NTNU – Trondheim
Norwegian University of
Science and Technology

Twitter Sentiment Analysis

Exploring the Effects of Linguistic Negation

Jørgen Faret
Johan Reitan

Master of Science in Computer Science
Submission date: June 2015
Supervisor: Bjørn Gamback, IDI
Co-supervisor: Lars Bungum, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Abstract

Twitter sentiment analysis, the process of automatically extracting sentiment conveyed by Twitter data, is a field that has seen a dramatic increase in research in recent times. This Master's Thesis presents a study of the effects of linguistic negation on Twitter sentiment analysis.

Current state-of-the-art solutions in Twitter sentiment analysis and negation scope detection have been explored. Furthermore, a corpus of English Twitter data (*tweets*) annotated for linguistic negation has been created, and an improved system for negation scope detection in Twitter sentiment analysis has been developed and evaluated.

Our research represents the first work that explores sophisticated negation scope detection methods on tweets. The system produces better results than what has been reported in other domains. It has been incorporated into a state-of-the-art Twitter sentiment analysis classifier and the effects have been compared to other solutions commonly used within this field.

The study shows that the inclusion of the developed negation scope detection method in a Twitter sentiment analysis system improves the performance on tweets containing negation. However, the sparse distribution of linguistic negation in tweets results in a marginal performance gain on general data.

Sammendrag

Twitter-sentimentanalyse, fagfeltet som omhandler å utvinne sentiment uttrykt i Twitterdata, har gjennomgått en dramatisk økning i mengden forskningsarbeid i de siste årene. Denne masteroppgaven presenterer et studie av effekten av lingvistisk negasjon på Twitter-sentimentanalyse.

Nåværende state-of-the-art-løsninger innen Twitter-sentimentanalyse og identifikasjon av negasjonsomfang har blitt utforsket. Videre har en samling av Twitterdata blitt annotert for lingvistisk negasjon, og et forbedret system for å identifisere negasjonsomfang i Twitter-sentimentanalyse har blitt utviklet og evaluert.

Vårt arbeid representerer det første som utforsker sofistikert identifisering av negasjonsomfang på Twitterdata, og produserer bedre resultater enn noen rapportert i andre domener. Dette systemet har blitt innlemmet i en state-of-the-art Twitter-sentimentklassifikator, og effektene har blitt sammenlignet med vanlig brukte løsninger innen feltet.

Studiet viser at innlemmelsen av det utviklede systemet for negasjonsomfangsidentifikasjon i et Twitter-sentimentanalysesystem forbedrer ytelsen på Twitterdata som inneholder negasjon. Den begrensede forekomsten av lingvistisk negasjon resulterer imidlertid i en beskjeden ytelsesgevinst på generell data.

Preface

This Master's Thesis has been conducted at the Department of Computer and Information Science at the Norwegian University of Science and Technology (NTNU), and concludes our Master of Science degrees in Computer Science. The thesis was supervised by Björn Gambäck and Lars Bungum.

Jørgen Faret and Johan Reitan
Trondheim, June 11, 2015

Acknowledgements

We would like to thank the organizers of the SemEval Twitter sentiment shared tasks, and in particular Sara Rosenthal (Columbia University, New York) and Preslav Nakov (Qatar Computing Research Institute), for providing sentiment annotated data sets. We would also like to thank the contributors to Carnegie Mellon University's ARK Tweet NLP project for making available several helpful Twitter natural language processing tools. Additionally, we thank Svetlana Kiritchenko, Xiaodan Zhu and Saif M. Mohammad of the National Research Council Canada for their prior polarity lexica and research, which was exceedingly valuable to our work.

We are particularly grateful to our supervisor Björn Gambäck and co-supervisor Lars Bungum for their patience, motivation, enthusiasm and meticulous feedback throughout the project.

Contents

List of Figures

List of Tables

Acronyms

CRF Conditional Random Fields. **11**, 12, 20, 34–37, 41, 43, 44, 53, 54, 64, 65, 68, 69, 80, 81

NB Naïve Bayes. **9**, 10, 11, 33

NLP Natural language processing. 2, 7, **12**, 18, 29, 34

NSD negation scope detection. 2, 4, 5, 17, 22, **34**, 35, 45, 56, 57, 63, 64, 66, 75, 76, 80–82

PCS Percentage of correctly classified scopes. **17**, 39, 63, 70, 81

POS Part-of-speech. 11, 12, **14**, 18, 29, 36, 37, 41, 53, 56, 58, 59, 69, 70, 75, 80, 82

RBF Radial basis function. **8**, 72

SA Sentiment analysis. **1**, 2, 7, 12, 25, 26, 35, 41, 63

SSP Shallow semantic parsing. 36, **38**

SVM Support Vector Machine. **7**, 8, 9, 33, 37, 56, 59, 60, 70, 72, 73

TF-IDF Term frequency-inverse document frequency. **13**, 56, 57

TSA Twitter sentiment analysis. **1**, 3–5, 20, 25, 27, 29–34, 36, 42–44, 55, 63, 64, 66, 74, 79, 81, 82

1. Introduction

With the recent growth of mobile information systems and the increased availability of smart phones, social media has become a large part of daily life in most societies. This development has entailed the creation of massive amounts of data: data which when analysed can be used to extract valuable information about a variety of subjects.

Sentiment analysis (SA), also known as *opinion mining* is the process of classifying the emotion conveyed by a text, for example as negative, positive or neutral. The data made available by social media has contributed to a burst of research activity within SA in recent times and a shift in the focus of the field towards this type of data. Information gained from applying SA to social media data has many potential usages, for instance, to help marketers evaluate the success of an ad campaign, to identify how different demographics have received a product release, to predict user behaviour, or to forecast election results [Tsakalidis et al., 2015].

1.1. Twitter Sentiment Analysis

A popular social medium is Twitter,[1] a micro-blogging site that allows users to write textual entries of up to 140 characters, commonly referred to as *tweets*. As of June 2015, Twitter has over 302 million monthly active users according to their homepage, whereof approximately 88 % have their tweets freely readable. Additionally, over 84 % of the users also have their location specified in their profiles [Beevolve, 2012], enabling the possibility of performing drill-down on geographic locations. Data created by Twitter is made available through Twitter's API, and represents a real-time information stream of opinionated data. Tweets can be filtered both by location and the time they were published. This has paved the way for a new sub-field of SA: Twitter sentiment analysis (TSA).

[1] `https://www.twitter.com`

Performing natural language processing on textual data from Twitter presents new challenges because of the informal nature of this data. Tweets often contain misspellings, and the constrictive limit of 140 characters encourages slang and abbreviations. Unconventional linguistic means are also used, such as capitalization or elongation of words to show emphasis. Additionally, tweets contain special features like *emoticons* and *hashtags* that may have an analytical value. Hashtags are labels used for search and categorisation, and are included in the text prepended by a "`#`". Emoticons are expressions of emotion, and can either be written as a string of characters e.g., "`:-)`", or as a unicode symbol. Finally, if a tweet is a reply or is directed to another Twitter user, *mentions* can be used by prepending a username with "`@`".

1.2. Motivation and Research Focus Area

In this project we explore the effect of applying sophisticated *negation scope detection* to Twitter sentiment analysis. To our knowledge, no previous work has been done in this regard.

The linguistic phenomenon of negation, described in Section 2.1.2, has been shown to play a significant role in SA. Councill et al. [2010] tested a sentiment classifier and found that including their negation classifier provided a 29.5 % improvement in F_1 score when classifying positive sentiment, and an 11.4 % improvement when classifying negative sentiment. Figure 1.1 graphs the effects on precision and recall of including negation handling when performing positive sentiment prediction, as recorded by Councill et al.

Kiritchenko et al. [2014] included a sophisticated solution for *handling* negated terms in their SemEval-2014 entry by creating tweet-specific sentiment lexica containing individual scores for terms in affirmative and negated contexts, but the state-of-the-art systems in TSA still employ a very simple solution for *identifying* which terms are negated, by marking as negated all words from a negation cue term to the next punctuation symbol.

In Section 6.2 we present a baseline experiment we have conducted: implementing a naïve negation scope detection solution, commonly used in TSA, in order to compare how it performs on the BioScope Corpus

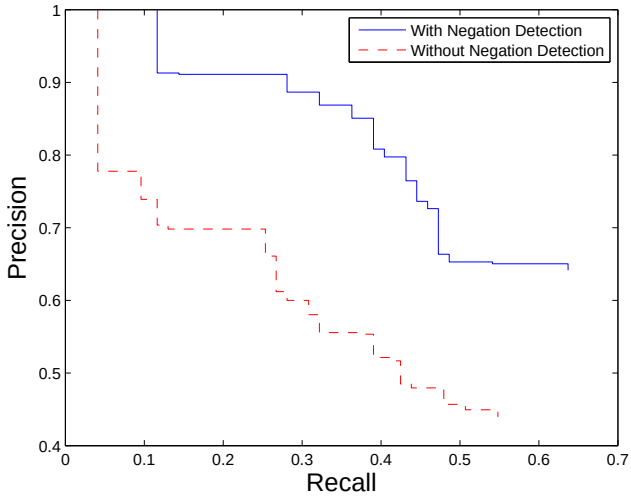


Figure 1.1.: The impact of negation handling on positive sentiment prediction [Councill et al., 2010]

(see Section 2.3.2) to existing, more sophisticated solutions, and show the potential for improvement in TSA. This naïve solution is then improved upon, and several experiments carried out with a more sophisticated approach to detecting the negation scope, both in isolation and embedded as part of a complete TSA system. The experiments show that naïve classification is slightly outperformed by existing alternative solutions and by our own, improved negation detector.

1.3. Project Goals

The goals for this project were four-fold, as follows:

G1: Research the State-of-the-Art in Twitter Sentiment Analysis

TSA has been the topic of several shared tasks hosted by the Association for Computational Linguistics.[2] We will base our state-of-the-art research mainly on the results from these tasks, and naturally focus more on submissions that achieved the best results.

The purpose of this goal is to form the knowledge basis required for developing a state-of-the-art TSA system.

G2: Create a Twitter Corpus Annotated for Negation

To allow for the use of supervised machine learning techniques when developing a negation classification system, a corpus annotated for the presence of negation is required. Several negation-annotated corpora have been created and are freely available, but none exist for the Twitter domain. As tweets contain unique characteristics, a classifier trained on out-of-domain data may struggle to classify Twitter data. Therefore, we aim to create a negation-annotated Twitter corpus.

G3: Develop a Negation Classifier

We aim to create a sophisticated negation classifier able to accurately classify the presence of negation in tweets. This classifier will then be implemented in the pipeline of a TSA system. This project goal also encompasses researching existing negation scope detection solutions and evaluating their transferability to TSA.

G4: Develop a Twitter Sentiment Classifier

The final goal is to create a state-of-the-art TSA system, incorporating the negation classifier developed in **G3**. The performance of using soph-

[2] `http://www.aclweb.org/`

isticated negation scope detection will then be explored and compared to the performance of the naïve approach commonly used in TSA.

1.4. Contributions

C1 A web-based linguistic negation annotation system.

C2 A corpus of Twitter data annotated for the presence of negation.

C3 The implementation of a Twitter negation scope detection system.

C4 The implementation of a Twitter sentiment classification system.

C5 A study into the effects of sophisticated negation scope detection in TSA compared to naïve solutions.

1.5. Thesis Structure

Relevant background theory for the project in addition to the tools and external data sets used are described in Chapter 2.

Chapter 3 presents the current state-of-the-art in TSA. Additionally, it contains a description of recent developments within the field of identifying linguistic negation, and a detailed look at some solutions, as well as an evaluation of their suitability for application in TSA.

In Chapter 4, we describe how a Twitter corpus annotated for the presence of linguistic negation was created: the NTNU Twitter Negation Corpus. The resulting corpus and its characteristics are also presented.

Chapter 5 contains the implementational details of two classification systems that have been created: a Twitter negation classifier and a TSA system. Experiments conducted on these systems and their results are presented in Chapter 6.

Finally, Chapter 7 contains an evaluation of the degree to which the project goals have been achieved, as well as the conclusions drawn and suggested future work.

2. Tools and Methods

This chapter contains a presentation of the background theory and technological tools relevant to this project, as well as the external data used in the conducted experiments.

2.1. Background Theory

Sentiment analysis (SA) comprises many concepts common to the whole field of natural language processing in addition to many concepts from machine learning: the most relevant of these are described in this section.

2.1.1. Machine Learning

Machine learning has become a cornerstone in the field of SA. Most well-performing systems incorporate some form of supervised machine learning; this is discussed further in Section 3.4. Here, we give a description of several machine learning algorithms relevant to the current state-of-the-art in sentiment analysis.

Support Vector Machines

The Support Vector Machine (SVM) classification algorithm was formally described by Cortes and Vapnik [1995]. The algorithm considers data points based on their spatial location, and attempts to split the feature space into optimal class segments. This division of the feature space is referred to as *training* the machine. A trained SVM can then be used for classification of new examples by assigning them a class based on which segment of the feature space they are located in.

In its basic form, it is an algorithm for linear classification of binary problems. When dealing with two linearly separable classes, the feature space is divided into class segments by creating a hyperplane with the

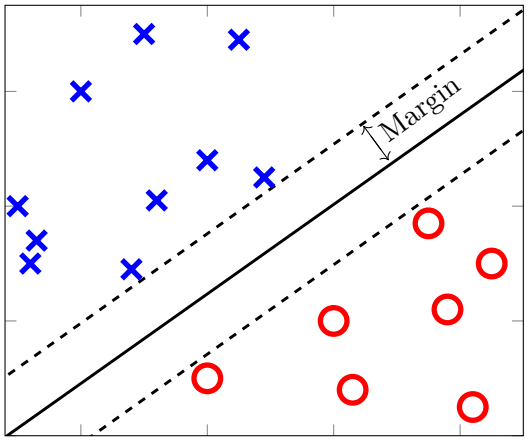


Figure 2.1.: An SVM linearly separating two classes.

largest possible margin between the two classes. This *margin maximization* is the essential concept of SVMs. The closest data points of both classes, parallel to the vector defining the hyperplane, constitute the *support vectors*. These give the algorithm its name. Figure 2.1 shows the hyperplane and support vectors in a two dimensional linear classification problem.

The algorithm can also be applied to problems where the examples are not linearly separable by allowing misclassified points. If no hyperplane exists to split all examples, the *soft margin* method introduces a slack variable that gives a penalty for each misclassified example [Cortes and Vapnik, 1995]. This penalty increases with the distance from the example's support vector. The slack variable governs the trade-off between classification errors and margin size.

To allow for non-linear classification, one can map the data into a higher dimensionality space by using the *kernel trick*. This is done by applying a *kernel function* to the data. The process is well explained in Fletcher [2009]. Popular kernel functions include radial basis function (RBF), Sigmoidal Kernel, and Polynomial Kernel. Figure 2.2 shows a data remapping done with an RBF kernel.

Using the RBF kernel function, there are two parameters that require adjustment for good performance: the regularization parameter C and

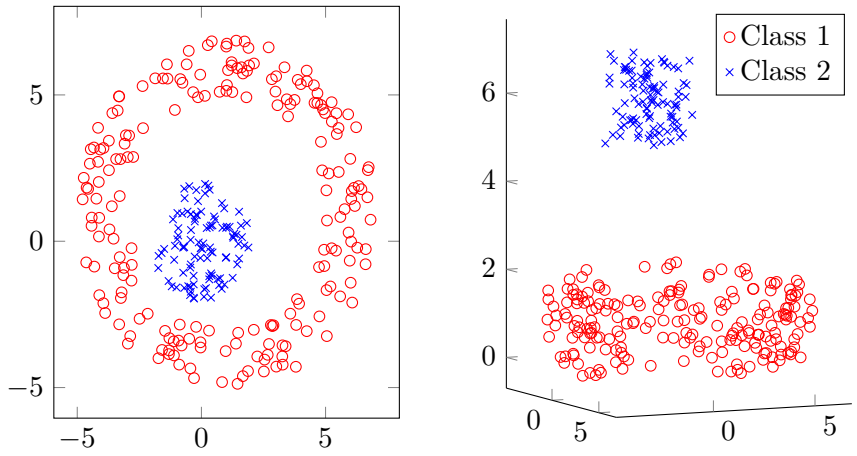


Figure 2.2.: Dichotomous data remapped using an RBF kernel function

the influence radius parameter γ. C is a penalty for misclassification, and controls the complexity of the decision surface. A low C makes a complex hyperplane that attempts to correctly classify most of the training samples (increasing the chance of overfitting), while increasing the C regularizes the hyperplane, resulting in a more generalized classifier. The γ controls the radius of influence for each training example, in an inverse manner. This means that a low γ makes each training example have an influence over a larger area, and a higher γ will result in the training examples only influencing themselves. These two parameters are highly dependent on each other.

Additionally, the SVM algorithm can be applied to classification problems featuring more than two classes. Commonly used methods include *one-against-one* and *one-against-the-rest*. Hsu and Lin [2002] provide a thorough comparison of these methods.

Naïve Bayes Classifier

Naïve Bayes (NB) classifiers, also known as Naïve Bayes Learners, are a relatively simple group of probabilistic classifiers based on Bayes' Theorem. For classification in certain domains, their performance has been shown to be comparable to much more complex machine learning algorithms, like

neural networks or decision-tree learners [Mitchell, 1997].

Given a document d, the task of assigning a class c to the document is done by looking at the probabilities of each class given the document, and finding the *maximum a posteriori* (MAP) probability estimate:

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(c|d)$$

The formula for each class can be rewritten using Bayes' Theorem:

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} \frac{P(d|c) \times P(c)}{P(d)}$$

Because the probability of the document is the same across all classes, the class that maximises the numerator is the same as the class that maximises the whole expression. The denominator can then be dropped:

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(d|c) \times P(c)$$

The document is represented as a vector of feature attributes:

$$d = a_1, a_2, a_3, ..., a_n$$

This gives the approach used by the NB classifier:

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c) \prod_{a} P(a|c)$$

The NB classifier assumes that all attribute values are conditionally independent. Despite this assumption, the classifier performs well on text classification tasks in practice.

Logistic Regression

Logistic regression (multinomial logistic regression for more than two possible output values) is another probabilistic classification method. Based on the Principle of Maximum Entropy, it seeks the model that best represents the available data, which is the model with the maximum information entropy. Because of this, logistic regression is often called maximum entropy classification.

Compared to the NB classifier, logistic regression does not assume conditional independence among the features. This makes the classifier more suited to text classification problems, since the features consisting of words are not conditionally independent.

The logistic regression classifier has been successfully applied to a wide range of text classification problems, including language detection, topic classification, and sentiment analysis [Vryniotis, 2013].

