ (see Hastie et al., 2009, p. 44). An LR model has the form

$$y = f(\mathbf{x}) = \beta_0 + \sum_{j=1}^{n} x_i \beta_j \tag{4.38}$$

Intercept

Coefficient

where β_0 is the *intercept* and $\beta_1, \beta_2, \dots, \beta_n$ are *coefficients*. Given a training set $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$, we estimate the parameters $\beta^T = (\beta_0, \beta_1, \dots, \beta_n)$. This is typically done by minimizing the residual sum (4.39)

$$\begin{aligned} RS(\beta) &= \sum_{i=1}^{l} L\left(y_i, f\left(\mathbf{x}_i\right)\right) \\ &= \sum_{i=1}^{l} L\left(y_i, \beta_0 + \sum_{j=1}^{n} x_{ij}\beta_j\right) \end{aligned} \tag{4.39}$$

Loss function

where $L\left(y_i, f\left(\mathbf{x}_i\right)\right)$ is a *loss function*, e. g. the squared error loss (4.40).

Loss Functions

A loss function $L(y_i, f(\mathbf{x}_i))$ allows for penalizing errors in prediction. By far the most common loss function in LR is squared error loss (see Hastie et al., 2009, p. 18).

SQUARED ERROR LOSS Squared error loss is defined as in Equation (4.40)

$$L(y_i, f(\mathbf{x}_i)) = (y_i - f(\mathbf{x}_i))^2 \tag{4.40}$$

where $y_i - f(\mathbf{x}_i) = y_i - \hat{y}_i$ are the *residuals*. Squared error loss may be dominated by outliers because $\sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i))$ may be heavily influenced by a few large $y_i - f(\mathbf{x}_i)$ values.

Residual

HUBER LOSS Huber loss (see Huber, 1964) is defined as in Equation (4.41)

$$L_\delta(y_i, f(\mathbf{x}_i)) = \begin{cases} \frac{1}{2}a_i^2 & \text{if } |a_i| \leqslant \delta \\ \delta(|a_i| - \frac{1}{2}\delta) & \text{else} \end{cases} \tag{4.41}$$

where $a_i = y_i - f(\mathbf{x}_i)$. Holland and Welsch (1977) choose $\delta = 1.345$ for a 95% asymptotic efficiency at the Gaussian distribution. In contrast to squared error loss, Huber loss does not suffer from the heavy influence of a few large a_i values and thus is more robust against outliers.

TUKEY'S BIWEIGHT Tukey's biweight or bisquare (see Holland and Welsch, 1977) is defined as in Equation (4.42)

$$L_\delta(y_i, f(\mathbf{x}_i)) = \begin{cases} a_i\left(1 - \left(\frac{a_i}{\delta}\right)^2\right)^2 & \text{if } |a_i| \leqslant \delta \\ 0 & \text{else} \end{cases} \tag{4.42}$$

where $a_i = y_i - f(\mathbf{x}_i)$. Holland and Welsch (1977) choose $\delta = 4.685$ for a 95% asymptotic efficiency at the Gaussian distribution. Just as Huber loss Tukey's biweight does not suffer from the heavy influence of a few large a_i values and thus is more robust against outliers.

Both Huber loss and Tukey's biweight are loss functions for M-estimators. Robust regression using Huber loss and Tukey's biweight are usually computed using iteratively reweighed least squares (see Holland and Welsch, 1977).

4.1.6 *Logistic Regression*

LogReg is a supervised, discriminative ML method based on LR (see Section 4.1.5) for predicting a categorical output instead of a real-valued one. The LR model (4.38) is transformed via the *logistic function* (4.43)

Logistic function

$$g(z) = \frac{1}{1+\exp(-z)} \tag{4.43}$$

resulting in the LogReg model (4.44):

$$\begin{aligned} y &= g\left(f\left(\mathbf{x}\right)\right) \\ &= \frac{1}{1+\exp\left(-\left(\beta_0 + \sum_{j=1}^{n} x_i\beta_j\right)\right)} \end{aligned} \tag{4.44}$$

A LogReg model's parameters β are usually fitted via maximum likelihood estimation (see Hastie et al., 2009, pp. 120–122).

4.2 FEATURE SCALING

When the j-th feature of $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ is not binary—i. e. either 0 or 1—it needs to be *scaled* (see Hsu et al., 2003), ideally in way such that

$$-1 \leqslant \mathbf{x}_k^j \leqslant 1 \tag{4.45}$$

We ensure Inequality (4.45) by normalizing $\mathbf{x}_k^j$ such that it has approximately zero mean. We do that by replacing $\mathbf{x}_k^j$ as shown in Equation (4.46):

$$\mathbf{x}_k^j \leftarrow \frac{\mathbf{x}_k^j - \mu^j}{s^j} \tag{4.46}$$

where μ^j is the j-th feature's distribution mean and s^j is its standard deviation.

4.3 FEATURE SELECTION METHODS

Methods for *FS*—or variable subset selection—retain only a subset of all features in some model while they eliminate the rest (see Hastie et al., 2009, p. 57). By removing redundant or irrelevant features, FS facilitates model interpretability and increases the model's ability to generalize by reducing overfitting (see Hastie et al., 2009, p. 57). Moreover, less complex models are usually faster to train and test. We now present two FS methods: IG (see Section 4.3.1) and χ^2 Test (see Section 4.3.2).

4.3.1 *Information Gain*

IG "measures the amount of information in bits about the class prediction, if the only information available is the presence [or absence] of a feature and the corresponding class distribution" (see Roobaert et al., 2006). More concretely, IG measures the expected decrease in entropy, i. e. the decrease in uncertainty associated with a certain random variable (see Mitchell, 1997, p. 57).

Applied to text classification, Yang and Pedersen (1997) define the IG of a term t regarding m classes $\{c_i\}_{i=1}^m$ as shown in Equation (4.47)

$$\begin{aligned} IG(t) = &- \sum_{i=1}^{m} p(c_i) \log_2 p(c_i) \\ &+ p(t) \sum_{i=1}^{m} p(c_i \mid t) \log_2 p(c_i \mid t) \\ &+ p(\bar{t}) \sum_{i=1}^{m} p(c_i \mid \bar{t}) \log_2 p(c_i \mid \bar{t}) \end{aligned} \tag{4.47}$$

where $p(t)$ is the probability of t and $p(\bar{t}) = 1 - p(t)$ is the probability of not t. The conditional probabilities are calculated using Bayes' theorem as shown in Equation (4.48)

$$p(c_i \mid t) = \frac{p(t \mid c_i) \cdot p(c_i)}{p(t)} \tag{4.48}$$

and analogous for $\bar{t}$.

4.3.2 χ^2 *Test*

χ^2 test is a statistical test that measures divergence between an observed and an expected distribution (see Forman, 2003). Applied to text classification, Yang and Pedersen (1997) define the χ^2 value of t regarding c as shown in Equation (4.49)

$$\chi^2(t, c) = \frac{N \cdot (AD - CB)^2}{(A + C) \cdot (B + D) \cdot (A + B) \cdot (C + D)} \tag{4.49}$$

Figure 4.: A 2×2 contingency matrix.

		Actual	
		c_i	$\neg c_i$
Predicted	c_i	tp_i	fp_i
	$\neg c_i$	fn_i	tn_i

where A is the number of times t occurs with c, B is the number of times t occurs not with c, C is the number of times c occurs not with t, D is the number of times neither c or t occur, and N is the number of documents. Yang and Pedersen (1997) then average $\chi^2(t, c)$ as shown in Equation (4.50):

$$\chi^2_{avg}(t) = \sum_{i=1}^{m} p(c_i) \cdot \chi^2(t, c_i) \tag{4.50}$$

4.4 EVALUATION MEASURES

In a multi-class classification setting, given n classes $C = \{c_1, c_2, \ldots, c_n\}$ as well as cases with their predicted and actual class, we can build 2×2 contingency matrices for every class c_i as shown in Figure 4: The cases in which the predicted class equals the actual class are called tp (*true positives*) and tn (*true negatives*). The cases in which the predicted class does not equal the actual class are called fp (*false positives*) and fn (*false negatives*).

True positives

True negatives

False positives

False negatives

For a single class c_i the *precision* P_i is then defined as in Equation (4.51):

Precision

$$P_i = \frac{tp_i}{tp_i + fp_i} \tag{4.51}$$

The *recall* R_i for c_i is defined as in Equation (4.52):

Recall

$$R_i = \frac{tp_i}{tp_i + fn_i} \tag{4.52}$$

We can combine precision with recall into *F measure* or F score, which is based on Van Rijsbergen (1979)'s *E measure* (see Manning and Schütze, 1999, p. 269). F measure F_i for c_i is defined as in Equation (4.53)

F measure

$$F_i = \frac{1}{\alpha\frac{1}{P_i} + (1 - \alpha)\frac{1}{R_i}} \tag{4.53}$$

Figure 5.: A 2×2 contingency matrix for micro-averaging.

		Actual	
		c_i	$\neg c_i$
Predicted	c_i	$\sum_{c_i} tp_i$	$\sum_{c_i} fp_i$
	$\neg c_i$	$\sum_{c_i} fn_i$	$\sum_{c_i} tn_i$

where α is a factor that determines the relative weighting of P_i and R_i. The larger α, the larger the influence of precision, and the smaller the influence of recall and vice versa. If $\alpha = 0.5$ Equation (4.53) simplifies to the harmonic mean of P_i and R_i as shown in Equation (4.54):

$$F_i = \frac{2P_iR_i}{P_i + R_i} \tag{4.54}$$

Another important evaluation measure for classification is *accuracy*. Accuracy A_i for c_i is defined as in Equation (4.55):

Accuracy

$$A_i = \frac{tp_i}{tp_i + fp_i + fn_i} \tag{4.55}$$

By convention, accuracy is usually presented in percent. E. g., an accuracy of 0.857 is written as 85.7. We follow this convention in our thesis.

We can calculate an overall performance measure M—e. g. overall accuracy A—either via *macro-averaging* or *micro-averaging*. In macro-averaging, one calculates an evaluation measure M_i for every class c_i and then calculates M as shown in Equation (4.56):

Macro-averaging
Micro-averaging

$$M = \frac{1}{n}\sum_{i=1}^{n} M_i \tag{4.56}$$

In micro-averaging, one first calculates an overall 2×2 contingency matrix as shown in Figure 5 and then calculates M based on this contingency matrix. Micro-averaging gives equal weight to every case, thus it is more dominated by large classes. Macro-averaging gives equal weight to every class, thus it is more dominated by small classes (see Manning and Schütze, 1999, p. 577). By default we use macro-averaging.

4.5 MODEL ASSESSMENT TECHNIQUES

The most widely used method for assessing a model's prediction accuracy, precision, recall, or F measure—e. g. for evaluation or parameter tuning—is *cross validation (CV)*. In this thesis we use K-fold CV (see Section 4.5.1) and leave-one-out CV (see Section 4.5.2).

Cross validation

4.5.1 *K-fold Cross Validation*

In a *K-fold CV* (see Hastie et al., 2009, pp. 241–242) we split the data into K parts. For $k = 1, \ldots, K$ we fit a model to $K - 1$ parts of the data and validate it on the held out k-th part. The K results are then averaged.

4.5.2 *Leave-one-out Cross Validation*

Leave-one-out CV (see Hastie et al., 2009, p. 242) is a special case of a K-fold CV, where $K = n$ and n is the number of instances in the data: We fit a model to $n - 1$ parts of the data and validate it on the held out n-th part. The n results are then averaged.

For estimating a model's prediction accuracy leave-one-out CV has low variance but may be biased. In contrast, K-fold CV is approximately unbiased but may have high variance (see Hastie et al., 2009, pp. 242–243).

Part II

SENTIMENT ANALYSIS IN DIFFERENT GENRES AND DOMAINS

5 GENRE AND DOMAIN DEPENDENCIES

In this chapter we verify our core hypothesis: SA is—just like NLP in general—genre and domain dependent.

We first describe our approach to SA in Section 5.1. In Section 5.2 we show that our SA approach performs differently when applied to different genres and domains. In Section 5.3 we show that different genres and domains differ in their textual characteristics, viz. their domain complexity. Finally, we relate differences in textual characteristics to differences in performance in Section 5.4[1].

5.1 APPROACH TO SENTIMENT ANALYSIS

This section describes our general approach to the SA subtasks that this thesis focuses on: polarity and subjectivity classification. We describe its underlying assumptions (see Section 5.1.1), its text representation (see Section 5.1.2), and our classifier choice (see Section 5.1.3).

5.1.1 *Assumptions*

Sentiment in text is—directly or indirectly—expressed in words or phrases, i. e. sequences of words. In Example (18) a. and Example (18) b. sentiment is expressed by the phrases "truly love" and "truly loves", respectively.

(18) a. I truly love wearing them.
b. She truly loves wearing them, too.

Word n-grams

Therefore, we capture sentiment via *word n-grams*. A word n-gram is a sequence of words of length n. Table 8 lists the word n-grams of Example (18a.) for $1 \leqslant n \leqslant 4$.

Text classification

Polarity and subjectivity classification may both be seen as instances of *text classification* or text categorization. Text classification attempts to assign e. g. a document or a sentence to at least one of two or more predefined classes or categories (see Manning and Schütze, 1999, p. 530). In the

[1] Section 5.4 is based on Remus and Ziegelmayer (2014).

[2] Tetragrams are also referred to as quad-, four-, or 4-grams.

Table 8.: Word n-grams of "I truly love wearing them.".

n	NOTATION	WORD n-GRAMS
1	unigrams	I, truly, love, wearing, them, .
2	bigrams	I truly, truly love, love wearing, wearing them, them .
3	trigrams	truly love, truly love wearing, love wearing them, wearing them .
4	tetragrams[2]	I truly love wearing, truly love wearing them, love wearing them .

Figure 6.: A bag-of-words of "I truly love wearing them." and "She truly loves wearing them, too.".

$$\left\{ \begin{array}{l} \text{I, truly, truly, love, wearing, wearing,} \\ \text{them, them, ., ., She, loves, ,, too} \end{array} \right\}$$

NLP research community the dominant approach to text classification is based on ML (see Sebastiani, 2002).

5.1.2 *Text Representation*

Bag-of-words

To apply ML algorithms to a text, it needs to be represented appropriately. We represent a text as a *bag-of-words* (Manning and Schütze, 1999, p. 237). We consider a bag-of-words as a multiset of word n-grams. A multiset disregards the order of its elements. While such a representation sacrifices information regarding a text's structure, it is invariant—thus robust—with respect to the positions of words and phrases. Therefore, bags-of-words have good generalization capabilities. Furthermore, the bag-of-words representation scheme is easily applicable to texts of different granularity: documents, sentences, or even phrases. A bag-of-words of Example (18) is shown in Figure 6.

Vocabulary
Dictionary

Given a *vocabulary*—a *dictionary*—that contains k unique word n-grams, we then encode a text as a k-dimensional vector $\mathbf{x} \in \mathbb{R}^k$. The i-th dimension of $\mathbf{x}$, i. e. $\mathbf{x}_i$, then encodes the absence (0) or presence (1) of the i-th vocabulary entry in the text. Generally, one could also encode a vocabulary entry's relative frequency in the text, or its tf-idf (see Manning and Schütze, 1999, p. 543) etc., but encoding an

$$\begin{array}{r}\text{love}\\ \text{loves}\\ \text{them}\\ \text{too}\\ \text{truly}\\ \text{wearing}\\ ,\\ .\\ \text{I}\\ \text{She}\end{array}\begin{pmatrix}1\\0\\1\\0\\1\\1\\0\\1\\1\\0\end{pmatrix}$$

Figure 7.: A vector encoding of "I truly love wearing them." in $\{0, 1\}^k$.

entry's absence or presence yields superior performance in polarity classification compared to schemes that encode an entry's frequency (see Pang et al., 2002). So instead of in $\mathbb{R}^k$ we encode vectors in $\{0, 1\}^k$. Figure 7 shows a vector encoding of Example (18a.) given a vocabulary that contains "love", "loves", "them", "too", "truly", "wearing", ",", ".", "I", "She", i. e. all unique word unigrams of Example (18).

5.1.3 *Classifier Choice*

While there is an abundance of discriminative and generative ML algorithms (see Chapter 4.1) available to build classifiers, SVMs (see Chapter 4.1.2) are a common choice for text classification. This is because SVMs can handle (i) high-dimensional feature spaces, (ii) many relevant features, and (iii) sparse feature vectors well (see Joachims, 1998).

To confirm that SVMs are in fact an adequate choice for text classification—and specifically for the SA subtasks polarity and subjectivity classification—we carry out preliminary experiments in which we compare SVMs to 3 other commonly used ML algorithms: NB classifiers (see Chapter 4.1.3), pruned DTs induced by C4.5 (see Chapter 4.1.4), and LogReg (see Chapter 4.1.6). All 4 ML algorithms are evaluated in a document-level polarity classification experiment on PD v2.0, in a sentence-level polarity classification experiment on SPD v1.0 as well as sentence-level subjectivity classification experiments on SD v1.0 and MPQA v2.0 (see

Table 9.: Accuracy of different ML algorithms on several SA gold standards.

GOLD STANDARD	SVM	NB	DT	LOGREG
MPQA v2.0	72.25	66.62	65.46	64.23
PD v2.0	86.6	64.35	71.25	81.85
SD v1.0	90.56	82.89	82.23	90.98
SPD v1.0	76.21	62.66	65.19	72.05
average	81.41	69.13	71.03	77.27

Table 10.: Accuracy of SVM models with (w/) and without (w/o) optimization of C on several SA gold standards.

GOLD STANDARD	W/	W/O
MPQA v2.0	72.25	69.43
PD v2.0	86.6	75.35
SD v1.0	90.56	89.3
SPD v1.0	76.21	74.56
average	81.41	77.16

Chapter 2.1). We use solely word unigrams as features. Table 9 shows the results on these 4 gold standards.

SVMs proved to be superior in all our experiments and also in comparative experiments of others (e. g. Pang et al., 2002; Go et al., 2009). Hence, SVMs are used for learning classifiers throughout this thesis.

Optimization of C

If not stated otherwise we use SVMs in their LibSVM implementation[3] with a linear kernel and their cost factor C chosen from {2.0E-3, 2.0E-2, 2.0E-1, 2.0, 2.0E1, 2.0E2, 2.0E3} via 10-fold CV on the training data (see Hsu et al., 2003). Optimization of C is time-consuming but usually leads to clear performance gains: Table 10 compares SVMs with optimization of C to SVMs without optimization of C on the same gold standards as before. When C is not optimized—but fixed to a medium value of 2.0—its accuracy on the aforementioned gold standards is on average 4.25 points lower compared to when C is optimized.

[3] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

In all our experiments throughout this thesis we represent texts as vector encodings of ordinary bags-of-words. The vocabularies—the features—are *induced* using a *data-driven* approach as described in Remus and Rill (2013). We do not make any assumption about which word n-grams carry sentiment and which do not. Therefore, we do not perform any (naïve) FS—we keep high- and low-frequency word n-grams just as we keep punctuations etc. From hereon we refer to this approach as *our SA approach*—or interchangeably *our SA method*.

Data-driven feature induction

Our SA approach

Our SA method

5.2 DIFFERENCES IN PERFORMANCE

Although genre and domain dependence may be distinguished from each other, they pose a similar problem to NLP in general and to SA specifically: the same method performs differently when applied to different genres and different domains.

We show that the same SA method—our approach as described in Section 5.1 using word unigrams as features—performs differently when applied to different genres and different domains. To that end, we carry out document-level polarity classification experiments, in which we compare results of our SA approach on gold standards (or subsets of gold standards) that originate from

1. the same genre and the same domain,
2. the same genre but a different domain,
3. a different genre but the same domain,
4. a different genre and different domain.

Consequently, we consider these gold standards as representative samples of their respective genres and domains. We hypothesize that the differences in performance are considerably smaller when varying neither genre nor domain compared with when varying the domain but not the genre. They are even larger when varying genre but not domain. And they are largest when varying both the genre and the domain (see Table 11).

Table 11.: Hypothesized differences in both performance and textual characteristics when varying either, both or neither genre and domain.

GENRE	DOMAIN	HYPOTHESIZED DIFFERENCE
not varied	not varied	"small"
not varied	varied	"medium"
varied	not varied	"medium" to "large"
varied	varied	"medium" to "large"

5.2.1 *Varying neither Genre nor Domain*

Before we investigate the differences in performance of our SA method when we vary either or both genre and domain, we investigate its behavior when we vary neither genre nor domain: we run experiments on RND (see Section 2.1.5).

For each of the 9 domains in RND we randomly divide the 2,000 available positive reviews in 2 runs of 1,000 positive reviews each; we do the same for the 2,000 available negative reviews. This experimental setup resembles a "treatment" group and a control group. We then evaluate our SA method on both runs in a 10-fold CV. Figure 8 shows the evaluation results. As hypothesized in Table 11 the average difference in performance between the domains in run 1 and run 2—i. e. when varying neither genre nor domain—is "small": 1.03±0.86 (minimum 0.1; maximum 2.25).

5.2.2 *Varying Domain*

We now investigate our SA method's behavior when we vary the domain but not the genre: we run experiments on MDSD v2.0 (see Section 2.1.2) and RND.

For each of the 10 domains in MDSD v2.0 we use all the available 1,000 positive and 1,000 negative reviews to evaluate our SA method in a 10-fold CV. Figure 8 and Figure 9 show the evaluation results. As hypothesized in Table 11 the average pairwise difference in performance between MDSD v2.0's domains—i. e. when varying domain but not genre—is "medium": 2.17±1.62 (0.1; 6.85). It is twice as large compared to when we vary neither genre nor domain (see Section 5.2.1).

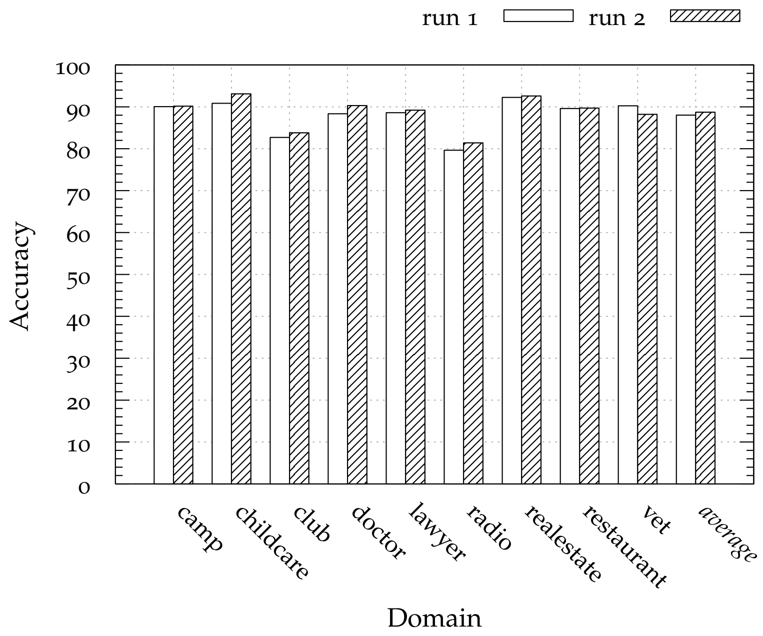


Figure 8.: Differences in performance of our SA method on RND when varying neither genre nor domain as well as when varying the domain but not the genre.

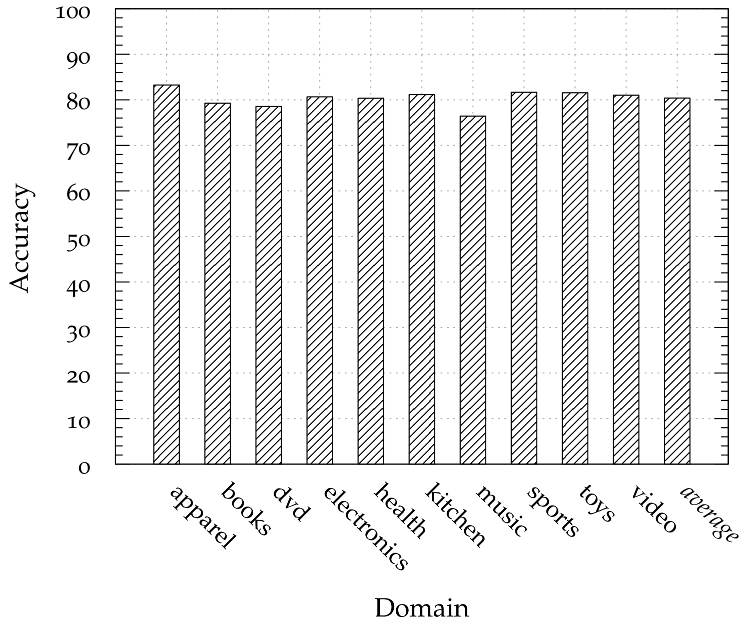


Figure 9.: Differences in performance of our SA method on MDSD v2.0 when varying the domain but not the genre.