Conditional Random Fields

Lafferty et al. [2001] define Conditional Random Fields (CRF) as follows:

Definition. *Let $G = (V, E)$ be a graph such that $\mathbf{Y} = (\mathbf{Y}_v)_{v \in V}$, so that $\mathbf{Y}$ is indexed by the vertices of G. Then $(\mathbf{X}, \mathbf{Y})$ is a **conditional random field** in case, when conditioned on $\mathbf{X}$, the random variables $\mathbf{Y}_v$ obey the Markov property with respect to the graph: $p(\mathbf{Y}_v|\mathbf{X}, \mathbf{Y}_w, w \neq v) = p(\mathbf{Y}_v|\mathbf{X}, \mathbf{Y}_w, w \sim v)$, where $w \sim v$ means that w and v are neighbors in G.*

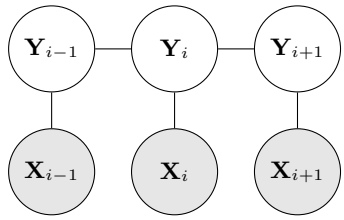


Figure 2.3.: Structure of a linear-chain CRF

This means that a CRF is an undirected graphical model consisting of the disjoint sets $\mathbf{X}$ and $\mathbf{Y}$, where $\mathbf{X}$ is the input variables and $\mathbf{Y}$ is the output variables. The conditional distribution $p(\mathbf{Y}|\mathbf{X})$ is then modelled:

$$p_\theta(\mathbf{y}|\mathbf{x}) \propto \exp\left(\sum_{e \in E,k} \lambda_k f_k(e, \mathbf{y}|_e, \mathbf{x}) + \sum_{v \in V,k} \mu_k g_k(v, \mathbf{y}|_v, \mathbf{x})\right),$$

where $\mathbf{x}$ is a data sequence, $\mathbf{y}$ a label sequence, and f_k and g_k are features (such as part-of-speech tags). The parameter $\theta = (\lambda_1, \lambda_2, \ldots; \mu_1, \mu_2, \ldots)$

is trained using gradient descent algorithms, or Quasi-Newton methods such as the L-BFGS algorithm [Nocedal, 1980].

Although the model can have any graphical structure, natural language processing tasks such as part-of-speech tagging are most concerned with sequences $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n)$ and $\mathbf{Y} = (\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_n)$, where $\mathbf{X}$ may be sequences of tokens and $\mathbf{Y}$ may be the corresponding sequences of part-of-speech tags. This sequence variant is often called a linear-chain CRF [Sutton and McCallum, 2011], and the structure is illustrated in Figure 2.3.

2.1.2. Natural Language Processing

Natural language processing (NLP) is a field in the Human-Machine Interaction area concerned with the use of human natural languages for communication with computers. Among the many topics of NLP, the following are particularly relevant in this project.

Linguistic Negation

Linguistic negation is a grammatical concept that encompasses devices used to reverse the truth value of propositions in language. Givón [1993] defines two forms of grammatical negation: *morphological* negation, where individual words are negated with an affix, and *syntactic* negation, where a set of words is negated by a word or phrase. Negators in syntactical negation, known as *negation cues* or *negation signals*, function as operators, with an associated affected scope of words [Morante and Sporleder, 2012]. Syntactic negation is what is most relevant within NLP, as well as textual data mining in general, and is what we mean throughout this report when referring to negation.

Tottie [1991] provides an extensive study of negation in written English language, and splits syntactic negation — or *clausal* negation as she denotes it — into two main categories: **rejections** of suggestions and **denials** of assertions.

Polanyi and Zaenen [2006] describe valence as "positive or negative attitude communicated by a lexical item". When looking at a segment of text, the segment's valence can be equated to its sentimental orientation. In the context of SA, negators often function as *valence shifters*, because

flipping the truth value of a proposition often also reverses, or significantly shifts, the valence it conveys. Valence shifters are terms that change the sentimental orientation of another set of terms, by changing the polarity and/or the evaluative intensity.

The most common linguistic negation cue in English is *not*, along with contractions created with it, such as *couldn't* or *isn't* [Tottie, 1991].

Bag-of-Words Model

A common way to represent text documents in a simplified manner is by using a bag-of-words model. The technique lists term occurrence and optionally the frequency of term occurrence, disregarding grammar and term order. Machine learning classifiers can use the resulting model directly as feature vectors.

Term Frequency-Inverse Document Frequency

Term frequency-inverse document frequency (TF-IDF) is a common term weighting scheme for the bag-of-words model, which lets us identify words in a collection of documents that can guide in deciding a document's topic. A term will have a high TF-IDF score if it rarely appears in the whole corpus, but appears often in the document at hand. As a result, very common words such as "the", "a", and "is" in English will be weighted lower and have little impact, instead of shadowing rarer and more interesting terms [Manning et al., 2008].

TF-IDF is calculated as:

$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)$$

where $\text{tf}(t, d)$ is the *term frequency*, and $\text{idf}(t, D)$ is the *inverse document frequency*. There are several variants of both tf and idf, but in their simplest forms, tf is the number of times a term t occurs in a document d, and the idf is:

$$\text{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}, \qquad |\{d \in D : t \in d\}| \neq 0$$

Here D is the entire corpus of documents, and N is the total number of documents in the corpus. As the number of documents where a term

appears increases, the ratio inside the logarithm will decrease towards 1, making the idf approach 0.

Part-of-Speech Tagging

Part-of-speech (POS) tagging is the process of categorizing the tokens of a sentence into the different parts of speech (such as nouns, verbs, adjectives and adverbs) based on their definitions as well as the contexts. This way, POS tagging attempts to solve the problem of word ambiguity. There are many POS taggers for regular languages trained on treebanks — particularly for the newswire domain — such as the Penn Treebank [Marcus et al., 1993]. However, the conversational language of Twitter causes an out-of-domain problem for these traditional POS taggers, degrading their performance. Gimpel et al. [2011] present a POS tagger tailored to Twitter.

Syntactic Parsing

Parsing in the sense of human languages is the analysis of a string of text using a set of grammar rules, ordering terms according to their syntactic relation to each other. The result is typically illustrated as parse trees, as seen in Figure 2.4. Only using a grammar, parsing can potentially produce an unlimited number of parse trees because of the substantial ambiguity of human languages. Statistical parsing and machine learning methods can be used in order to solve this ambiguity [Collins, 2003; Ratnaparkhi, 1999]. Figure 2.4 is an example of two possible parses for a problematic sentence, where each parse is correct, but has a different meaning.

Dependency Parsing

The task of dependency parsing consists of identifying the main verb of a sentence, and all other syntactic units being either directly or indirectly dependent on the verb. The dependency relations are based on syntactic rules, and do not rely on the meaning [Hudson, 2010]. Figure 2.5 is an example of a dependency parsed sentence where the root of the tree is the main verb, and the remaining words depend on the root either directly or indirectly.

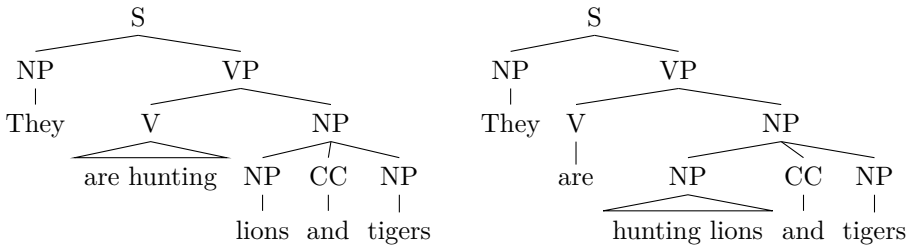


Figure 2.4.: Two syntactic parse trees for the same sentence, demonstrating human language ambiguity

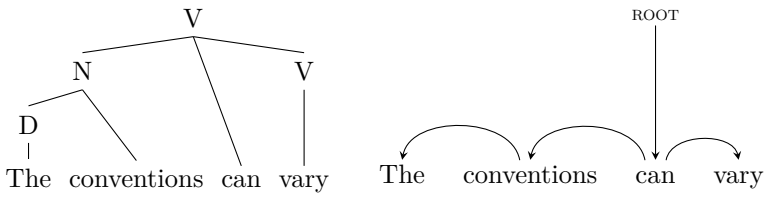


Figure 2.5.: Two representations of a dependency parsed sentence

2.1.3. Classification Scoring Metrics

Several different metrics exist for evaluating the performance of a classifier. This section presents the scoring metrics used in this project.

Precision

In a binary classification task, precision is a measure of the rate of positively predicted samples that are correct. Precision is defined as follows:

$$precision = \frac{t_p}{t_p + f_p}$$

t_p denotes the number of true positives and f_p denotes the number of false positives.

Recall

Recall is also a binary classification scoring metric, and measures the ratio of total positive samples that were correctly identified by a classifier. It is defined as follows:

$$recall = \frac{t_p}{t_p + f_n}$$

t_p denotes the number of true positives and f_n denotes the number of false negatives.

F-Measure

F-measure, is a weighted combination of precision and recall [Rijsbergen, 1979]. In its general form, it follows the formula:

$$\mathrm{F}_\beta = \frac{(1 + \beta^2) \times recall \times precision}{(\beta^2 \times precision) + recall}$$

The value of β defines the relationship between precision and recall: The F_β score weights recall more than precision by a factor of β. The most commonly used β value is 1. This weights precision and recall evenly, and is known as the *harmonic mean*. The F-measure with this value is known as the F_1 score, and is defined as:

$$\mathrm{F}_1 = \frac{2 \times recall \times precision}{precision + recall}$$

Averaging Single-Class Scores

When evaluating the performance of a non-binary classifier — a classifier with more than two possible class labels — it is common to report the performance as an average of one or several of the aforementioned metrics, such as F_1 score, across a subset of, or all possible labels. This can be done by taking the *macro-average*, or the *micro-average* of the metric, across the set of labels.

Micro-averaging is calculated by aggregating all of the relevant measures out of true positives, false positives, true negatives and false negatives and calculating the metric as if one was calculating the metric for a single label.

Macro-averaging completely disregards class imbalance. It is calculated by taking the average of the classification metric outputs for each label, equally weighing each label, regardless of its number of samples.

More formally: consider a binary evaluation metric $B(t_p, f_p, t_n, f_n)$, that is calculated based on the number of true positives (t_p), true negatives (t_n), false positives (f_p) and false negatives (f_n). Micro and macro-average scores for a given label γ with t_{p_γ}, f_{p_γ}, t_{n_γ}, f_{n_γ} denoting the number of true positives, false positives, true negatives, and false negatives for the metric B, follow the formulae [Tsoumakas et al., 2010]:

$$B_{micro} = B\left(\sum_{\gamma=1}^{q} t_{p_\gamma}, \sum_{\gamma=1}^{q} f_{p_\gamma}, \sum_{\gamma=1}^{q} t_{n_\gamma}, \sum_{\gamma=1}^{q} f_{n_\gamma}\right)$$

$$B_{macro} = \frac{1}{q}\sum_{\gamma=1}^{q} B\left(t_{p_\gamma}, f_{p_\gamma}, t_{n_\gamma}, f_{n_\gamma}\right)$$

Percentage of Correctly Classified Scopes

For several classification tasks where the output is a sequence, classification metrics that only consider individual units regardless of their order are insufficient. Percentage of correctly classified scopes (PCS) is a metric for measuring the performance of a scope classifier, for example a negation scope detection system.

A scope is considered correctly classified if, for a given negation cue, every token in its associated scope has been correctly marked as in a negation scope. The evaluation metric *accuracy* is defined as the number of correctly classified samples divided by the total number of samples. The PCS metric can thus be considered an accuracy measure.

2.2. Tools

Many external tools and libraries were necessary for the realization of this project. The following lists the most important of these.

2.2.1. Tweet NLP

Tweet NLP[1] is a collection of tools for common NLP tasks tailored to Twitter's conversational language. The tools included are a tokenizer, a part-of-speech (POS) tagger [Owoputi et al., 2013], hierarchical word clusters, and a dependency parser for tweets [Kong et al., 2014].

The POS tagger is an important tool used in many NLP operations, including the dependency parser, and for building feature vectors for a classifier (see Section 3.4.4). The tagger included in Tweet NLP is reported to have an accuracy above 93 % [Owoputi et al., 2013].

The dependency parser achieved 80 % unlabeled attachment accuracy [Kong et al., 2014].

2.2.2. Scikit-learn

`Scikit-learn` [Pedregosa et al., 2011] is a framework for the Python programming language that offers machine learning models as well as tools for performing preprocessing and data analysis. The `Scikit-learn` project is focused on providing state-of-the-art implementations for a wide range of machine learning methods, with particular attention to performance, consistent API and good documentation. The documentation is simplified in order to present to inexperienced readers the key points of a topic, while pointing experts to more in-depth information. The following highlights some key aspects of the framework.

Transformer

`Transformer` objects are an essential part of performing machine learning with `Scikit-learn`, and are objects that may "clean, reduce, expand, or generate feature representations". They are among other things used to create a matrix representation of the feature set for a set of samples.

Pipeline

`Scikit-learn` provides a framework for pipelining machine learning tools,[2] making it easy to chain tasks such as preprocessing and feature extrac-

[1] `http://www.ark.cs.cmu.edu/TweetNLP/`

[2] `http://scikit-learn.org/stable/modules/pipeline.html`

tion together with a machine learning algorithm in a tidy manner. The `Pipeline` framework is also able to perform grid search for parameters across all the *estimators* — the main API of `Scikit-learn`. An estimator is any object that learns from data, e.g., a classifier, or a `Transformer` object that extracts/filters useful features from raw data in the `Pipeline`. A pipeline can be trained as a whole, resulting in an object that performs all the necessary preprocessing, feature extraction and classification when given raw data.

Feature Union

Also included in the `Scikit-learn` `Pipeline` framework is the `FeatureUnion`, which is useful for feature extraction. Similarly to the `Pipeline`, it is used for chaining `Transformer` objects, but instead of passing the resulting data on to the next `Transformer`, all the `Transformers` are passed the same input. Each `Transformer` then produces its respective feature vectors — in parallel if desired. All the resulting vectors are concatenated into a final feature matrix.

Grid Search

One of the benefits of using the `Pipeline` framework is that it allows performing a parameter grid search across all the estimators in the `Pipeline`. The parameters can be provided for each estimator as ranges or sets of values to be evaluated. This grid search across preprocessing, feature extraction, and classifier parameters can be useful, for example for choosing a combination of vectorisation n-grams that works well for some classifier parameters.

2.2.3. Pandas

`Pandas` [McKinney, 2012] is an open-source library providing high-performance data structures and data analysis tools for the Python programming language. It also includes tools for efficiently reading and writing data between in-memory data structures and different textual file formats, such as comma-separated value files.

2.2.4. CRFSuite

`CRFSuite` [Okazaki, 2007] is an implementation of the Conditional Random Fields (CRF) sequence classifier, described in Section 2.1.1. It is written in the `C++` programming language. Each CRF classifier object takes as input the parameters $C1$ and $C2$ for setting the coefficients for L1 and L2 level normalization, respectively. The implementation includes a SWIG[3] API interface for other high-level programming languages, such as Python.

2.2.5. Mechanical Turk

Amazon Mechanical Turk[4] is a marketplace for work that requires human intelligence. It allows for performing a large amount of *Human Intelligence Tasks* by providing a large workforce consisting of real people. Within the context of this project, it is a useful tool for manually annotating a large number of tweets, for example as positive or negative.

2.3. External Data Sets

Three external data sets were used in the experiments conducted in this project: a sentiment-annotated data set collection, the SemEval-2014 Twitter Sentiment Data Sets, and two negation-annotated corpora, the BioScope and SFU Review corpora.

2.3.1. SemEval-2014 Twitter Sentiment Data Sets

Several data sets were created for subtask B of the Twitter sentiment analysis (TSA) task of SemEval-2014 (described in Section 3.3), created using Mechanical Turk. The data sets include a Twitter training set, a Twitter development set, two Twitter testing sets in addition to two out-of-domain testing sets: one consisting of SMS text messages and one comprising entries from LiveJournal,[5] a social networking service where users can keep a blog, journal or diary. Each sample is labeled either

[3] `http://www.swig.org/`
[4] `https://www.mturk.com/`
[5] `http://www.livejournal.com/`

positive, negative or objective/neutral. The size of each data set, along with its label distribution, is shown in Table 2.1.

Corpus	Positive	Negative	Objective/ Neutral	Total
Twitter2013-train	3,662	1,466	4,600	9,728
Twitter2013-dev	575	340	739	1,654
Twitter2013-test	1,575	601	1,640	3,816
SMS2013-test	492	394	1,207	2,093
Twitter2014-test	982	202	669	1,853
Twitter2014-sarcasm	33	40	13	86
LiveJournal2014-test	427	304	411	1,142

Table 2.1.: Data sets provided for SemEval-2014 Subtask B

Due to Twitter's privacy policy, the Twitter data sets cannot be distributed directly, but are downloaded with a script that combines tweets with their sentiment label using tweet IDs. Because of this, tweets that have been deleted since the data sets' creation are unavailable for download, and the available data sets are currently smaller than they were originally. Table 2.2 shows the number of available tweets in the data sets as of when they were downloaded by us, November 2014.

Data set	Number of Tweets
Twitter2013-train	8,026
Twitter2013-test	3,109
Twitter2014-test	1,533
Twitter2014-sarcasm	86

Table 2.2.: Number of available tweets in SemEval-2014 Twitter Sentiment Data Sets as of November 2014

2.3.2. BioScope Corpus

The BioScope Corpus [Vincze et al., 2008] is a collection of bio-medical textual data annotated for speculation and linguistic negation. The data

set is intended as a research resource to aid in the field of biomedical textual data mining, and was created because the detection of speculative and negative assertions is an essential part of that field. It was the first extensive, freely available negation-annotated corpus and allowed for the use of supervised learning in negation scope detection. It consists of three sub-corpora:

- Medical free texts
- Biological full papers
- Biological scientific abstracts

The respective number of sentences in the free text, full paper and abstracts sub-corpora are 6,383, 2,670 and 11,871.

The characteristics of the medical free text sub-corpus differs significantly from the other two sub-corpora. The free texts are radiology reports containing mainly short and concise sentences, with an average sentence length of 7.73 tokens, while the average sentence lengths in the full paper and abstracts sup-corpora are 26.24 and 26.43 tokens, respectively. The rate of negation, though, is similar across the entire corpus: 13.6 % of sentences in the free texts, 12.7 % of sentences in the full papers, and 13.5 % of sentences in the abstracts contain negation.

Figure 2.6 shows an example sentence from the data set.