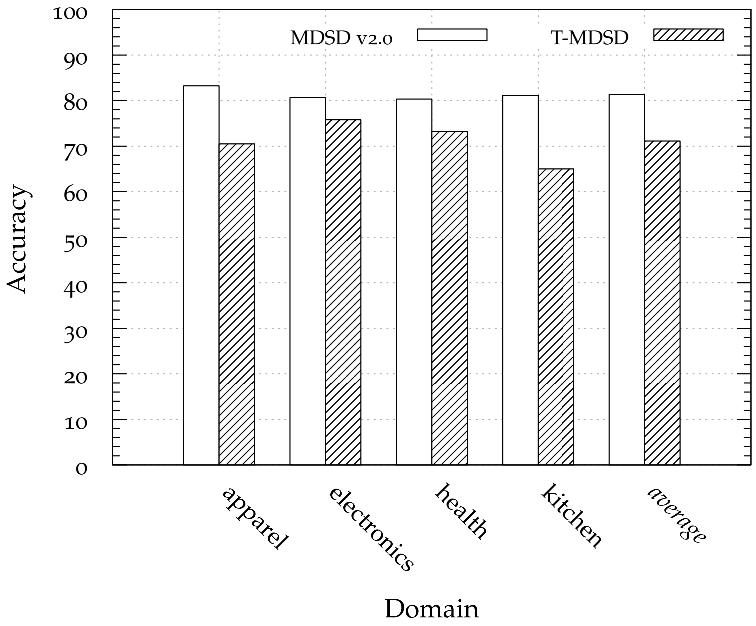


Figure 10.: Differences in performance of our SA method on MDSD v2.0 and T-MDSD when varying the genre but not the domain as well as when varying both genre and domain.

The average pairwise differences in performance between RND's domains are even larger: 4.58±3.54 (0.1; 12.6) for run 1 and 4±3.87 (0; 11.25) for run 2 (see Section 5.2.1). They are four times as large compared to when we vary neither genre nor domain (see Section 5.2.1).

5.2.3 *Varying Genre*

We now investigate our SA method's behavior when we vary the genre but not the domain: we run experiments on MDSD v2.0 and T-MDSD (see Section 2.1.10).

For each of the 4 domains found both in MDSD v2.0 and T-MDSD—apparel, electronics, health & personal care, and kitchen & housewares—we use all the available 1,000 positive and 1,000 negative reviews and tweets to evaluate our SA method in a 10-fold CV. Figure 10 shows the evaluation results. As hypothesized in Table 11 the average pairwise difference in performance between MDSD v2.0's reviews and T-MDSD's tweets for equal domains—i. e. when varying genre but not domain—is "large": 10.23±5.16 (4.85; 16.15). It is at least twice as large compared to when we vary the domain but not the genre (see Section 5.2.2).

5.2.4 *Varying Genre and Domain*

Finally, we investigate our SA method's behavior when we vary both genre and domain. We run experiments as described in Section 5.2.3.

Figure 10 shows the evaluation results. As hypothesized in Table 11 the average pairwise difference in performance between MDSD v2.0's reviews and T-MDSD's tweets for unequal domains—i. e. when varying genre and domain—is "large": 10.67±3.84 (5.35; 18.25). It is slightly larger compared to when we vary genre but not domain (see Section 5.2.3).

5.3 DIFFERENCES IN TEXTUAL CHARACTERISTICS

Intuitively, different genres and domains differ in their vocabulary and in the way their vocabulary is used, i. e. they differ in their textual characteristics. To confirm our intuition, we measure differences in domain complexity of the same datasets as chosen in Section 5.2, i. e. of gold standards (and subsets of gold standards) that originate from

1. the same genre and the same domain,
2. the same genre but a different domain,
3. a different genre but the same domain,
4. a different genre and different domain.

We approximate domain complexity using percentage of rare words, word richness[4], and relative entropy (see Chapter 3.2.1) as well as homogeneity (see Chapter 3.2.2). In this section our domain complexity approximations are computed on word unigrams.

Analogously to Section 5.2 we hypothesize that the differences in domain complexity are considerably smaller when varying neither genre nor domain compared with when varying the domain but not the genre. They are even larger when varying the genre but not the domain. And they are largest when varying both genre and domain (see Table 11).

[4] From hereon we refer to word richness by its more common notation: type/token ratio.

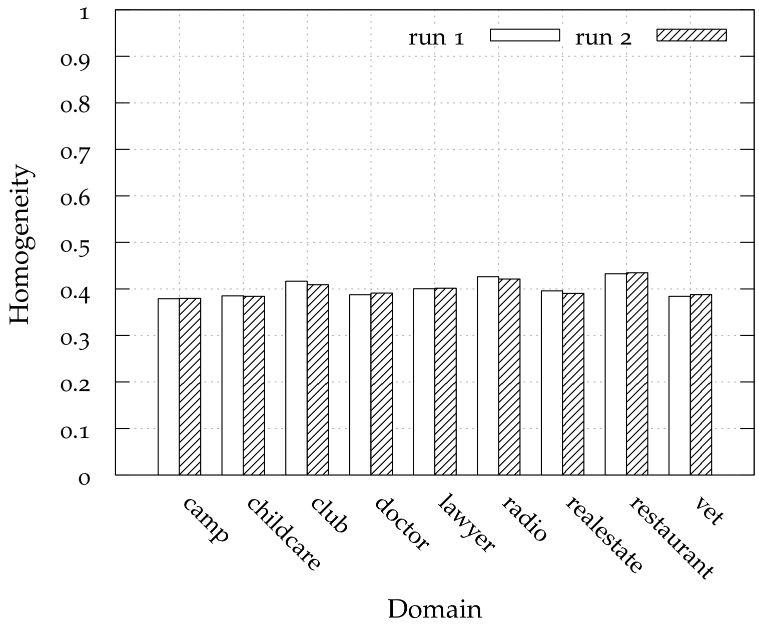


Figure 11.: Differences in homogeneity of RND when varying neither genre nor domain.

5.3.1 *Varying neither Genre nor Domain*

When we vary neither genre nor domain as in Section 5.2.1 the differences in textual characteristics, viz. domain complexity, are small. Exemplarily, Figure 11 shows the differences in homogeneity.

The average difference in homogeneity between run 1 and run 2 is 0.003±0.003 (0.001; 0.008). Approximately the same applies to percentage of rare words, type/token ratio, and relative entropy.

5.3.2 *Varying Domain*

When we vary the domain but not the genre as in Section 5.2.2 the differences in domain complexity are larger compared with when we vary neither domain nor genre (see Section 5.3.1). Exemplarily, Figure 12 shows the differences in homogeneity.

The average pairwise difference in homogeneity between MDSD v2.0's domains is 0.032±0.021 (0.002; 0.077). In percentage of rare words it is 0.002±0.002 (0; 0.005). In type/-token ratio it is 0.016±0.012 (0; 0.034). In relative entropy it is 0.024±0.018 (0; 0.05).

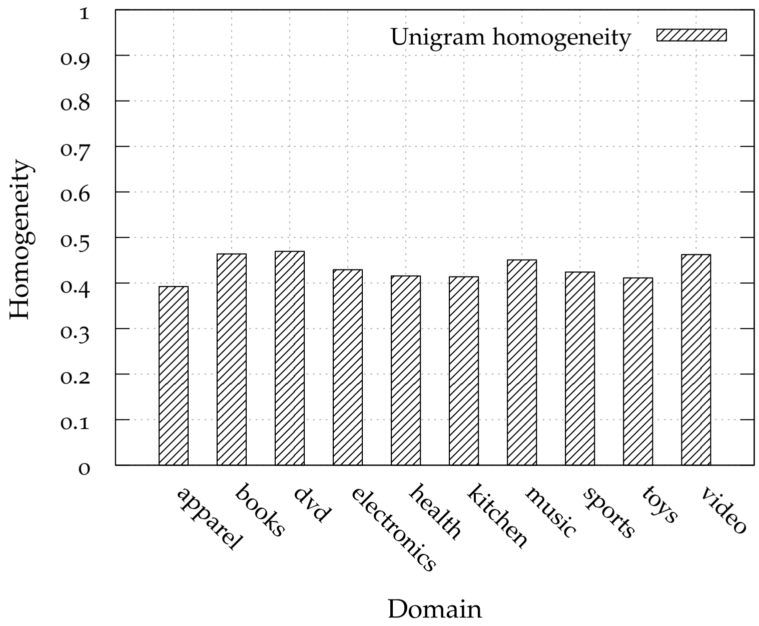


Figure 12.: Differences in homogeneity of MDSD v2.0 when varying the domain but not the genre.

5.3.3 *Varying Genre*

When we vary the genre but not the domain as in Section 5.2.3 the differences in domain complexity are larger compared with when we do not vary the genre but the domain (see Section 5.3.2). Exemplarily, Figure 13 shows the differences in homogeneity.

The average pairwise difference in homogeneity between MDSD v2.0's reviews and T-MDSD's tweets for equal domains is 0.035±0.023 (0.012; 0.067). In percentage of rare words average it is 0.021±0.014 (0.003; 0.036). In type/token ratio it is 0.072±0.042 (0.014; 0.111). In relative entropy it is 0.067±0.009 (0.059; 0.08).

5.3.4 *Varying Genre and Domain*

When we vary both genre and domain as in Section 5.2.4 the differences in domain complexity are on par compared with when we vary the genre but not the domain (see Section 5.3.3). Exemplarily, Figure 13 shows the differences in homogeneity.

The average pairwise difference in homogeneity between MDSD v2.0's reviews and T-MDSD's tweets for unequal do-

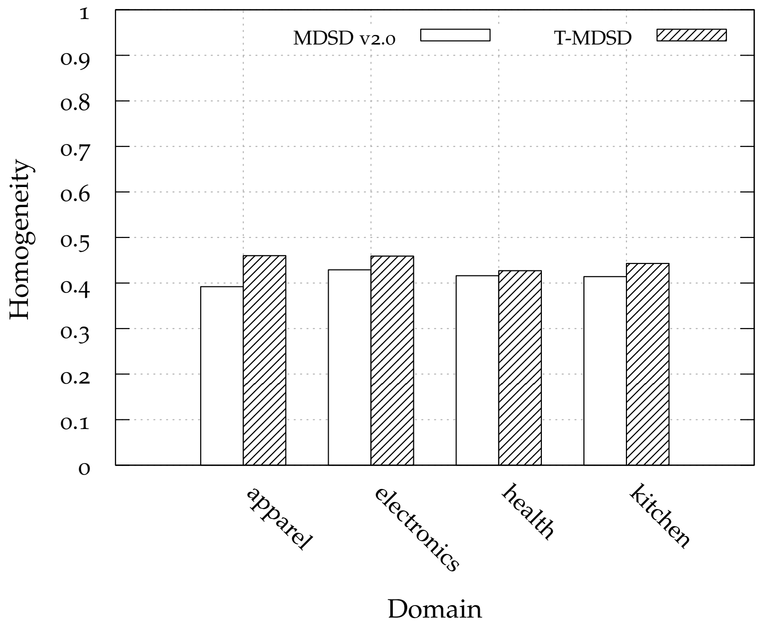


Figure 13.: Differences in homogeneity of MDSD v2.0 and T-MDSD when varying the genre but not the domain as well as differences in homogeneity when varying both genre and domain.

mains is 0.035±0.018 (0.002; 0.067). In percentage of rare words it is 0.024±0.012 (0.002; 0.037). In type/token ratio it is 0.079±0.031 (0.018; 0.11). In relative entropy it is 0.066±0.012 (0.049; 0.086).

5.4 PERFORMANCE ESTIMATION

A question that immediately arises from Section 5.2 and Section 5.3 is whether there is a relation between the differences in performance and the differences in textual characteristics, viz. the differences in domain complexity. To answer this question, we measure their correlation.

In this section, whenever we speak of accuracies we refer to accuracies of our SA approach when not varying the genre but varying the domain (see Section 5.2.2). Whenever we speak of domain complexity measurements we refer to the corresponding domain complexity measurements when not varying the genre but varying the domain.

Table 12 shows the Pearson correlation r between accuracies of our SA approach and domain complexity measurements. All correlations—except of relative entropy—

Table 12.: Pearson correlation r between domain complexity measurements and accuracies as well as r's significance level p.

DOMAIN COMPLEXITY MEASURE	r	p
Percentage of rare unigrams	-0.673	0.023
Unigram type/token ratio	-0.723	0.012
Unigram relative entropy	-0.425	0.192
Unigram homogeneity	-0.708	0.015

are strong ($|r| > 0.67$) and statistically significant ($p < 0.05$). From Table 12 we learn that

- the smaller the percentage of rare unigrams, i. e. the less hapax legomena and dis legomena,
- the smaller the unigram type/token ratio, i. e. the more tokens per type,
- the smaller the unigram relative entropy, i. e. the farther the distribution from a uniform distribution and
- the smaller the unigram homogeneity value, i. e. the more homogeneous the corpus,

the higher the accuracy of our SA method.

Relative entropy's correlation with accuracy is neither strong ($|r| = 0.425$) nor statistically significant ($p > 0.05$). It exhibits an irregular behavior. As shown in Figure 14 accuracy peaks when relative entropy is mid-range. Accuracy is lowest when relative entropy is largest. Accuracy is mid-range when relative entropy is smallest. Further investigations in these matters are left to future work.

However, given such strong ($|r| > 0.67$) and statistically significant correlations ($p < 0.05$) of the 3 other domain complexity measures we perform LRs (see Section 4.1.5): We fit LR models using squared error loss with single domain complexity measurements as predictors[5] and single accuracies as responses. To evaluate the LR models, we measure their mean residual standard error (MRSE) in leave-one-domain-out CVs (see Chapter 4.5.2). Table 13 shows the

[5] We do not use more than one predictor in our LR models in accordance with Harrell (2001, p. 61), who suggests to obey the rule of thumb $p < n/10$ where p is the number of predictors and n is the total sample size. In our leave-one-domain-out CV experiments $n = 9$ (and hence $1 > 9/10$).

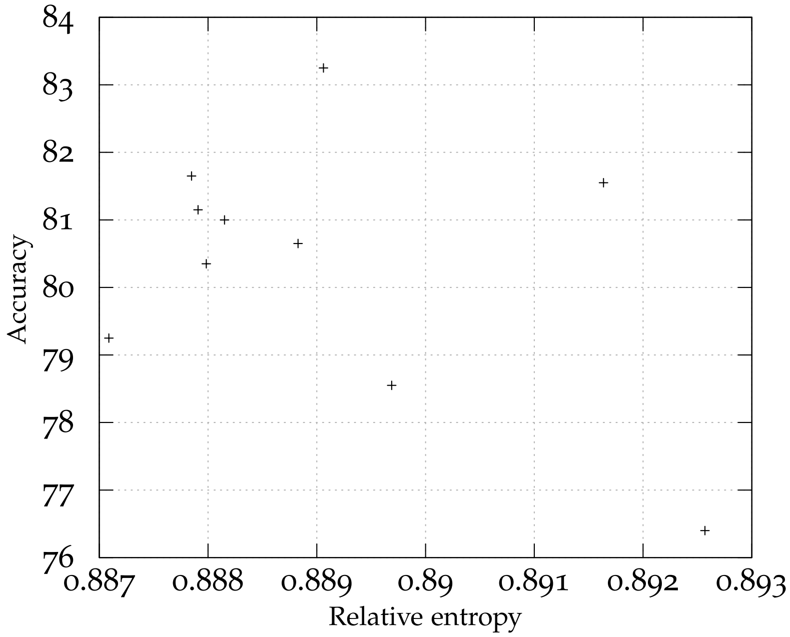


Figure 14.: Accuracy vs. relative entropy.

Table 13.: MRSEs of ordinary LR models fitted using squared error loss in leave-one-domain-out CVs with domain complexity measurements as predictors and accuracies as responses.

PREDICTOR	MRSE	p
Percentage of rare unigrams	1.238	0.033
Unigram type/token ratio	1.116	0.018
Unigram relative entropy	1.837	0.221
Unigram homogeneity	*1.058*	0.007

resulting MRSEs as well as the significance level p of the predictor's influence on the response. All predictors' influences on the response—except of relative entropy—are statistically significant ($p < 0.05$). From Table 13 we learn that—analogously to the correlations we found—3 out of 4 domain complexity measures allow us to accurately estimate our SA method's performance based solely on domain complexity measurement. Homogeneity appears to be the most informative domain complexity measure: it yields the smallest MRSE (1.058).

As we can see in Figure 15 our data contains (at least) one outlier: the domain MUSIC with an accuracy of 76.4 and a homogeneity of 0.451. Outliers such as MUSIC affect the

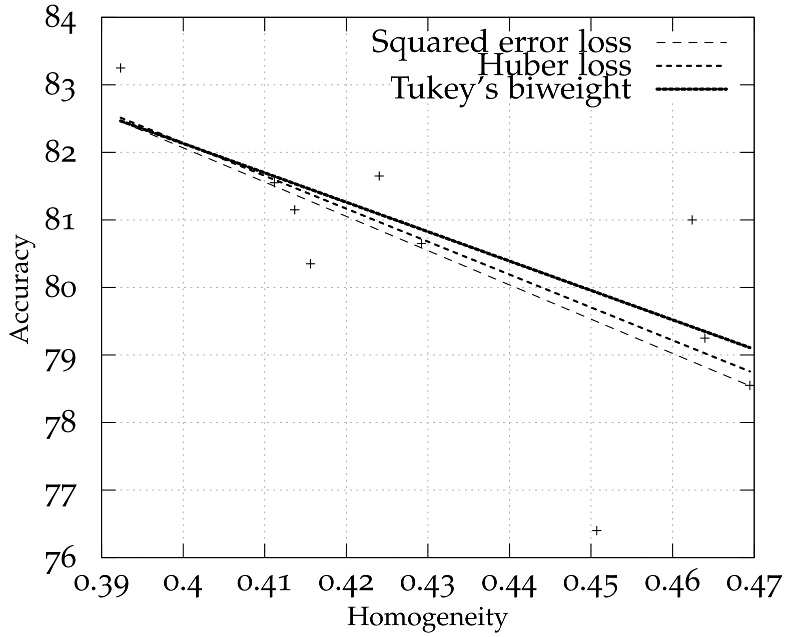


Figure 15.: Accuracy vs. homogeneity as well as LR models fitted using squared error loss, Huber loss, and Tukey's biweight.

slope of the LR fit. We counteract outliers by employing loss functions that are more robust than ordinary squared error loss: Huber loss and Tukey's biweight (see Chapter 4.1.5). Using these robust loss functions leads to small improvements in estimating accuracy—i. e. reduces the MRSEs—as shown in Table 14.

Figure 15 depicts LR models fitted to our data using squared error loss, Huber loss, and Tukey's biweight. Both robust LRs are less influenced by outliers than ordinary LR. Thus, they result in a more accurate fit of the data, especially when applied to subsamples of the data as in our leave-one-domain-out CVs.

Additional predictors

Performance estimation does not only work for SVM models based on word unigrams, but also for SVM models based on higher order word n-grams, i. e. SVM models based on word uni- and bigrams, and SVM models based on word uni, bi-, and trigrams: we just use higher order word n-gram domain complexity measurements as *additional predictors* in our LR models. To estimate the accuracy of e. g. an SVM model based on word uni- and bigrams, we measure both word unigram relative entropy and word bigram relative entropy, or both unigram type/token ratio and bigram type/token ratio etc. These additional predictors are either

Table 14.: MRSEs of robust LR models fitted using Huber loss and Tukey's biweight in leave-one-domain-out CVs with domain complexity measurements as predictors and accuracies as responses.

PREDICTOR	HUBER	TUKEY'S
Percentage of rare unigrams	1.205	1.208
Unigram type/token ratio	1.054	1.073
Unigram relative entropy	1.959	2.050
Unigram homogeneity	*1.027*	1.082

kept separately or *averaged*. Averaging predictors, e. g. averaging word uni-, bi-, and trigram relative entropy, results in a single predictor in our LR models. Keeping predictors separately results in multiple predictors in our LR models.

Separately kept and averaged predictors

We then proceed as described earlier. Results of the accuracy estimation for SVM models based on word uni- and bigrams are shown in Table 15. Results of the accuracy estimation for SVM models based on word uni-, bi-, and trigrams are shown in Table 16.

For accuracy estimation of SVM models based on word uni- and bigrams using percentage of rare words as separate predictors and an LR model fitted using Tukey's biweight yields the smallest MRSE (0.472). For accuracy estimation of SVM models based on word uni-, bi-, and trigrams using type/token ratio as separate predictors and an LR model fitted using Tukey's biweight yields the smallest MRSE (0.634).

Discussion

We showed that we are able to estimate our SA approach's accuracy on a certain gold standard based solely on the gold standard's textual characteristics, viz. its domain complexity. Domain complexity measures allow us to determine how much an SVM model can learn from the gold standard's data, in which certain features—word n-grams—occur, re-occur, or do not re-occur.

However, our performance estimates are not 100% accurate. On average we over- or underestimate our SA approach's performance by about 1 accuracy point. This is because an ML-based classifier's—e. g. an SVM model's—

Table 15.: MRSEs of LR models fitted using squared error loss, Huber loss, and Tukey's biweight in leave-one-domain-out CVs with domain complexity measurements as predictors and accuracies of SVM models based on word uni- and bigrams as responses. PRW denotes percentage of rare word n-grams, TTR denotes word n-gram type/token ratio, H_{rel} denotes word n-gram relative entropy, Hom denotes word n-gram homogeneity. "sep" denotes separately kept predictors, "avg" denotes averaged predictors.

predictor(s)		squared	huber	tukey's
PRW	sep	0.94	0.506	*0.472*
	avg	0.963	0.905	0.907
TTR	sep	0.942	0.591	0.579
	avg	0.921	0.777	0.765
H_{rel}	sep	0.902	0.882	0.87
	avg	1.604	1.514	1.464
Hom	sep	1.02	1.063	1.067
	avg	0.927	0.925	0.958

Table 16.: MRSEs of LR models fitted using squared error loss, Huber loss, and Tukey's biweight in leave-one-domain-out CVs with domain complexity measurements as predictors and accuracies of SVM models based on word uni-, bi- and trigrams as responses.

predictor(s)		squared	huber	tukey's
PRW	sep	1.143	0.867	0.738
	avg	0.913	0.943	0.928
TTR	sep	1.048	0.747	*0.634*
	avg	0.781	0.713	0.75
H_{rel}	sep	1.002	1.022	1.049
	avg	1.43	1.429	1.620
Hom	sep	1.037	0.996	0.904
	avg	0.877	0.854	0.894

discriminative power ultimately also depends on a gold standard's *non-textual characteristics*, e. g. whether it contains erroneous labels, its size, and its class boundary complexity (see Chapter 3.2.3).

Non-textual characteristics

To transfer an LR model for performance estimation, viz. our *performance estimator* from the gold standard(s) it is trained on to another, their non-textual characteristics have to similar. If their non-textual characteristics are not similar our performance estimator will most likely suffer from an average accuracy loss larger than 1 accuracy point.

Performance estimator

Lessons Learned

We learned a few lessons while carrying out our experiments in this section: (i) We are able to estimate the word n-gram homogeneity of a joint corpus C_{joint} consisting of corpora C_i, C_j through the homogeneity of corpora C_i, C_j and their similarity via LR. (ii) We are also able to estimate the actual word $\{1, \dots, k\}$-gram homogeneity out of the separate word unigram homogeneity, word bigram homogeneity, $\dots$, word k-gram homogeneity by simply averaging them.

Related Work

Ponomareva and Thelwall (2012a) estimate the accuracy loss when transferring their SA method from a source to a target domain via domain complexity and domain similarity measures. Van Asch and Daelemans (2010) estimate the accuracy loss when transferring a POS tagger from one domain to another via domain similarity measures. Blitzer et al. (2007) compute an $\mathcal{A}$-distance proxy and show that it correlates with accuracy loss when transferring their SA method from a source to a target domain.

Future Work

Future work includes to estimate our SA approach's lower and upper bounds of accuracy instead of estimating its accuracy directly. To learn more about the relation between our SA approach's accuracy on a certain gold standard and its domain complexity, we consider experiments in which we control certain textual characteristics. E. g., we limit the vocabulary to a certain size by discarding sentences that contain uncommon words (see Chapter 3.3.2). Thereby, we create subsets of gold standards on which we then evaluate our SA approach and investigate its behavior. However, this strategy bears the risk of introducing arbitrary effects that we cannot control.

5.5 CONCLUSION

In this chapter we described our ML-based approach to the SA subtasks that we focus on—polarity and subjectivity classification. We described its underlying assumptions, its text representation, and our classifier choice (see Section 5.1).

We showed that our SA approach performs differently when applied to gold standards (or subsets of gold standards) that originate from different genres and domains. Moreover, we verified our hypothesis that differences in performance are small when we vary neither genre nor domain; that differences in performance are larger when we do vary the domain; and that differences in performance are even larger when we vary the genre or both genre and domain (see Section 5.2).

We then showed that gold standards that originate from different genres and domains differ in their textual characteristics, viz. their domain complexity. Moreover, we verified our hypothesis that differences in domain complexity behave similar to the differences in performance (see Section 5.3).

Finally, we showed that there is a clear relationship between our SA approach's performance on a certain gold standard and its domain complexity. We used this relationship to estimate our SA approach's accuracy on a gold standard based solely on its domain complexity (see Section 5.4). In summary, we verified the core hypothesis of our thesis, i. e. SA is genre and domain dependent.

6 MODEL SELECTION AND FEATURE ENGINEERING

> Wesentliche Aufgabe eines Ingenieurs ist es, für technische Probleme mit Hilfe naturwissenschaftlicher Erkenntnisse Lösungen zu finden, und sie unter den jeweils gegebenen Einschränkungen stofflicher, technologischer und wirtschaftlicher Art in optimaler Weise zu verwirklichen.
>
> — *Pahl and Beitz (1986, p. 1)*

In this chapter we use textual characteristics—viz. domain similarity (see Chapter 3.1), domain complexity (see Chapter 3.2) and readability (see Chapter 3.3)—in a variety of applications: domain complexity will guide us in model selection for in-domain polarity classification in Section 6.1. Domain similarity and domain complexity will be used in domain adaptation (DA) for cross-domain polarity classification in Section 6.2. Readability will be used in feature engineering for subjectivity classification in Section 6.3.

6.1 DOMAIN COMPLEXITY-BASED MODEL SELECTION

It is surprisingly hard to outperform a text classification model that is based solely on word unigrams (e. g. Bekkerman and Allan, 2004). However, often there is room for improvement over such word unigram models, e. g. using *higher order word* n*-grams* such as word bi- or trigrams as additional features or by performing word n-gram FS.

Higher order word n*-grams*

Some studies argue in favor of using models that combine word unigram and higher order word n-gram representations (e. g. Riloff et al., 2006), some studies argue in favor of using word unigram representations alone (e. g. Scott and Matwin, 1999; Bekkerman and Allan, 2004; Moschitti and Basili, 2004). Several studies suggest aggressive word n-gram FS (e. g. Rogati and Yang, 2002), other studies suggest conservative word n-gram FS, especially for SVMs (e. g. Brank et al., 2002). All studies underpin their suggestions by empirical results based on different gold standards. Therefore, we cannot be sure whether to opt for one or another.

For in-domain document-level polarity classification on MDSD v2.0 using our SA approach (see Chapter 5.1), domain complexity will guide us in 2 model selections: Section 6.1.1 describes how domain complexity can guide us in deciding which word n-gram model order to employ in an SVM. Section 6.1.2 describes how domain complexity can guide us in deciding whether to employ aggressive or conservative word n-gram FS in an SVM model. Both Section 6.1.1 and Section 6.1.2 are based on Remus and Ziegelmayer (2014).

6.1.1 *Word n-gram Model Order*

In their study on the use of higher order word n-grams as features for text classification, Bekkerman and Allan (2004) conclude that

> "for an unrestricted text categorization task one would probably not expect dramatic effects of using [word] bigrams. However, in domains with severely limited lexicons and high chances of constructing stable phrases the bigrams can be useful."

Furthermore they state that some

> "corpora are 'simple' enough so only a few extracted keywords [i. e. word unigrams] can do the entire job of distinguishing between categories."

Corpus simplicity

While Bekkerman and Allan (2004) stay vague on their notion of *corpus simplicity*, we believe it concurs with our notion of domain complexity: If a dataset has low domain complexity, it is less difficult for a classifier to learn an accurate model and vice versa (see Chapter 5.4). We assume that e. g. if a dataset's word unigram domain complexity is low, but its word bigram domain complexity is high, it is likely that an SVM model based on word unigrams outperforms an SVM model based on both word uni- and bigrams. If a dataset's word uni- and bigram domain complexity is low, but its word trigram domain complexity is high, it is likely that an SVM model based on word uni- and bigrams outperforms an SVM model based on word uni-, bi-, and trigrams etc.

Model selector

If this assumption holds, given a dataset, an algorithm for model selection, viz. a *model selector* may—based on domain complexity—automatically decide what word n-gram model order to employ in an SVM for this dataset. E. g., whether to employ a first order SVM model based on word unigrams or a second order SVM model based on word uni- and bigrams. Our model selector estimates the accuracies of these SVM models for a given dataset (see Chapter 5.4) and chooses the SVM model that yields the highest estimated accuracy as shown in Pseudocode 1.

Pseudocode 1: Model selector for word n-gram model order.

```
input: dataset
for $n$ = 1, 2, ..., k {
    estimate accuracy of an SVM model based on word {1, ..., $n$}-grams on dataset
}
output: $n$ that yields the highest estimated accuracy
```

Evaluation

We evaluate our model selector in a leave-one-domain-out CV on MDSD v2.0's 10 domains, in which for each run we train our model selector on 9 domains and decide what word n-gram model order to employ in an SVM for the remaining 1 domain.

DATA We decide between first, second, and third order word n-gram SVM models, i. e. between SVM models based on word unigrams, uni- and bigrams, or uni-, bi-, and trigrams. To produce data for our leave-one-domain-out CV, we evaluate 3 SVM models per domain in 10-fold CVs: one SVM model based on word unigrams, one SVM model based on word uni- and bigrams, and one SVM model based word uni-, bi-, and trigrams. The evaluation results are shown in Table 17.

SVM models based word uni- and bigrams always outperform SVM models based solely on word unigrams. SVM models based on word uni-, bi-, and trigrams outperform SVM models based on word uni- and bigrams only for 1 domain: MUSIC.

Table 17.: Accuracy of SVM models based on word unigrams, word uni- and bigrams, or word uni-, bi-, and trigrams on MDSD v2.0.

DOMAIN	UNI	UNI-BI	UNI-BI-TRI
apparel	83.25	85.55	85.05
books	79.25	79.65	79.5
dvd	78.55	79.8	79.25
electronics	80.65	82.05	81.6
health	80.35	83.55	83.45
kitchen	81.15	82.1	81.85
music	76.4	78.45	78.9
sports	81.65	83	82.95
toys	81.55	83.15	82.75
video	81	81.65	81.65
average	80.38	81.9	81.7

EXPERIMENTS We vary 3 parameters of our model selector's accuracy estimation. (i) We compare 4 predictors: percentage of rare words, type/token ratio, relative entropy, and homogeneity (see Chapter 3.2). (ii) We compare separately kept and averaged predictors (see Chapter 5.4). (iii) We compare 3 LR loss functions: squared error loss, Huber loss, and Tukey's biweight (see Chapter 4.1.5). The evaluation results of our leave-one-domain-out CV are shown in Table 18.

RESULTS AND DISCUSSION Our model selector yields an average accuracy between 60–100 when deciding between first or second order. It yields an average accuracy between 40–90 when deciding between second or third order. It yields an overall accuracy between 65–95.

The most reliable model selector uses averaged homogeneity as predictor and fits the LR model using Huber loss: It yields an average accuracy of 100 when deciding between first or second order. It yields an average accuracy of 90 when deciding between second or third order. Thus, it yields an overall average accuracy of 95.

Note that for our data a naïve baseline also yields an overall average accuracy of 95: A naïve model selector that

Table 18.: Accuracy of our model selectors for word n-gram model order. PRW denotes percentage of rare word n-grams, TTR denotes word n-gram type/token ratio, H_{rel} denotes word n-gram relative entropy, Hom denotes word n-gram homogeneity. "sep" denotes separately kept predictors, "avg" denotes averaged predictors. "1–2" denotes first vs. second order, "2–3" denotes second vs. third order.

PREDICTOR		LOSS FUNCTION					
		Squared		Huber		Tukey's	
		1–2	2–3	1–2	2–3	1–2	2–3
PRW	sep	100	80	90	70	80	50
	avg	100	80	100	70	100	40
TTR	sep	90	90	90	80	90	80
	avg	100	80	100	60	90	40
H_{rel}	sep	60	80	60	70	60	70
	avg	90	80	90	70	80	60
Hom	sep	100	80	100	80	90	70
	avg	100	80	100	90	90	90

always decides for second order yields an average accuracy of 100 when deciding between first or second order. It yields an average accuracy of 90 when deciding between second or third order. Thus, its overall average accuracy is also 95.

6.1.2 *Word n-gram Feature Selection*

We face 2 questions when we perform word n-gram FS (see Chapter 4.3):

1. Which FS method should we use?
2. How many features should we select?

We answer question 1 up front: as FS method we use IG (see Chapter 4.3.1), because it has been shown that IG is superior to other FS methods for word n-gram based text classification (see Yang and Pedersen, 1997; Forman, 2003), e. g. χ^2 (see Chapter 4.3.2).

We answer question 2 analogously to Section 6.1.1: for a given dataset a model selector may—based on its domain complexity—estimate how many features to select for an SVM model based on word n-grams.

Evaluation

As in Section 6.1.1 we evaluate our model selector—which we develop in our experiments—in a leave-one-domain-out CV on MDSD v2.0's 10 domains, in which for each run we train our model selector on 9 domains and estimate how many features to select for an SVM model based on word n-grams for the remaining 1 domain.

DATA FS methods such as IG produce an implicit ranking with the most predictive features ranked highest and the least predictive features ranked lowest. To employ FS via IG, we have to determine a *cut off (CO)*: features ranked above the CO are kept, while features ranked below the CO are discarded.

Cut off

To produce data for our leave-one-domain-out CV, for each domain we determine the CO for which the accuracy of our SVM model peaks. First we rank a domain's word unigrams via IG. We then set the CO to 1, 2, . . . , 100% of the domain's original word unigram vocabulary size. If it is set to 1% we keep its 1% highest ranked word unigrams, if it is set to 2% we keep its 2% highest ranked word unigrams etc. For each of the resulting 100 word unigram vocabularies we evaluate an SVM model based on this word unigram vocabulary in a 10-fold CV. We call the CO for which our SVM model's accuracy peaks *ideal CO*. Table 19 shows evaluation results of SVM models based on word unigrams with and without FS via IG. FS is based on the ideal CO.

Ideal cut off

With FS using the ideal CO the average accuracy is 1.12 higher than without FS. Ideal COs are scattered: 85% (13,139 word unigram types) is the most conservative FS and 2% (506 word unigram types) is the most aggressive FS. The *average ideal CO* is 45% (8,045 word unigram types).

Average ideal cut off

EXPERIMENTS Table 20 shows the Pearson correlation r between domain complexity measurements of MDSD v2.0's 10 domains and their ideal CO. Relative entropy correlates strongest with ideal CO ($r = -0.35$): The smaller the domain's relative entropy, the larger its ideal CO. Thus, the less uniform a domain's word unigram distribution, the more of its word unigrams are kept as features in our SVM

Table 19.: Accuracy of SVM models based on word unigrams with and without FS via IG based on the ideal CO in percent and word unigram types.

domain	without	with	Δ	ideal co
apparel	83.25	83.6	0.35	78% (7,927)
books	79.25	80.45	1.2	11% (3,136)
dvd	78.55	80.55	2	60% (18,169)
electronics	80.65	81.75	1.1	85% (13,139)
health	80.35	80.85	0.5	90% (11,778)
kitchen	81.15	82.8	1.65	17% (2,214)
music	76.4	78.45	2.05	2% (506)
sports	81.7	82.55	0.85	19% (2,715)
toys	81.5	82.45	0.95	4% (564)
video	81.05	81.6	0.55	80% (20,301)
average	80.39	81.51	1.12	45% (8,044.9)

Table 20.: Pearson correlation r between domain complexity measurements and ideal COs as well as r's significance level p.

domain complexity measure	r	p
Percentage of rare word unigrams	-0.08	0.814
Word unigram type/token ratio	-0.119	0.727
Word unigram relative entropy	-0.35	0.291
Word unigram homogeneity	-0.095	0.78

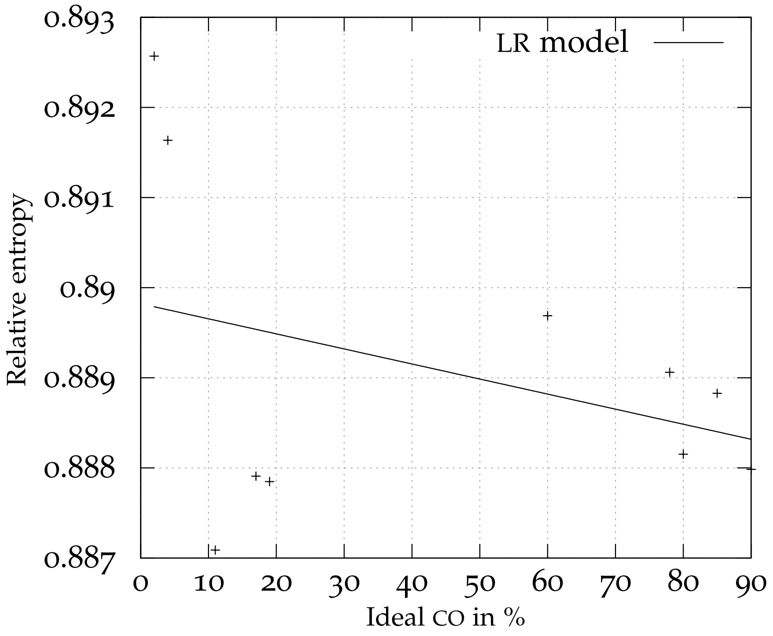


Figure 16.: Relative entropy vs. ideal CO.

model. Figure 16 plots relative entropy vs. ideal CO. Additionally, it shows an LR model fitted to the data using squared error loss. It achieves no perfect fit, but it still roughly estimates ideal CO.

Given its correlation with the ideal CO, we use as our model selector a robust LR model with relative entropy as single predictor and ideal CO as response. We compare LR models fitted using Huber loss and Tukey's biweight. Table 21 shows the evaluation results of our leave-one-domain-out CV for Huber loss, Table 22 shows the evaluation results for Tukey's biweight.

RESULTS AND DISCUSSION Our model selector over- or underestimates a domain's ideal CO on average by 40%. Still, SVM models with FS using the estimated CO outperform SVM models without FS in 6 out of 10 domains. SVM models with FS using the estimated CO yield an average accuracy of 80.55 (Huber loss) and 80.67 (Tukey's biweight). Without FS average accuracy is 80.39. Thus, FS using the estimated CO yields an average accuracy gain of 0.16 and 0.28, respectively.

Compared with SVM models with FS using the ideal COs (81.51), using the estimated CO performs 0.95 and 0.83 lower, respectively. SVM models with FS using the average ideal

Table 21.: Ideal COs and COs estimated by our model selector fitted using Huber loss as well as accuracies of SVM models based on word unigrams using FS based on ideal and estimated CO.

domain	ideal		estimated		Δ	
	co	a	co	a	co	a
apparel	78%	83.6	41%	83.50	37%	-0.1
books	11%	80.45	74%	78.25	63%	-2.2
dvd	60%	80.55	38%	80.10	22%	-0.45
electronics	85%	81.75	42%	80.45	43%	-1.3
health	90%	80.85	46%	80.50	44%	-0.35
kitchen	17%	82.8	59%	82.30	42%	-0.5
music	2%	78.45	39%	77.65	37%	-0.8
sports	19%	82.55	60%	81.75	41%	-0.8
toys	4%	82.45	37%	81.00	33%	-1.45
video	80%	81.6	47%	80.00	33%	-1.6
average	45%	81.51	48%	80.55	40%	-0.95

Table 22.: Ideal COs and COs estimated by our model selector fitted using Tukey's biweight as well as accuracies of SVM models based on word unigrams using FS based on ideal and estimated CO.

domain	ideal		estimated		Δ	
	co	a	co	a	co	a
apparel	78%	83.6	41%	83.5	37%	-0.1
books	11%	80.45	78%	79.2	67%	-1.25
dvd	60%	80.55	37%	80.1	23%	-0.45
electronics	85%	81.75	42%	80.45	43%	-1.3
health	90%	80.85	46%	80.5	44%	-0.35
kitchen	17%	82.8	62%	82.45	45%	-0.35
music	2%	78.45	39%	77.65	37%	-0.8
sports	19%	82.55	62%	81.85	43%	-0.7
toys	4%	82.45	37%	81	33%	-1.45
video	80%	81.6	47%	80	33%	-1.6
average	45%	81.51	49%	80.67	40%	-0.83

CO (45%) yield an average accuracy of 80.56, which is on par with SVM models with FS using the estimated COs (80.55–80.67).

6.1.3 *Conclusion and Future Work*

Based on our findings from Section 6.1.1 and Section 6.1.2 we conclude that domain complexity—given a task such as document-level in-domain polarity classification, a corresponding gold standard such as MDSD v2.0 and an SA approach—can guide us in model selection, e. g. whether to opt for an SVM model based on word unigrams, word uni- and bigrams, or word uni-, bi-, and trigrams as well as whether to opt for aggressive or conservative word n-gram FS.

In future work domain complexity may guide us in feature engineering: whether to use super- or sub-word character n-gram representations (see Raaijmakers and Kraaij, 2008) instead of word n-gram representations; whether to use non-binary word n-gram weighting, e. g. weighting using tf-idf (see Manning and Schütze, 1999, p. 543) or distributional information (see Xue and Zhou, 2009); or whether to employ non-lexical features, e. g. POS tags or dependency parses. Further future work may consider nonlinear regression models.

6.2 DOMAIN SIMILARITY- AND DOMAIN COMPLEXITY-BASED DOMAIN ADAPTATION

Domain adaptation

Source domain
Target domain

Domain complexity variance

A lot of SA research focuses on *DA* algorithms, which minimize the performance loss when transferring a model from one *source domain* to another *target domain* (see Aue and Gamon, 2005; Blitzer et al., 2007; Pan et al., 2010). Only recently, Ponomareva and Thelwall (2012a) proposed to model the accuracy loss in cross-domain polarity classification based on domain similarity and *domain complexity variance*, i. e. the domain complexity difference between source and target domain. Ponomareva and Thelwall (2012b) hypothesized that the optimal parameter setting of a graph-based DA algorithm is related to their notions of domain similarity and domain complexity.

We pick up on this intuitive idea: we exploit domain similarity and domain complexity variance to tailor a given source domain training set to a given target domain via *Instance Selection (IS)* (see Remus, 2012).

Instance selection

6.2.1 *Related Work*

Our work resembles Plank and van Noord (2011)'s study on parsing, in which training instances are selected based on their similarity to a given test set. Close to our work is Jiang and Zhai (2007), who propose a general instance weighting framework for DA.

Work on DA in SA focuses on polarity classification: Aue and Gamon (2005) observed early that it is nontrivial to customize classifiers to new domains without accepting significant accuracy loss. Blitzer et al. (2007) proposed Structural Correspondence Learning (SCL), by which they determine pivot features to link source and target domain features. Tan et al. (2007) use labeled in-domain examples to train a classifier, then classify informative but unlabeled out-of-domain examples and finally re-train their classifier leveraging the newly created training data. In contrast, Li and Zong (2008) do not adapt classifiers to new domains, but exchange knowledge among them. They pool features common to different domains and use meta-learning to join classifiers trained in different domains. Pan et al. (2010) bridge the gap between different domains via Spectral Feature Alignment (SFA). Ponomareva and Thelwall (2012b) propose graph-based DA.

Work on DA in SA beyond polarity classification encompasses domain-specific expansion of sentiment lexicons (see Kanayama and Nasukawa, 2006; Qiu et al., 2009; Gindl et al., 2010) as well as cross-domain opinion holder extraction (see Wiegand and Klakow, 2012) and cross-domain opinion target extraction (see Jakob and Gurevych, 2010).

6.2.2 *Method*

Our DA scheme—IS—is based on 2 assumptions: (i) When learning a target domain model from both source and target domain instances, the source domain instances that are most similar to the target domain instances are more "informative", i. e. likely to improve target domain model quality. (ii) The *reduction factor* r by which the original source domain training set size is reduced can be determined auto-

Reduction factor r

matically by measuring domain similarity between source and target domain as well as their domain complexity variance. We estimate $r_{d_{src},d_{tgt}}$ as shown in Equation (6.1)

$$\tilde{r}_{d_{src},d_{tgt}} = 1.0 - \left(\alpha \cdot s_{d_{src},d_{tgt}} + \beta \cdot |\Delta c_{d_{src},d_{tgt}}|\right) \tag{6.1}$$

where $s_{d_{src},d_{tgt}}$ is the domain similarity between source domain d_{src} and target domain d_{tgt}. $\Delta c_{d_{src},d_{tgt}} = c_{d_{src}} - c_{d_{tgt}}$ is the domain complexity variance of d_{src} and d_{tgt} (see Ponomareva and Thelwall, 2012a). α and β are essentially scaling parameters for domain similarity and domain complexity variance, respectively. *α and β*

Similarity of domains d_i and d_j is measured as pairwise JS divergence (see Chapter 3.1.1) between d_i and d_j's word unigram distributions. JS divergence was given preference over other divergence measures, e. g. KL divergence, skew divergence, or Renyi divergence (see Chapter 3.1.1), because it provided good results across several recent studies (e. g. Plank and van Noord, 2011; Ponomareva and Thelwall, 2012a). Word unigram distributions—i. e. word unigram probabilities—are estimated via the word unigrams' relative frequencies. We also experimented with Simple Good-Turing smoothing (see Gale and Sampson, 1995) of these probability estimates, but found it not beneficial for our purposes. Domain complexity is measured as word unigram homogeneity (see Chapter 3.2.2).

The rationale behind Equation (6.1) is as follows: The more dissimilar the domains, i. e. the larger $s_{d_{src},d_{tgt}}$, the smaller $\tilde{r}_{d_{src},d_{tgt}}$. Thus, the more the original source domain training set size is reduced. Moreover, the more the domain complexity varies among source and target domain, i. e. the larger $|\Delta c_{d_{src},d_{tgt}}|$, the smaller $\tilde{r}_{d_{src},d_{tgt}}$. The former part follows from our intuition that a target domain model benefits more from training instances drawn from a similar source domain than from training instances drawn from a dissimilar source domain. The latter part follows from Ponomareva and Thelwall (2012a)'s observation that learning an accurate model for d_{tgt} is harder when d_{tgt}'s domain complexity is higher than d_{src}'s domain complexity. On the one hand, if d_{src} is much less complex than d_{tgt}, we cannot learn enough from the source domain about the target domain. On the other hand, if d_{src} is much more complex than d_{tgt}, we learn much from the source domain that will not be useful within the target domain.

Source domain training instances are then selected by IS as follows: First, they are ranked according to their similarity to the target domain. Similarity is measured as JS divergence between a source domain instance's word unigram distribution and the target domain's word unigram distribution. Secondly, only the top $100 \cdot \tilde{r}_{d_{src},d_{tgt}}$%-ranked instances are kept, while the rest is discarded. Thereby, the source domain distribution is tailored to the target domain distribution.

6.2.3 *Evaluation*

To evaluate IS, we carry out cross-domain polarity classification experiments on document-level in a semi-supervised setting: given a "normal" amount of source domain training data and little target domain training data, we aim at learning a target domain model that is as accurate as possible.