2.3.3. SFU Review Corpus

The Simon Fraser University (SFU) Review Corpus [Konstantinova et al., 2012] is a collection of over 400 documents of movie, book and consumer product reviews annotated at the token level for the presence of linguistic negation and speculation. In total, it contains over 17,000 annotated sentences, and was the first publicly available negation-annotated corpus released from the review domain. Figure 2.7 shows an example sentence from the corpus.

```
<sentence id="S26.8">
  These findings
  <xcope id="X26.8.2">
    <cue type="speculation" ref="X26.8.2">indicate that</cue>
    <xcope id="X26.8.1">
      corticosteroid resistance in bronchial asthma
      <cue type="negation" ref="X26.8.1">can not</cue>
      be explained by abnormalities in corticosteroid
      receptor characteristics
    </xcope>
  </xcope>.
</sentence>
```

Figure 2.6.: Example sentence from the BioScope Corpus

```
<SENTENCE>
    <W>I</W>
    <cue ID="15" type="speculation">
        <W>would</W>
    </cue>
    <xcope ID="16">
        <ref COMMENTS="" ID="17" SRC="15"/>
        <W>definitely</W>
        <cue COMMENTS="" ID="11" type="negation">
            <W>not</W>
        </cue>
        <xcope ID="12">
            <ref ID="13" SRC="11"/>
            <W>recommend</W>
            <W>it</W>
            <W>to</W>
            <W>anybody</W>
        </xcope>
    </xcope>
</SENTENCE>
```

Figure 2.7.: Example sentence from the SFU Review Corpus

3. Related Work

This chapter first looks at the recent development within the field of sentiment analysis (SA) and discusses a shift of the field towards Twitter data. The state-of-the-art in Twitter sentiment analysis (TSA) is presented based on the International Workshop on Semantic Evaluation (SemEval). Related work within negation scope detection is also presented.

3.1. Literature Review Method

Our method of literature search has been centred around the SemEval shared tasks on TSA. As a starting point, we considered the systems that performed best in the shared tasks. In addition, we received a number of introductory papers from our supervisors. This approach has given us considerable freedom to continuously pursue the topics we deem interesting, as we discover them.

An alternative approach to searching and reviewing literature that we initially considered is the Structured Literature Review (SLR) procedure as explained by Kofod-Petersen [2012]. The method consists of defining research questions, quality criteria and a search protocol, and then executing said protocol step-by-step, assessing each paper according to the criteria. Among the advantages of this method is the wide area of research covered, which can reveal existing solutions that could otherwise have been overlooked. The wide coverage may also avoid research bias, and the results of the review may be useful for the research community as an overview of the area in question.

We did not see the need for performing an SLR for the sake of making a contribution to the research community, because Selmer & Brevik [2013] already did extensive work in that regard. Additionally, SLR requires spending time on tasks that neither contribute to our project nor our understanding of the field of research, such as performing and documenting

quality assessment of the research material.

There are some possible weaknesses to our chosen approach. Because several SemEval papers refer to similar sets of literature, as well as each other's systems, there is a possibility of ending up with a limited literature space resulting in a biased perception of the field. We have kept this in mind throughout the process, however, and have been careful to perform our own literature searches when researching important topics.

3.2. Recent Developments Within Sentiment Analysis

Exploring popular opinion on various subjects has always been an important part of humans' information gathering behaviour. Where one in the past needed to conduct surveys to learn about opinion trends, for instance to conduct political polls, the availability of online data expressing sentiment has allowed for non-intrusive data mining to extract this information.

Over the last decade, there has been a substantial increase in the amount of work done in the field of SA. Surveys conducted by Pang and Lee [2008] and Liu and L. Zhang [2012] give a good overview of the state-of-the-art at the respective points in time. The work in the field of SA has largely followed the available data, both in terms of the amount of work done and the focus area. Figure 3.1 shows the amount of hits for queries (3.1) and (3.2) on Google Scholar,[1] displaying a shift of the field towards Twitter data in recent years.

```
"opinion mining" OR "sentiment analysis"
      OR "opinion classification"                    (3.1)
```

```
"opinion mining" OR "sentiment analysis"
      OR "opinion classification"
AND "micro-blogging" OR "microblogging" OR "twitter"   (3.2)
```

[1] http://scholar.google.com

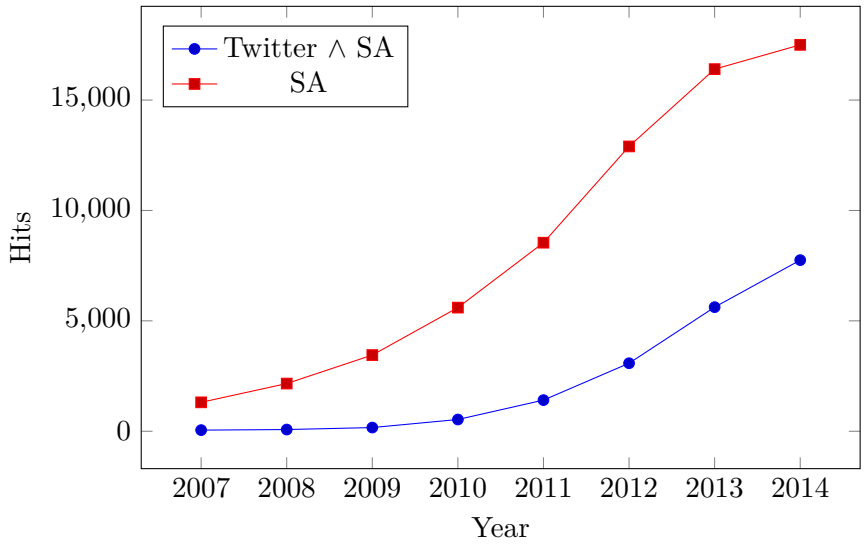


Figure 3.1.: Google Scholar hits for queries *SA*: (3.1) and *Twitter* ∧ *SA*: (3.2), for articles published from 2007 up to and including 2014

3.3. International Workshop on Semantic Evaluation

The International Workshop on Semantic Evaluation (SemEval)[2] is a series of evaluations of computational semantic language analysis systems. In recent years, it has been hosted annually. Each iteration of SemEval has a set of tasks. Tasks are hosted by experts within the field of study, who assist participants by providing resources such as training data and facilitating communication between teams. SemEval-2013 and SemEval-2014 both included tasks for TSA, see Nakov et al. [2013] and Rosenthal et al. [2014]. Additionally, SemEval-2015 has two TSA shared tasks, and two TSA-related shared tasks will be included in SemEval-2016. Recent SemEvals have yielded significant improvements to the state-of-the-art of TSA. This will be discussed in Section 3.4.

The TSA tasks in SemEval-2013 and SemEval-2014 included two sub-

[2] `http://aclweb.org/aclwiki/index.php?title=SemEval_Portal`

#	Team Name	Twitter-2014	Twitter-2013	SMS	Sarcasm	Live-Journal
1	TeamX[a]	70.96	72.12	57.36	56.50	69.44
2	Coooll[b]	70.14	70.40	67.68	46.66	72.90
3	RTRGO[c]	69.95	69.10	67.51	47.09	72.20
4	NRC-Canada[d]	69.85	70.75	70.28	58.16	74.84
5	TUGAS[e]	69.00	65.64	62.77	52.87	69.79
6	CISUC_KIS[f]	67.95	67.56	65.90	55.49	74.46
7	SAIL[g]	67.66	66.80	56.98	57.26	69.34
8	Swiss Chocolate[h]	67.54	64.81	66.43	49.46	73.25
9	Synalp-Empathic[i]	67.43	63.65	62.54	51.06	71.75
10	Think_Positive[j]	67.04	68.15	63.20	47.85	66.96

[a] Fuji Xerox
[b] Harbin Technical Institute
[c] Retres Co and Gothenburg U.
[d] National Research Council Canada
[e] INESC-ID em Lisboa
[f] U. Coimbra
[g] U. Southern California
[h] ETH Zürich
[i] U. Lorraine
[j] IBM Research Brazil

Table 3.1.: The top 10 ranking submissions for SemEval-2014 Subtask B

tasks: a term-level subtask (Subtask A) where the aim was to classify the contextual polarity of a term in a tweet and an expression-level subtask (Subtask B) where the aim was to correctly classify the overall polarity of whole tweets. Subtask B is the one relevant to our project, and is the one we will focus on. Throughout the remainder of this report, when we refer to SemEval-2014, we are referring to Subtask B, unless explicitly stated otherwise.

In addition to providing training, development, and test data sets of annotated tweets, the task hosts also provide out-of-domain data sets to test the versatility and generalisability of the created submissions. The data sets provided for SemEval-2013 and SemEval-2014 are described in greater detail in Section 2.3.1. The top ranking submissions for SemEval-2014 are displayed in Table 3.1.

3.4. State-of-the-Art in Twitter Sentiment Analysis

In this section we present the current state-of-the-art in TSA by breaking the field down into several areas.

The typical approach to TSA uses a supervised machine learning system including three main steps: preprocessing, feature extraction, and training the classifier. To reduce noise and remove unnecessary information, the preprocessing step consists of a variety of filters for, e.g., normalizing URLs and elongated words. Features for the classifier are extracted using sentiment scores from polarity lexica, statistics from metacommunicative expressions specific to conversational language such as emoticons and hashtags, as well as natural language processing information including bag-of-words, part-of-speech tags and word clusters. Finally, training the classifier is usually a matter of performing a grid search over the parameter space for selecting the most suitable parameters for a supervised machine learning model.

3.4.1. Tweet Preprocessing

Common preprocessing tasks in TSA include filtering out or normalizing URLs and user mentions, because these items have minimal information value in the context of sentiment classification. Agarwal et al. [2011] perform this normalization by substituting user mentions with the tag `||T||` and URLs with the tag `||U||`.

Another Twitter-specific syntactic feature is prefixing tweets with "RT" to indicate that the following part of the tweet is a *retweet* — a repost of previous content. A simple way of handling this is to remove the "RT" string from the tweet.

It is also common to normalize elongated words, e.g., *cooooooll*, *soooooooo*, or *happyyyyyyy* by substituting letters that occur many times sequentially with one or two occurrences of the letter.

It was previously quite common to filter out hashtags [Selmer & Brevik 2013]. The assumption behind this is that hashtags when used as intended — i.e., to categorize posts by topic — offer little information of value. Mohammad [2012] show through experiments that hashtags add sentiment-semantic information to tweets by indicating the tone of the

message or the writer's emotions. The following tweet in relation to the *Occupy Wall Street* (OWS) movement is an example of this:

We are fighting for the 99 % that have been left behind.
#OWS #anger

Go et al. [2009] perform the typical preprocessing steps: URL, user mention and word-elongation normalization, and achieve a reduction of the feature space dimensionality by 45.85 % after constructing their feature vector further down the classification pipeline.

3.4.2. Prior Polarity Lexica

Prior polarity lexica consist of words with their corresponding prior polarity values. These prior polarity values are each word's *out-of-context semantic polarity associations* — a score representing its sentimental orientation. A feature set similar to the feature set in Table 3.2 is typically generated for each polarity lexicon. These prior polarity based features are among the most important in TSA along with the vectorized tweet tokens, as shown in several ablation studies [Leal et al., 2014; Kiritchenko et al., 2014; Kouloumpis et al., 2011].

Feature	**Description**
Total Score	$\sum_{\text{token}\in\text{tweet}} score(token)$
Maximum Score	$\max_{\text{token}\in\text{tweet}} score(token)$
Last Subjective Term Score	Score of the last term with $score > 0$

Table 3.2.: The associated set of features for each prior polarity lexicon in a state-of-the-art feature set

Many different prior polarity lexica exist. Miura et al. [2014] describe a solution for SemEval-2014 based heavily on the use of polarity lexica, and divide them into two categories: formal and informal. Formal lexica contain only correctly spelled terms that are formal words in written English (e.g., present in the dictionary), while informal lexica also include erroneously spelled words and slang. Prior polarity lexica used in TSA include *The MPQA Subjectivity Lexicon* [Wilson et al., 2005], *SentiWordNet*

[Baccianella et al., 2010], *AFINN-111* [Nielsen, 2011], *Bing Liu's Opinion Lexicon* [Ding et al., 2008], *NRC Emotion Lexicon* [Mohammad and Turney, 2010], *NRC Hashtag Sentiment Lexicon*, and *Sentiment140 Lexicon* [Kiritchenko et al., 2014]. The first three are formal lexica, while the rest are informal.

The Sentiment140 and NRC Hashtag Sentiment lexica were created by Kiritchenko et al. [2014] for their solution to SemEval-2014. They are particularly interesting because they include two entries for each term: a negated and an affirmative context entry, each with a corresponding sentiment score. They were automatically created using a large corpus of tweets collected by the NRC Canada group, labelled as positive or negative by the presence of a set of polarity-indicating emoticons and hashtags for Sentiment140 and Hashtag Sentiment, respectively. Then, the tweets in the lexica were annotated with negative contexts following the work of Pang et al. [2002]: a negated context is created from a negation word (e.g., *no*, *shouldn't*), to the first punctuation mark encountered. The two lexica were created by calculating the *point-wise mutual information* [Bouma, 2009] of each of the positive and negative tweets, where values from tokens in a negated context were added to the negative context lexica while the rest were added to the affirmative context lexica.

3.4.3. Negation Handling

The ability to handle linguistic negation of terms — described in Section 2.1.2 — is an important aspect of sentiment classification. When using supervised learning, negated terms are typically represented as a feature in the feature vector. Thus, the problem of handling negators is limited to identifying the amount of other terms the negator affects, also known as the *scope* of the negator. A simple approach, when for instance using a bag-of-words word n-gram feature, is to prefix the negated terms with `_not_`. A naïve, and still quite widely used approach in TSA, is to set the scope of negation as all terms from the negator until the first punctuation encountered: Section 6.2 contains a more detailed description of this solution.

If the model includes a prior polarity lexicon of terms and their sentiment values, the process of handling negation becomes more complex. An intuitive approach is to invert the polarity of the sentiment score of the

negated terms. For instance, if a word with a sentiment score of 0.5 was located in a negated context, the polarity would then be shifted to -0.5. This assumption has been shown to be incorrect [Kiritchenko et al., 2014]. Positive terms, e.g., *fantastic*, when negated tend to shift their polarity and decrease their intensity (*not fantastic*), while negative terms tend to stay negative with a reduced intensity (e.g., *terrible* ⇒ *not terrible*).

Arguably the biggest development of the state-of-the-art of TSA following SemEval-2014 was the negation handling technique using the lexical approach introduced by Kiritchenko et al. [2014], described in the previous section. The lexica provided a significant performance increase, shown by their ablation study: the two lexica contributed an average of 5.83 % to the F_1 score of their system across the five datasets (including the sarcasm data set).

3.4.4. Feature Extraction

Many submissions for SemEval-2014 based their feature matrix on the top submission from SemEval-2013 [Mohammad et al., 2013]. Tang et al. [2014] define this feature set as the state-of-the-art feature set, labelling it STATE. We will adopt this definition in our project. The STATE feature set includes most typically used features, and is built up as follows for each tweet:

- *N-grams:* The presence of word n-grams (where $1 \leq n \leq 4$) and character n-grams (where $3 \leq n \leq 5$).
- *Sentiment Lexica:* A set of sentiment lexica, each with a group of features as shown in Table 3.2
- *Clusters:* The presence of words from each of the 1000 clusters from the Twitter NLP tool (see Section 2.2.1).
- *All-Caps:* The number of words with all letters capitalized.
- *Elongated Words:* The number of elongated words.
- *Emoticons:* The presence of positive and negative emoticons.
- *Punctuation:* The number of contiguous sequences of periods, question marks or exclamation points.

- *Negation:* The number of individual negated terms within the tweet. Negated terms are also marked as being in a negated context for sentiment lexicon look-ups.

3.4.5. Machine Learning Algorithms

Most well-performing systems for TSA use a supervised machine learning-based classifier. The submissions to SemEval-2014 follow this trend; almost every submitted system uses a machine learner, and all of the top performing systems use machine learning [Rosenthal et al., 2014].

Rank	**Team Name**	**Algorithm**
1	TeamX	Logistic Regression
2	Cooolll	Support Vector Machine — Linear
3	RTRGO	Stochastic Gradient Descent
4	NRC-Canada	Support Vector Machine — Linear
5	TUGAS	Logistic Regression
6	CISUC	Support Vector Machine — RBF
7	SAIL	Naïve Bayes
8	Swiss-chocolate	Support Vector Machine — Linear
9	Synalp-Empathic	Support Vector Machine (SMO)
10	Think_Positive	Neural Networks

Table 3.3.: The classification algorithms used by the top ranking systems for SemEval-2014 Subtask B

Table 3.3 shows the classification algorithms used by the ten top ranking submissions to SemEval-2014. Sequential Minimal Optimization (SMO) [Platt, 1998] is an optimizing variation for training the Support Vector Machine (SVM) classifier (the Synalp-Empathic group does not specify which SVM kernel they use), and as made apparent by Table 3.3, SVM and Logistic Regression classifiers were popular choices in SemEval-2014. These are both fast machine learning models, and SVM is considered the state-of-the-art for text classification and high-dimensional sparse feature spaces [Kiritchenko et al., 2014].

The Naïve Bayes classifier is considered one of the traditional text classification models, and has been proven strong for sentiment classification

of micro-blogs [Bermingham and Smeaton, 2010]. It has, however, only been used by one of the top-10 ranking teams, *SAIL*, and they use a special algorithm: a decision tree with Naïve Bayes classifiers on each leaf. The fall in popularity of Naïve Bayes may be caused by the increasingly complex and conglomerated feature matrix.

3.4.6. SemEval-2015

The papers from the SemEval-2015 shared task on TSA appeared as we were writing the last sentences of this thesis, and so could not be included in any of the comparisons or studied in detail, although the task overview paper [Rosenthal et al., 2015] actually mentions negation as one of the areas that this year's systems focused on. It is interesting to note, however, that almost all of the participating systems that included some treatment of negation in fact used the naïve approach described in Section 6.2, that is, to mark as negated all terms from a detected negation cue to the next punctuation. One system [Z. Zhang et al., 2015] included an even simpler treatment, just using a binary feature to flag the presence or absence of a negation cue in a tweet.

One of the top-ranked systems, KLUEless [Plotnikova et al., 2015], used a heuristic of assigning a negation cue a scope over the 4 following tokens. A number that compares well with the 3.8 average tokens in the negation scope for the NTNU Twitter Negation Corpus (see Table 4.3). Only one of the TSA systems in SemEval-2015 utilized a more sophisticated treatment of valence shifters (Section 2.1.2): Cerezo-Costas and Celix-Salgado [2015] trained a CRF-based classifier to detect the scope of what they call "denier particles" (i.e., negation) and "reversal verbs" (e.g., avoid, prevent, solve), that reverse the polarity of the terms in their scope. Their system did not perform that well over all on the shared task, but was the best-performing one on the 2014 tweet sarcasm data set.