Experimental Setup

Our experimental setup is as follows: As classifiers we employ SVMs using a linear kernel with their cost parameter C set to 2.0 without any further optimization. We refrain from optimizing C because training and testing the large number of models in our evaluation is too time consuming. We model polarity using word unigrams or word uni- and bigrams as described in Chapter 5.1. As gold standard for cross-domain document-level polarity classification we use MDSD v2.0 (see Chapter 2.1.2).

For all $10!/(10-2)! = 90$ source domain–target domain pairs from MDSD v2.0 we use 2,000 labeled source domain instances (1,000 positive and 1,000 negative) and 200 labeled target domain instances (100 positive and 100 negative) for training. 1,800 labeled target domain instances (900 positive and 900 negative) are used for testing. If required by the method 2,000 unlabeled target domain instances are available for training. This is a typical *semi-supervised* DA setting similar to Daumé III et al. (2010b)'s setup.

Semi-supervised DA

According to Daumé III (2007) there are several natural baselines in DA: (i) SrcOnly, in which only labeled source domain instances are used for training, (ii) TgtOnly, in which only labeled target domain instances are used for training, and (iii) All, in which both labeled source and

SrcOnly

TgtOnly

All

labeled target domain instances are used for training.

IS

Sample size-normalized domain similarity

IS is trained on the same data as ALL, i. e. all available labeled source and target domain instances. To estimate domain similarity, IS uses 200 labeled source domain instances and 200 labeled target domain instances. In contrast to using the full amount of 2,000 labeled source domain and 2,000 labeled target domain instances instances, this enables symmetric domain similarity estimation across all d_{src}–d_{tgt} pairs in our semi-supervised setting, in which only 200 labeled target domain instances are available: d_i–d_j's domain similarity equals d_j–d_i's domain similarity, independently of whether d_i or d_j is the source or target domain, respectively. To estimate source and target domain complexity, IS uses 200 labeled instances and 1,800 unlabeled instances each. We use only 200 labeled instances because in our semi-supervised setting only 200 labeled target domain instances are available. To improve the estimate of domain complexity, we add another 1,800 unlabeled instances. We assume that unlabeled instances are readily available. Both domain similarity and domain complexity are normalized with respect to sample size. *Sample size-normalized domain similarity* is computed similar to sample size-normalized domain complexity (see Chapter 3.2.4): We sample a 1,000 word window from each the source and the target domain instances and compute their word unigram probability distributions. We iterate for 10,000 times and average the resulting word unigram probability distributions. Finally, we compute the domain similarity between the averaged distributions as described above. Furthermore, we "normalize" domain similarity between a source domain instance and the target domain with respect to the source domain instances's document length. Thereby, we give longer documents additional preference when ranking them. We do this because we expect longer documents to contain more information than shorter ones.

EA

EA++

We compare IS with the baselines SRCONLY, TGTONLY, and ALL. Additionally, we compare IS with Daumé III (2007)'s EasyAdapt (EA), and Daumé III et al. (2010a)'s EA++. Both EA and EA++ are light-weight and state-of-the-art DA algorithms that operate via feature space augmentation and were theoretically analyzed in the framework of co-regularization (see Daumé III et al., 2010b). Because the authors of EA and EA++ only provide results for 2 domain pairs of MDSD v2.0, we re-implemented EA and EA++.

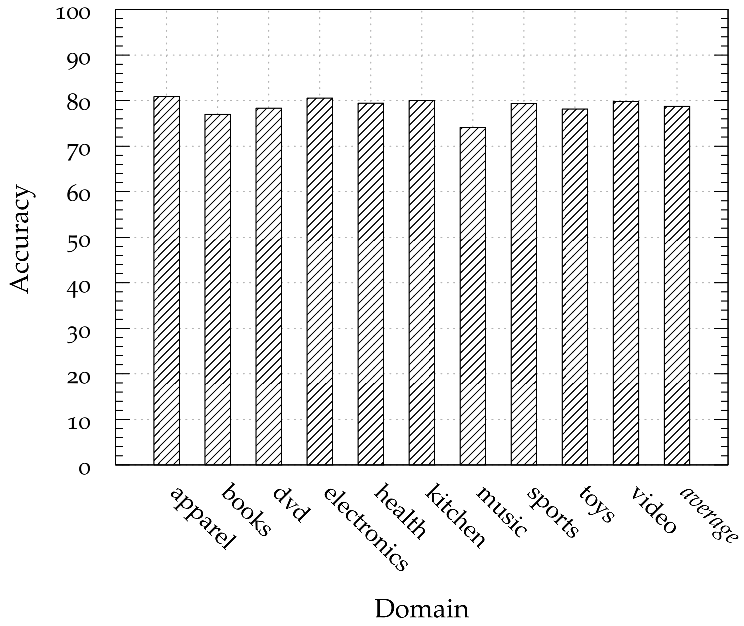


Figure 17.: In-domain accuracy of SVM models based on word unigrams on MDSD v2.0.

Unsupervised DA

We also evaluate IS in an *unsupervised DA* setting, in which there are no labeled target domain training instances available. Essentially, we combine IS and SrcOnly, resulting in IS-SrcOnly. IS-SrcOnly is then compared with SrcOnly.

IS-SrcOnly

Moreover, we establish an upper bound for IS: We compute the optimal reduction factor for each source domain–target domain pair by repeatedly evaluating IS while varying r from 0 to 1 with a step size of 0.01. Using the optimal reduction factor for all domain pairs yields an average accuracy of 75.77. The average optimal reduction factor is 0.8, which is chosen for a sanity check described in the next paragraph.

$IS_{r=0.8}$

IS_{random}

Finally, we perform 2 sanity checks: (i) $IS_{r=0.8}$ selects source domain instances using a fixed reduction factor $r = 0.8$. (ii) IS_{random} sets $\tilde{r}_{d_{src},d_{tgt}}$ randomly and then selects source domain instances without any ranking.

Results

Figure 17 shows in-domain accuracies on MDSD v2.0's 10 domains achieved by SVM models based on word unigrams in 10-fold CVs. Achieving or even exceeding in-domain accuracy is the ultimate goal of DA.

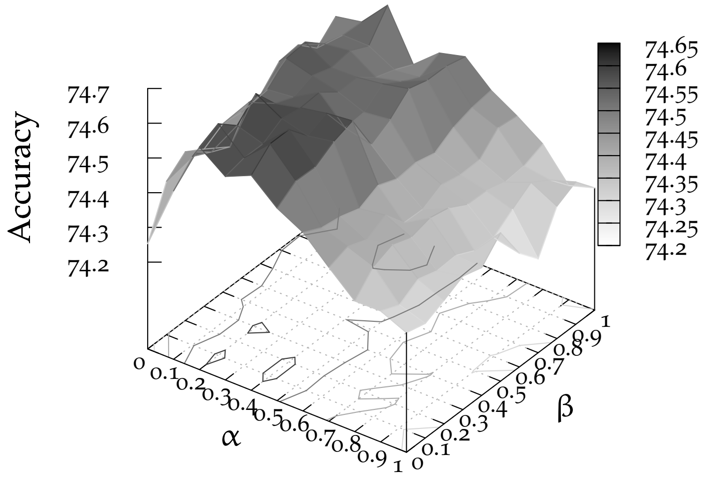


Figure 18.: Behavior of IS for varying α and β.

For each of the 90 source domain–target domain pairs we carry out a cross-domain polarity classification experiment. In line with previous work on MDSD v2.0 (e. g. Pan et al., 2010; Ponomareva and Thelwall, 2012a,b) we measure performance using accuracy (see Chapter 4.4). The level of statistical significance is determined by *stratified shuffling*, which is an approximate randomization test (see Noreen, 1989) run with 2^{20} = 1,048,576 iterations as recommended by Yeh (2000). Because listing accuracies for all domain pairs and all DA methods is not feasible due to space restrictions, we report only the average accuracy across all domain pairs.

Stratified shuffling

A priori we do not know how to scale domain similarity and domain complexity variance, i. e. how to set α and β. Therefore, we run IS using different parameter settings. Figure 18 shows the behavior of IS when varying the scaling parameters α and β with $\alpha \in [0, 1]$ and $\beta \in [0, 1]$ and a step size of 0.1. These scaling intervals were chosen to allow $\tilde{r}$ a range of $[0.5, 1]$. The best overall result (74.62) is achieved by setting $\alpha = 0.3$ and $\beta = 0.2$ (see Figure 18), which from now on is the default setting. Setting $\alpha = 0$ while varying β leads to a minimum accuracy of 74.29 and a maximum accuracy of 74.59. Setting $\beta = 0$ while varying α leads to a minimum accuracy of 74.39 and a maximum

Table 23.: Cross-domain accuracies of DA methods averaged over MDSD v2.0's 90 domain pairs. SVM models are based on word unigrams or word uni- and bigrams. SVM models are trained with and without optimization of C.

METHOD	MODEL		
	uni	uni-bi	uni-bi
	without		with
SrcOnly	72.2	74.36	74.88
TgtOnly	68.42	68.96	69.04
All	74.25	76.61	77.17
IS	74.62	76.9	76.95
IS-SrcOnly	72.27	74.4	74.51
EA	74.02	75.83	76.36
EA++	74.5	75.95	76.41

accuracy of 74.5. Thus, both domain similarity—scaled by α—and domain complexity—scaled by β—influence $\tilde{r}$ positively. Improvements over all 3 baselines—SrcOnly, TgtOnly, and All—are achieved for all scaling parameter settings where $\alpha > 0$ and $\beta > 0$ except when $\alpha = 1$ or $\beta = 0.7$. Thus, IS is beneficial even without fine-tuning its scaling parameters.

Table 23 shows the evaluation results. When using word unigrams as features, IS (74.62) performs significantly better than SrcOnly (72.2, $p < 0.005$), TgtOnly (68.42, $p < 0.005$), All (74.25, $p < 0.05$) and EA (74.02, $p < 0.005$). IS also outperforms $\text{IS}_{\text{random}}$ (71.98, $p < 0.005$) and $\text{IS}_{r=0.8}$ (74.31, $p > 0.05$). IS also outperforms EA++ (74.5, $p > 0.05$). IS-SrcOnly (72.27, $p > 0.05$) outperforms SrcOnly.

For document-level polarity classification SVM models based on word uni- and bigrams generally yield higher accuracies than SVM models based on word unigrams (see Section 6.1.1). Optimizing SVM models' hyperparameter C usually leads to clear performance gains (see Chapter 5.1.3). Therefore, we re-evaluate all DA methods (i) based on word uni- and bigrams; (ii) with and without optimization of the SVM models' hyperparameter C. Evaluation results are shown in Table 23.

When using both word uni- and bigrams as features but without optimization of C, IS (76.9) outperforms its baselines SrcOnly (74.36, $p < 0.005$), TgtOnly (68.96, $p < 0.005$) and All (76.61, $p < 0.005$), just as IS-SrcOnly (74.4) outperforms its baseline SrcOnly (74.36, $p > 0.05$). However, with optimization of C, All (77.17) outperforms IS (76.95) by a small but statistically significant margin, just as SrcOnly (74.88) outperforms IS-SrcOnly (74.51) by a small but statistically significant margin. This may be because IS' scaling parameters α and β were determined without optimization of C. Further investigations in these phenomena are left to future work. IS with (76.9) and without (76.95) optimization of C always outperforms EA (75.83, $p < 0.005$ and 76.36, $p < 0.005$) and EA++ (75.95, $p < 0.005$ and 76.41, $p < 0.005$).

Note that there are 6 source domain–target domain pairs for which IS is obliged to decrease performance, because their performance peaks for $r = 1.0$, i. e. when there is no source domain training set size reduction at all. These domain pairs are books–dvd, books–sports, electronics–sports, electronics–video, toys–electronics, toys–video.

Comparison

Technically, we cannot compare IS to "traditional" DA methods like SCL (see Blitzer et al., 2007), SFA (see Pan et al., 2010), OPTIM-SOCAL, and RANK-SOCAL (see Ponomareva and Thelwall, 2012b), because—just as EA and EA++—IS is essentially a preprocessing step that is agnostic to both the employed ML algorithm as well as feature types and their representation. Therefore, it is likely that IS is also beneficial to more advanced DA techniques beyond simple bag-of-word models. For completeness, Figure 19 nevertheless compares IS, IS-SrcOnly, EA, and EA++ as well as SrcOnly, TgtOnly, and All to the aforementioned algorithms, all of which were evaluated on 4 domains—dvd, books, electronics, kitchen—of MDSD v2.0, resulting in 12 domain pairs.

Although IS generally performs worse than more sophisticated DA methods, it rivals their performance on some domain pairs: it outperforms SCL in 3 out 12 domains, it outperforms OPTIM-SOCAL in 4 out of 12 domains and it outperforms SFA in 2 out of 12 domains.

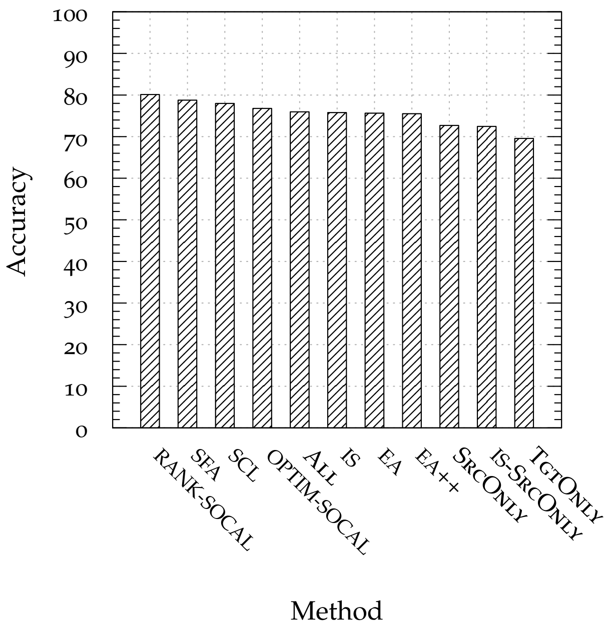


Figure 19.: Cross-domain accuracy of DA methods on 12 MDSD v2.0 domain pairs.

A Case Study

In addition to our evaluation in Section 6.2.3, we evaluate IS in a case study on all $9!/(9-2)! = 72$ source domain–target domain pairs from RND (see Chapter 2.1.5). We compare IS with its baselines SrcOnly, TgtOnly, and All. For all DA methods we use SVM models based on word unigrams and the same experimental setup as before. IS yields an average accuracy of 81.55. It outperforms SrcOnly (77.45, $p < 0.005$), TgtOnly (76.19, $p < 0.005$) and All (81.3, $p > 0.05$).

Lessons Learned

We conclude our evaluation with a few "lessons learned": (i) The "normalization" with respect to document length is not mandatory, but improves performance slightly. (ii) We also experimented with a slightly modified version of Equation (6.1) as shown in Equation (6.2)

$$\tilde{r}_{d_{src},d_{tgt}} = 1.0 - \left(\alpha \cdot s_{d_{src},d_{tgt}} + \beta \cdot \Delta c_{d_{src},d_{tgt}}\right) \tag{6.2}$$

but found using $|\Delta c_{d_{src},d_{tgt}}|$ instead of $\Delta c_{d_{src},d_{tgt}}$ to provide slightly more reliable estimates of $r_{d_{src},d_{tgt}}$, although the overall difference is almost negligible. (iii) EA and

EA++ do not benefit from IS, i. e. a combination of EA and EA++ and IS does not improve over EA and EA++ alone (see Remus, 2012). This is because source domain training instances discarded by IS contain viable information with respect to the unlabeled instances used in EA++. Furthermore, EA and EA++ augment—i. e. increase—the feature space. Discarding source domain training instances then leads to an even sparser feature space. (iv) IS did not improve accuracy in a *supervised DA* setting in which there is an equally large amount of labeled training instances from both source and target domain available. (v) We were not able to estimate r via LR. (vi) Selecting the most similar source domain training instances from all source domains instead of a single source domain in the vein of Plank and van Noord (2011) did not improve accuracy.

Supervised DA

6.2.4 *Conclusion and Future Work*

Our contribution is two-fold: (i) We proposed a novel lightweight approach to DA—viz. IS—and showed that it yields small but statistically significant improvements over several natural baselines and achieves competitive results to other state-of-the-art DA schemes in cross-domain polarity classification. (ii) We demonstrated that it is possible to estimate IS's parameter settings using domain similarity and domain complexity variance.

Future work includes (i) to fine-tune the estimation of r—e. g. via nonlinear regression of r; via estimation functions other than Equation (6.1) and Equation (6.2); via estimation of α and β—(ii) the combination of IS with more sophisticated DA schemes like SCL or SFA and (iii) the estimation of other algorithms' parameters using domain complexity and domain similarity. Apart from that, we are also interested in whether IS generalizes to other NLP tasks beyond polarity classification.

Furthermore, we will also investigate whether IS is beneficial for EA++ when selecting unlabeled target domain instances that are most similar to a "mixture" of the given labeled source and labeled target domain instances.

6.3 readability-based feature engineering

Textual characteristics—viz. domain complexity and domain similarity measures—can guide model selection in both in-domain and cross-domain document-level polarity classification (see Section 6.1 and Section 6.2). In this section we investigate how textual characteristics—viz. readability measures (see Chapter 3.3)—can be used in feature engineering for sentence-level subjectivity classification.

In a pilot study Remus (2011) improved the quality of sentence-level subjectivity classification by employing readability gradings as additional features in SVM models. Han et al. (2013) confirmed that SA benefits from being informed by readability gradings (see Chapter 3.3.2) and readability indicators (see Chapter 3.3.2). In this section we build upon Remus (2011)'s pilot study and extend it.

6.3.1 *Motivation*

While the meaning of Example (19)

(19) Nanometer-sized single crystals, or single-domain ultrafine particles, are often referred to as nanocrystals.[1]

is quite difficult to grasp, Example (20)

(20) Wills and Kate got into marriage mode.[2]

is easier to understand. This is because Example (19) not only exhibits a more complex syntactic structure than Example (20), but also uses domain-specific terminology with which many readers would not be familiar. Note that on the one hand, Example (20) is more colloquial than Example (19). On the other hand, Example (19) conveys a fact—nanometer-sized single crystals are called nanocrystals—while Example (20) conveys factual information—Prince William and Princess Kate will marry—and non-factual information—the bridal pair notably changed their usual routine. Both lexical and syntactical complexity relate to the notion of readability (see Chapter 3.3).

[1] Example (19) is taken from `http://en.wikipedia.org/Nanoparticles` (accessed January 8th, 2011).

[2] Example (20) is adapted from `http://www.thesun.co.uk/sol/homepage/news/3338590/Wills-and-Kate-in-marriage-mode.html` (accessed January 8th, 2011).

Therefore, we pose the following hypothesis: There is a relation between subjectivity in natural language text and its readability. We assume that knowing about a text's readability yields valuable information regarding its subjectivity.

6.3.2 *Related Work*

To the best of our knowledge readability has not been used to assess subjectivity in natural language text before Remus (2011)'s study. However, readability was used to e. g. evaluate the quality of user-created documents such as reviews(see Hoang et al., 2008), grade their helpfulness (see O'Mahony and Smyth, 2010) and summarize their sentiment (see Nishikawa et al., 2010).

The identification of features for subjectivity classification was studied extensively. Wiebe (2000) learns subjective adjectives using clustering based on distributional similarity. Wiebe et al. (2001) learn subjectivity clues using collocations. Riloff and Wiebe (2003) and Riloff et al. (2003) learn subjective nouns by bootstrapping extraction patterns. Wiebe et al. (2004) study how their previously identified features work together in concert. Riloff et al. (2006) learn complex lexical features and remove unnecessary ones using a feature subsumption hierarchy.

For subjectivity classification Yu and Hatzivassiloglou (2003) use NB models based on word uni-, bi-, and trigrams as well as POS tags and the polarity of words. Pang and Lee (2004) use a graph-based formalism. Wiebe and Riloff (2005) create high-precision but low-recall classifiers to distinguish subjective and objective sentences. Subsequently, they use these sentences to self-train an NB model. Raaijmakers and Kraaij (2008) use sub- and super-character n-grams as features in SVM models. Das and Bandyopadhyay (2009) use genetic algorithms. Wang and Liu (2011) use calibrated expectation maximization.

6.3.3 *Experiments*

We follow our assumption that knowing about a text's readability yields valuable information regarding its subjectivity: Initially, we analyze a gold standard for sentence-level subjectivity classification regarding its readability. We then use readability for feature engineering on this gold standard: given a sentence, we measure its readability, use it as feature, and classify the sentence as being either subjective or objective. Finally, we evaluate our engineered features on 2 more gold standards for sentence-level subjectivity classification.

Data Analysis

We analyze DSRC (see Chapter 2.1.1) regarding readability differences between subjective and objective sentences. DSRC contains 2,384 subjective sentences and 3,721 objective sentences. With binary subjectivity classification experiments in mind, we balance DSRC's subjective and objective sentences using undersampling, leaving us with 2,384 subjective and 2,384 objective sentences.

We contrast the readability of DSRC's subjective and objective sentences using 9 readability indicators—number of monosyllabic words, number of polysyllabic words, vocabulary complexity, i. e. number of uncommon words, sentence length in words, average word length in characters per sentence, noun/verb ratio, number of nominal forms, parse tree branching factor, and parse tree depth (see Chapter 3.3.2)—and 7 readability gradings—DRI, EL, FKS, FI, FORCAST, NREI, and SMOG grading (see Chapter 3.3.1). Table 24 shows the results.

Average number of monosyllabic words is notably different between subjective and objective sentences in DSRC: subjective sentences contain on average 0.45 less monosyllabic words than objective sentences, but 0.15 more polysyllabic words. Moreover, subjective sentences contain on average 0.21 uncommon words more than objective sentences. Subjective sentences are on average 0.26 words shorter than objective sentences. Parse trees of objective sentences are on average 0.46 nodes deeper than parse trees of subjective sentences.

Readability grading differences between subjective and objective sentences in DSRC are less pronounced than readability indicator differences. Notably different are NREI (0.48), EL (0.15), and FKS (0.15).

In summary, subjective sentences are on average shorter than objective sentences; they contain longer words; they contain more uncommon words; they contain less nominal forms. According to FI and NREI subjective sentences

Table 24.: Minima, maxima, averages, and standard deviations of readability indicator and readability grading measurements of subjective (Subj.) and objective (Obj.) sentences from DSRC as well as accuracies of SVM models on DSRC using as only feature the readability measurement.

READABILITY MEASURE	MINIMUM		MAXIMUM		AVERAGE			ST.DEV.		ACCURACY
	Subj.	Obj.	Subj.	Obj.	Subj.	Obj.	Δ	Subj.	Obj.	
Number of monosyllabic words	0	0	124	57	11.92	12.36	0.447	8.18	7.13	50.15
Number of polysyllabic words	0	0	64	25	5.38	5.23	0.154	3.93	3.33	50.13
Number of uncommon words	0	0	62	31	6.72	6.50	0.214	4.11	3.62	50.90
Sentence length in words	1	1	190	83	17.82	18.08	0.263	11.49	9.65	49.01
Average word length in characters	2.29	2.5	9.25	29	4.53	4.43	0.101	0.74	0.86	52.88
Noun/verb ratio	0	0	11	26	1.61	1.71	0.095	1.04	1.29	50.92
Number of nominal forms	0	0	65	27	5.32	5.74	0.427	3.81	3.26	51.45
Parse tree branching factor	1.2	1.2	1.64	1.68	1.49	1.49	0.005	0.04	0.05	52.61
Parse tree depth	3	3	39	34	10.91	11.37	0.455	4.42	4.26	51.60
DRI	-2.26	-2.22	41.7	17.75	3.71	3.66	0.058	2.61	2.18	49.96
EL	0	0	64	25	5.38	5.23	0.154	3.93	3.33	50.13
FKS	-10.49	-14.81	85.39	36.86	7.36	7.21	0.152	5.73	4.89	49.71
FI	3.24	3.24	34.48	17.01	5.98	6.05	0.065	1.9	1.61	50.06
FORCAST	11.08	14.16	20.43	20.43	19.12	19.07	0.052	0.87	0.78	49.98
NREI	-58.07	-62.77	-14.98	-15.14	-32.63	-32.15	0.477	4.04	3.92	53.32
SMOG grading	3.13	3.13	11.47	8.34	5.4	5.39	0.008	0.84	0.76	48.63

are less "readable" than objective sentences. DRI, EL, FKS, FORCAST, and SMOG signalize the opposite.