3.5. Negation Scope Detection

The natural language processing sub-field of identifying the scope of linguistic negation, or negation scope detection (NSD), presented in Section 2.1.2, is relatively young. Originally, the main area of application

was biomedical texts, such as clinical reports and discharge summaries, as correctly identifying negation plays a critical role when automatically extracting information from this data. Early solutions were typically rule-based, such as the NegFinder and NegEx systems developed by Mutalik et al. [2001] and Chapman et al. [2001], respectively, which both heavily incorporate the use of regular expressions.

The BioScope corpus [Vincze et al., 2008] is a corpus of biomedical texts annotated for the presence of negation and speculation (see Section 2.3.2). It was the first extensive negation-annotated corpus and significantly boosted research on NSD, in large part due to facilitating for the use of supervised machine learning. Morante and Daelemans [2009] developed a meta-classifying NSD system on this corpus that as of 2010 was considered the state-of-the-art.

The Conference on Computational Natural Language Learning (CoNLL) hosted a shared task in 2010 with a sub-task aimed at detecting speculation cues and their affected scope [Farkas et al., 2010], a very similar task to NSD. The BioScope corpus was among the corpora used for training and evaluation of this sub-task. Morante and Daelemans submitted a new system to this task and achieved the best F_1-score of all participating teams, but it is difficult to compare the performance to their system released in 2009, as it was as far we could find not evaluated for NSD.

Additionally, NSD has been the focus of a shared task hosted by The Joint Conference on Lexical and Computational Semantics, *SEM 2012 [Morante and Blanco, 2012]. The hosts of this shared task created a new negation-annotated data set based on the literary works of Sir Arthur Conan Doyle, named the CD-SCO data set. Most well-performing submissions to both tasks used a supervised machine learning-based approach.

The area of application for research on NSD has in recent times shifted focus towards SA. Wiegand et al. [2010] performed a survey exploring the effects of negation on SA, concluding that it is "highly relevant". Moilanen and Pulman [2007] were among the first to implement an SA system with a sophisticated NSD mechanism, labeling the phenomenon *polarity reversal*. Their approach was focused on linguistic syntactic composition. Councill et al. [2010] created a small negation-annotated corpus consisting of customer reviews from Google Products and used it to develop a Conditional Random Fields (CRF)-based negation classification system.

The effect of this system on a sentiment classifier was explored, and it was reported to provide a "dramatic" increase in performance. Konstantinova et al. [2012] also created a customer review corpus annotated for negation, the SFU Review Corpus, described in Section 2.3. As opposed to the corpus created by Councill et al., containing only 2,111 sentences in total, this corpus contains over 17,000 annotated sentences and poses no size concerns for most purposes.

The remainder of this chapter explores in detail three solutions to handling linguistic negation: Morante and Daelemans' solution using meta-learning, Councill et al.'s CRF-based solution, and a shallow semantic parsing-based solution created by Zhu et al. We also evaluate the solutions' suitability for TSA.

3.5.1. Using Meta-Learning

Morante and Daelemans [2009] describe a negation scope detection system that uses meta-learning for classification. Their algorithm consists of two steps:

1. Signal identification
2. Scope identification

Feature	**Description**
Lemma	The canonical base form
POS	The part-of-speech tag
Chunk tags	The chunk tag of token to the left and right

Table 3.4.: The *Base* feature set for each token in the meta-learning negation scope detection system

The first phase of the algorithm uses a decision tree to classify if a token is the first token of a negation signal, inside a negation signal, or outside it, as a negation signal may consist of several words. The tokens are first preprocessed with the GENIA tagger,[3] a biomedical text processing

[3] `http://www.nactem.ac.uk/tsujii/GENIA/tagger/`

tool that splits the text into chunks and labels tokens with part-of-speech (POS) and Named-Entity tags. The chunking process segments a sentence into a set of constituents, such as noun phrase or verb phrase. These chunks are an important feature, and a reason some describe the algorithm as a *chunking-based approach*. The feature set used for the classifier is shown in Table 3.4. We label this feature set the *Base* feature set for each token in this system.

The second phase of the algorithm is performed by a meta-learner that uses predictions made by three classifiers to predict scope classes. The three classifiers providing input to the meta-learner are an SVM, a k-nearest neighbour classifier and a CRF classifier.

An instance represents a pair of a negation signal and a token. The meta-learning classifier is run on all instances. All tokens in a sentence are paired with all negation signals in the same sentence, identified in phase one. The feature set used by the three classifiers for each instance (signal s, token t) consists of:

- *Base* features of t
- *Base* features of one token to the left, and three tokens to the right
- s as a chain of words
- Chain of POS, chain of chunk tags and distance in number of tokens for all tokens between t and s

The meta-learner is also a CRF. For each instance it has a feature set similar to the first three classifiers that also contains their prediction results for the current token and the neighbouring tokens. After some post-processing, the scopes are identified using the labels assigned to each token.

The system was trained on the abstracts BioScope sub-corpus, and evaluated by Morante and Daelemans on the free text and full paper BioScope sub-corpora. The results are shown in Table 3.5. PCS stands for percentage of correctly classified scopes.

3.5.2. Using Shallow Semantic Parsing

Zhu et al. [2010] propose a rule-based approach to negation scope identification using shallow semantic parsing (SSP), also known as semantic

Corpus	Precision	Recall	F_1	PCS
Biological full papers	0.722	0.697	0.709	0.410
Medical free texts	0.864	0.821	0.842	0.708

Table 3.5.: System scores when running the meta-learning classifier on the medical free text and biological full paper BioScope subcorpora

role labelling.

shallow semantic parsing (SSP) is based around a predicate word. Given a parse tree and a predicate, SSP recognizes and maps all of the constituents in the sentence into their corresponding semantic arguments. Zhu et al. changed the algorithm slightly: they set the negation signal as the predicate, and then they find the negated scope by identifying the correct argument(s). An example parse tree is shown in Table 3.2.

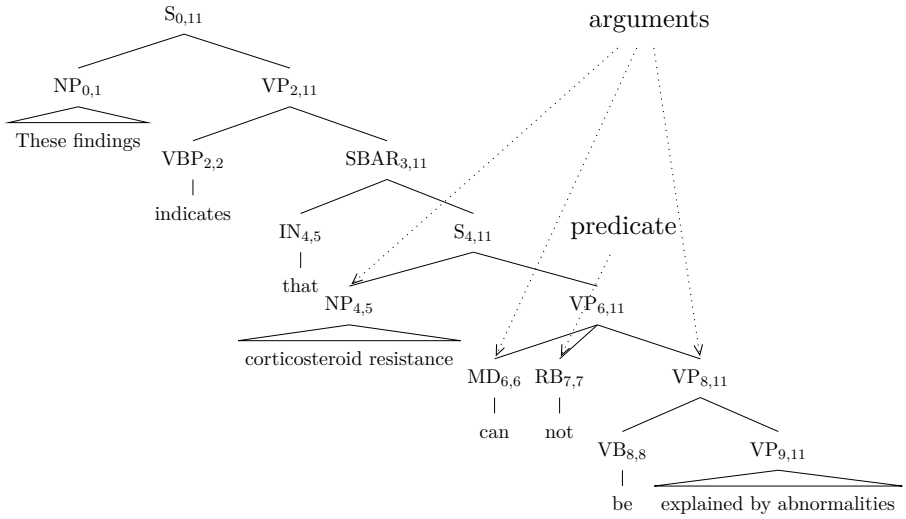


Figure 3.2.: A parse tree showing the arguments identified by the SSP algorithm applied to an example sentence.

They formulated two heuristic rules. Given a negation signal and its

negation scope which covers $\text{word}_m, \ldots, \text{word}_n$:

1. The negation signal itself and all of its ancestral constituents are non-arguments.

2. If constituent X is an argument of the given negation signal, then X should be the highest constituent dominated by the scope of $\text{word}_m, \ldots, \text{word}_n$.

The first rule is intended to prevent an argument covering the negation signal, while the second rule is meant to prevent overlapping arguments.

The negation scope algorithm consists of two main steps followed by a postprocessing step:

The first step is argument pruning. This step is done to filter out constituents that are very unlikely to be arguments of the negation signal by employing a heuristic. The algorithm is similar to the heuristic algorithm proposed by Xue and Palmer [2004]. It designates the negation signal as the current node and iteratively moves up one level in the tree and collects the current node's siblings. The algorithm terminates when it reaches the root. All the collected constituents are passed to the argument identification phase.

Argument identification is the second step, and is done by using a binary classifier — an implementation of the SVMLight algorithm.[4] Each argument is represented by a feature vector, and the classifier is given two possible labels: positive (an identified argument) or negative (not an identified argument). Many different features sets are tested, for a more detailed description see Zhu et al. [2010].

The system is also able to automatically identify negation signals. This is done using supervised learning; they do not state more precisely how they perform this. Like Morante and Daelemans, Zhu et al. train their classifier on the abstracts sub-corpus of the BioScope corpus, and evaluate the classifier on the full paper and free text sub-corpora, achieving the results shown in Table 3.6. Percentage of correctly classified scopes (PCS) denotes percentage of correctly classified scopes.

Corpus	Precision	Recall	F_1	PCS
Biological full papers	0.582	0.563	0.572	0.640
Medical free texts	0.806	0.822	0.814	0.898

Table 3.6.: System scores when running the SSP-based classifier on the medical free text and biological full paper BioScope sub-corpora

Feature	Description
Word	The lower case token string.
POS	The part-of-speech of the token.
Right Dist.	The linear token-wise distance to the nearest explicit negation cue to the right of the token.
Left Dist.	The linear token-wise distance to the nearest explicit negation cue to the left of the token.
Dep1 POS	The part-of-speech of the first order dependency of the token.
Dep1 Dist.	The minimum number of dependency relations that must be traversed to from the first order dependency head of the token to an explicit negation cue.
Dep2 POS	The part-of-speech of the second order dependency of the token.
Dep2 Dist.	The minimum number of dependency relations that must be traversed to from the second order dependency head of the token to an explicit negation cue.

Table 3.7.: Features for each token used by the CRF classifier

3.5.3. Using Conditional Random Fields

Councill et al. [2010] created a machine learning-based solution for identifying the scope of negation for improved SA. The system used an upstream dependency parser: an implementation of the MaltParser [Nivre et al., 2007]. The parser handles tokenization, performs POS-tagging, and creates a dependency parse tree. This is used to generate the feature vector shown in Table 3.7. Note that *Right/Left Dist.* means the distance between tokens in a sentence, while *Dep1/Dep2 Dist.* means distance between tokens in the dependency parse tree i.e., the amount of edges that must be traversed. The feature set was used to train a CRF classifier.

Because the intended usage area was predicting sentiment in online customer reviews, and it was assumed that the intended domain would likely contain language patterns that were uncommon in the text of professional biomedical bindings, a new data set was created by annotating a sample of 268 product reviews, containing 2,111 sentences whereof 679 contained negation.

Councill et al. only evaluate the system on the full paper sub-corpus of the BioScope Corpus, because this was deemed most similar to the text from the intended usage domain, customer reviews, using 5-fold cross validation. Additionally, they evaluate the system on the created customer review corpus using 7-fold cross validation. Results are shown in Table 3.8. PCS denotes percentage of correctly classified scopes.

Corpus	**Precision**	**Recall**	**F_1**	**PCS**
Reviews	0.819	0.782	0.800	0.398
BioScope	0.808	0.708	0.755	0.537

Table 3.8.: Cross validation scores for the CRF classifier on the BioScope full paper sub-corpus and the created customer review corpus

[4] `http://svmlight.joachims.org/`

3.5.4. Evaluation of Systems for TSA

When working with Twitter texts, a problem that arises is that the data can be very unstructured, containing slang, misspellings and various unconventional linguistic means, as described in Section 1.1. This requires a robust sentiment classifier that does not rely on features that are not reliably present in tweets, such as correct grammatical structure and use of punctuation.

The two phases of the negation scope detection algorithms can be split up and considered independently. That is to say, given that the features used in both phases of the algorithms can be recreated, it would for instance be possible to use the signal finding algorithm proposed by Morante and Daelemans with the scope finding algorithm of Councill et al.

Cue Detection

We could not find any previous work related to negation cue detection on Twitter data; most work has been done on biomedical texts. Two general approaches are used in linguistic negation cue detection: classification using machine learning, or a lexicon of explicit negation cues.

A simplification to consider is only looking at cues comprised of one word, as it is reasonable to assume that most cues take this form. Zhu et al. found that 96.57 % of the cues in the BioScope corpus consist of one word. This turns the classification problem into a binary one, as opposed to the three-class system used by Morante and Daelemans.

An intuitive approach is using a lexicon. Twitter data may make this difficult because the system must be able to identify cues written with misspellings and/or using slang. An approach to solving this problem is using the clusters created by Owoputi et al. [2013] (see Section 2.2.1) and consider the whole cluster for each token in the negation cue lexicon, e.g., the lexicon shown in Table 5.1. The cluster for the token *can't* is shown in Figure 3.3.

Scope Identification

The solution proposed by Zhu et al. may be difficult to transfer to TSA because performing a semantic parse on the data is difficult to do accurately, and errors in the early stages of the classification pipeline are amplified by

can't cant couldn't cannot cnt couldnt can't kant icant cudnt can't can't couldn't caint canny knt cantt cldnt culdnt carnt ican't cannt icnt cn't cannae can't could'nt caant cudn't cany -can't couldent #cant canna kan't coudn't canot cann't could't cain't canttt cldn't couldn't cyah cannnot coudnt ckant cant't couldn't caaant

Figure 3.3.: Tweet NLP cluster for the token *can't*

the following steps. The fact that it performs poorly in terms of F_1 score also makes it an unattractive solution to explore further, despite performing well on percentage of perfectly classified scopes. PCS is not overly important, as TSA makes extensive use of bag-of-words representations.

The concept of meta-learning should be well suited for handling noisy and unstructured data, as the classifiers working in cooperation provide robustness through redundancy. Despite this, Morante and Daelemans's system presents a similar problem to Zhu et al.'s: the phrase-chunk features generated by the GENIA tagger create requirements for the grammatical structure of sentences.

As Councill et al. put it: "[...] our work represents a simplified approach [to Morante and Daelemans] that replaces machine-learned cue prediction with a lexicon of explicit negation cues, and uses only a single CRF to predict negation scopes, with a more comprehensive model that includes features from a dependency parser." The simplicity of their system also makes it adaptable: when trained on user reviews and tested on the BioScope Corpus, the system was able to learn some negation signals that were not in its signal lexicon indirectly through their feature set.

The system scores on the biological full paper BioScope sub-corpus are shown in Table 3.9. CRF denotes the Conditional Random Fields-based scope detection identifier created by Councill et al. [2010], MetaLearn denotes the meta-learning solution created by Morante and Daelemans [2009], while SSP denotes the shallow semantic parsing-based solution created by Zhu et al. [2010]. The results are difficult to directly compare, because Councill et al. evaluated their system using cross-validation, unlike Zhu et al. and Morante and Daelemans who trained their systems on the abstracts sub-corpus. Nonetheless, the CRF classifier achieves an

Classifier	Precision	Recall	F_1	PCS
MetaLearn	0.722	0.697	0.709	0.410
SSP	0.582	0.563	0.572	0.640
CRF*	0.808	0.708	0.755	0.537

Table 3.9.: A comparison of system scores on the biological full paper BioScope sub-corpus. The asterisk (*) denotes that the system was evaluated using cross-validation.

impressive score.

The intended usage area of Councill et al.'s system is customer reviews, and the ability to handle the language of web users was a central requirement throughout the development of the system. This language, albeit typically more structured than Twitter language, is very different to that found in biomedical texts. The existence of a well-performing dependency parser for Twitter (see Section 2.2.1), allows for the implementation of the simplified CRF-based scope detection system created by Councill et al. These factors lead us to believe that this solution is the most well-suited to applying to TSA.

4. Annotation

Performing negation scope detection (NSD) using supervised machine learning requires a corpus of tweets annotated for negation cues and scopes. There are negation scope corpora available for other domains such as the BioScope corpus [Vincze et al., 2008] for the biomedical domain, and the SFU Review Corpus [Konstantinova et al., 2012] for customer product reviews written by web users. Although the BioScope corpus is of high annotation quality, the language differs from the typical Twitter language, as it is quite technical and contains very few spelling mistakes. The product review corpus is closer to our domain, since the authors are web users that write in an informal language containing misspellings.

Tweets contain additional linguistic means (as explained in Section 1.1) that distinguish them from the available corpora to the extent that we decided to develop a corpus consisting of annotated Twitter data.

One way to perform the annotation work is using crowdsourcing services such as Amazon Mechanical Turk (see Section 2.2.5), an effective way to achieve sufficient redundancy for measuring inter-annotator agreement, allowing for evaluation of annotations. Crowdsourcing services are hired to publish tasks for their workers to solve, in return for a compensation. This crowdsourcing technique was used in Mohammad and Turney [2010] to annotate emotions in tweets. Our initial plan was to use Mechanical Turk or Crowdflower[1] to annotate tweets for negation. As we developed the annotation guidelines (see Appendix A) for the crowdsource workers, however, we realized that the task would become too complicated and specific for the crowdsource services' general set of survey tools. In order to easily annotate tweets for negation, we developed a web application for this specific purpose.

Initially, we intended the application to be used by several people, in a crowdsourcing manner. This way, we hoped to avoid doing the annotations

[1] `http://www.crowdflower.com/`

ourselves — making them independent from the authors — and to get a higher level of redundancy with the help of more people. After having problems contacting Crowdflower, and discovering Mechanical Turk's US-only payment solution, we tried getting in contact with linguists through our university, but none were available. Because of time constraints, we finally decided to perform the annotation work ourselves.

4.1. Annotation System

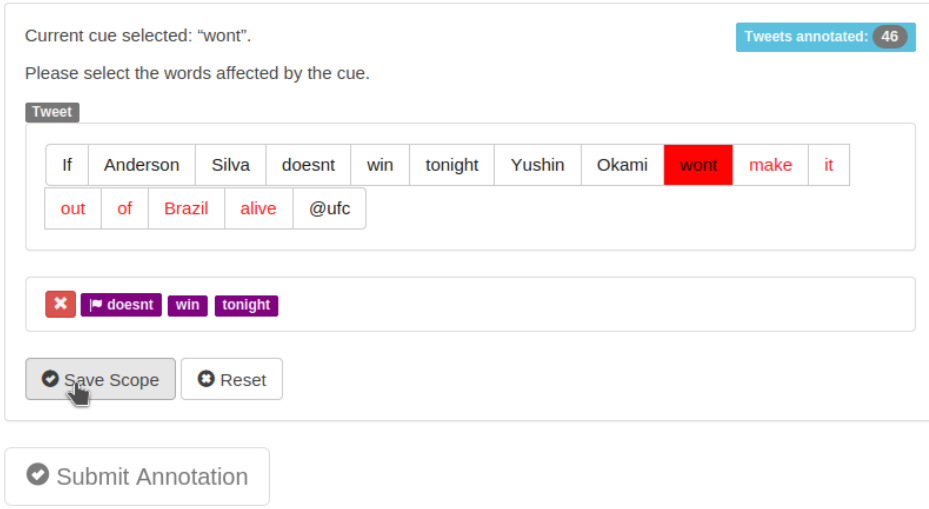


Figure 4.1.: Annotation system user interface.

Figure 4.1 shows the user interface created for annotating tweets. When the page is loaded, the application queries a tokenized tweet from the database and the tweet is displayed using a container of HTML buttons, where each button represents a token. The user can then click a token to mark it as a negation cue, and click the corresponding tokens to mark the cue's scope. When the user has selected all tokens in the scope, (s)he can press "save scope", to save the marked cue and scope. Saved scopes are displayed below the tweet, and can be removed if erroneous. The user repeats this until all instances of negation have been identified, and

presses “submit annotation” to save the annotation and query a new tweet. The system has a label displaying the current user’s number of annotated tweets, and a high score list to create a satisfying user experience.

4.2. Annotation Database

When the server receives a request for a new tweet to annotate, the query shown in Figure 4.2 is executed. The query excludes any tweets the current user has already annotated (lines 6–9), counts the number of annotations per tweet (line 3), and excludes tweets with two or more annotations. Finally, the server selects a tweet with the greatest number of annotations. This ensures that tweets with one annotation are selected first, completing the required number of redundant annotations per tweet.

```
1   SELECT
2   tweet.id,
3   COUNT(annotation.id) AS num_annotations
4   FROM tweet
5   LEFT OUTER JOIN annotation ON (tweet.id = annotation.tweet_id)
6   WHERE NOT (tweet.id IN (
7   SELECT annotation_user.tweet_id AS tweet_id
8   FROM annotation annotation_user
9   WHERE annotation_user.user_id = <user_id>))
10  GROUP BY tweet.id
11  HAVING COUNT(annotation.id) < 2
```

Figure 4.2.: SQL query for a tweet to annotate

Tweets are tokenized and stored as individual tokens in the database as illustrated by the Entity-Relationship diagram shown in Figure 4.3. When a user submits an annotation using the user interface, any selected scopes are stored as comma-separated lists of integers. Initially, the system was intended for multiple users to contribute to the annotation effort, requiring an email address, first and last names, and a password. However, as explained in the introduction to this chapter, we decided to perform the annotations ourselves, thus the user system is more extensive than necessary.

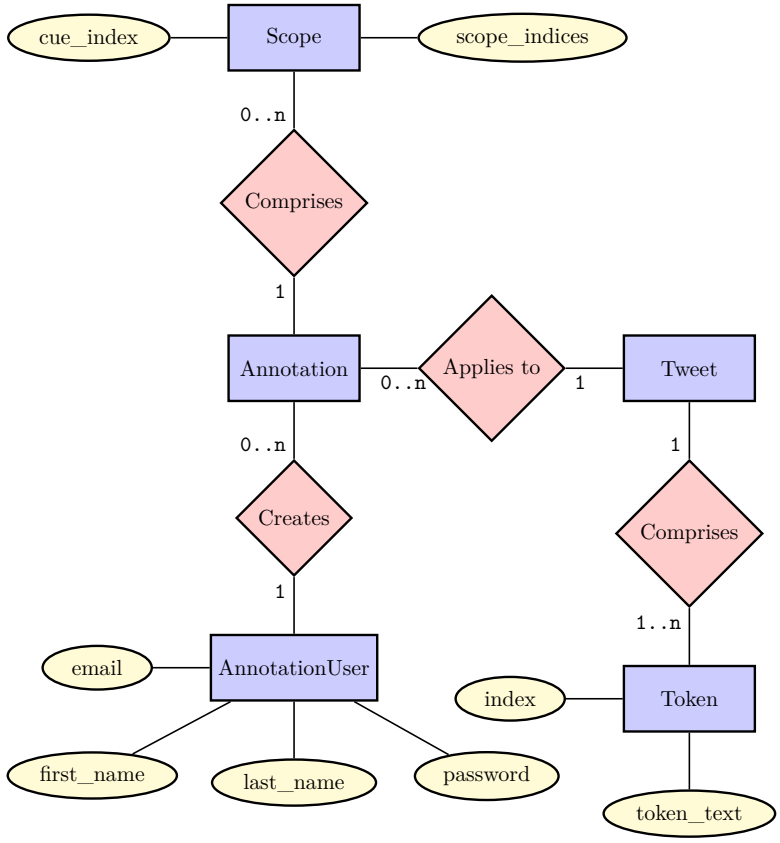


Figure 4.3.: ER diagram for the annotation database

The data set was exported from the database to XML format using the `lxml`[2] Python library. Figure 4.4 displays the structure of an example exported tweet. Scopes are linked with their corresponding cues using the "src" attribute.

[2] `http://lxml.de/`

```
<tweet id="17109">
  <cue id="436">
    <token>Don't</token>
  </cue>
  <scope id="44" src="436">
    <token>ask</token>
    <token>me</token>
    <token>about</token>
    <token>someone</token>
    <token>I</token>
    <cue id="437">
      <token>don't</token>
    </cue>
    <scope id="43" src="437">
      <token>care</token>
      <token>about</token>
    </scope>
  </scope>
</tweet>
```

Figure 4.4.: Annotated tweet in XML format

4.3. Inter-Annotator Agreement

To measure the agreement between the annotators, we calculated the token-wise and full scope agreements. The token-wise agreement score is the number of tokens both annotators agree upon divided by the total number of tokens in the dataset. This measure is quite unbalanced as the number of tokens in an affirmative context greatly outnumbers the number of tokens in a negated context as shown in Table 4.3. In order to calculate a full scope agreement score, we match every cue each of the annotators have marked. If a cue is marked by one annotator, but not by the other, a conflict is recorded. Otherwise, if the cues match, the corresponding scopes are matched using token indices. An agreement is recorded if both scopes match completely, while any difference results in a conflict. After

matching all tokens, the final score is calculated as follows:

$$full\_scope\_agreement = \frac{num\_agreed}{num\_agreed + num\_conflicts}$$

The resulting scores are presented in Table 4.1.

Token-wise agreement	98.9 %
Full scope agreement	73.8 %

Table 4.1.: Inter-annotator agreement

4.4. Conflict Resolution

Using the same scope matching method as in Section 4.3, we reviewed all conflicting scopes, discussing what should appear in the final corpus. In most cases, one of the annotators had made a mistake, and the conflict was easily resolved. Some sentence constructions were consistently in conflict, but were clarified after discussing with our supervisors. The most notable conflicts in doubt were sentences containing:

- Verb phrases with specification or condition. In the example "**dont** care what people say *when we're together*", the emphasized part was a case of doubt. It was decided to include this part in the negation scope because it is a part of the verb phrase (see the guidelines in Appendix A.2.5).
- Idioms such as in the example "**no** matter how good or bad you think your life is", or suggestions like "Why **not** make the best of your day!". The question was whether to include such constructs in the corpus. We decided to include them, concentrating on the syntax, and not letting the semantics raise doubt.

4.5. Resulting Corpus

4000 tweets downloaded using Twitter's API were annotated by the authors of this thesis, giving an annotation redundancy of two. Tweets were annotated in accordance to the guidelines in Appendix A.

General statistics about the tweets in the final corpus can be seen in Table 4.2, while statistics relating to negation are shown in Table 4.3. Tottie [1991] states that the frequency of negation in written English is 12.8 %. It is interesting to note that the frequency of tweets containing negation, 13.5 %, is quite close to this number.

It is uncommon to negate several separate scopes in the same tweet, as the average number of cues per negated tweet is quite close to 1. The average scope size is less than 4, suggesting that simple expressions are negated more often than long ones, and is possibly an indicator that simple language is used. For comparison, the scope length of the biological full paper sub-corpus of the BioScope corpus, which is known to contain complex language, is 8.8 on average [Morante and Daelemans, 2009]. Additionally, the average number of tokens per sentence in the full paper BioScope sub-corpus is 26.2, while the same statistic for our Twitter corpus is 10.2, something that may also serve to indicate simpler language. This indication is strengthened by the distribution of negation cues, which contains relatively few unique terms.

Number of tweets	4,000
Total number of tokens	61,172
Average number of tokens in tweet	15.3
Average number of tokens in sentence	10.2

Table 4.2.: General statistics of the NTNU Twitter Negation Corpus

Total number of scopes	615	
Number of tweets containing negation	539	(≈13.5 %)
Average number of cues per negated tweet	1.14	
Average number of tokens in scope	3.8	

Table 4.3.: Negation statistics of the NTNU Twitter Negation Corpus

Cue	Occurrences		Cue	Occurrences	
not	157	(9)	couldn't	4	
don't	100	(6)	wasn't	4	
no	90	(35)	cannot	3	(1)
never	48	(1)	dnt	3	
can't	48	(1)	none	2	(1)
didn't	24		shouldn't	2	
doesn't	19		aint	2	
won't	15	(1)	ain	2	
cant	14		hasn't	2	
dont	14	(1)	isnt	1	(1)
haven't	9		would'nt	1	
nothing	7	(15)	neeeever	1	
isn't	7		cudnt	1	
without	6	(6)	wont	1	
wouldn't	5		neva	1	
ain't	5	(1)	weren't	1	
didnt	5		neither	1	
aren't	4	(1)	eint	1	
idk	4	(3)			

Table 4.4.: Number of cue occurrences in the NTNU Twitter Negation Corpus. The number of times a token appears a non-cue (if any) is shown in parentheses.

5. Architecture

Two classifiers have been created: a classifier to detect the scope of negation by binarily classifying each token, and a sentiment classifier. The negation classifier is used in the feature extraction process for the sentiment classifier. Both have been implemented using the Python programming language.

5.1. Negation Classifier

The binary negation classifier created is a Twitter-tailored implementation of the Conditional Random Fields (CRF)-based system described by Councill et al. [2010] with one change: the dependency distance from the current token to the closest negation cue for each token has been added to the feature set.

The classification algorithm consists of two steps: negation cue detection and scope identification. Cue detection is performed using a pattern-matching approach with a lexicon of explicit cues. Additionally, all strings ending with *n't* are matched. These cues are adopted from Councill et al. [2010], and are displayed in Table 5.1. A dependency-based CRF classifier is then used to binarily classify each token as either being in a negated or affirmative context. When training the classifier, no automatic cue identification is performed, as actual cues from the annotated data are used.

5.1.1. Tweet Preprocessing

Tweets to be classified are first preprocessed with the TweeboParser dependency parser [Kong et al., 2014]. This parser first tokenizes each tweet, then produces part-of-speech (POS) tags for each token, and finally dependency parses each tweet, labeling each token with its dependency head.

hardly	lack	lacking	lacks	neither
nor	never	no	nobody	none
nothing	nowhere	not	without	aint
cant	cannot	darent	dont	doesnt
didnt	hadnt	hasnt	havent	havnt
isnt	mightnt	mustnt	neednt	oughtnt
shant	shouldnt	wasnt	wouldnt	*n't

Table 5.1.: Lexicon of negation cues. **n't* denotes all words with the suffix *n't*.

5.1.2. Feature Set

The feature set for each token is shown in Table 5.2. Linear token-wise distance is the number of tokens from one token to another, in the order they appear in a sentence. Dependency distance is calculated as the minimum number of edges that must be traversed in a dependency tree to move from one token, represented as a node, to another. The dependency tree is treated as a bi-directional graph, meaning that the traversals between nodes in the tree to measure dependency distance can be done both ways. A distance of 0 signifies that a token is a negation cue.

The classifier takes a parameter, *max distance*, that specifies the maximum distance to be considered. This applies to both linear distance and dependency distance. All distances above the max distance parameter are equivalent, and considered "far away".

5.1.3. System Implementation

We use the implementation of CRF created by Okazaki [2007], `CRFSuite` (see Section 2.2.4) with a Python binding created by Peng and Korobov [2014]. The classifier has been wrapped in classes to comply with the conventions of the `Scikit-learn` framework, allowing for easy interaction with the sentiment classifier, in addition to allowing for the use of `Scikit-learn`'s grid search and classifier scoring functions.

Feature	Description
Word	The lower-case token string
POS	The token's part-of-speech tag
Right Distance	The linear token-wise distance to the nearest negation cue to the right of the token
Left Distance	The linear token-wise distance to the nearest negation cue to the left of the token
Dep Distance	The minimum number of edges that must be traversed to reach the nearest negation cue from the token
Dep1 POS	The part-of-speech tag of the token's first order dependency head
Dep1 Distance	The minimum number of edges that must be traversed to reach the nearest negation cue from the token's first order dependency head
Dep2 POS	The part-of-speech tag of the token's second order dependency head
Dep2 Distance	The minimum number of edges that must be traversed to reach the nearest negation cue from the token's second order dependency head

Table 5.2.: Negation classifier feature set for each token

5.2. Sentiment Classifier

A Twitter sentiment analysis (TSA) system has been developed, using the `Scikit-learn` framework, described in Section 2.2.2. The machine learning pipeline includes three steps: preprocessing, feature extraction, and either training the classifier or classifying samples, as illustrated in Figure 5.1.

The Twitter2013-train data set (presented in Section 2.3) includes annotations with the labels `objective` and `objective-OR-neutral`. We simplify the classification task by treating both of these classes as `neutral`. Our system thus treats sentiment classification as a three-class problem (`positive`/`negative`/`neutral`).

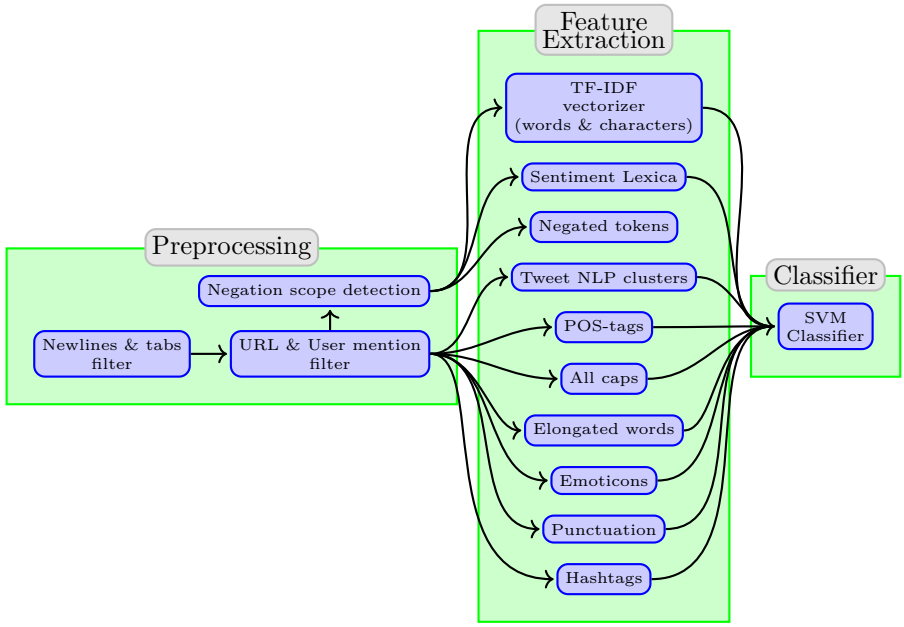


Figure 5.1.: The sentiment classification machine learning pipeline

5.2.1. Tweet Preprocessing

The preprocessing step performs some of the text-associated tasks described in Section 3.4.1 using regular expressions, and produces as output a cleaned corpus of tweets ready for feature extraction.

Tweet Text Filtering

Newline and tab characters are substituted with spaces in order to avoid parsing problems in other parts of the pipeline. In order to standardize URLs, we substitute URLs with the string "`http://someurl`" using a slightly modified regular expression by *@stephenhay*,[1] also matching URLs starting with "www" instead of a protocol specifier:

```
(?:www\.|(?:https?|ftp)://)[^\s/$.?#].[^\s]*
```

[1] `https://mathiasbynens.be/demo/url-regex`

Similarly, user mentions (user names starting with an "@" character) are substituted with the string "*@someuser*".

Negation Scope Detection

A selection of features in the feature set requires data where linguistic negation has been identified. These features are labelled *context-sensitive features*. Negation scope detection (NSD) of the filtered tweets is therefore performed as a tweet preprocessing step, and the output of this is sent to the context-sensitive features.

The system includes two mechanisms for performing NSD: a naïve method, marking every token between an identified negation cue and the first punctuation encountered as negated, and a sophisticated method, using the NSD system described in Section 5.1. Cues for the naïve method are identified with the same lexical approach that is used in the NSD classifier, using the cues shown in Table 5.1.

5.2.2. Feature Set

The feature set is represented as a `Scikit-learn FeatureUnion` that builds the feature matrix for the classifier. It contains a `Transformer` for each part of the STATE feature set, presented in Section 3.4.4.

We group the feature set into two categories, distinguishing between features that are affected by linguistic negation, *context-sensitive features*, and features that are unaffected, *context-insensitive features*. The complete feature set is shown in Table 5.3; the top four features displayed are context-sensitive, while the rest are context-insensitive.

Context-Sensitive Features

The context-sensitive features consist of several sentiment lexica, two term frequency-inverse document frequency (TF-IDF) vectorizers, and the number of negated tokens.

There are two TF-IDF vectorizers: one for word n-grams and one for character n-grams. Both ignore common English stop words, convert all characters to lower case, and select the $1,000$ features with the highest TF-IDF scores. In order to account for negation in the vectorized text, tokens in a negation scope are appended the string `_NEG`.