Feature Engineering

We now know that there are measurable readability differences between subjective and objective sentences in DSRC. Therefore, we use readability indicators and readability gradings as well as distributions of word frequency classes, POS tags, and dependency relations for feature engineering on DSRC: i. e., using DSRC, we identify features that discriminate between subjective and objective sentences.

We use all features within SVM models whose hyperparameters' C are optimized. All SVM models are evaluated in 10-fold CVs on DSRC.

READABILITY MEASURES We use as potential features 9 readability indicators and 7 readability gradings. Therefore, there are $\sum_{k=1}^{9} \binom{9}{k} = 511$ and $\sum_{k=1}^{7} \binom{7}{k} = 127$ possible feature combinations, respectively. We evaluate each feature combination. Due to space restrictions we only report the best performing feature combinations with respect to accuracy, and the accuracy of each single feature. Table 24 shows the evaluation results using single features.

The best performing feature combination of the 7 readability gradings consists of DRI, EL, FI, NREI, and SMOG: it yields an accuracy of 54.45, compared with an accuracy of 53.63 when using all 7 readability gradings.

The best performing feature combination of the 9 readability indicators consists of number of number of polysyllabic words, monosyllabic words, average word length in characters per sentence, number of uncommon words, number of nominal forms, parse tree branching factor, and parse tree depth: it yields an accuracy of 58.47, compared with an accuracy of 58.15 when using all 9 readability indicators.

Combining the best feature combinations of readability gradings and readability indicators yields an accuracy of 58.45, compared with an accuracy of 57.92 accuracy when using all 9 readability indicators and all 7 readability gradings. We conclude that—based on readability measures alone—we are able to distinguish between subjective and objective sentences well above chance level.

WORD FREQUENCY CLASS DISTRIBUTION Meier (1964, p. 69)'s observation motivates using the distribution of word frequency classes (see Chapter 3.3.2) as feature for sentence-level subjectivity classification:

> "Im gesamten Bereich echter Abwehr drängt sich in den bewegenden Augenblicken der Wortschatz sprachlicher Äußerungen (...) vorwiegend auf die Wortformen der Häufigkeits-Stufen I bis III zusammen."

Meier (1964) states that when we are aroused, we tend to use only high-frequency words, i. e. words that fall into "high" frequency classes, e. g. frequency class 1 to 3. We capitalize Meier (1964, p. 69)'s observation and use a sentence's word frequency class distribution as a feature for subjectivity classification.

Because we do not know how many word frequency classes to consider in a sentence's word frequency class distribution, we tune this parameter: We use as only feature in an SVM model the word frequency class distribution. Thereby, we vary the number of word frequency classes to consider from 1 to 27 with a step size of 1. We always add a "miscellaneous" word frequency class, in which all words fall that do not fall in any other frequency class, either because they are unknown, or because their word frequency class is larger. Word frequency classes are determined using a corpus from the Wortschatz project: `eng_news_2009`[3]. The evaluation results are shown in Figure 20.

Accuracy increases for an increasing number of word frequency classes, peaks for 23 word frequency classes at 57.9 and drops for more than 23 word frequency classes.

SYNTAX DISTRIBUTION Pak and Paroubek (2010)'s observe that objective and subjective texts differ in their use of syntactical devices, e. g. (i) objective texts contain more nouns, (ii) objective texts contain more verbs in the third person, (iii) subjective texts contain less comparative adjectives and more superlative adjectives, (iv) subjective texts contain more personal pronouns etc. Their observations motivate using POS tag and dependency relation distributions as features for sentence-level subjectivity classification.

[3] `http://corpora.uni-leipzig.de/`

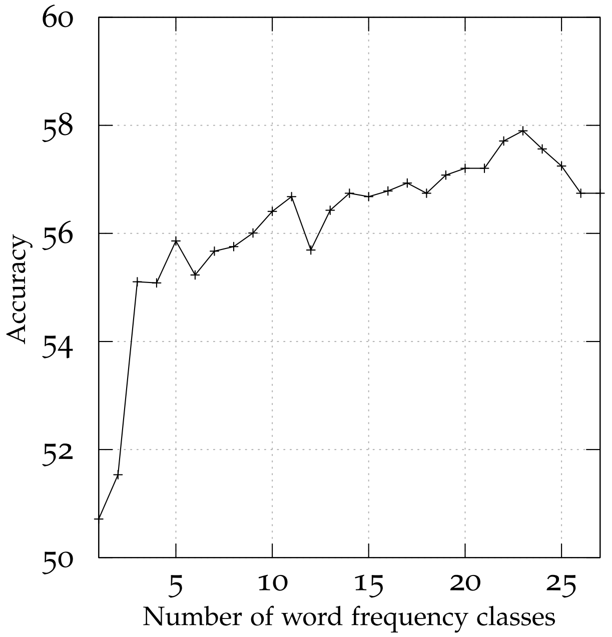


Figure 20.: Accuracy vs. number of word frequency classes.

Using as only feature the POS tag distribution yields an accuracy of 65.71. Using the dependency relation distribution yields an accuracy of 66.13. Using both POS tag and dependency relation distribution as features yields an accuracy of 67.12.

FEATURES IN CONCERT Table 25 shows the evaluation results when all features are used in concert, both with and without word unigrams as features[4]. Our baseline features are POS tag, dependency relation, and word frequency class distributions. The level of statistically significant difference (see Section 6.2.3) to our baseline features is indicated by $\star\star$ ($p < 0.005$) and $\star$ ($p < 0.05$).

Without word unigrams, using readability indicators and readability gradings in addition to our baseline features yields the best overall accuracy: 69.26 ($p < 0.005$). With word unigrams, our baseline features alone yield the best overall accuracy: 74.24 ($p < 0.01$). However, without our baseline features, using readability gradings and readabil-

[4] We refrain from using word bigrams as features, because an SVM model based solely on word unigrams yields an accuracy of 73.34 while an SVM model based on word uni- and bigrams yields an accuracy of 72.79.

Table 25.: Accuracy of engineered feature sets on DSRC with and without word unigrams. RI denotes readability indicators, RG denotes readability gradings, POS denotes POS tag distribution, DR denotes dependency relation distribution and FC denotes word frequency class distribution.

FEATURE SET	WITHOUT	WITH
$\mathrm{RI}_{\mathrm{best}}$	58.47	73.8
$\mathrm{RG}_{\mathrm{best}}$	54.45	73.05
$\mathrm{RI}_{\mathrm{best}}$, $\mathrm{RG}_{\mathrm{best}}$	58.45	74.03
$\mathrm{RI}_{\mathrm{all}}$	58.15	73.74
$\mathrm{RG}_{\mathrm{all}}$	53.63	72.98
$\mathrm{RI}_{\mathrm{all}}$, $\mathrm{RG}_{\mathrm{all}}$	57.92	73.63
POS	65.71	73.74
DR	66.13	73.82
FC	57.9	73.15
POS, DR	67.12	74.18
POS, DR, FC	67.94	74.24
POS, DR, FC, $\mathrm{RI}_{\mathrm{best}}$	68.63	73.97
POS, DR, FC, $\mathrm{RG}_{\mathrm{best}}$	68.74⋆	73.95
POS, DR, FC, $\mathrm{RI}_{\mathrm{best}}$, $\mathrm{RG}_{\mathrm{best}}$	68.68	73.91
POS, DR, FC, $\mathrm{RI}_{\mathrm{all}}$	69.2	74.05
POS, DR, FC, $\mathrm{RG}_{\mathrm{all}}$	69.08	73.91⋆
POS, DR, FC, $\mathrm{RI}_{\mathrm{all}}$, $\mathrm{RG}_{\mathrm{all}}$	69.26⋆⋆	74.01

Table 26.: Accuracy of engineered feature sets on MPQA v2.0 and SD v1.0.

feature set	mpqa v2.0	sd v1.0
Word unigrams	72.25	90.56
+ RI_{best}, RG_{best}	72.86⋆	90.69
+ RI_{all}, RG_{all}	73.01⋆⋆	90.63
+ FC, POS, DR	72.74	90.88
+ RI_{all}, RG_{all}, FC, POS, DR	73.22	90.73

ity indicators as additional features increases accuracy compared with using word unigrams as features alone.

Evaluation

To evaluate whether readability indicators and readability gradings as well as word frequency class, POS tag, and dependency relation distributions yield valuable information regarding a text's subjectivity or not, we perform a sentence-level subjectivity classification on MPQA v2.0 and SD v1.0. All models are evaluated in 10-fold CVs. MPQA v2.0 contains 5,380 subjective and 4,352 objective sentences; we balance MPQA v2.0 using undersampling leaving us with 4,352 subjective and 4,352 objective sentences. SD v1.0 contains 5,000 subjective and 5,000 objective sentences and hence does not require balancing. As baseline features we use word unigrams. We compare with that baseline the best performing—engineered—feature combinations as determined in the previous section.

In contrast to Remus (2011)'s pilot study, we employ linear kernels instead of RBF kernels in our SVM models. We also tune the SVM's hyperparameter C as described in Chapter 5.1.

results and discussion Table 26 shows evaluation results of sentence-level subjectivity classification on SD v1.0 and MPQA v2.0. All features added to our baseline increase accuracy, i. e. they all yield valuable information regarding a sentence's subjectivity.

On MPQA v2.0 an SVM model that uses as features word unigrams, all 9 readability indicators, all 7 readability gradings as well as word frequency class, POS tag, dependency

Table 27.: Accuracy of different subjectivity classification methods on MPQA v2.0 and SD v1.0.

DATASET	STUDY	ACCURACY
MPQA v2.0	Riloff et al. (2006)	74.9
	Raaijmakers and Kraaij (2008)	82.5
	Wang and Liu (2011)	71.5
SD v1.0	Pang and Lee (2004)	90–92
	Wang and Liu (2011)	90

relation distributions yields the best accuracy: 73.22. On SD v1.0 an SVM model that uses as features word unigrams as well as word frequency class, POS tag, dependency relation distributions performs best: an accuracy of 90.88.

Although the improvements in accuracy are small, both readability indicators and readability gradings as well as POS tag, dependency relation, and word frequency class distributions contribute positively to sentence-level subjectivity classification models.

COMPARISON We briefly compare the results we achieve using our engineered features to those reported in the literature (see Table 27): our models rank midfield both on MPQA v2.0 and SD v1.0.

6.3.4 *Conclusion and Future Work*

We have shown that using readability gradings, readability indicators and distributions of words and syntactical devices as features in addition to word unigrams yields accuracy improvements in sentence-level subjectivity classification. However, the accuracy improvements are small and often not statistically significant. Therefore, we were not able to validate our hypothesis that there is a relation between a text's readability and its subjectivity. Certainly, not only words but also meta-characteristics of words—e. g. their number of syllables—and meta-characteristics of sentences—e. g. their length—as well as the use of certain syntactical devices are related to a text's subjectivity. Based on such meta-characteristics alone—viz. readability measures—we were able to distinguish between subjective and objective sentences well above chance level.

Our results are based on Remus (2011)'s pilot study. However, we deviated from Remus (2011)'s pilot study in several important aspects:

- Remus (2011) used Wilson et al. (2005)'s subjectivity clues as features. In contrast, we did not incorporate any lexical resources (see Remus and Rill, 2013).

- Remus (2011) used only readability gradings. In contrast, we used both readability gradings and their "components", i. e. readability indicators. We also used more complex readability gradings as described in Oelke et al. (2010) and word frequency class distributions motivated by Meier (1964).

- Remus (2011) used RBF kernels in their SVM models and did not optimize the SVMs' hyperparameter C. In contrast, we used a linear kernel (see Chapter 4.1.2) but optimized the SVMs' hyperparameter C.

Future work includes (i) to further study the relation between readability gradings and readability indicators and (ii) to investigate whether we can predict a certain readability grading's or readability indicator's usefulness for sentence-level subjectivity classification based on its distribution among the objective and the subjective training instances (see Table 24).

7 NEGATION MODELING

Ich bin der Geist, der stets verneint!
Und das mit Recht; denn alles, was entsteht,
Ist wert, daß es zugrunde geht;
Drum besser wär's, daß nichts entstünde.
So ist denn alles, was ihr Sünde,
Zerstörung, kurz, das Böse nennt,
Mein eigentliches Element.

— *Johann Wolfgang von Goethe,*
Faust: Der Tragödie erster Teil

Negations as "Don't" in Example (21)

(21) *Don't* ask me!

are at the core of human language. Therefore, negations are commonly encountered in NLP tasks, e. g. textual entailment (e. g. Herrera et al., 2005; Delmonte et al., 2005). Negations are expressed via *negation words*—also referred to as *negation signals*—e. g. "don't x", "no findings of x", or "rules out x" and via morphology, e. g. the morphs "x-free", "x-less", or "un-x". In SA negation plays a special role (see Wiegand et al., 2010). Whereas Example (22) expresses positive sentiment, the only slightly different Example (23) expresses negative sentiment:

Negation word

Negation signal

(22) They are ⟨comfortable to wear⟩$^+$.

(23) They are ⟨*not* ⟨~~comfortable to wear~~⟩$^+$⟩$^-$.[1]

Therefore, attention is paid to negations frequently in compositional semantic approaches to SA (e. g. Moilanen and Pulman, 2007; Choi and Cardie, 2008; Neviarouskaya et al., 2009; Klenner et al., 2009; Remus and Hänig, 2011; Socher et al., 2012), as well as in bag of words-based ML techniques (e. g. Pang et al., 2002; Pak and Paroubek, 2010; Mohammad et al., 2013).

Research on *negation scopes (NSs)* and *negation scope detec-*

Negation scope

Negation scope detection

[1] In this work, struck out ~~words~~ are considered as negated.

tion (NSD) was primarily driven by biomedical NLP, particularly research on the detection of absence or presence of certain diseases in biomedical text. One of the most prominent studies in this field is Morante and Daelemans (2009), which identifies negation words and their scope using a variety of ML techniques and features. Only quite recently, the impact of NSD on SA became of increasing interest: Jia et al. (2009); Carrillo de Albornoz et al. (2010); Carrillo-de Albornoz and Plaza (2013); Johansson and Moschitti (2013) detect NSs using parse trees, typed dependencies, semantic role labeling and/or manually defined negation words. Hogenboom et al. (2011) compare several baselines for NSD, e. g. they consider as NS the rest of the sentence following a negation word, or a fixed window of 1 to 4 words following, preceding or around a negation word. Councill et al. (2010); Lapponi et al. (2012) study NSD based on Conditional Random Fields (CRFs). All these studies concur in their conclusion that SA—or more precisely polarity classification—benefits from NSD.

We model NSs in word n-gram feature space systematically and adopt recent advances in NSD. We believe this endeavor is worthwhile, because it allows machines to learn by themselves how negations modify the meaning of words, instead of being taught by manually defined and often ad hoc rules. As before in this thesis our study focuses on a data-driven ML-based approach to SA that operates in word n-gram feature space and does not rely on lexical resources, e. g. prior polarity dictionaries like SentiWordNet (see Esuli and Sebastiani, 2006). While various methods and features have been proposed for SA, such data-driven word n-gram models proved to be still competitive in recent studies (e. g. Barbosa and Feng, 2010; Agarwal et al., 2011; Saif et al., 2012).

This chapter is based on Remus (2013b) and is structured as follows: In the next section we describe our approach to modeling and representing negation in data-driven ML-based SA. In Section 7.2 we evaluate our approach in experiments for several SA subtasks and discuss their results. Additionally, we compare the effectivity of negation modeling (NM) in different domains. Finally, we draw conclusions and point out possible directions for future work in Section 7.3.

7.1 APPROACH

We now describe our approach to implicitly and explicitly modeling and representing negation in word n-gram feature space for data-driven ML-based SA. When *explicitly* modeling negation, we incorporate our knowledge of negation into the model; when *implicitly* modeling negation, we do not.

Implicit vs. explicit NM

7.1.1 *Implicit Negation Modeling*

As pointed out in Wiegand et al. (2010), negations are often *implicitly* modeled via higher order word n-grams, e. g. word bigrams (such as "*n't* return"), word trigrams (such as "*lack of* padding"), and word tetragrams[2] (such as "*denied* sending wrong size") etc. That aside, higher order word n-grams also implicitly capture other linguistic phenomena, e. g. comparatives ("larger than", "too much").

7.1.2 *Explicit Negation Modeling*

Although it is convenient, there is a drawback to solely relying on higher order word n-grams when trying to capture negations, i. e. modeling negations solely implicitly: long NSs as shown in Example (24) occur frequently (see Section 7.2.3), but typical word n-grams ($n < 5$) are not able to properly capture them.

(24) The leather straps have *never* ~~worn~~ ~~out~~ ~~or~~ ~~broken~~.[3]

Here e. g. a word trigram captures "never worn out" but not "never (...) broken". While a word 5-gram is able to capture "never (...) broken", learning models using word n-gram features with $n \geqslant 3$ usually leads to very sparse representations, depending on how much training data is available and how homogeneous (see Chapter 3.2.2) this training data is. In such cases learning from the training data what a certain higher order word n-gram contributes to the model is then backed up by only very little to almost none empirical findings. Therefore, we model negations also *explicitly*.

[2] Tetragrams are also referred to as quad-, four-, or 4-grams.

[3] Except Example 28 all examples in this chapter are taken or adapted from MDSD v2.0's reviews on apparel.

Negation Scope Detection

Vital to explicit NM is NSD. In Example (25) we need to detect that "stand up to laundering very well" is in the scope of "don't":

(25) They *don't* ~~stand up to laundering very well~~, in that they shrink up quite a bit.

For that purpose we employ a simple regular expression-based NSD, viz. NegEx[4] (see Chapman et al., 2001) and LingScope[5] (see Agarwal and Yu, 2010), a sophisticated CRF-based NSD trained on the BioScope corpus (see Vincze et al., 2008). NegEx was chosen as a strong *NSD baseline*. Its detected NSs are similar to a weak baseline NSD method frequently used (e. g. Pang et al., 2002; Mohammad et al., 2013): consider all words following a negation word as negated, up to the next punctuation. LingScope was chosen to represent the *state-of-the-art* in NSD. Moreover, both NegEx and LingScope are publicly available.

NSD baseline

NSD state-of-the-art

To improve NSD, we expand *contractions* like "can't" to "can not", "didn't" to "did not" etc. Note that while NegEx considers the negation itself to be part of the NS, we do not. NegEx's NSs are adjusted accordingly.

Contraction

Representation in Feature Space

Once NSs are detected, negated and non-negated word n-grams need to be explicitly represented in feature space. Therefore, we resort to a representation inspired by Pang et al. (2002), who create a new feature `NOT_f` when feature `f` is preceded by a negation word, e. g. "not" or "isn't".

Let $\mathcal{W} = \{w_i\}, i = 1, \dots, d$ be our word n-grams and let $\mathcal{X} = \{0, 1\}^d$ be our word n-gram feature space of size d, where for $x_j \in \mathcal{X}$, $x_{j_k} = 1$ denotes the presence of w_k and $x_{j_k} = 0$ denotes its absence. For each feature x_{j_k} we introduce an additional feature $\check{x}_{j_k}$ that encodes whether w_k appears negated ($\check{x}_{j_k} = 1$) or non-negated ($\check{x}_{j_k} = 0$). Thus, we obtain an *augmented feature space* $\check{\mathcal{X}} = \{0, 1\}^{2d}$. In $\check{\mathcal{X}}$ we are now able to represent whether a word n-gram

Augmented feature space

- w is present (encoded as $[1, 0]$),
- w is absent ($[0, 0]$),

[4] http://code.google.com/p/negex/

[5] http://sourceforge.net/projects/lingscope/

$$\begin{array}{r c}
\text{bit} & \multirow{10}{*}{} \\
\end{array}
\begin{array}{r}
\text{bit} \\
\text{don't} \\
\text{down} \\
\sout{\text{laundering}} \\
\text{quite} \\
\text{shrink} \\
\sout{\text{stand}} \\
\sout{\text{up}}/\text{up} \\
\sout{\text{very}} \\
\sout{\text{well}}
\end{array}
\begin{pmatrix}
1,0 \\
1,0 \\
0,0 \\
0,1 \\
1,0 \\
1,0 \\
0,1 \\
1,1 \\
0,1 \\
0,1
\end{pmatrix}$$

Figure 21.: A representation of "They don't stand up to laundering very well, in that they shrink up quite a bit." in $\breve{\mathcal{X}}$.

- w is present and negated ($[0, 1]$) or
- w is present both negated and non-negated ($[1, 1]$).

Representing an Example

Assume we employ naïve tokenization that simply splits at white spaces, ignore punctuation characters like "." and ",", and extract the presence and absence of $\mathcal{W}_{\text{uni}} = \{$"bit", "don't", "down", "laundering", "quite", "shrink", "stand", "up", "very", "well"$\}$, i. e. $\mathcal{W}_{\text{uni}}$ is our word unigram vocabulary. Representing Example (25) in $\breve{\mathcal{X}}$ results then in a stylized feature vector as shown in Figure 21.

Note the difference between "laundering" and "up" in Figure 21: while "laundering" is present only once and is negated and thus is represented as $[0, 1]$, "up" is present twice—once negated and once non-negated—and thus is represented as $[1, 1]$.

7.2 evaluation

We evaluate our NM approach in 3 common SA subtasks: in-domain and cross-domain document-level polarity classification (see Section 7.2.1) as well as sentence-level polarity classification (see Section 7.2.2).

Our setup for all experiments is—just as in previous experiments of our thesis—as follows: As classifiers we employ SVMs, but just as in Remus (2013b) we refrain from optimizing the SVMs' hyperparameter C—instead we fix C to 2.0.

As features we use word uni-, bi-, and trigrams extracted from the data[6]. Word bi- and trigrams model negation implicitly as described in Section 7.1.1. We perform no FS—neither stop words nor punctuation characters are removed because we do not make any assumption about which word n-grams carry sentiment and which do not. Additionally, we explicitly model the negation of these word uni-, bi-, and trigrams as described in Section 7.1.2. This is different from Pang et al. (2002)'s approach, who "(...) consider bigrams (and n-grams in general) to be an orthogonal way to incorporate context.". Explicitly modeling negation of higher order word n-grams—e. g. word bi- and trigrams—allows for learning that there is a difference between "doesn't work well" in Example (26) and "doesn't work" in Example (27),

(26) The stand *doesn't* ~~work~~ ~~well~~.

(27) The stand *doesn't* ~~work~~.

just as an ordinary word uni- and bigram model allows for learning the difference between "work" and "work well".

The in-domain document-level and sentence-level polarity classification experiments are construed as 10-fold CVs. As performance measure we report accuracy to be comparable to other studies (see Section 7.2.4). The level of statistically significant difference (see Section 6.2.3) to the corresponding base model without NM is indicated by ⋆⋆ ($p < 0.005$) and ⋆ ($p < 0.05$).

7.2.1 *Document-level Polarity Classification*

As gold standard for our in- and cross-domain document-level polarity classification experiments we use MDSD v2.0 (see Chapter 2.1.2).

[6] We also experimented with word tetragrams, but found that they do not contribute to the SVM models' discriminative power. This is not surprising, because in all used gold standards most word tetragrams appear only once. Their word tetragram distribution's relative entropy (see Chapter 3.2.1) is greater than 0.99, i. e. word tetragrams are almost uniformly distributed.

Table 28.: In-domain accuracy of NM averaged over all 10 domains from MDSD v2.0.

BASE MODEL	NSD	EXPLICIT NM FOR		
		uni	uni-bi	uni-bi-tri
uni	none	78.77		
	LingScope	80.06⋆⋆		
	NegEx	79.57⋆		
uni-bi	none	81.37		
	LingScope	81.73	**81.93**⋆⋆	
	NegEx	81.53	81.58	
uni-bi-tri	none	81.27		
	LingScope	81.65⋆	81.55	81.59⋆
	NegEx	81.28	81.3	81.28

In-domain

The evaluation results of our in-domain document-level polarity classification experiments averaged over all 10 domains from MDSD v2.0 are shown in Table 28.

A word uni- and bigram base model, LingScope for NSD and explicitly modeling negations for word uni- and bigrams yields the best overall result (81.93). This result is statistically significant different ($p < 0.005$) from the result the corresponding base model achieves using word uni- and bigrams alone (81.37).