Feature	Description
Word n-grams	Presence or absence of contiguous token sequences
Character n-grams	Presence or absence of contiguous character sequences
Sentiment lexica	The feature set for each sentiment lexicon
Negated tokens	Number of negated tokens
Clusters	The presence or absence of tokens from each of the 1000 clusters from the CMU Twitter-NLP word clusters
POS	Number of each part-of-speech tag
All caps	Number of tokens with all characters capitalized
Elongated words	The number of words with a character repeated more than two times
Emoticons	The number of positive and number of negative emoticons
Punctuation	Number of contiguous sequences of exclamation or question marks
Hashtags	Number of hashtags

Table 5.3.: Sentiment classifier feature set for each tweet

The feature set includes a transformer for sentiment lexicon score calculation. All prior polarity lexica are handled by this `Transformer`. The *NRC Hashtag Sentiment Lexicon* and *Sentiment140 Lexicon* [Kiritchenko et al., 2014] contain sentiment scores for words in negated contexts. For lookups, the first negated word in a negation scope is appended with the string `_NEGFIRST`, and the rest with `_NEG`. The actual sentiment lexica feature vectors are adopted from Kiritchenko et al. [2014], and contain:

- the number of tokens with $score(w) \neq 0$
- the total score $= \sum_{w \in tweet} score(w)$
- the maximal score $= \max_{w \in tweet} score(w)$
- the score of the last token in the tweet

We also use the *MPQA Subjectivity Lexicon*, *Bing Liu's Opinion Lexicon*, and *NRC Emotion Lexicon*. These classify the words instead of assigning a numerical sentiment value, so we use a value of +1 for *positive* words, and a value of −1 for *negative* words. Since the *MPQA Subjectivity Lexicon* assigns the words a strong or weak degree of sentiment, we use scores of $+/-2$ and $+/-1$ for strong and weak degrees of sentiment respectively. Also adopted from Kiritchenko et al. [2014], the resulting feature vectors are as follows:

- the sum of positive scores for tweet tokens in affirmative contexts
- the sum of negative scores for tweet tokens in affirmative contexts
- the sum of positive scores for tweet tokens in negated contexts
- the sum of negative scores for tweet tokens in negated contexts

The *negated tokens* feature is simply the number of tokens in a negated context.

Context-Insensitive Features

Instead of adding only the presence of clusters in the cluster feature, like Kiritchenko et al., we count the number of occurrences for each cluster, and represent them with a feature. Using the POS tagger from the Tweet NLP collection (see Section 2.2.1), the number of each POS-tag is also in the feature set. The emoticons feature is the number of happy and sad emoticons, and whether the last token in a tweet is a happy or a sad one.

The all-caps, elongated words, punctuation and are created according to the STATE feature set. All the matrices from the different parts of the feature extraction are concatenated column-wise into the final feature matrix, and scaled in order to be suitable as input to a classifier.

5.2.3. Classifier

The classifier step declares which classifier to use, along with its default parameters. We use a Support Vector Machine (SVM) classifier because it is a state-of-the-art learning algorithm proven effective on text categorization tasks, and robust on large feature spaces (see Section 3.4.5). We

use `SVC`, an SVM implementation from `Scikit-learn`, based on libsvm [Chang and Lin, 2011]. It is passed the resulting feature matrix from the feature extraction step, with which it creates the decision space if training, or classifies samples if predicting. The process of grid searching for the optimal SVM parameters is described in Section 6.4.1.

5.2.4. Live Tweet Classification

In order to apply our classifier to live Twitter data, we wrap the classifier in a web application, depicted in Figure 5.2. The application uses the Django web framework[2] to allow a user to query Twitter for a search phrase. The resulting tweet hits are classified using a pre-trained classifier, and presented to the user indicating their sentiment polarities. The total distribution of polarity is displayed as a graph to give the user an impression of the overall opinion of the tweets matching the specified query.

[2] `https://djangoproject.com`

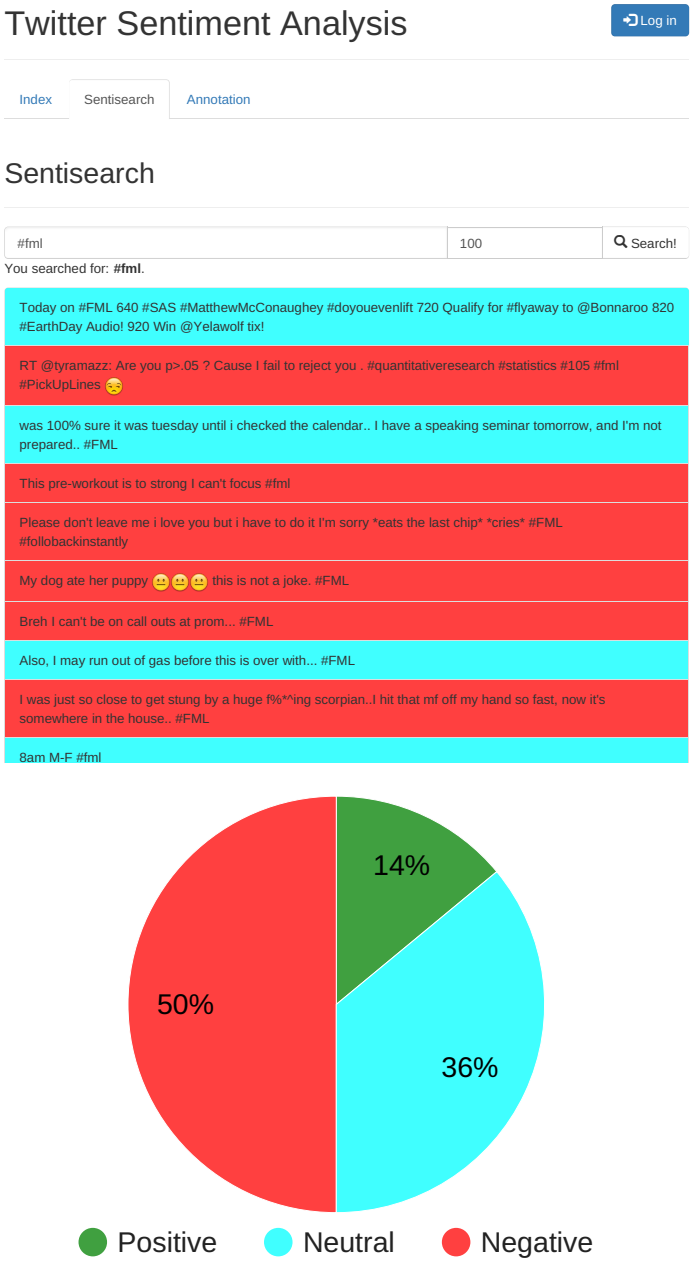


Figure 5.2.: Screenshot of the search sandbox

6. Experiments

Three sets of experiments have been conducted. A baseline negation experiment was executed at the beginning of the project, exploring the performance of a naïve negation scope detection (NSD) solution, commonly used in state-of-the-art Twitter sentiment analysis (TSA) systems. Additionally, experiments were conducted on each of the two classifiers that have been created. The classifiers are described in Chapter 5.

6.1. Evaluation Measures

For evaluating classifier performance, both when performing negation and sentiment classification, we adopt the standard metrics common in the respective fields. All of the evaluation metrics described in this section are presented in Section 2.1.3.

6.1.1. Negation Classification Scoring

In NSD each token is classified as being either in an affirmative or a negated context. NSD is thus a binary classification problem, where the positive label is that a token is *negated*. It is normal to report the *precision*, *recall*, and F_1 scores for this label. Additionally, it is common to report the percentage of correctly classified scopes (PCS). We report these metrics in all of our NSD experiments.

6.1.2. Sentiment Classification Scoring

We treat sentiment classification as a three-class task, with the labels `positive`, `negative`, and `objective/neutral`. For classification scoring in sentiment analysis, it is common to report the *precision*, *recall*, and F_1 for each class, in addition to the *macro-average* of each aforementioned

metric across all classes. Because the F_1 score is a harmonic mean of *precision* and *recall*, it is sometimes sufficient to report this metric. We follow this convention in our experiments, reporting all of the mentioned metrics across all labels in the most important experiments, and a macro-averaged F_1 score elsewhere. Additionally, we report the *support*, for several experiments on the sentiment classifier: the number of samples for each label in the test set.

6.2. Baseline Negation Experiment

To explore the potential for improvement within NSD in the current state-of-the-art in TSA we implemented a naïve NSD solution on the BioScope Corpus and compared the results to the systems described in Section 3.5 [Councill et al., 2010; Zhu et al., 2010; Morante and Daelemans, 2009]. The NSD algorithm implemented in our experiment, shown in Algorithm 6.1, is a common solution in the current state of TSA, and is used by, among others, the three top ranking solutions in SemEval-2014 Subtask B [Miura et al., 2014; Tang et al., 2014; Günther et al., 2014]: When a negation signal is detected, all terms from the signal to the first punctuation are considered negated. We have chosen to work with the medical full paper sub-corpus of the BioScope Corpus, because it is the one most similar to text found in social media, and because all of the existing systems we compare performance to have been evaluated on this corpus.

6.2.1. Experiment Description

To implement the naïve scope detection, a Python script that parses the XML data, and searches through the sentences for the tokens shown in Table 5.1, marking them as a cue, was created. All tokens between a marked cue and punctuation are considered negated.

Results from evaluating the naïve scope identifier with 5-fold cross-validation, along with the results from the solutions described in Section 3.5 when tested on the same data, are shown in Table 6.1. CRF denotes the Conditional Random Fields-based scope detection identifier created by Councill et al. [2010], MetaLearn denotes the meta-learning

Algorithm 6.1 Naïve negation scope detection algorithm

```
negated ← false
tokens ← all tokens in the corpus
for all t in tokens do
  if negated then
    t ← t + "_NOT"
  end if
  if t in negation_signals then
    negated ← true
  end if
  if t in punctuation then
    negated ← false
  end if
end for
```

Classifier	Precision	Recall	F_1	PCS
MetaLearn	0.722	0.697	0.709	0.410
SSP	0.582	0.563	0.572	0.640
CRF	0.808	0.708	0.755	0.537
Naïve	0.583	0.688	0.631	0.437

Table 6.1.: System scores when tested on the BioScope biological full papers sub-corpus

solution created by Morante and Daelemans [2009], while SSP denotes the shallow semantic parsing-based solution created by Zhu et al. [2010]. Notably, the naïve negation scope detection algorithm is a strong baseline, which actually outperforms the shallow parsing approach on F_1 score and the meta-learning approach on correctly classified scopes.

6.3. Experiments on Negation Classifier

The created Conditional Random Fields (CRF) negation classifier, described in Section 5.1, was developed and evaluated on the NTNU Twitter Negation Corpus. More information about this corpus is available in

Chapter 4. The data set was split into two subsets: a development subset and an evaluation subset. The development subset consists of 3000 tweets, while the evaluation subset consists of 1000 tweets. The split was stratified, meaning that the ratio of tweets containing negation is the same in both subsets. This was done to ensure more reliable training and testing, considering the heavy label imbalance of the NTNU Twitter Negation Corpus. As noted in Section 4.5, only 13.5 % of the tweets contain negation.

6.3.1. Naïve Performance

Similarly to the experiment conducted in Section 6.2, we also implemented naïve NSD on the NTNU Twitter Negation Corpus, to establish a baseline for NSD on the corpus.

Two variants were implemented: marking only words between the cue *not* and punctuation as negated, and marking words between any cue in the cue dictionary, shown in Table 5.1 on page 54, and punctuation as negated. The former was implemented as this is a method still used by some TSA systems. Results when applying the two methods to the entire NTNU Twitter Negation Corpus are shown in Table 6.2. The two cue detection mechanisms, identifying only *not* or identifying any word in the cue dictionary, are denoted by *not* and *dictionary*, respectively.

Cue Detection	**Precision**	**Recall**	**F_1**	**PCS**
Not	0.654	0.277	0.389	0.123
Dictionary	0.591	0.962	0.733	0.431

Table 6.2.: Naïve negation scope detection performance on the NTNU Twitter Negation Corpus

We hereafter consider the performance of the naïve NSD solution using the complete cue dictionary as the baseline score for NSD on the NTNU Twitter Negation Corpus.

6.3.2. Cue Detection

The developed negation classifier performs cue detection using a lexicon-based approach, as described in Section 5.1. The performance scores for this cue detection method on the NTNU Twitter Negation Corpus are shown in Table 6.3.

Precision	**Recall**	**F_1**
0.873	0.976	0.922

Table 6.3.: Standard lexicon-based cue detection performance

In Section 3.5.4 we speculate that negation cue detection, or *signal finding*, may prove challenging on Twitter data due to a significant number of cues possibly being slang or misspelled words. The table of cue occurrences, Table 4.4 on page 52 shows that this assumption proved false for the NTNU Twitter Negation Corpus. All of the most frequent cues are correctly spelled and present in the dictionary, with the exception of missing apostrophes in contractions, something accounted for by the negation cue lexicon. This result is in line with that of Hu et al. [2013], who noted that tweets tend to be surprisingly formal in that they often follow grammatical norms and use standard lexical items.

Inspection of cue detection output shows that one of the main areas the classifier struggles with is the separation of words that are used both as a negator and an exclamation. This is illustrated by the relatively high recall in comparison to the precision. Table 4.4 also shows this, displaying the number of times a word that occurs as a cue occurs as a non-cue. Transitively, the table also displays which cues have been misclassified most frequently, as a pure pattern-matching approach is used. By far the most significant of these is the word *no*, occuring 35 times as a non-cue; often it occurs as a determiner and functions as a negator, such as in the phrase "there were no letters this morning", but it may occur as an exclamation, e.g., in the phrases "No, I'm not ready yet" and "No! Don't touch it".

Despite the high recall, it is reasonable to believe that cue outliers such as *dnt neva*, or *cudnt* could be detected by using word-clusters. We expanded the lexicon of negation cues to contain the whole set of Tweet NLP

Precision	Recall	F_1
0.535	0.992	0.695

Table 6.4.: Lexicon-based cue detection performance using Tweet NLP Clusters

word clusters created by Owoputi et al. [2013] (see Section 2.2.1) for each word in the lexicon. Results from this are shown in Table 6.4. Recall is increased by 0.016, while precision suffers a dramatic decrease of 0.338, resulting in an overall decreased F_1 of 0.227. This shows that expanding the cue lexicon to include Tweet NLP word clusters is unsuitable. This is due to the clusters being too inclusive; it may be that more finely-grained word clusters could achieve an increased recall without considerably decreasing precision.

6.3.3. Parameter Grid Search

To identify the best-performing parameters, a grid search was performed on the development data subset on the L1 and L2 CRF penalty coefficient parameters, $C1$ and $C2$, in addition to the *max distance* parameter (described in Section 5.1.2). Grid search was performed using 7-fold stratified cross validation, and the identified parameter-set was a $C1$, $C2$ and *max distance* of 0.1, 1, and 7, respectively. The parameter space used for the grid search is shown in Table 6.5, and the best-performing parameters are highlighted.

Max distance	5	6	7	8	9	10
C1	10^{-4}	10^{-3}	10^{-2}	0.1	1	10
C2	10^{-4}	10^{-3}	10^{-2}	0.1	1	10

Table 6.5.: Negation classifier grid search parameter space

6.3.4. Classifier Performance

The classifier performance with the parameter set selected through grid search was evaluated on the held-out evaluation data set. Results from the 7-fold cross validation on the development data set, and the test run on the evaluation data set are shown in Table 6.6.

Data Set	Precision	Recall	F_1	PCS
Evaluation	0.972	0.923	0.853	0.645
Development	0.849	0.891	0.868	0.663

Table 6.6.: Negation classifier performance

The classifier achieves very good results. The run on the evaluation data set produces an F_1 score of 0.853, considerably outperforming the baseline system's F_1 score of 0.733. It also outperforms Councill et al.'s scores when they apply a very similar system to their created customer review corpus and achieve an F_1 score of 0.800.

The trained negation classifier was also tested on the evaluation data subset with *gold standard* cue detection, meaning that the classifier was tested with perfect negation cue identified. The resulting performance scores are shown in Table 6.7.

Precision	Recall	F_1	PCS
0.841	0.956	0.895	0.663

Table 6.7.: Negation classifier performance with gold standard cues

6.3.5. Out-of-Domain Performance

Although the negation classifier created is a Twitter-tailored implementation of the system described by Councill et al. [2010] with minor modifications — the addition of the current token's dependency distance to the closest negation cue to the feature set for each token — the fact that a different CRF implementation is used in addition to a different part-of-speech (POS)-tagger and dependency parser may lead to considerable differences

in performance. To give an indication of the generalizability of the classifier, the classifier has been evaluated on the BioScope and SFU Review corpora. Like Councill et al., we only evaluate on the biological full paper sub-corpus of the BioScope corpus, as negation systems have typically encountered the most difficulty when processing this sub-corpus. An important thing to note is that a POS-tagger and dependency parser tailored towards Twitter language has been used in all cases. The performance of the classifier relative to existing systems, compared to the performance of the classifier on the NTNU Twitter Negation corpus, also gives a measure of the general difficulty of negation classification on Twitter data. These corpora are all described in Section 2.3.

The results are shown in Table 6.8. As expected, the classifier struggles the most when applied to the BioScope corpus, as this corpus has generally proven difficult to classify for negation (see Section 3.5), and is quite dissimilar to the data the classifier was trained on. The classifier is outperformed in terms of F_1 by the systems created by Morante and Daelemans [2009] and Councill et al. [2010] who achieve F_1 scores of 0.709 and 0.755 respectively, but performs slightly better than the system created by Morante and Daelemans in terms of PCS, shown in Table 3.5 on page 38, and Table 3.8 on page 41. The modest F_1 score can likely be explained by the use of upstream preprocessing tools tailored towards Twitter language, a language that differs significantly from that found in biomedical texts.

Corpus	**Precision**	**Recall**	**F_1**	**PCS**
BioScope	0.660	0.610	0.634	0.426
SFU Review	0.668	0.874	0.757	0.435

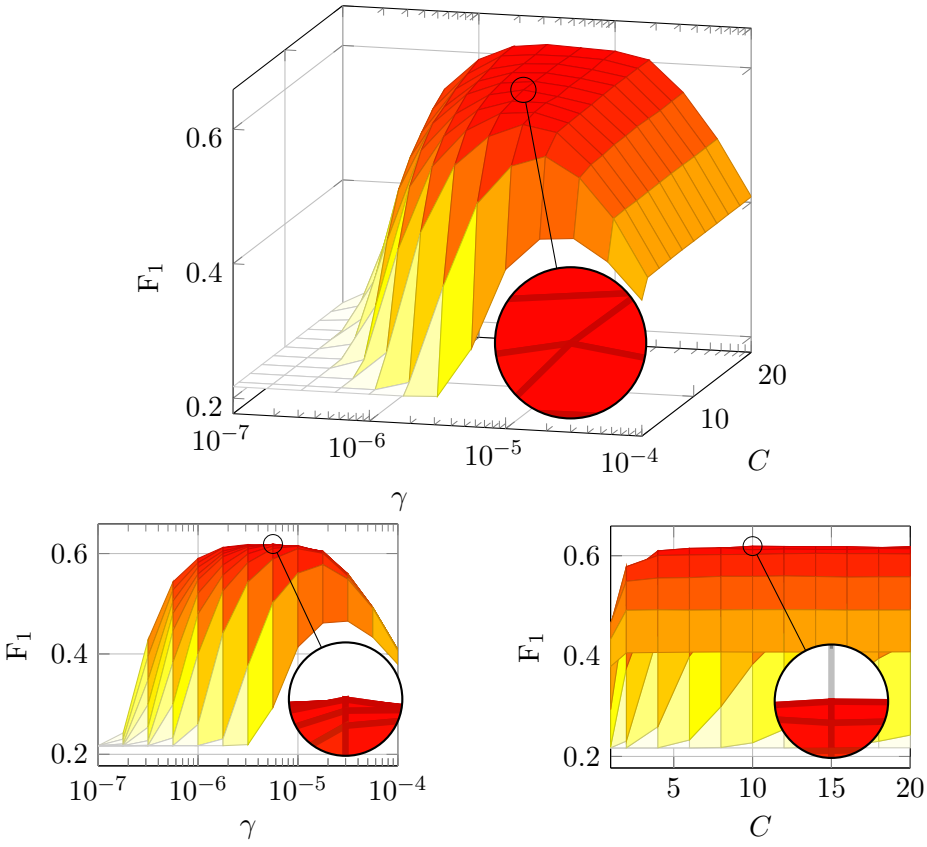
Table 6.8.: Negation classifier performance on alternative data sets

6.4. Experiments on Sentiment Classifier

A Support Vector Machine (SVM) was created, as described in Section 5.2, and trained on the Twitter2013-train data set. The test results were generated using the Twitter2013-test and Twitter2014-test data sets. All the data sets are described in Section 2.3.1.

6.4.1. Parameter Grid Search

The entire Twitter2013-train data set was used to find the C and γ parameters from a grid search using K-fold cross validation with $K = 10$. A K of 10 is a popular choice when the classifier has a steep learning curve, as is the case with ours. As the dataset is quite imbalanced, the data for each fold is split using stratification.



(a) With respect to γ

(b) With respect to C

Figure 6.1.: SVM grid search F_1 scores for C and γ

Using the negation scope classifier with the parameters identified in Section 6.3, a grid search was performed to determine the parameters for

the SVM classifier used for sentiment classification. A preliminary coarse search proved that the radial basis function (RBF) kernel yielded the best results, although most state-of-the-art sentiment classification systems use a linear kernel, as seen in Table 3.3 on page 33.

After an initial broad search, a finer parameter space was examined, displayed in Figure 6.1. The surface plots display the effect of the C and γ parameters on the classifier's F_1 score for the different parameter combinations explored. Figure 6.1a and Figure 6.1b show the same plot from different angles to more clearly see the impact of the γ and C parameters, respectively. Circles mark the maximum score in each plot.

The combination of parameters that scored best from the grid search was $C = 10$ and $\gamma \approx 5.6 * 10^{-6}$. As C increases beyond 10, there are no notable changes in terms of F_1 score. The combination of a small γ and higher values of C means that the classifier is quite generalized, and that increasing the C — thus regularizing further — makes no difference. It also suggests that the data is noisy, requiring a great deal of generalization.

Label	**Precision**	**Recall**	**F_1**	**Support**
`positive`	0.863	0.589	0.700	805
`neutral`	0.568	0.872	0.688	572
`negative`	0.717	0.487	0.580	156
avg / total	0.738	0.684	0.684	1533

(a) Twitter2014-test

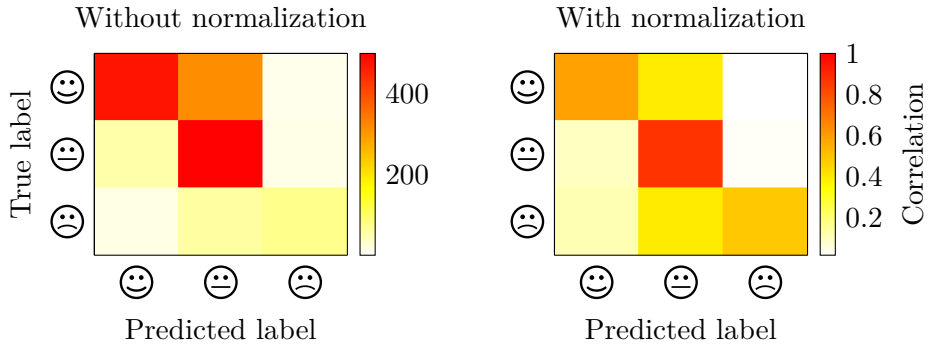
Label	**Precision**	**Recall**	**F_1**	**Support**
`positive`	0.851	0.581	0.691	1273
`neutral`	0.627	0.898	0.739	1369
`negative`	0.711	0.426	0.533	467
avg / total	0.731	0.697	0.688	3109

(b) Twitter2013-test

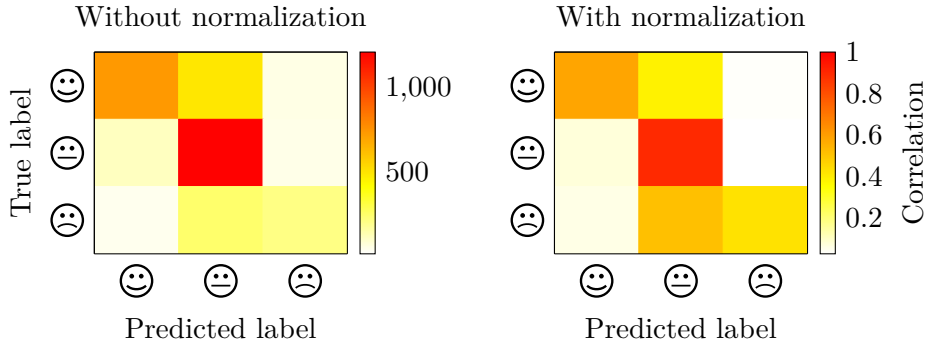
Table 6.9.: Sentiment classifier performance

6.4.2. Classifier Performance

An SVM was trained on the Twitter2013-train data set using the parameters identified through grid searching, and tested on the Twitter2014-test and Twitter2013-test data sets (presented in Section 2.3), scoring as shown in Table 6.9a and Table 6.9b.



(a) Twitter2014-test data set



(b) Twitter2013-test data set

Figure 6.2.: Confusion matrices for the sentiment classifier

Figure 6.2a and Figure 6.2b show confusion matrices for the Twitter2014-test and Twitter2013-test data sets, respectively. If all samples of the test sets were classified correctly, there would be perfect correlation between the true and predicted labels for each class, and the diagonal would be completely red. What can be seen from the confusion matrices, however

(the normalized versions make this more apparent), is that the classifier is quite biased towards the `neutral` label (illustrated with 😐). This can be seen from the warm colours in the `positive` and `negative` true label cells of the `neutral` column of predicted label.

From Table 6.9a and Table 6.9b, we can see that the classifier is least accurate classifying `negative` samples. In fact, Figure 6.2b shows that on the Twitter2013-test data set, `negative` samples are classified as `neutral` in most cases.

Both of these weaknesses are likely an effect of the imbalanced training set, where `neutral` samples greatly outnumber `negative` ones.

6.4.3. Ablation Study

The results of an ablation study of the developed TSA classifier are shown in Table 6.10. In order to investigate differences in feature impact, we include scores on the Twitter2013-test data set. This data set is also more than twice the size of the Twitter2014-test data set (see Table 2.1), which may help reduce the effect of unfortunate random factors.

Most apparently, removal of the sentiment lexica feature is the single ablation that has the greatest impact on classifier performance. This is especially the case on the Twitter2013-test data set. The cause for this may be the fact that the most important lexica (Sentiment140 and NRC Hashtag Sentiment, described in Section 3.4.2), were created at the same time as the Twitter2013-test data set, and could be more accurate on the particular language used in that period of time.

Interestingly, the feature for character n-grams seems to damage the performance on the Twitter2014-test data set slightly, although on the Twitter2013-test data set, it makes a positive contribution. This may suggest that the feature is sensitive to certain details that appeared after the development of the Twitter2013-train data set, but it is more likely that the decrease in performance is merely caused by noise in the data.

The majority of the count features do not impose considerable changes in performance individually. In fact, the all-caps count feature actually decreases the classification performance on both test data sets, most likely only introducing noise. An interesting observation on the count features is that the Tweet NLP cluster count feature has a very large impact on classification performance, as anticipated. Tweets contain many misspellings

Features	Twitter2014-test	Twitter2013-test
All	0.684	0.688
All \ word n-grams	0.672	0.674
All \ character n-grams	0.688	0.676
All \ both n-grams	0.664	0.667
All \ sentiment lexica	0.665	0.657
All \ Tweet NLP cluster counts	0.666	0.677
All \ part-of-speech tag counts	0.684	0.685
All \ all caps counts	0.685	0.689
All \ elongated word counts	0.682	0.687
All \ emoticon counts	0.681	0.688
All \ punctuation counts	0.682	0.688
All \ hashtag counts	0.684	0.688
All \ negation counts	0.684	0.688
All \ all counts features	0.665	0.671

Table 6.10.: Sentiment classifier ablation study. $B \setminus A$ denotes all items in B except all items in A, and all scores are in F_1.

and unusual abbreviations and expressions, and the purpose of this feature is to make generalizations by counting the occurrences of clusters that include similar words.

6.4.4. Effect of Negation Scope Detection

In Table 6.12, we present an experiment on the effects of performing NSD on several variations of the sentiment classification system and Twitter sentiment data sets.

The first six rows contain results from evaluation using the full Twitter-2013-train and Twitter2014-test data sets, whereas the remaining rows contain results from evaluation using only a subset of both data sets: tweets that contain negation, as predetermined by our NSD system. When evaluating the system on these subsets, the differences between the NSD

methods used by the system are more prominent.

A higher ratio of sentiment-bearing tweets (with `positive` or `negative` labels) was expected to appear in these filtered data sets, because negators often function as valence shifters, as discussed in Section 2.1.2, and the presence of valence shifters indicates the presence of valence. Table 6.11 shows that this is indeed the case.

The rows of Table 6.12 are grouped into four segments, where each segment shows scores for a classifier using no, naïve and sophisticated NSD. The segments represent different combinations of features and data sets tested. We evaluate the system using all features, as well as using only *context-sensitive* features; features that are affected by linguistic negation: word n-grams, character n-grams, sentiment lexica and negation counts. These are described in more detail in Section 3.4.4. Additionally, we evaluate the system on the entire train and test sets as well as only the subsets containing negation.

Using all features, our sophisticated solution scores marginally better than the naïve one by an F_1 measure of 0.009. In every case, taking negation into account using either the naïve or the sophisticated method improves the F_1 score considerably. The sophisticated method improves more clearly upon the naïve method on the negated part of the data, with F_1 improvements ranging between 0.03 and 0.04.

Data Set	`positive`	`neutral`	`negative`	**Total**
Twitter2013-train	467	335	446	1248
Twitter2014-test	103	49	65	217

Table 6.11.: Sentiment data sets with only tweets containing negation

Tweets	**Features**	**Negation**	**Precision**	**Recall**	**F_1**
All	All	No	0.730	0.659	0.653
All	All	Naïve	0.738	0.676	0.675
All	All	Sophisticated	0.738	0.684	0.684
All	CS	No	0.705	0.618	0.601
All	CS	Naïve	0.728	0.663	0.662
All	CS	Sophisticated	0.729	0.667	0.665
Neg	All	No	0.598	0.599	0.585
Neg	All	Naïve	0.653	0.654	0.644
Neg	All	Sophisticated	0.675	0.682	0.673
Neg	CS	No	0.609	0.604	0.586
Neg	CS	Naïve	0.648	0.654	0.633
Neg	CS	Sophisticated	0.681	0.696	0.672

Table 6.12.: Comparison of sentiment classification results using different methods, data and feature sets. CS denotes *context-sensitive* features, and Neg denotes *only tweets containing negation*.

7. Discussion

This chapter contains an evaluation of the project and the conclusions we draw from the results, in addition to suggested future work.

7.1. Evaluation

At the beginning of this project, we formulated four goals, described in Section 1.3. They form a natural chain of dependency, as each goal builds on all the previous. In this section we evaluate the degree to which each goal has been accomplished.

G1: Research the State-of-the-Art in Twitter Sentiment Analysis

The area of Twitter sentiment analysis (TSA) was explored, with special attention to the several shared tasks that have been hosted for this field. The related works discovered formed the basis for most of the work performed throughout the project.

G2: Create a Twitter Corpus Annotated for Negation

A Twitter corpus annotated for the presence of linguistic negation, the NTNU Twitter Negation Corpus, was successfully created, although the annotation process was not performed exactly as planned. As discussed in Chapter 4, the original plan was to benefit from the power of crowdsourcing in order to produce a large corpus, as well as achieving a high level of redundancy while performing the annotation independently of the thesis authors. However, for reasons explained in Chapter 4, the annotation work was ultimately performed by the authors. As a consequence, the resulting corpus has an annotation redundancy of 2 and the moderate number of 4000 samples.

G3: Develop a Negation Classifier

A negation scope detection (NSD) system was developed consisting of two parts: a negation cue detector and a negation scope classifier. The negation cue detector uses a relatively simple lexicon lookup that yields a high recall, but a precision that could benefit from a more sophisticated detection method. This is discussed further in Section 7.3. Despite that the cue detector has potential for improvement, the negation scope classifier performs very well.

G4: Develop a Twitter Sentiment Classifier

A sentiment classifier for Twitter data was developed, incorporating several features that benefit from the NSD system. The results confirm that taking negation into account in general improves the sentiment classification performance significantly, and that using a sophisticated NSD system slightly improves the performance further.

7.2. Conclusions

When testing the cue detection mechanism of the Conditional Random Fields (CRF)-based NSD system that we developed, we observed that cue identification on Twitter's conversational language presented challenges not present in other domains we studied. As discussed in Section 6.3.2, we observed that the variation in cues used was quite low, but that a selection of tokens occurring as a cue had significant ambiguity as to what part-of-speech they occurred as, and thus whether they functioned as a cue. This was mainly the case for the term *no*, exemplified by tweets such as "no dude, i'm stuck in the house..." and "no!!! cant believe I lost to justin again!". This use of *no* as an exclamation proved to be quite frequent in the annotated corpus created. Other problematic terms include *nothing* and *not*, both occurring several times in idioms that neither contain nor constitute negations.

This resulted in a modest cue detection precision of 0.873. Conversely, Morante and Daelemans [2009] achieve a cue detection precision of 100 % on all sub-corpora of the BioScope corpus. Nevertheless, our system

achieved a high cue detection recall score of 0.976, resulting in an overall NSD F_1 score of 0.922.

The complete NSD system produces better results than any we have observed in other domains: an F_1 score of 0.853 and a PCS score of 0.645. This suggests that the CRF machine learner was able to identify the trend of certain dictionary cues being misclassified.

The characteristics of the negation-annotated corpus created in this project are discussed in Section 4.5. Several statistics for this corpus, such as the average number of tokens per sentence and the average number of tokens per negation scope, indicate that the corpus contains relatively simple language. This, along with the fact that our developed NSD system produces better results than we have seen in other domains, leads us to believe that performing NSD in the Twitter domain may be less challenging than in other domains it has been applied to. The high score of the naïve NSD solution on tweets strengthens this belief.

A comparison of the effects of the naïve and sophisticated NSD solutions on the TSA classifier treating only tweets containing negation was performed. The negation classifier was first run on the Twitter2013-train and the Twitter2014-test data sets, and the tweets classified to contain negation were held out. The performance of both solutions was then recorded, trained on the Twitter2013-train and tested on the Twitter2014-test held out data sets. The sophisticated NSD-system achieved an F_1 score of 0.673, a performance increase of 0.029 in comparison to the naïve system, which achieved an F_1 score of 0.644.

Although this result shows that incorporating sophisticated NSD is effective when classifying tweets that contain linguistic negation, the low frequency of these tweets — roughly 13 % — causes this performance increase to be slightly overshadowed by other features when classifying random samples. Through experiments conducted on the complete TSA classifier, we observed that including a NSD-handling mechanism in general had significant positive impact compared to the performance when including none, and that the performance difference between the naïve and the sophisticated variants was noticeable, but not very significant: an increase in F_1 measure from 0.675 to 0.684. The limited performance difference can also be attributed to the strength of the naïve NSD system — achieving an F_1 score of 0.733 — relative to the sophisticated system,

which achieves an F_1 measure of 0.853.

7.3. Future Work

As discussed in the previous section, a number of potential negation cues proved problematic for cue detection in the developed NSD classifier. This problem was limited to a select few; an overview of the number of times a term that occurred as a cue also occurred as a non-cue is shown in Table 4.4, on page 52. As we used a pure pattern-matching approach, this table also illustrates the occurrences of misclassified cues. A more sophisticated cue detection mechanism should be able to counteract this problem, perhaps incorporating part-of-speech tags as a feature. As the problem is limited to a subset of cues, developing a cue detector with a special mechanism for these cases may be worth considering.

A Twitter corpus annotated for both sentiment and negation would be a valuable resource to measure the effects of linguistic negation in TSA. This would allow for evaluating the performance of a sentiment classifier, and the impact of different features, with *gold standard* negation scope detection, thus displaying the maximum possible performance gain with perfect negation handling. This could, for instance, be done by applying the negation annotation system developed in this project on a SemEval Twitter sentiment data set.

An important thing to note is that this study is based on the current state-of-the-art features used in TSA, as opposed to taking a *bottom-up* approach. The impact of a more sophisticated NSD system on this feature set has been studied, but it may be that other features could be hand-crafted to better take advantage of a well-performing NSD system. An example of this is the *context-sensitive* prior polarity lexica used in this project, lexica that contain two entries for each term: the term's prior polarity score in both an affirmative and a negated context. These are created using a naïve NSD solution, and could possibly be made more accurate by employing a more sophisticated NSD solution when they are created.

Bibliography

Agarwal, A., Xie, B., Vovsha, I., Rambow, O. & Passonneau, R. (2011). Sentiment analysis of Twitter data. In *Proceedings of the workshop on languages in social media* (pp. 30–38). LSM '11. Portland, Oregon: Association for Computational Linguistics.

Baccianella, S., Esuli, A. & Sebastiani, F. (2010). SentiWordNet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. In *Proceedings of the seventh international conference on language resources and evaluation (LREC'10)* (pp. 2200–2204). Valletta, Malta.

Beevolve. (2012, October 10). An exhaustive study of Twitter users across the world. Retrieved December 18, 2014, from `http://www.beevolve.com/twitter-statistics/#c1`

Bermingham, A. & Smeaton, A. F. (2010). Classifying sentiment in microblogs: is brevity an advantage? In *Proceedings of the 19th ACM international conference on information and knowledge management* (pp. 1833–1836). ACM. Toronto, Canada.

Bouma, G. (2009). Normalized (pointwise) mutual information in collocation extraction. In *Proceedings of the biennial GSCL conference* (pp. 31–40). Potsdam, Germany.

Cerezo-Costas, H. & Celix-Salgado, D. (2015, June). Gradiant-analytics: training polarity shifters with CRFs for message level polarity detection. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval-2015)* (pp. 539–544). Denver, Colorado: Association for Computational Linguistics.