Cross-domain

For all 10!/(10-2)! = 90 source domain–target domain pairs from MDSD v2.0 2,000 labeled source domain instances (1,000 positive and 1,000 negative) and 200 labeled target domain instances (100 positive and 100 negative) are used for training. 1,800 labeled target domain instances (900 positive and 900 negative) are used for testing. If required by the method, 2,000 unlabeled target domain instances are available for training. This is a typical semi-supervised DA setting as described in Chapter 6.2.

We employ 3 methods for cross-domain polarity classification: ALL, IS, and EA++ (see Chapter 6.2). Table 29 shows the evaluation results for ALL, Table 30 shows the evaluation results for IS and Table 31 shows the evaluation results for EA++.

Table 29.: Cross-domain accuracy of NM in All averaged over all 90 domain-pairs from MDSD v2.0.

BASE MODEL	NSD	EXPLICIT NM FOR		
		uni	uni-bi	uni-bi-tri
uni	none	74.25		
	LingScope	75.46⋆⋆		
	NegEx	75.35⋆⋆		
uni-bi	none	76.61		
	LingScope	77.23⋆⋆	**77.31**⋆⋆	
	NegEx	77.18⋆⋆	77.13⋆⋆	
uni-bi-tri	none	76.44		
	LingScope	77.01⋆⋆	77.13⋆⋆	77.12⋆⋆
	NegEx	76.97⋆⋆	76.83⋆⋆	76.81⋆⋆

Table 30.: Cross-domain accuracy of NM in IS averaged over all 90 domain-pairs from MDSD v2.0.

BASE MODEL	NSD	EXPLICIT NM FOR		
		uni	uni-bi	uni-bi-tri
uni	none	74.62		
	LingScope	75.75⋆⋆		
	NegEx	75.53⋆⋆		
uni-bi	none	76.90		
	LingScope	77.74⋆⋆	**77.75**⋆⋆	
	NegEx	77.43⋆⋆	77.38⋆⋆	
uni-bi-tri	none	76.72		
	LingScope	77.52⋆⋆	77.63⋆⋆	77.61⋆⋆
	NegEx	77.30⋆⋆	77.24⋆⋆	77.19⋆⋆

For All, just like for in-domain polarity classification, a word uni- and bigram base model, LingScope for NSD and explicitly modeling negations for word uni- and bigrams yields the best overall result (77.31, $p < 0.005$). The same applies to IS (77.75, $p < 0.005$). For EA++, a word uni- and bigram base model, NegEx for NSD and explicitly modeling negations for word unigrams yields the best overall result (77.5, $p < 0.005$). A word uni-, bi-, and trigram base model, LingScope for NSD and explicitly modeling negations for word unigrams performs almost as good and yields 77.48 accuracy ($p < 0.005$).

Table 31.: Cross-domain accuracy of NM in EA++ averaged over all 90 domain-pairs from MDSD v2.0.

BASE MODEL	NSD	EXPLICIT NM FOR		
		uni	uni-bi	uni-bi-tri
uni	none	74.5		
	LingScope	75.46⋆⋆		
	NegEx	75.1⋆⋆		
uni-bi	none	75.95		
	LingScope	76.42⋆⋆	76.54⋆⋆	
	NegEx	**77.5**⋆⋆	77.34⋆⋆	
uni-bi-tri	none	75.75		
	LingScope	76.22⋆⋆	77.46⋆⋆	**77.48**⋆⋆
	NegEx	76.14⋆⋆	77.19⋆⋆	77.08⋆⋆

Table 32.: Accuracy of NM on SPD v1.0.

BASE MODEL	NSD	EXPLICIT NM FOR		
		uni	uni-bi	uni-bi-tri
uni	none	74.56		
	LingScope	75.85⋆⋆		
	NegEx	75.08		
uni-bi	none	77.69		
	LingScope	77.93	77.55	
	NegEx	77.72	77.36	
uni-bi-tri	none	77.62		
	LingScope	77.85	77.99	**78.01**⋆
	NegEx	77.71	77.23	77.36

7.2.2 *Sentence-level Polarity Classification*

As gold standards for sentence-level polarity classification we use SPD v1.0 (see Chapter 2.1.8) and SE-2007-T14D (see Chapter 2.1.6). Evaluation results for SPD v1.0 are shown in Table 32, evaluation results for SE-2007-T14D are shown in Table 33.

For SPD v1.0, a word uni-, bi-, and trigram base model, using LingScope for NSD and explicitly modeling negations for word uni-, bi-, and trigrams yields the best result (78.01, $p < 0.05$). For SE-2007-T14D, a word uni- and bigram base model, NegEx for NSD and explicitly modeling negations

Table 33.: Accuracy of NM on SE-2007-T14D.

BASE MODEL	NSD	EXPLICIT NM FOR		
		uni	uni-bi	uni-bi-tri
uni	none	64.73		
	LingScope	65.18		
	NegEx	64.73		
uni-bi	none	66.43		
	LingScope	66.79	66.88	
	NegEx	66.96	**67.41**⋆	
uni-bi-tri	none	66.07		
	LingScope	66.7	66.52	66.7
	NegEx	66.52	66.61	66.61

Table 34.: Precision, recall and f-score of NSD methods.

NSD METHOD	PRECISION	RECALL	F-SCORE
LingScope	0.696	0.656	0.675
NegEx	0.407	0.5	0.449

for word uni- and bigrams yields the best result (67.41, $p < 0.05$).

7.2.3 *Discussion*

Intuitively, explicit NM benefits from high quality NSD: the more accurate the NSD, the more accurate the explicit NM. This intuition is met by our results. As shown by Agarwal and Yu (2010), LingScope is often more accurate than NegEx on biomedical data. This also applies to review data: We evaluated LingScope and NegEx on 500 sentences that were randomly extracted from SPD v1.0 and annotated for their NSs by us. Table 34 shows the evaluation results. LingScope clearly outperforms NegEx with respect to precision and recall.

So although genre and domain of BioScope's data—on which LingScope and NegEx were trained and tested—differ from genre and domains of MDSD v2.0 and SPD v1.0, models learned using LingScope as NSD yield the best or almost best results for all our SA subtasks evaluated on MDSD v2.0 and SPD v1.0. On another genre—SE-2007-T14D's news headlines—models learned using NegEx as NSD yield the best results. SE-2007-T14D is different because it contains almost no negations (see Table 35). Compared with MDSD v2.0 and SPD v1.0 it contains more verbal negations which LingScope does not detect, but NegEx does, e. g. "denies" as in Example (28):

(28) China *denies* ~~reports of North Korean apology~~.

Compared with ordinary word n-gram models that do not model negation ($n = 1$) or model negation only implicitly ($2 \leqslant n \leqslant 3$), word n-gram models that additionally model negation explicitly achieve statistically significant improvements—given an accurate NSD method.

To shed some light on the differences between the results for the evaluated SA subtasks and their corresponding gold standards, we analyze how many and what kind of NSs the NSD methods detect (see Table 35). Naturally, these findings rely on LingScope's and NegEx's definition of what an NS actually is. According to both LingScope's and NegEx's implementation an NS may encompass coordinations like "or" as in Example (24). Generally, LingScope detects more negations than NegEx. NSs detected by LingScope are on average shorter than those detected by NegEx, thus they are more precise. While LingScope and NegEx detect negations in about 67% of all documents in MDSD v2.0, only about 20% of all sentences in SPD v1.0 contain detected negations. Not surprisingly, even less NSs are detected in SE-2007: only about 3% of all headlines contain detected negations.

Note that only very little NSs have length 1, i. e. span 1 word unigram, but many NSs have length 4 or longer, i. e. span 4 word unigrams or more. That confirms the need for explicit NM as mentioned in Section 7.1.2, but also hints at a data sparsity problem: certain parts of word n-grams in the scope of negations may re-occur, but the same NS basically never appears twice. For MDSD v2.0 and LingScope as NSD, on average each NS overlaps only on 0.18 positions with each other NS. Thus, overlaps as shown in Example (29) and Example (30) where "buy" appears in both NSs are scarce:

(29) *Don't* ~~buy these shoes for running~~!

(30) Do *not* ~~buy them~~ unless you like getting blisters.

Table 35.: Statistics of NSs: # number of detected NSs, $\bar{\#}$ average number of detected NSs per document/sentence, $w/$ percentage of documents/sentences with detected NSs, $\bar{l}$ average NS length in tokens, $l = 1, 2, 3, \geqslant 4$ distribution of detected NSs of the according length.

GOLD STANDARD	NSD	#	$\bar{\#}$	$w/$	$\bar{l}$	$l = 1$	$l = 2$	$l = 3$	$l \geqslant 4$
MDSD v2.0	LingScope	3,187.5	1.6	67.4%	6.6	1.4%	13.5%	12.7%	72.5%
	NegEx	2,971.2	1.5	67.3%	10.7	1.8%	6.6%	8.4%	83.2%
SPD v1.0	LingScope	2,339	0.2	20.5%	6.8	2.2%	9.8%	13.8%	74.2%
	NegEx	2,085	0.2	19.6%	12.1	1.9%	3.8%	5.9%	88.3%
SE-2007-T14D	LingScope	33	0.03	2.9%	3.2	3.0%	51.5%	9.1%	36.4%
	NegEx	31	0.03	2.8%	4.6	16.1%	9.7%	6.5%	67.7%

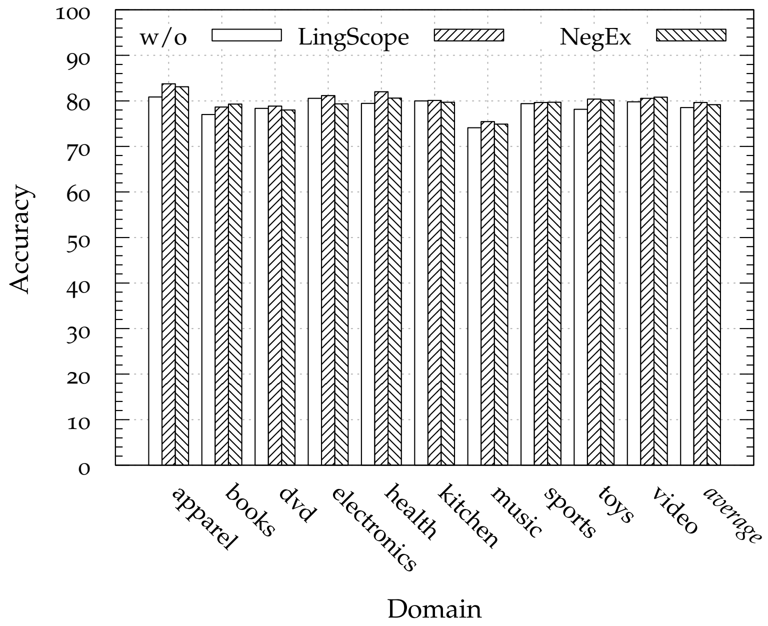


Figure 22.: Accuracy of SVM models on different domains from MDSD v2.0 with and without NM. LingScope and NegEx are used as NSD methods.

The picture is similar for SPD v1.0 with an overlap in on average 0.22 positions and worse for SE-2007-T14D with an overlap in on average 0.02 positions.

Efficiency in Different Domains

We now compare the efficiency of our NM approach in different domains of MDSD v2.0. Figure 22 shows the accuracies SVM models yield based on word unigrams and NM for word unigrams using either LingScope or NegEx as NSD.

Explicit NM using LingScope improves accuracy for 10/10 domains. Explicit NM using NegEx improves accuracy for 7/10 domains. Explicit NM using LingScope outperforms explicit NM using NegEx for 7/10 domains. We did not find any apparent (and statistically significant) correlations between the accuracy gain that explicit NM yields and any of the statistics shown in Table 35.

Lesson Learned

We conclude our discussion with a "lesson learned": Apart from the negation representation scheme we used through-

WORD STATE	P/N/A	P/PN/A	P/PN/N/A
present	$[1,0]$	$[1,0]$	$[1,0]$
present-negated	$[0,1]$	$[1,1]$	$[1,1]$
negated	$[0,1]$	$[1,1]$	$[0,1]$
absent	$[0,0]$	$[0,0]$	$[0,0]$

Table 36.: Representations of $w \in \mathcal{W}$ in $\breve{\mathcal{X}}$ when w is present, present both negated and non-negated (present-negated), present and negated (negated), and absent as described in Section 7.1.2.

out this—viz. P/PN/N/A[7]—we also experimented with 2 other negation representation schemes—viz. P/N/A and P/PN/A[8]. Table 36 summarizes all 3 negation representation schemes. We found that P/PN/N/A is generally superior to P/N/A and P/PN/A.

7.2.4 *Comparison*

For sentence-level polarity classification on SPD v1.0, our best performing model (78.01) outperforms 3 state-of-the-art models: Nakagawa et al. (2010)'s dependency tree-based CRFs (77.3), Socher et al. (2012)'s linear matrix-vector recursion (77.1), and Socher et al. (2011)'s semi-supervised recursive autoencoders (77.7). It is beaten by Socher et al. (2012)'s matrix-vector recursive neural network (79) and Wang and Manning (2012)'s SVM models with NB features (79.4).

For sentence-level polarity classification on SE-2007-T14D, we mimic the evaluation scheme of Strapparava and Mihalcea (2007)'s SemEval-2007 Task 14: Instead of a binary polarity classification as described in Section 7.2.2 we perform a ternary polarity classification, in which a polarity intensity of $[-100, -50]$ is mapped to negative, $(-50, 50)$ is mapped to neutral and $[50, 100]$ is mapped to positive. Our models are trained on 250 headlines from the trial data and tested on the remaining 1,000 headlines from the test data. Our best performing model yields an accuracy of

[7] P/PN/N/A stands for present, present-negated, negated, and absent.

[8] P/N/A stands for present, negated, and absent; P/PN/A stands for present, present-negated, and absent.

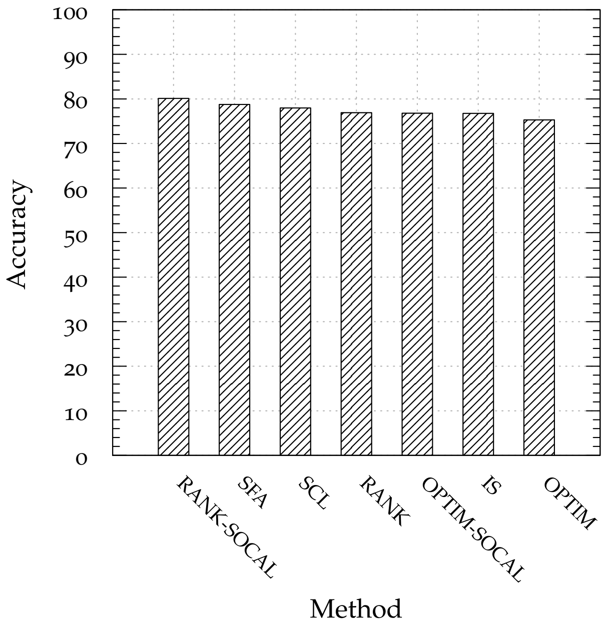


Figure 23.: Cross-domain accuracy of DA methods and IS with explicit NM on MDSD v2.0.

59.9. It outperforms all systems that originally participated in SemEval-2007 Task 14 (29–55.1).

For in-domain document-level polarity classification on MDSD v2.0, Ponomareva and Thelwall (2012a) report results for 7 domains (dvd, books, electronics, health, kitchen, music, toys) out of the 10 domains we used in our experiments. Their SVM models use word unigrams and word stem bigram as features and yield 80.29 average accuracy; on the same 7 domains our best performing model yields 81.49 average accuracy.

For cross-domain document-level polarity classification on MDSD v2.0, our best performing model is inferior compared to more sophisticated DA methods—SCL (see Blitzer et al., 2007), SFA (see Pan et al., 2010) as well as RANK, OPTIM, RANK-SOCAL, and OPTIM-SOCAL (see Ponomareva and Thelwall, 2012b)—all of which are evaluated on 4 domains (dvd, books, electronics, kitchen) out of the 10 domains we used in our experiments. Their evaluation results are shown in Figure 23.

In summary, a purely data-driven ML-based SA approach with NM for word n-grams proves to be competitive in several common SA subtasks.

7.3 CONCLUSION AND FUTURE WORK

We conclude that data-driven ML-based SA models that operate in word n-gram feature space benefit from explicit NM. In turn, explicit NM benefits from (i) high quality NSD like LingScope and (ii) modeling not only negation of word unigrams, but also of higher order word n-grams, especially word bigrams.

These insights suggest that explicitly modeling semantic compositions is promising for data-driven ML-based SA. Given appropriate scope detection methods, our approach may for example easily be extended to model other *valence shifters* (see Polanyi and Zaenen, 2006), e. g. intensifiers like "very" or "many", or *hedges* (see Lakoff, 1973) like "may" or "might", or even implicit negation in the absence of negation words (see Reyes and Rosso, 2013). Our approach is also easily extensible to other word n-gram weighting schemes aside from encoding pure presence or absence, e. g. weighting using relative frequencies or tf-idf. The feature space then simply becomes $\breve{\mathcal{X}} = \mathbb{R}^{2d}$.

Valence shifter

Hedge

Note that modeling negations explicitly may be particular interesting in environments with strong restrictions of physical memory or computing power. Then, word unigram models with NM for word unigrams may present a viable alternative to word uni- and bigram models, as the resulting feature spaces are much smaller. E. g., the average feature space size of word unigram models for MDSD v2.0's 10 domains is $|\mathcal{X}_{\text{uni}}| = 17822.6$, so $|\breve{\mathcal{X}}_{\text{uni}}| = 2 \cdot |\mathcal{X}_{\text{uni}}| = 35645.2$. In contrast, the average feature space size for word uni- and bigram models is $|\mathcal{X}_{\{\text{uni, bi}\}}| = 114891.1$. Thus, $|\mathcal{X}_{\{\text{uni, bi}\}}|$ is more than 3 times larger than $|\breve{\mathcal{X}}_{\text{uni}}|$, while the achieved accuracies are similar.

Future work encompasses model fine-tuning, e. g. accounting for NSs in the scope of other negations as in Example (31)

(31) I ⟨*don't* ~~care that they are~~ ⟨~~*not*~~ ~~really leather~~⟩⟩.

and employing generalization methods (see Turian et al., 2010) such as word grouping via Brown clustering (see Brown et al., 1992) or distributional semantics (see Baroni and Lenci, 2010) to tackle data sparsity when learning the effects of negations, modeled both implicitly and explicitly.

Part III

APPLICATION & DISCUSSION

8 A CASE STUDY

In this chapter we carry out a case study on document-level polarity classification with previously unused data—SE-2013-T2BD—and apply our findings from Chapter 5, Chapter 6 and Chapter 7. For SE-2013-T2BD we

1. estimate our SA approach's accuracy (see Section 8.1).
2. let domain complexity guide us in model selection (see Section 8.2), viz. we decide
   a) what word n-gram model order to employ in our SA approach (see Section 8.2.1).
   b) whether to employ aggressive or conservative word n-gram FS in our SA approach (see Section 8.2.2).
3. add word n-gram NM to our SA approach (see Section 8.3).

8.1 PERFORMANCE ESTIMATION

To transfer our performance estimator (see Chapter 5.4) and model selectors (see Chapter 6.1.1 and Chapter 6.1.2) from the dataset they are trained on—i. e. MDSD v2.0—to a new dataset—i. e. SE-2013-T2BD—their non-textual characteristics have to be similar (see Chapter 5.4): e. g. they have to be designated for the same SA subtask and they have to be of similar size. Therefore, we modify SE-2013-T2BD to match MDSD v2.0's properties:

1. We binarize SE-2013-T2BD, i. e. we ignore the neutral class. We transform the hitherto ternary classification problem (positive vs. negative vs. neutral) into a binary classification problem (positive vs. negative).
2. We balance SE-2013-T2BD by undersampling the majority class, which leaves us with 2,076 positive and 2,076 negative instances.

From hereon we refer to this binarized, balanced version of SE-2013-T2BD as SE-2013-T2BD*.

Table 37.: Domain complexity of SE-2013-T2BD*.

DOMAIN COMPLEXITY	$n = 1$	$n = 2$	$n = 3$
Percentage of rare n-grams	0.782	0.956	0.992
n-gram type/token ratio	0.55	0.939	0.989
n-gram relative entropy	0.916	0.994	0.999
n-gram homogeneity	0.483	0.915	0.984

Table 38.: Accuracy of our SA approach based on word unigrams, word uni- and bigrams as well as word uni-, bi-, and trigrams on SE-2013-T2BD*.

BASED ON	ESTIMATED	ACTUAL
Word unigrams	77.18	75.29
Word uni- and bigrams	77.37	75.82
Word uni, bi-, and trigrams	77.53	75.92

Recall that our performance estimator and model selectors are based on LR models that use as predictor a dataset's domain complexity. Table 37 shows SE-2013-T2BD*'s domain complexity measurements for word uni-, bi-, and trigrams.

To estimate our SA approach's accuracy on SE-2013-T2BD*, we train a performance estimator on MDSD v2.0's 10 domains. We use Huber loss as loss function, averaged homogeneity as predictor, and accuracy as response (see Chapter 5.4). Using SE-2013-T2BD*'s homogeneity measurements as inputs, our performance estimator outputs our SA approach's accuracies as shown in Table 38.

Table 38 also shows the accuracies that our SA approach actually achieves in 10-fold CVs. Our performance estimator overestimates our SA approaches' accuracies between 1.55 and 1.89 accuracy points.

8.2 DOMAIN COMPLEXITY-BASED MODEL SELECTION

8.2.1 *Word n-gram Model Order*

To decide for SE-2013-T2BD* what word n-gram model order to employ in our SA approach, i. e. whether to employ an SVM model based on word unigrams, word uni- and bigrams, or word uni-, bi-, and trigrams, we use our performance estimator in a model selector for word n-gram model order (see Chapter 6.1.1).

According to the estimated accuracies, an SVM model based on word uni-, bi-, and trigrams performs better than one based on word uni- and bigrams, which in turn performs better than the one based on word unigrams alone. According to the actual accuracies, the same is true. Therefore, our model selector correctly selects the SVM model based on word uni-, bi-, and trigrams.

8.2.2 *Word n-gram Feature Selection*

To decide for SE-2013-T2BD⋆ whether to employ aggressive or conservative word n-gram FS, we train a model selector on MDSD v2.0's 10 domains. We use Tukey's biweight as loss function, relative entropy as predictor, and ideal CO as response (see Chapter 6.1.2).

Using the relative entropy measurement of SE-2013-T2BD⋆ as input, our model selector estimates a CO of -156%, which of course is impossible: we underestimate the CO[1]. Therefore, we fall back to using the average ideal CO of 45% (see Chapter 6.1.2). Using the average ideal CO for word unigram FS via IG yields an accuracy of 76.52, compared with an accuracy of 75.29 when performing no FS. The actual ideal CO is 16% and yields an accuracy of 77.34 when using only word unigrams as features and 76.15 when using word uni-, bi-, and trigrams as features.

8.3 NEGATION MODELING

In Section 8.2 we decided for SE-2013-T2BD⋆ to

- employ an SVM model based on word uni-, bi-, and trigrams,
- employ an aggressive word n-gram FS in the SVM model, i. e. to reduce its word unigram vocabulary to 45% of its original size by selecting the most discriminative features via IG,

in our SA approach.

[1] The model selector is trained on relative entropy values ranging from 0.887 to 0.893. SE-2013-T2BD⋆'s relative entropy is 0.782 and hence is "out of range".

Table 39.: Accuracy of our SA approach on SE-2013-T2BD* with and without FS as well as with and without explicit NM.

FS	NM	
	with	without
with	77.1	76.52
without	76.21	75.92

We now add NM to our SA approach (see Chapter 7). We use LingScope for NSD as it generally yields the best results for document-level polarity classification (see Chapter 7.2.1). Because we do not know for which word n-gram to model negation explicitly, we model it for all word n-grams in our model, i. e. for word uni-, bi-, and trigrams. Table 39 shows the evaluation results of our SA approach with and without explicit NM as well as with and without FS.

Compared with our SA approach without explicit NM, our SA approach with explicit NM gains 0.29 in accuracy. When we additionally perform FS (see Section 8.2.2), we gain another 0.89 in accuracy.

8.4 CONCLUSION

In our case study we successfully transferred the core methods of our thesis—domain complexity-based performance estimation, domain complexity-based model selection and NM—to SE-2013-T2BD*, an SA gold standard hitherto unused in our thesis.

Based on its domain complexity, we estimated our SA approach's accuracy on SE-2013-T2BD*. Using our model selector for word n-gram model order, we then decided to employ an SVM model based on word uni-, bi-, and trigrams (75.92 accuracy). Using our model selector for word n-gram FS, we then decided to employ quite aggressive word n-gram FS by reducing SE-2013-T2BD*'s word unigram vocabulary to 45% of its original size via IG (76.52 accuracy). Finally, we added explicit NM for word uni-, bi-, and trigrams (77.1 accuracy).

9 SUMMARY AND CONCLUSION

> Bah, the latest news is not the last.
> — *Samuel Beckett,*
> *The Unnamable*

Our thesis investigated genre and domain dependencies in SA. We conclude by revisiting the core hypothesis that prompted our research and by summarizing our findings.

As outlined in Chapter 1, SA—and NLP in general—faces several challenges. A major challenge is NLP's dependence on certain contextual parameters, e. g. the point of time at which something is expressed or the social-cultural background against which something is expressed. Most importantly, many NLP methods are dependent on the genre and the domain in which something is expressed. Therefore, our core hypothesis is: *SA is genre and domain dependent*.

In Chapter 2 we introduced SA gold standards that function as representative samples of certain genres and domains. In Chapter 3 we discussed textual characteristics that characterize these gold standards and uncover similarities and dissimilarities between them. In Chapter 4 we described the ML algorithms that we use to approach the SA subtasks addressed in this thesis: polarity and subjectivity classification.

In Chapter 5 we then presented a prototypical SA approach: a supervised, data-driven ML model that is based solely on lexical features, viz. word n-grams. In a first step, we then *validated our core hypothesis*: We showed that different genres and domains differ in their textual characteristics, viz. their domain complexity. We also showed that our SA approach performs differently on gold standards that originate from differing genres and domains, but performs similarly on gold standards that originate from resembling genres and domains. We found a strong linear relation between our SA approach's accuracy on a particular gold standard and its domain complexity. Based solely on a gold standard's domain complexity, we were then able to *estimate our SA approach's accuracy* on this particular gold standard.

In a second step, we used domain complexity measures and domain similarity measures to exploit genre and domain specifics for two tasks. (i) We let domain complexity measures *guide us in prototypical model selection* processes for in-domain document-level polarity classification (see Chapter 6.1): to decide what word n-gram model order to employ in an SVM and to decide whether to employ aggressive or conservative word n-gram FS for an SVM model. (ii) We let domain complexity and domain similarity measures *guide us in a semi-supervised DA* scenario for which we proposed a novel DA scheme: IS (see Chapter 6.2). Subsequently, we applied IS to cross-domain document-level polarity classification. In a third step, we *exploited readability measures for feature engineering* (see Chapter 6.3). Based on the observation that human beings tend to use mainly high-frequency words when they are aroused, we *proposed a novel readability measure*: the distribution of word frequency classes. Subsequently, we adopted several readability measures for sentence-level subjectivity classification.

In Chapter 7 we *generalized a framework for modeling and representing negation* in ML-based SA. Subsequently, we applied this framework to in-domain and cross domain polarity classification on document-level and in-domain polarity classification on sentence-level. We investigated (i) the relationship between implicit and explicit NM, (ii) the influence of NSD methods, and (iii) the efficiency of NM in different domains. Finally, we carried out a *case study* in Chapter 8, in which we *successfully transferred the core methods* of our thesis—domain complexity-based performance estimation, domain complexity-based model selection, and NM—to a gold standard that originates from a genre and domain hitherto not used in this thesis.

9.1 SCIENTIFIC CONTRIBUTION

As discussed in Chapter 1, genre and domain dependence is not a particular property of SA and its subtasks but applies to many NLP techniques. Therefore, certain findings of our thesis may generalize beyond SA and may apply to NLP in general. To investigate which findings generalize and which do not is left to future work (see Section 9.2):

our thesis focused on SA, which is a prime example of genre and domain dependence (see Chapter 1.1).

In general, our thesis contributes to a methodology to deal with genre and domain specifics in SA. This methodology (i) enables us to estimate "how far we can get" using a prototypical SA approach and (ii) guides us "how to choose the right tools for the job". In particular, our scientific contributions—which lead to several publications—are:

- We deepened the understanding about a prototypical, supervised ML-based SA approach and its relation to the gold standard it is trained and tested on:
  - We found that data-driven models are superior to dictionary-based models (see Remus and Rill, 2013).
  - We introduced several domain complexity measures (see Bank et al., 2012; Remus, 2012) and investigated their sample size-dependence (see Remus and Bank, 2012).
  - We estimated our SA approach's accuracy using domain complexity measures (see Remus and Ziegelmayer, 2014).
  - We let domain complexity measures guide us in prototypical model selection processes (see Remus and Ziegelmayer, 2014).
  - We let domain complexity and domain similarity measures guide us in DA (see Remus, 2012).
- We used readability measures for feature engineering (see Remus, 2011).
- We generalized a framework for modeling and representing negation in ML-based SA (see Remus, 2013a,b).

9.2 LIMITATIONS AND FUTURE WORK

With each decision we made in our thesis there come certain limitations regarding the conclusions we can draw from our results:

- Initially, we experimented with ML techniques other than SVMs, but we later focused solely on SVMs. Therefore, we cannot be sure whether our findings generalize beyond SVMs.

- We focused on SA subtasks that may be seen as text classification problems, i.e. polarity and subjectivity classification. Other SA subtasks, e.g. opinion holder or opinion target extraction, may be seen as sequential labeling problems. Therefore, we cannot conclude that our findings generalize beyond SA classification subtasks.

- There is a gap between what some writer of some text intended to convey to the readers of its text—the writer's intention—and what some reader actually perceives when reading the text—the reader's perception (see Maks and Vossen, 2013). We often used gold standards that were labeled via distant supervision, e.g. MDSD v2.0 and T-MDSD. Distant supervision relies on some proxy, e.g. stars given to a review by its author or emoticons added to a tweet by its author. Therefore, such gold standards are labeled according to the writer's intention rather than the reader's perception. However, the reader's perception is what we are interested in most of the times.

From these limitations immediately follows what could be future work. For the small picture future work is outlined in Chapter 5, Chapter 6 and Chapter 7. For the big picture the most important future work is to investigate whether our findings generalize beyond

- SA classification subtasks, e.g. to SA sequential labeling subtasks.

- SA classification subtasks to other NLP classification tasks, e.g. topic classification.

- simple to more sophisticated SA approaches, e.g. deep learning (e.g. Socher et al., 2013).

- supervised to unsupervised ML-based SA, e.g. via topic models (e.g. Mei et al., 2007; Lin and He, 2009).

- SVMs to other ML algorithms, e.g. NB.

In this regard, it is also important to investigate the relation between a gold standard's domain complexity and its class boundary complexity (see Chapter 3.2.3).

EPILOGUE

When we analyze what people say, think, or feel—manually or automatically—we should always bear in mind that it is our obligation to preserve and to protect their—our—*privacy*. It is up to every researcher and practioner to decide in each and every situation whether to tap SA's and NLP's full potential or not.

BIBLIOGRAPHY

Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. Sentiment analysis of twitter data. In *Proceedings of the Workshop on Languages in Social Media (LSM*, pages 30–38, 2011. (Cited on p. 114)

Shashank Agarwal and Hong Yu. Biomedical negation scope detection with conditional random fields. *Journal of the American Medical Informatics Association*, 17(6):696–701, 2010. (Cited on p. 116 und 122)

Rehan Akbani, Stephen Kwek, and Nathalie Japkowicz. Applying support vector machines to imbalanced datasets. In *Proceedings of the 10th European Conference on Machine Learning (ECML)*, pages 39–50, 2004. (Cited on p. 19)

Anthony Aue and Michael Gamon. Customizing sentiment classifiers to new domains: A case study. In *Proceedings of the 5th International Conference on Recent Advances in Natural Language Processing (RANLP)*, 2005. (Cited on p. 90 und 91)

Alexandra Balahur and Ralf Steinberger. Rethinking sentiment analysis in the news: From theory to practice and back. In *Proceeding of the 1st Workshop on Opinion Mining and Sentiment Analysis (WOMSA)*, 2009. (Cited on p. 6)

Alexandra Balahur and Marco Turchi. Multilingual sentiment analysis using machine translation? In *Proceedings of the 3rd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis (WASSA)*, pages 52–60, 2012. (Cited on p. 11)

Alexandra Balahur, Ralf Steinberger, Mijail Kabadjov, Vanni Zavarella, Erik Van Der Goot, Matina Halkia, Bruno Pouliquen, and Jenya Belyaeva. Sentiment analysis in the news. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC)*, 2010. (Cited on p. 6)

Mathias Bank, Robert Remus, and Martin Schierle. Textual characteristics for language engineering. In *Proceedings of* *the 8th International Conference on Language Resources and Evaluation (LREC)*, pages 515–519, 2012. (Cited on p. 25, 28, 29, 30, 31, und 137)

Luciano Barbosa and Junlan Feng. Robust sentiment detection on twitter from biased and noisy data. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING)*, pages 36–44, 2010. (Cited on p. 114)

Marco Baroni and Alessandro Lenci. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721, 2010. (Cited on p. 128)

Ron Bekkerman and James Allan. Using bigrams in text categorization. Technical Report IR-408, Department of Computer Science, University of Massachusetts, 2004. (Cited on p. 81 und 82)

Douglas Biber. *Variation Across Speech and Writing*. Cambridge University Press, 1988. (Cited on p. 8)

John Blitzer, Mark Dredze, and Fernando C.N. Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 440–447, 2007. (Cited on p. 16, 78, 90, 91, 98, und 127)

Kenneth Bloom, Navendu Garg, and Shlomo Argamon. Extracting appraisal expressions. In *Proccedings of the 2007 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technology*, pages 308–315, 2007. (Cited on p. 7)

Janez Brank, Marko Grobelnik, Natasa Milic-Frayling, and Dunjia Mladenic. Interaction of feature selection methods and linear classification models. In *Internation Conference on Machine Learning (ICML) Workshop on Text Learning*, 2002. (Cited on p. 81)

Eric Breck, Yejin Choi, and Claire Cardie. Identifying expressions of opinion in context. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2683–2688, 2007. (Cited on p. 7)

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. Word-sense disambiguation using statistical methods. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 264–270, 1991. (Cited on p. 11)

Peter F. Brown, Peter V. Desouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479, 1992. (Cited on p. 128)

Erik Cambria, Catherine Havasi, and Amir Hussain. SenticNet 2: A semantic and affective resource for opinion mining and sentiment analysis. In *Proceedings of the 25th Florida Artificial Intelligence Research Society Conference (FLAIRS)*, pages 202–207, 2012a. (Cited on p. 11)

Erik Cambria, Daniel Olsher, and Kenneth Kwok. Sentic activation: A two-level affective common sense reasoning framework. In *Proceedings of the 26th National Conference on Artificial Intelligence (AAAI)*, 2012b. (Cited on p. 11)

Jorge Carrillo-de Albornoz and Laura Plaza. An emotion-based model of negation, intensifiers, and modality for polarity and intensity classification. *Journal of the American Society for Information Science and Technology (JASIST)*, 64(8):1618–1633, 2013. (Cited on p. 114)

Jorge Carrillo de Albornoz, Laura Plaza, and Pablo Gervás. A hybrid approach to emotional sentence polarity and intensity classification. In *Proceedings of the 14th Conference on Computational Natural Language Learning (CoNLL)*, pages 153–161, 2010. (Cited on p. 114)

John S. Caylor, Thomas G. Sticht, Lynn C. Fox, and J. Patrick Ford. Methodologies for determining reading requirements of military occupational specialties. Technical Report 73-5, HUMRO Western Division, 1973. (Cited on p. 36)

Wendy W. Chapman, Will Bridewell, Paul Hanbury, Gregory F. Cooper, and Bruce G. Buchanan. A simple algorithm for identifying negated findings and diseases in discharge summaries. *Journal of Biomedical Informatics*, 34 (5):301–310, 2001. (Cited on p. 116)

Yejin Choi and Claire Cardie. Learning with compositional semantics as structural inference for subsentential sentiment analysis. In *Proceedings of the 13th Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 793–801, 2008. (Cited on p. 113)

Yejin Choi, Claire Cardie, Ellen Riloff, and Siddharth Patwardhan. Identifying sources of opinions with conditional random fields and extraction patterns. In *Proceedings of the Conference on Human Language Technology (HLT) and the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 355–362, 2005. (Cited on p. 7)

Yejin Choi, Eric Breck, and Claire Cardie. Joint extraction of entities and relations for opinion recognition. In *Proceedings of the 11th Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 431–439, 2006. (Cited on p. 7)

Jurgen A.H.R. Claassen. The gold standard: not a golden standard. *BMJ*, 330(7500):1121, May 2005. (Cited on p. 15)

Kevyn Collins-Thompson and Jamie Callan. A language modeling approach to predicting reading difficulty. In *Proccedings of the 2004 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technology (NAACL-HLT)*, 2004. (Cited on p. 37)

Paul Cook and Suzanne Stevenson. Automatically identifying changes in the semantic orientation of words. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC)*, pages 28–34, 2012. (Cited on p. 2)

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, 3rd edition, 2009. (Cited on p. 51)

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995. (Cited on p. 43 und 44)

Isaac G. Councill, Ryan McDonald, and Leonid Velikovich. What's great and what's not: Learning to classify the scope of negation for improved sentiment analysis. In *Proceedings of the 2010 Workshop on Negation and Speculation in Natural Language Processing (NeSp-NLP)*, pages 51–59, 2010. (Cited on p. 114)

David Crystal. *A Dictionary of Linguistics and Phonetics*. Blackwell, 6th edition, 2008. (Cited on p. 8, 9, 15, und 28)

Hamish Cunningham. A definition and short history of language engineering. *Natural Language Engineering*, 5 (1):1–16, 1999. (Cited on p. 3)

Antonio R. Damasio. Emotions and feelings – a neurobiological perspective. In Antony S.R. Manstead, Nico H. Frijda, and Agneta Fischer, editors, *Feelings and Emotions: The Amsterdam Symposium*, pages 49–57. Cambridge University Press, 2004. (Cited on p. 4 und 5)

Amitava Das and Sivaji Bandyopadhyay. Subjectivity detection using genetic algorithm. In *Proceeding of the 1st Workshop on Opinion Mining and Sentiment Analysis (WOMSA)*, 2009. (Cited on p. 102)

Hal Daumé III. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 256–263, 2007. (Cited on p. 93 und 94)

Hal Daumé III, Abhishek Kumar, and Avishek Saha. Frustratingly easy semi-supervised domain adaptation. In *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing (DANLP)*, pages 53–59, 2010a. (Cited on p. 94)

Hal Daumé III, Abhishek Kumar, and Avishek Saha. Co-regularization based semi-supervised domain adaptation. In *Proceedings of Neural Information Processing Systems (NIPS)*, pages 256–263, 2010b. (Cited on p. 93 und 94)

Marie-Catherine de Marneffe, Bill MacCartney, Christopher D. Manning, et al. Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC)*, pages 449–454, 2006. (Cited on p. 23)

Rodolfo Delmonte, Sara Tonelli, Marco Aldo Piccolino Boniforti, and Antonella Bristot. Venses – a linguistically-based system for semantic evaluation. In *Proceedings of the 1st PASCAL Challenge Workshop on Recognizing Textual Entailment (RTE)*, pages 49–52, 2005. (Cited on p. 113)

Gerard Escudero, Lluis Màrquez, and German Rigau. An empirical study of the domain dependence of supervised word sense disambiguation systems. In *Proceedings of Joint Conference on Empirical Methods in Natural Language Processing (EMNLP) and Very Large Corpora (VLC)*, pages 172–180, 2000. (Cited on p. 2)

Andrea Esuli and Fabrizio Sebastiani. SentiWordNet: A publicly available lexical resource for opinion mining. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC)*, pages 417–422, 2006. (Cited on p. 4 und 114)

Irving E. Fang. The "easy listening formula". *Journal of Broadcasting & Electronic Media*, 11(1):63–68, 1966. (Cited on p. 35)

James N. Farr, James J. Jenkins, and Donald G. Paterson. Simplification of flesch reading ease formula. *Journal of Applied Psychology*, 35(5):333–337, 1951. (Cited on p. 36)

George Forman. An extensive empirical study of feature selection metrics for text classification. *The Journal of Machine Learning Research (JMLR)*, 3:1289–1305, 2003. (Cited on p. 55 und 85)

William A. Gale and Geoffrey Sampson. Good-turing frequency estimation without tears. *Journal of Quantitative Linguistics*, 2(3):217–237, 1995. (Cited on p. 37 und 92)

Dmitriy Genzel and Eugene Charniak. Variation of entropy and parse trees of sentences as a function of the sentence number. In *Proceedings of the 8th Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 65–72, 2003. (Cited on p. 38 und 39)

Stefan Gindl, Albert Weichselbraun, and Arno Scharl. Cross-domain contextualisation of sentiment lexicons. In *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI)*, pages 771–776, 2010. (Cited on p. 91)

Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. CS224N project report, Stanford University, 2009. (Cited on p. 21, 22, 42, und 64)

Kristina Gulordava and Marco Baroni. A distributional similarity approach to the detection of semantic change in the google books ngram corpus. In *Proceedings of the 2011 Geometrical Models for Natural Language Semantics Workshop (GEMS)*, 2011. (Cited on p. 2)

Robert Gunning. *The Technique of Clear Writing*. McGraw-Hill New York, 1952. (Cited on p. 35)

Michael A.K. Halliday. Language as system and language as instance: The corpus as a theoretical construct. In Jan Svartvik, editor, *Directions in Corpus Linguistics: Proceedings of Nobel Symposium 82*, pages 61–77, Stockholm, 4–8 August 1991. (Cited on p. 25)

Qi Han, Junfei Guo, and Hinrich Schütze. Codex: Combining an svm classifier and character n-gram language models for sentiment analysis on twitter text. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval)*, pages 520–524, 2013. (Cited on p. 101)

Frank E. Harrell. *Regression Modeling Strategies: with Applications to Linear Models, Logistic Regression, and Survival Analysis*. Springer, 2001. (Cited on p. 73)

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning*. Springer Series in Statistics. Springer, 2nd edition, 2009. (Cited on p. 37, 41, 52, 53, 54, 55, und 58)

Vasileios Hatzivassiloglou and Kathleen R. McKeown. Predicting the semantic orientation of adjectives. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 174–181, 1997. (Cited on p. 5)

Marc D. Hauser, Noam Chomsky, and William T. Fitch. The faculty of language: What is it, who has it, and how did it evolve? *Science*, 298:1569–1579, 2002. (Cited on p. 11)

Michael J. Heilmann, Kevyn Collins-Thompson, Jamie Callan, and Maxine Eskenazi. Combining lexical and grammatical features to improve readability measures for first and second language texts. In *Proccedings of the 2007 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technology*, pages 460–467, 2007. (Cited on p. 37)

Jesús Herrera, Anselmo Penas, and Felisa Verdejo. Textual entailment recognition based on dependency analysis and wordnet. In *Proceedings of the 1st PASCAL Challenge Workshop on Recognizing Textual Entailment (RTE)*, pages 21–24, 2005. (Cited on p. 113)

Tin Kam Ho and Mitra Basu. Complexity measures of supervised classification problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):289–300, 2002. (Cited on p. vii, 30, und 31)

Linh Hoang, Jung-Tae Lee, Young-In Song, and Hae-Chang Rim. A model for evaluating the quality of user-created documents. In *Proceedings of the 4th Asia Information Retrieval Conference on Information Retrieval Technology (AIRS)*, pages 496–501, 2008. (Cited on p. 102)

Alexander Hogenboom, Paul van Iterson, Bas Heerschop, Flavius Frasincar, and Uzay Kaymak. Determining negation scope and strength in sentiment analysis. In *Proceedings of the 2011 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 2589–2594, 2011. (Cited on p. 114)

Paul W. Holland and Roy E. Welsch. Robust regression using iteratively reweighted least-squares. *Communications in Statistics*, 6(9):813–827, 1977. (Cited on p. 53)

Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, 2002. (Cited on p. 48)

Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. A practical guide to support vector classification, 2003. (Cited on p. 54 und 64)

Peter J. Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964. (Cited on p. 53)

Eva Hudlicka. To feel or not to feel: The role of affect in human–computer interaction. *International Journal of Human-Computer Studies*, 59(1):1–32, 2003. (Cited on p. 1)

Niklas Jakob and Iryna Gurevych. Extracting opinion targets in a single- and cross-domain setting with conditional random fields. In *Proceedings of the 15th Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1035–1045, 2010. (Cited on p. 7 und 91)

Lifeng Jia, Clement Yu, and Weiyi Meng. The effect of negation on sentiment analysis and retrieval effectiveness. In *Proceedings of the 18th Conference on Information and Knowledge Management (CIKM)*, pages 1827–1830, 2009. (Cited on p. 114)

Jing Jiang and ChengXiang Zhai. Instance weighting for domain adaptation in NLP. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 264–271, 2007. (Cited on p. 91)

Thorsten Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning (ECML)*, pages 137–142, 1998. (Cited on p. 63)

Richard Johansson and Alessandro Moschitti. Relational features in fine-grained opinion analysis. *Computational Linguistics*, 39(3):473–509, 2013. (Cited on p. 7 und 114)

Hiroshi Kanayama and Tetsuya Nasukawa. Fully automatic lexicon expansion for domain-oriented sentiment analysis. In *Proceedings of the 11th Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 355–363, 2006. (Cited on p. 91)

Jason S. Kessler and Nicolas Nicolov. Targeting sentiment expressions through supervised ranking of linguistic configurations. In *Proceedings of the 3rd International Conference on Weblogs and Social Media (ICWSM)*, pages 90–97, 2009. (Cited on p. 7)

Adam Kilgarriff. Comparing corpora. *International Journal of Corpus Linguistics*, 6(1):97–133, 2001. (Cited on p. 29 und 30)

Adam Kilgarriff and Tony Rose. Measures for corpus similarity and homogeneity. In *Proceedings of the 3rd Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 46–52, 1998. (Cited on p. 25 und 29)

Soo-Min Kim and Eduard Hovy. Identifying opinion holders for question answering in opinion texts. In *Proceedings of AAAI-05 Workshop on Question Answering in Restricted Domains*, pages 1367–1373, 2005. (Cited on p. 7)

Soo-Min Kim and Eduard Hovy. Extracting opinions, opinion holders, and topics expressed in online news media text. In *Proceedings of the Workshop on Sentiment and Subjectivity in Text*, pages 1–8, 2006. (Cited on p. 7)

J. Peter Kincaid, Robert P. Fishburne Jr., Richard L. Rogers, and Brad S. Chissom. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. Technical Report RBR 8-75, Chief of Naval Technical Training, 1975. (Cited on p. 35)

George R. Klare. Assessing readability. *Reading Research Quarterly*, 10(1):62–102, 1974. (Cited on p. 34 und 35)

Dan Klein and Christopher D. Manning. Fast exact inference with a factored model for natural language parsing. *Advances in Neural Information Processing Systems*, 15:3–10, 2003. (Cited on p. 23)

Manfred Klenner, Stefanos Petrakis, and Angela Fahrni. Robust compositional polarity classification. In *Proceedings of the 7th International Conference on Recent Advances in Natural Language Processing (RANLP)*, pages 180–184, 2009. (Cited on p. 113)

Eugene F. Krause. *Taxicab Geometry: An Adventure in Non-Euclidean Geometry*. Dover Publications, 1987. (Cited on p. 28)

Ulrich H.-G. Kreßel. Pairwise classification and support vector machines. In Bernhard Schölkopf, Christopher J.C. Burges, and Alexander J. Smolja, editors, *Advances in Kernel Methods – Support Vector Learning*, pages 255–268. MIT Press, 1999. (Cited on p. 48)

Solomon Kullback and Richard A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22 (1):79–86, 1951. (Cited on p. 26)

Shibamouli Lahiri, Prasenjit Mitra, and Xiaofei Lu. Informality judgment at sentence level and experiments with formality score. In *Proceedings of the 12th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*, pages 446–457, 2011. (Cited on p. 1)

George Lakoff. Hedging: A study in media criteria and the logic of fuzzy concepts. *Journal of Philosophical Logic*, 2:458–508, 1973. (Cited on p. 128)