Chang, C.-C. & Lin, C.-J. (2011). LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, *2*, 27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Chapman, W. W., Bridewell, W., Hanbury, P., Cooper, G. F. & Buchanan, B. G. (2001). A simple algorithm for identifying negated findings and diseases in discharge summaries. *Journal of biomedical informatics*, *34*(5), 301–310.

Collins, M. (2003). Head-driven statistical models for natural language parsing. *Computational Linguistics*, *29*(4), 589–637.

Cortes, C. & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*(3), 273–297.

Councill, I. G., McDonald, R. & Velikovich, L. (2010). What's great and what's not: learning to classify the scope of negation for improved sentiment analysis. In *Proceedings of the workshop on negation and speculation in natural language processing* (pp. 51–59). NeSp-NLP '10. Uppsala, Sweden: Association for Computational Linguistics.

Ding, X., Liu, B. & Yu, P. S. (2008). A holistic lexicon-based approach to opinion mining. In *Proceedings of the 2008 international conference on web search and data mining* (pp. 231–240). WSDM '08. Palo Alto, California, USA: ACM.

Farkas, R., Vincze, V., Móra, G., Csirik, J. & Szarvas, G. (2010). The CoNLL-2010 shared task: learning to detect hedges and their scope in natural language text. In *Proceedings of the fourteenth conference on computational natural language learning—shared task* (pp. 1–12). Association for Computational Linguistics. Uppsala, Sweden.

Fletcher, T. (2009). Support vector machines explained, tutorial paper. `http://www.tristanfletcher.co.uk/SVM%20Explained.pdf`. Retrieved June 11, 2015.

Gimpel, K., Schneider, N., O'Connor, B., Das, D., Mills, D., Eisenstein, J., Heilman, M., Yogatama, D., Flanigan, J. & Smith, N. A. (2011). Part-of-speech tagging for Twitter: annotation, features, and experiments. In *Proceedings of the 49th annual meeting of the Association for Computational Linguistics: human language technologies: short papers* (Vol. 2, pp. 42–47). Association for Computational Linguistics. Portland, Oregon.

Givón, T. (1993). *English grammar: a function-based introduction*. John Benjamins Publishing.

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

Günther, T., Vancoppenolle, J. & Johansson, R. (2014, August). RTRGO: enhancing the GU-MLT-LT system for sentiment analysis of short messages. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval-2014)* (pp. 497–502). Dublin, Ireland: Association for Computational Linguistics and Dublin City University.

Hsu, C.-W. & Lin, C.-J. (2002). A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, *13*(2), 415–425.

Hu, Y., Talamadupula, K., Kambhampati, S. et al. (2013). Dude, srsly?: the surprisingly formal nature of Twitter's language. In *Proceedings of the 7th international AAAI conference on weblogs and social media (ICWSM)*. Boston, USA.

Hudson, R. (2010). An encyclopedia of word grammar and English grammar. Retrieved December 16, 2014, from `http://tinyurl.com/wg-encyc`

Kiritchenko, S., Zhu, X. & Mohammad, S. M. (2014). Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research*, *50*, 723–762.

Kofod-Petersen, A. (2012). How to do a Structured Literature Review in computer science. Document released as a guide to performing a Structured Literature Review at NTNU. Retrieved December 19, 2014, from `https://research.idi.ntnu.no/aimasters/files/SLR_HowTo.pdf`

Kong, L., Schneider, N., Swayamdipta, S., Bhatia, A., Dyer, C. & Smith, N. (2014). A dependency parser for tweets. In *Proceedings of the conference on empirical methods in natural language processing* (pp. 1001–1012). EMNLP '14. Doha, Qatar.

Konstantinova, N., de Sousa, S. C., Díaz, N. P. C., López, M. J. M., Taboada, M. & Mitkov, R. (2012). A review corpus annotated for negation, speculation and their scope. In *Proceedings of the eighth international conference on language resources and evaluation (LREC'12)* (pp. 3190–3195). Istanbul, Turkey.

Kouloumpis, E., Wilson, T. & Moore, J. (2011). Twitter sentiment analysis: the good the bad and the OMG! In *Proceedings of the fifth international AAAI conference on weblogs and social media* (Vol. 11, pp. 538–541). Barcelona, Spain.

Lafferty, J. D., McCallum, A. & Pereira, F. C. N. (2001). Conditional Random Fields: probabilistic models for segmenting and labeling sequence data. In *Proceedings of the eighteenth international conference on machine learning* (pp. 282–289). ICML '01. Williamstown, MA, USA: Morgan Kaufmann Publishers Inc.

Leal, J., Pinto, S., Bento, A., Gonçalo Oliveira, H. & Gomes, P. (2014, August). CISUC-KIS: tackling message polarity classification with a large and diverse set of features. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval-2014)* (pp. 166–170). Dublin, Ireland: Association for Computational Linguistics and Dublin City University.

Liu, B. & Zhang, L. (2012). A survey of opinion mining and sentiment analysis. In C. C. Aggarwal & C. Zhai (Eds.), *Mining text data* (pp. 415–463). Springer US.

Manning, C. D., Raghavan, P. & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge, UK: Cambridge University Press.

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

McKinney, W. (2012). Pandas: a Python data analysis library. Retrieved May 13, 2015, from `http://pandas.pydata.org`

Mitchell, T. M. (1997). *Machine learning* (1st ed.). New York, NY, USA: McGraw-Hill, Inc.

Miura, Y., Sakaki, S., Hattori, K. & Ohkuma, T. (2014, August). TeamX: a sentiment analyzer with enhanced lexicon mapping and weighting scheme for unbalanced data. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval-2014)* (pp. 628–632). Dublin, Ireland: Association for Computational Linguistics and Dublin City University.

Mohammad, S. M. (2012, July). #Emotional tweets. In **SEM 2012: the first joint conference on lexical and computational semantics – volume 1: proceedings of the main conference and the shared task, and volume 2: proceedings of the sixth international workshop on semantic evaluation (SemEval-2012)* (pp. 246–255). Montréal, Canada: Association for Computational Linguistics.

Mohammad, S. M., Kiritchenko, S. & Zhu, X. (2013, June). NRC-Canada: building the state-of-the-art in sentiment analysis of tweets. In *Proceedings of the seventh international workshop on semantic evaluation exercises (SemEval-2013)*. Atlanta, Georgia, USA.

Mohammad, S. M. & Turney, P. D. (2010). Emotions evoked by common words and phrases: using Mechanical Turk to create an emotion lexicon. In *Proceedings of the NAACL HLT 2010 workshop on computational approaches to analysis and generation of emotion in text* (pp. 26–34). Association for Computational Linguistics. Los Angeles, California.

Moilanen, K. & Pulman, S. (2007). Sentiment composition. In *Proceedings of the recent advances in natural language processing international conference* (pp. 378–382). RANLP '07. Borovets, Bulgaria.

Morante, R. & Blanco, E. (2012). * SEM 2012 shared task: resolving the scope and focus of negation. In *Proceedings of the first joint conference on lexical and computational semantics-volume 1: proceedings of the main conference and the shared task, and volume 2: proceedings of the sixth international workshop on semantic evaluation* (pp. 265–274). Association for Computational Linguistics. Montreal, Canada.

Morante, R. & Daelemans, W. (2009, June). A metalearning approach to processing the scope of negation. In *Proceedings of the thirteenth conference on computational natural language learning (CoNLL-2009)* (pp. 21–29). Boulder, Colorado: Association for Computational Linguistics.

Morante, R. & Sporleder, C. (2012). Modality and negation: an introduction to the special issue. *Computational Linguistics*, *38*(2), 223–260.

Mutalik, P. G., Deshpande, A. & Nadkarni, P. M. (2001). Use of general-purpose negation detection to augment concept indexing of medical documents a quantitative study using the umls. *Journal of the American Medical Informatics Association*, *8*(6), 598–609.

Nakov, P., Rosenthal, S., Kozareva, Z., Stoyanov, V., Ritter, A. & Wilson, T. (2013, June). Semeval-2013 task 2: sentiment analysis in twitter. In *Second joint conference on lexical and computational semantics (*SEM), volume 2: proceedings of the seventh international workshop on semantic evaluation (SemEval-2013)* (pp. 312–320). Atlanta, Georgia, USA: Association for Computational Linguistics.

Nielsen, F. Å. (2011). A new ANEW: evaluation of a word list for sentiment analysis in microblogs. In *Proceedings of the 1st workshop on making sense of microposts (#MSM2011)* (pp. 93–98). Kaunas, Lithuania.

Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S. & Marsi, E. (2007). MaltParser: a language-independent system for data-driven dependency parsing. *Natural Language Engineering*, *13*(02), 95–135.

Nocedal, J. (1980). Updating quasi-Newton matrices with limited storage. *Mathematics of computation*, *35*(151), 773–782.

Okazaki, N. (2007). CRFsuite: a fast implementation of conditional random fields (CRFs). Retrieved May 13, 2015, from `http://www.chokkan.org/software/crfsuite/`

Owoputi, O., O'Connor, B., Dyer, C., Gimpel, K., Schneider, N. & Smith, N. A. (2013). Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of the 2013 conference of the North American Chapter of the Association for Computational Linguistics: human language technologies* (pp. 380–390). Atlanta, GA, USA.

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

Pang, B., Lee, L. & Vaithyanathan, S. (2002). Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the conference on empirical methods in natural language processing - volume 10* (pp. 79–86). EMNLP '02. Philadelphia, PA, USA: Association for Computational Linguistics.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. & Duchesnay, E. (2011). Scikit-learn: machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Peng, T. & Korobov, M. (2014). Python-crfsuite. Retrieved May 1, 2015, from `https://github.com/tpeng/python-crfsuite`

Platt, J. (1998). Fast training of Support Vector Machines using sequential minimal optimization. In B. Schoelkopf, C. Burges & A. Smola (Eds.), *Advances in kernel methods - support vector learning*. MIT Press.

Plotnikova, N., Kohl, M., Volkert, K., Evert, S., Lerner, A., Dykes, N. & Ermer, H. (2015, June). KLUEless: polarity classification and association. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval-2015)* (pp. 619–625). Denver, Colorado: Association for Computational Linguistics.

Polanyi, L. & Zaenen, A. (2006). *Contextual valence shifters* (J. G. Shanahan, Y. Qu & J. Wiebe, Eds.). Springer.

Ratnaparkhi, A. (1999). Learning to parse natural language with maximum entropy models. *Machine Learning*, *34*(1-3), 151–175.

Rijsbergen, C. J. V. (1979). *Information retrieval* (2nd). Newton, MA, USA: Butterworth-Heinemann.

Rosenthal, S., Nakov, P., Kiritchenko, S., Mohammad, S., Ritter, A. & Stoyanov, V. (2015, June). SemEval-2015 task 10: sentiment analysis in Twitter. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval-2015)* (pp. 451–463). Denver, Colorado: Association for Computational Linguistics.

Rosenthal, S., Ritter, A., Nakov, P. & Stoyanov, V. (2014, August). SemEval-2014 task 9: sentiment analysis in Twitter. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval-2014)* (pp. 73–80). Dublin, Ireland: Association for Computational Linguistics and Dublin City University.

Selmer, Ø., Brevik, M., Gambäck, B. & Bungum, L. (2013, June). NTNU: domain semi-independent short message sentiment classification. In *Second joint conference on lexical and computational semantics (*SEM), volume 2: proceedings of the seventh international workshop on semantic evaluation (SemEval-2013)* (pp. 430–437). Atlanta, Georgia, USA: Association for Computational Linguistics.

Sutton, C. & McCallum, A. (2011). An introduction to conditional random fields. *Machine Learning*, *4*(4), 267–373.

Tang, D., Wei, F., Qin, B., Liu, T. & Zhou, M. (2014, August). Coooolll: a deep learning system for Twitter sentiment classification. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval-2014)* (pp. 208–212). Dublin, Ireland: Association for Computational Linguistics and Dublin City University.

Tottie, G. (1991). *Negation in English speech and writing: a study in variation*. Academic Press.

Tsakalidis, A., Papadopoulos, S., Cristea, A., Kompatsiaris, Y. et al. (2015). Predicting elections for multiple countries using Twitter and polls. *Intelligent Systems, IEEE*, *30*(2), 10–17.

Tsoumakas, G., Katakis, I. & Vlahavas, I. (2010). Mining multi-label data. In O. Maimon & L. Rokach (Eds.), *Data mining and knowledge discovery handbook* (pp. 667–685). Springer US.

Vincze, V., Szarvas, G., Farkas, R., Móra, G. & Csirik, J. (2008). The BioScope corpus: biomedical texts annotated for uncertainty, negation and their scopes. *BMC bioinformatics*, *9*(Suppl 11), S9.

Vryniotis, V. (2013, November 20). Machine learning tutorial: the max entropy text classifier. Retrieved December 16, 2014, from `http://blog.datumbox.com/machine-learning-tutorial-the-max-entropy-text-classifier/`

Wiegand, M., Balahur, A., Roth, B., Klakow, D. & Montoyo, A. (2010). A survey on the role of negation in sentiment analysis. In *Proceedings of the workshop on negation and speculation in natural language processing* (pp. 60–68). Association for Computational Linguistics.

Wilson, T., Hoffmann, P., Somasundaran, S., Kessler, J., Wiebe, J., Choi, Y., Cardie, C., Riloff, E. & Patwardhan, S. (2005). OpinionFinder: a system for subjectivity analysis. In *Proceedings of HLT/EMNLP 2005 Interactive Demonstrations* (pp. 34–35). Association for Computational Linguistics. Vancouver, Canada.

Xue, N. & Palmer, M. (2004). Calibrating features for semantic role labeling. In *Proceedings of the 2004 conference on empirical methods in natural language processing* (pp. 88–94). EMNLP '04. Barcelona, Spain.

Zhang, Z., Wu, G. & Lan, M. (2015, June). ECNU: multi-level sentiment analysis on Twitter using traditional linguistic features and word embedding features. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval-2015)* (pp. 561–567). Denver, Colorado: Association for Computational Linguistics.

Zhu, Q., Li, J., Wang, H. & Zhou, G. (2010). A unified framework for scope learning via simplified shallow semantic parsing. In *Proceedings of the 2010 conference on empirical methods in natural language processing* (pp. 714–724). EMNLP '10. Cambridge, Massachusetts: Association for Computational Linguistics.

A. Annotation Guidelines

This document contains instructions for annotating the presence of linguistic negation in tweets.

A.1. Task Description

The task is to identify linguistic negation in tweets. Linguistic negation consists of a negation cue, a word that creates the negated context; and a negated scope, the words inside the negated context. An example sentence is "The weather is not very nice today". In this sentence, *not* is the negation cue, and *very nice* is the negated scope. See Appendix A.3 for example annotated sentences.

An important point to note is that we are not interested in identifying cases where an individual word is negated by a prefix (*morphological* negation), such as *unfortunate* or *abnormal*.

The task is two-fold:

1. Identify negation cues. This is to be done based on a list of words that we have created, shown in Table A.1.

2. Identify the negated scope, meaning the affected words, for each negation cue.

Guidelines for performing these steps are described in Section A.2.

hardly	lack	lacking	lacks	neither
nor	never	no	nobody	none
nothing	nowhere	not	without	aint
cant	cannot	darent	dont	doesnt
didnt	hadnt	hasnt	havent	havnt
isnt	mightnt	mustnt	neednt	oughtnt
shant	shouldnt	wasnt	wouldnt	*n't

Table A.1.: Lexicon of negation cues. **n't* denotes all words with the suffix *n't*.

A.2. Guidelines

A.2.1. General Principles

1. Negation cues (e.g., the words *never*, *no*, or *not* in its various forms) should not be included in the negated scope. For example, in the sentence, "It was not good", *good* is annotated as the negated scope.

2. Annotate the minimum scope of negation, including only the portion of the text being negated semantically. When in doubt, prefer simplicity.

3. Many tweets do not contain any form of linguistic negation, and no annotation is to be done in these cases.

A.2.2. Slang and Misspellings

As the data to be annotated comes from Twitter, the language used is likely to contain many cases of misspelled words, slang and unconventional linguistic means. If a slang, misspelled or otherwise out-of-dictionary word occurs, try to interpret the word to the best of your ability taking into account the context. If the word is still incomprehensible, it is to be ignored. This is especially important for negation cues, e.g., if the term *cnt* appears in a context where it is obvious that the author means *can't*, it is to be annotated as a negation cue. For example the sentence "i cnt believe its not butter".

A.2.3. Adjectives/Adverbs

When considering adjectives in noun phrases, do not annotate the entire noun phrase if only the adjective is being negated. For instance, "Not top-drawer cinema, but still good...": *top drawer* is negated, but *cinema* is not, since it is still *cinema*, just not *top-drawer*.

The same is true for adverbs in verb phrases. If only the adverb is directly negated, only annotate the adverb itself. E.g., "Not only was it great", or "Not quite as great": in both cases the subject still *is great*, so just *only* and *quite* should be annotated, respectively. However, there are cases where the intended scope of adverbial negation is greater, e.g., the adverb phrase *just a small part* in "Tony was on stage for the entire play. It was not just a small part".

A.2.4. Noun Phrases

Typically entire noun phrases are annotated as within the scope of negation if a noun within the phrase is negated. For example, in the sentence, "This was not a review" the string *a review* is annotated. This is also true for more complex noun phrases, e.g., "This was not a review of a movie that I watched" should be annotated with the scope *a review of a movie that I watched*.

A.2.5. Verb Phrases

If a verb is directly negated, annotate the entire verb phrase as negated, e.g., *appear to be red* would be marked in "It did not appear to be red".

A.3. Examples

This section contains examples of tweets annotated to show linguistic negation. Negation cues are shown in **bold font** and negated scopes are underlined.

1. They may have a SuperBowl in Dallas, but Dallas **ain't** winning a SuperBowl. **Not** with that quarterback and owner. @S4NYC @RasmussenPoll

2. victor cruz **aint** do the salsa ALL game...he's overdue....its the 4th quarter...its TIME

3. Bad times when u missed Waterloo road last thursday when in Tenerife and **dont** realise it was the last episode :(#notfair

4. **Should'nt** have watched The Grey last night because walking down a dark driveway when the sun decides to play hookie is **not** a good time.

5. So far I have drafted Aaron Rodgers at #9 overall, and then at #16 in the 2nd round, Matt Forte. **Not** bad.

6. @Darien_Smalling the Pro Bowl definitely **isn't** better and NBA All-Star game is trash until 4th qtr

7. Dishonoured is awesome! **Not** quite sure where to rank it. Journey and Walking dead are one and two. I think it's 4th behind Mass Effect 3.

8. @Raiders_Spurs Reggie **isn't** interested in dealing picks like the old regime. **Can't** see him parting with anything other than a 6th or 7th