Emanuele Lapponi, Jonathon Read, and Lilja Øvrelid. Representing and resolving negation for sentiment analysis. In *Proceedings of the 2012 IEEE 12th International Conference on Data Mining Workshops (ICDMW 2012) Workshop on Sentiment Elicitation from Natural Text for Information Retrieval and Extraction (SENTIRE)*, pages 687–692, 2012. (Cited on p. 114)

David Lee. Genres, registers, text types, domains, and styles: Clarifying the concepts and navigating a path through the bnc jungle. *Language Learning & Technology*, 5(3):37–72, 2001a. (Cited on p. 7 und 8)

Lillian Lee. On the effectiveness of the skew divergence for statistical language analysis. In *In Proceedings of the 8th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 65–72, 2001b. (Cited on p. 27)

Shoushan Li and Chengqing Zong. Multi-domain sentiment classification. In *Proceedings of 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 257–260, 2008. (Cited on p. 91)

Chenghua Lin and Yulan He. Joint sentiment/topic model for sentiment analysis. In *Proceedings of the 18th Conference on Information and Knowledge Management (CIKM)*, pages 375–384, 2009. (Cited on p. 12 und 138)

Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1): 145–151, 1991. (Cited on p. 26)

Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. Text classification using string kernels. *The Journal of Machine Learning Research*, 2: 419–444, 2002. (Cited on p. 47)

Stephanie Lukin and Marilyn Walker. Really? well. apparently bootstrapping improves the performance of sarcasm and nastiness classifiers for online dialogue. In *Proceedings of the Workshop on Language Analysis in Social Media (LASM)*, pages 30–40, 2013. (Cited on p. 9)

Isa Maks and Piek Vossen. Sentiment analysis of reviews: Should we analyze writer intentions or reader perceptions. In *Proceedings of the 9th International Conference on Recent Advances in Natural Language Processing (RANLP)*, pages 415–419, 2013. (Cited on p. 138)

Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, 1999. (Cited on p. 22, 23, 49, 50, 56, 57, 61, 62, und 90)

G. Harry McLaughlin. SMOG grading: A new readability formula. *Journal of Reading*, 12(8):639–646, 1969. (Cited on p. 34, 36, und 38)

Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and Cheng-Xiang Zhai. Topic sentiment mixture: Modeling facets and opinions in weblogs. In *Proceedings of the 16th International Conference on World Wide Web (WWW)*, pages 171–180, 2007. (Cited on p. 138)

Helmut Meier. *Deutsche Sprachstatistik*, volume 1. Georg Olms, 1964. (Cited on p. 105, 106, und 111)

James Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London*, 209: 415–446, 1909. (Cited on p. 46)

Rada Mihalcea and Stephen Pulman. Characterizing humour: An exploration of features in humorous texts. In *Proceedings of the 8th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*, pages 337–347, 2007. (Cited on p. 10)

Rada Mihalcea and Carlo Strapparava. Making computers laugh: Investigations in automatic humor recognition. In *Proceedings of the Conference on Human Language Technology (HLT) and the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 531–538, 2005. (Cited on p. 10)

Rada Mihalcea, Carmen Banea, and Janyce Wiebe. Learning multilingual subjective language via cross-lingual projections. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 976–83, 2007. (Cited on p. 11)

Tom M. Mitchell. *Machine Learning*. McGraw-Hill, 1997. (Cited on p. 37, 49, 50, 51, 52, und 55)

Saif Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval)*, pages 321–327, 2013. (Cited on p. 113 und 116)

Karo Moilanen and Stephen Pulman. Sentiment composition. In *Proceedings of the 6th International Conference on Recent Advances in Natural Language Processing (RANLP)*, pages 378–382, 2007. (Cited on p. 113)

Roser Morante and Walter Daelemans. A metalearning approach to processing the scope of negation. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL)*, pages 21–29, 2009. (Cited on p. 114)

Katharina Morik, Peter Brockhausen, and Thorsten Joachims. Combining statistical learning with a knowledge-based approach – a case study in intensive care monitoring. In *Proceedings of the 16th International Conference on Machine Learning (ICML)*, pages 268–277, 1999. (Cited on p. 47)

Alessandro Moschitti and Roberto Basili. Complex linguistic features for text classification: A comprehensive study. In *Proceedings of the 26th European Conference on IR Research (ECIR)*, pages 181–196, 2004. (Cited on p. 81)

Tetsuji Nakagawa, Kentaro Inui, and Sadao Kurohashi. Dependency tree-based sentiment classification using crfs with hidden variables. In *Proccedings of the 2010 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technology*, pages 786–794, 2010. (Cited on p. 126)

Tetsuya Nasukawa and Jeonghee Yi. Sentiment analysis: Capturing favorability using natural language processing. In *Proceedings of the 2nd International Conference on Knowledge Capture (K-CAP)*, pages 70–77, 2003. (Cited on p. 4 und 5)

Alena Neviarouskaya, Helmut Prendinger, and Mitsuru Ishizuka. Compositionality principle in recognition of fine-grained emotions from text. In *Proceedings of the 3rd International Conference on Weblogs and Social Media (ICWSM)*, pages 278–281, 2009. (Cited on p. 113)

Andrew Y. Ng and Michael I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in Neural Information Processing Systems (NIPS)*, pages 841–848, 2001. (Cited on p. 41)

Hitoshi Nishikawa, Takaaki Hasegawa, Yoshihiro Matsuo, and Genichiro Kikui. Optimizing informativeness and readability for sentiment summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 325–330, 2010. (Cited on p. 102)

Eric W. Noreen. *Computer Intensive Methods for Testing Hypothesis – An Introduction.* John Wiley and Sons, Inc., 1989. (Cited on p. 96)

Daniela Oelke, David Spretke, Andreas Stoffel, and Daniel A. Keim. Visual readability analysis: How to make your writings easier to read. In *Proceedings of IEEE Symposium on Visual Analytics Science and Technology (VAST)*, pages 123–130, 2010. (Cited on p. 38 und 111)

Michael P. O'Mahony and Barry Smyth. The readability of helpful product reviews. In *Proceedings of the 23rd Florida Artificial Intelligence Research Society Conference (FLAIRS)*, 2010. (Cited on p. 102)

Gerhard Pahl and Wolfgang Beitz. *Konstruktionslehre*. Springer, 2nd edition, 1986. (Cited on p. 25 und 81)

Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC)*, 2010. (Cited on p. 106 und 113)

Sinno Jialin. Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Zheng Chen. Cross-domain sentiment classification via spectral feature alignment. In *Proceedings of the 19th International Conference on World Wide Web (WWW)*, pages 751–760, 2010. (Cited on p. 90, 91, 96, 98, und 127)

Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2004. (Cited on p. 5, 18, 21, 102, und 110)

Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 115–124, 2005. (Cited on p. 20)

Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2 (1-2):1–135, 2008. (Cited on p. 4)

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 79–86, 2002. (Cited on p. 63, 64, 113, 116, und 118)

Barbara Plank and Gertjan van Noord. Effective measures of domain similarity for parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1566–1576, 2011. (Cited on p. 26, 91, 92, und 100)

Livia Polanyi and Annie Zaenen. Contextual valence shifters. In James G. Shanahan, Yan Qu, and Janyce

Wiebe, editors, *Computing Attitude and Affect in Text: Theory and Application*, volume 20 of *The Information Retrieval Series. Computing Attitude and Affect in Text: Theory and Applications*, pages 1–9. Springer, Dordrecht, 2006. (Cited on p. 128)

Natalia Ponomareva and Mike Thelwall. Biographies or blenders: Which resource is best for cross-domain sentiment analysis? In *Proceedings of the 13th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*, pages 488–499, 2012a. (Cited on p. vii, 28, 30, 78, 90, 92, 96, und 127)

Natalia Ponomareva and Mike Thelwall. Do neighbours help? an exploration of graph-based algorithms for cross-domain sentiment classification. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing (EMNLP) and Computational Natural Language Learning (CoNLL)*, pages 655–665, 2012b. (Cited on p. 90, 91, 96, 98, und 127)

Richard D. Powers, William A. Sumner, and Bryant E. Kearl. A recalculation of four adult readability formulas. *Journal of Educational Psychology*, 49(2):99–105, 1958. (Cited on p. 35)

Geoffrey K. Pullum and Barbara C. Scholz. Recursion and the infinitude claim. *Recursion in Human Language*, 104: 113–138, 2010. (Cited on p. 11)

Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. Expanding domain sentiment lexicon through double propagation. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1199–1204, 2009. (Cited on p. 91)

Guang Qiu, Xiaofei He, Feng Zhang, Yuan Shi, Jiajun Bu, and Chun Chen. DASA: Dissatisfaction-oriented advertising based on sentiment analysis. *Expert Systems with Applications*, 37(9):6182–6191, 2010. (Cited on p. 1)

Uwe Quasthoff, Matthias Richter, and Chris Biemann. Corpus portal for search in monolingual corpora. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC)*, pages 1799–1802, 2006. (Cited on p. 31)

Uwe Quasthoff, Sabine Fiedler, and Erla Hallsteinsdóttir, editors. *Frequency Dictionary English*. Leipziger Universitätsverlag, 2012. (Cited on p. 39)

J. Ross Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986. (Cited on p. 50)

John R. Quinlan. *C4.5: Programs for Machine Learning*, volume 1. Morgan kaufmann, 1993. (Cited on p. 50)

Stephan Raaijmakers and Wessel Kraaij. A shallow approach to subjectivity classification. In *Proceedings of the 2nd International Conference on Weblogs and Social Media (ICWSM)*, pages 216–217, 2008. (Cited on p. 90, 102, und 110)

Robert Remus. Improving sentence-level subjectivity classification through readability measurement. In *Proceedings of the The 18th International Nordic Conference of Computational Linguistics (NODALIDA)*, pages 168–174. NEALT, 2011. (Cited on p. 34, 101, 102, 109, 111, und 137)

Robert Remus. Domain adaptation using domain similarity- and domain complexity-based instance selection for cross-domain sentiment analysis. In *Proceedings of the 2012 IEEE 12th International Conference on Data Mining Workshops (ICDMW 2012) Workshop on Sentiment Elicitation from Natural Text for Information Retrieval and Extraction (SENTIRE)*, pages 717–723, 2012. (Cited on p. vii, 29, 91, 100, und 137)

Robert Remus. ASVUniOfLeipzig: Sentiment analysis in twitter using data-driven machine learning techniques. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval)*, pages 450–454, 2013a. (Cited on p. 137)

Robert Remus. Modeling and representing negation in data-driven machine learning-based sentiment analysis. In *Proceedings of the 1st International Workshop on Emotion and Sentiment in Social and Expressive Media (ESSEM)*, pages 22–33, 2013b. (Cited on p. 114, 117, und 137)

Robert Remus and Mathias Bank. Textual characteristics of different-sized corpora. In *Proceedings of the 5th Workshop* *on Building and Using Comparable Corpora (BUCC)*, pages 156–160, 2012. (Cited on p. 31 und 137)

Robert Remus and Christian Hänig. Towards well-grounded phrase-level polarity analysis. In *Proceedings of the 12th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*, pages 380–392, 2011. (Cited on p. 113)

Robert Remus and Sven Rill. Data-driven vs. dictionary-based approaches for machine learning-based sentiment analysis. In *Proceedings of the International Conference of the German Society for Computational Linguistics and Language Technology (GSCL)*, number 8105 in LNCS, pages 176–183. Springer, 2013. (Cited on p. 65, 111, und 137)

Robert Remus and Dominique Ziegelmayer. Learning from domain complexity. In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC)*, 2014. (Cited on p. 61, 82, und 137)

Alfréd Rényi. On measures of entropy and information. In *In Proceedings of the 4th Berkeley Symposium on Mathematical Statistics and Probability*, pages 547–561, 1961. (Cited on p. 27)

Antonio Reyes and Paolo Rosso. Making objective decisions from subjective data: Detecting irony in customer reviews. *Decision Support Systems*, 53(4):754–760, 2012. (Cited on p. 9)

Antonio Reyes and Paolo Rosso. On the difficulty of automatically detecting irony: Beyond a simple case of negation. *Knowledge and Information Systems*, 35(2):1–20, 2013. (Cited on p. 9 und 128)

Ellen Riloff and Janyce Wiebe. Learning extraction patterns for subjective expressions. In *Proceedings of the 8th Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 105–112, 2003. (Cited on p. 102)

Ellen Riloff, Janyce Wiebe, and Theresa Wilson. Learning subjective nouns using extraction pattern bootstrapping. In *Proceedings of the 7th Conference on Natural language learning (CoNLL)*, pages 25–32, 2003. (Cited on p. 102)

Ellen Riloff, Siddharth Patwardhan, and Janyce Wiebe. Feature subsumption for opinion analysis. In *Proceedings of the 11th Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 440–448, 2006. (Cited on p. 17, 81, 102, und 110)

Monica Rogati and Yiming Yang. High-performing feature selection for text classification. In *Proceedings of the 11th Conference on Information and Knowledge Management (CIKM)*, pages 659–661, 2002. (Cited on p. 81)

Danny Roobaert, Grigoris Karakoulas, and Nitesh V. Chawla. Information gain, correlation and support vector machines. In Isabelle Guyon, Steve Gunn, Massoud Nikravesh, and Lotfi A. Zadeh, editors, *Feature Extraction, Foundations and Applications*, pages 463–470. Springer, 2006. (Cited on p. 55)

Hassan Saif, Yulan He, and Harith Alani. Alleviating data sparsity for twitter sentiment analysis. In *Proceedings of the 2nd Workshop on Making Sense of Microposts (#MSM)*, 2012. (Cited on p. 114)

Christian Scheible and Hinrich Schütze. Unsupervised sentiment analysis with a simple and fast Bayesian model using Part-of-Speech feature selection. In *Proceedings of the 1st Workshop on Practice and Theory of Opinion Mining and Sentiment Analysis (PATHOS)*, pages 269–273, 2012. (Cited on p. 12)

Bernhard Schölkopf, Chris Burges, and Vladimir Vapnik. Extracting support data for a given task. In *Proceedings of the 1st International Conference on Knowledge Discovery & Data Mining (KDD)*, pages 252–257, 1995. (Cited on p. 48)

Thomas Scholz and Stefan Conrad. Integrating viewpoints into newspaper opinion mining for a media response analysis. In *Proceedings of the 11th Conference on Natural Language Processing (KONVENS)*, pages 30–38, 2012. (Cited on p. 10)

Sarah E. Schwarm and Mari Ostendorf. Reading level assessment using support vector machines and statistical language models. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 523–530, 2005. (Cited on p. 37)

Sam Scott and Stan Matwin. Feature engineering for text classification. In *Proceedings of the 16th International Conference on Machine Learning (ICML)*, 1999. (Cited on p. 81)

Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys (CSUR)*, 34(1): 1–47, 2002. (Cited on p. 62)

Yohei Seki, Koji Eguchi, Noriko Kando, and Masaki Aono. Multi-document summarization with subjectivity analysis at DUC 2005. In *Proceedings of the Document Understanding Conference (DUC)*, 2005. (Cited on p. 1)

Satoshi Sekine. The domain dependence of parsing. In *Proceedings of the 5th Conference on Applied Natural Language Processing (ANLP)*, pages 96–102, 1997. (Cited on p. 2)

Luo Si and Jamie Callan. A statistical model for scientific readability. In *Proceedings of the 10th Conference on Information and Knowledge Management (CIKM)*, pages 574–576, 2001. (Cited on p. 37)

Edgar A. Smith. Devereux readability index. *The Journal of Educational Research*, 54(8):298–303, 1961. (Cited on p. 35)

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y Ng, and Christopher D. Manning. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 16th Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 151–161, 2011. (Cited on p. 126)

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing (EMNLP) and Computational Natural Language Learning (CoNLL)*, pages 1201–1211, 2012. (Cited on p. 113 und 126)

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 18th Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1631–1642, 2013. (Cited on p. 138)

Gerard Steen. Genres of discourse and the definition of literature. *Discourse Processes*, 28(2):109–120, 1999. (Cited on p. 8)

Carlo Strapparava and Rada Mihalcea. SemEval-2007 task 14: Affective text. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval)*, pages 70—-74, 2007. (Cited on p. 19 und 126)

John M. Swales. *Genre Analysis: English in Academic and Research Settings*. Cambridge University Press, 1990. (Cited on p. 8)

Songbo Tan, Gaowei Wu, Huifeng Tang, and Xueqi Cheng. A novel scheme for domain-transfer problem in the context of sentiment analysis. In *Proceedings of the 16th Conference on Information and Knowledge Management (CIKM)*, pages 979–982, 2007. (Cited on p. 91)

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 575–584, 2010. (Cited on p. 5 und 15)

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proccedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technology (NAACL-HLT)*, pages 173–180, 2003. (Cited on p. 22)

Oren Tsur, Dmitry Davidov, and Ari Rappoport. ICWSM – a great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews. In *Proceedings of the 4th International Conference on Weblogs and Social Media (ICWSM)*, pages 162–169, 2010. (Cited on p. 9)

Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 384–394, 2010. (Cited on p. 128)

Peter D. Turney. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 417–424, 2002. (Cited on p. 2)

Vincent Van Asch and Walter Daelemans. Using domain similarity for performance estimation. In *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing (DANLP)*, pages 31–36, 2010. (Cited on p. 26 und 78)

Cornelis J. Van Rijsbergen. *Information Retrieval*. Buttersworth, London, 2nd edition, 1979. (Cited on p. 56)

Vladimir Vapnik. *The Nature of Statistical Learning*. Springer New York, NY, 1995. (Cited on p. 43)

Vladimir Vapnik. *Statistical Learning Theory*. Wiley, 1998. (Cited on p. 43)

Veronika Vincze, György Szarvas, Richárd Farkas, György Móra, and János Csirik. The BioScope corpus: Biomedical texts annotated for uncertainty, negation and their scopes. *BMC Bioinformatics*, 9(Suppl 11):S9, 2008. (Cited on p. 116)

Dong Wang and Yang Liu. A cross-corpus study of unsupervised subjectivity identification based on calibrated EM. In *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis (WASSA)*, pages 161–167, 2011. (Cited on p. 12, 102, und 110)

Sida Wang and Christopher D. Manning. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 90–94, 2012. (Cited on p. 126)

Jason Weston and Chris Watkins. Multi-class support vector machines. Technical Report CSD-TR-98-04, Royal Holloway University of London, May 1998. (Cited on p. 48)

Janyce Wiebe. Learning subjective adjectives from corpora. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI)*, pages 735–741, 2000. (Cited on p. 102)

Janyce Wiebe and Ellen Riloff. Creating subjective and objective sentence classifiers from unannotated texts. In *Proceedings of the 6th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*, pages 486–497, 2005. (Cited on p. 102)

Janyce Wiebe, Theresa Wilson, and Matthew Bell. Identifying collocations for recognizing opinions. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL) Workshop on Collocation: Computational Extraction, Analysis, and Exploitation*, pages 24–31, 2001. (Cited on p. 102)

Janyce Wiebe, Eric Breck, Chris Buckley, Claire Cardie, Paul Davis, Bruce Fraser, Diane Litman, David Pierce, Ellen Riloff, Theresa Wilson, David May, and Mark Maybury. Recognizing and organizing opinions expressed in the world press. In *Proceedings of Association for the Advancement of Artificial Intelligence 2004 Spring Symposium on New Directions in Question Answering*, 2003. (Cited on p. 5)

Janyce Wiebe, Theresa Wilson, Rebecca Bruce, Matthew Bell, and Melanie Martin. Learning subjective language. *Computational Linguistics*, 30(3):277–308, 2004. (Cited on p. 2, 5, und 102)

Janyce Wiebe, Theresa Wilson, and Claire Cardie. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 1(2):165–210, 2005. (Cited on p. 17)

Michael Wiegand and Dietrich Klakow. Convolution kernels for opinion holder extraction. In *Proccedings of the 2010 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technology*, pages 795–803, 2010. (Cited on p. 7 und 47)

Michael Wiegand and Dietrich Klakow. Generalization methods for in-domain and cross-domain opinion holder extraction. In *Proceedings of the 13th Conference of the European Chapter of the Association for computational Linguistics (EACL)*, pages 325–335, 2012. (Cited on p. 91)

Michael Wiegand, Alexandra Balahur, Benjamin Roth, Dietrich Klakow, and Andrés Montoyo. A survey on the role of negation in sentiment analysis. In *Proceedings of the*

2010 Workshop on Negation and Speculation in Natural Language Processing (NeSp-NLP), pages 60–68, 2010. (Cited on p. 113 und 115)

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Human Language Technology (HLT) and the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 347–354, 2005. (Cited on p. 111)

Theresa Wilson, Zornitsa Kozareva, Preslav Nakov, Alan Ritter, Sara Rosenthal, and Veselin Stoyanov. SemEval-2013 task 2: Sentiment analysis in twitter. In *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval)*, 2013. (Cited on p. 20)

Xiao-Bing Xue and Zhi-Hua Zhou. Distributional features for text categorization. *IEEE Transactions on Knowledge and Data Engineering*, 21(3):428–442, 2009. (Cited on p. 90)

Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning (ICML)*, pages 412–420, 1997. (Cited on p. 55, 56, und 85)

David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 189–196, 1995. (Cited on p. 11)

Alexander Yeh. More accurate tests for the statistical significance of result differences. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING)*, pages 947–953, 2000. (Cited on p. 96)

Hong Yu and Vasileios Hatzivassiloglou. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of the 8th Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 129–136, 2003. (Cited on p. 1 und 102)

Xiaojin Zhu. Semi-supervised learning literature survey. Technical Report 1530, University of Wisconsin-Madison, 2006. (Cited on p. 42)

Li Zhuang, Feng Jing, and Xiao-Yan Zhu. Movie review mining and summarization. In *Proceedings of the 15th Conference on Information and Knowledge Management (CIKM)*, pages 43–50, 2006. (Cited on p. 7)

Part IV

APPENDIX

A DATA

Table 40.: T-MDSD keywords.

DOMAIN	KEYWORDS
apparel	abercrombie & fitch, american apparel, american eagle, andrew christian, armani, bealls, ben sherman, birkenstock, boxer briefs, burlington, calvin klein, camel active, carhartt, crocs, desigual, dickies, eastpak, ed hardy, forever 21, forever21, fred perry, gucci, h&m, hollister, hugo boss, joop, kenneth cole, lacoste, levis, louis vuitton, new balance, nike, old navy, paul frank, prada, quiksilver, ralph lauren, ray bans, reebok, river island, speedo, superdry, tk maxx, tommy hilfiger, topshop, true religion, united colors of benetton, victoria’s secret
electronics	belkin, deskjet, garmin, inspiron, ipod, linksys, mac mini, optiplex, os x, osx, panasonic, ps3, samsung, sandisk, sd card, sony, toshiba, walkman, wii, zune
health	acetaminophen, advil, alka-seltzer, allergies, anti-anxiety, aspirin, benadryl, blood pressure monitor, casein, cetaphil, cholesterol, cocoa butter, cotton swabs, hair dryer, humidifier, lubrication, melatonin, microdermabrasion, nutritions, oat flour, old spice, oral b, oral-b, pedometer, razor, silk epil, skin cleanser, sodium, thyroid, tunnel syndrome, tylenol, vitamins

Continued on the next page

DOMAIN	KEYWORDS
kitchen	blender, bodum, breville, brita, coffeemaker, cooker, cookware, corkscrew, crockery, cuisinart, cutlery, delonghi, diffuser, dinnerware, dish washer, dishwasher, electric kettle, fiestaware, foreman, french press, fryer, frying pan, humidifier, iron chef, juicer, kitchen aid, kitchenaid, le creuset, oven, percolator, saucepan, spin brush, stove, tea infuser, toaster, zilch

SELBSTSTÄNDIGKEITSERKLÄRUNG

Hiermit erkläre ich, die vorliegende Dissertation selbständig und ohne unzulässige fremde Hilfe angefertigt zu haben. Ich habe keine anderen als die angeführten Quellen und Hilfsmittel benutzt und sämtliche Textstellen, die wörtlich oder sinngemäß aus veröffentlichten oder unveröffentlichten Schriften entnommen wurden, und alle Angaben, die auf mündlichen Auskünften beruhen, als solche kenntlich gemacht. Ebenfalls sind alle von anderen Personen bereitgestellten Materialen oder erbrachten Dienstleistungen als solche gekennzeichnet.

Leipzig, den 24. April 2015

. . . .