Sentiment-Analyse von Twitter-Daten mit Hilfe von distributioneller Semantik

Bachelor-Thesis von Qi Shao aus Darmstadt
Tag der Einreichung:

1. Gutachten: Prof. Dr. Chris Biemann
2. Gutachten: Eugen Ruppert

TECHNISCHE UNIVERSITÄT DARMSTADT

Fachbereich Informatik

Sentiment-Analyse von Twitter-Daten mit Hilfe von distributioneller Semantik

Vorgelegte Bachelor-Thesis von Qi Shao aus Darmstadt

1. Gutachten: Prof. Dr. Chris Biemann
2. Gutachten: Eugen Ruppert

Tag der Einreichung:

Erklärung zur Bachelor-Thesis

Hiermit versichere ich, die vorliegende Bachelor-Thesis ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 6. April 2015

(Q. Shao)

Abstract

This bachelor thesis describes an experiment using lexical expansion (JoBimText) and weighting factors to determine the sentiment of tweets. The aim of the experiment is to investigate whether with multi-expansion the classification result compared to results of previous papers can be improved. The evaluation shows that with expansion despite higher noise a better F_1 value can be achieved. The best result was obtained with the hybrid method "konstant", which considers the number and score of returned words of the expansion.

Abstract

Diese Bachelor-Thesis beschreibt ein Experiment, um mit Hilfe von Lexical Expansion (JoBimText) und Gewichtungsfaktoren, das Sentiment eines Tweets zu bestimmen. Das Ziel des Experiments ist, zu untersuchen, ob mit Multiexpansion das Klassifikationsergebnis gegenüber früheren Arbeiten verbessert werden kann. Die Evaluation zeigt, dass mit Expansion trotz höheren Rauschens ein besserer F_1 Wert als ohne erreicht wird. Das beste Ergebnis wurde mit der Hybrid-Methode "konstant", die Anzahl und Score der Expanson berücksichtigt, erzielt.

Contents

1 Einleitung

Nach der Erfindung des ersten Computers (ABC 1939 [23]; ENIAC 1946 [26]) und der schnellen Verbreitung des Internets, gibt es soziale Beziehungen zwischen Menschen nicht nur in der realen, sondern auch in der virtuellen Welt. Die Leute sprechen miteinander durch Chatsoftware, sagen ihre Meinung und veröffentlichen ihre Stimmung auf einem Mikroblog. Hinter den sozialen Netzwerken stehen Firmen wie Facebook oder Twitter, die in kurzer Zeit einen hohen Wert erlangt haben. Z.B. Twitter ist seit dem 7. November 2013 an der New York Stock Exchange gelistet und hat an diesem Tag über 2 Milliarden Dollar Marktkapitalisierung erreicht.

Twitter ist einer der zur Zeit beliebtesten Mikroblogs. Die Anzahl seiner Benutzer ist riesig. Es gibt mehr als 40 Millionen monatlich "aktive Benutzer", die im Twitter System eingeloggt sind[1]. Dabei fällt eine riesige Menge von Textinformationen an, 500 Millionen Tweets werden jeden Tag geschrieben[2]. Twitter ist eine Plattform die eine große Menge von Benutzer- und Textinformationenen sammelt. Die Informationen sind ähnlich wie Kommentare in Foren, Blog Nachrichten oder Produktbeschreibungen von Medien und enthalten einen Teil einer Beschreibung, Ansicht oder eines Gefühlsausdrucks. So ein "Short-Text" (nicht mehr als 140 Zeichen pro Text erlaubt) [27] umfasst einige typische Merkmale [5] [8] [13] [20]:

- Jeder Text ist kurz, enthält nur wenige Informationen aber es gibt eine große Menge davon.
- Die Texte haben Eigenschaften von Echtzeit-Updates und dynamischen Veränderungen.
- Die Texte sind nicht normativ, d.h. sie enthalten viele Grammatikfehler, Umgangssprache, nicht formale Abkürzungen, Emoticons, usw.
- Die Texte haben halbstrukturierte Merkmale, und enthalten häufig den Zeitpunkt der Veröffentlichung, Bewertungen und Anzahl der Weiterleitungen.

Die besonderen Merkmale, die schnell wachsende Entwicklung und die große Anzahl und Komplexität von sozialen Beziehungen auf Mikroblogs bzw. Twitter erhöhen die Schwierigkeiten bei der Analyse von Text, bietet aber auch neue Chancen.

1.1 Problemstellung und Zielsetzung

Im Bereich Sentiment Analysis für Mikroblogging wurde schon viel geforscht. Eine der verwendeten Methoden ist die "Lexical-Based" Methode. Die Methode verwendet zum klassifizieren von Datensätzen eine Liste von Worten, wobei jedem Wort der Wert positiv oder negativ zugeordnet ist. Weil die Größe eines Lexikons beschränkt ist, gibt es viele Wörter die nicht im Lexikon enthalten sind, aber trotzdem einen Einfluss auf den Sentiment eines Satzes haben. Weil ein Wort entweder im Lexikon enthalten ist, oder nicht, gibt es keine Abstufungen von positiven und negativen Bedeutungen. Deswegen ist es schwierig, eine leicht positive Meinung als positiv zu klassifizieren. Daher wäre es Hilfreich, wenn man ein erweitertes Lexikon hätte, um eine abgestufte Bedeutung für ein Wort nachschlagen zu können.

[1] http://www.ilias.de/docu/goto_docu_pg_42176_2222.html
[2] https://about.twitter.com/company

Im Rahmen dieser Arbeit wurden Tweets mit Hilfe von JoBimText (eine distributive Semantik) klassifiziert. Ein Klassifkationsverfahren gilt dabei als ”gut”, wenn es Tweets so in die Klassen positiv, negativ und neutral einteilt, wie es ein Mensch auch machen würde. Ein Ziel dieser Arbeit ist es, gegenüber der ”Lexical-Based” Methode mit Expansion (mit Hilfe von JoBimText) eine Verbesserung zu erreichen. Des Weiteren wurden Tests mit Hilfe des Tools ”Weka” durchgeführt.

1.2 Aufbau der Arbeit

- Klassifizieren mit Methoden:
  Es wurde Trainingsdaten benutzt, um einige Parameter und Grenzwertpunkte festzulegen. Die URL (z.B. http://example.de) und der Twitter Benutzername (z.B. @example, alle Worte die mit dem @ Zeichen anfangen) werden zuerst gefiltert. Dann wurde jeder Tweet in eine Liste von Worten zerlegt. Die aus der Zerlegung erhaltenen Worte wurden mit Hilfe von Filterdaten (stopwords) gefiltert. Dann wurde für jedes dieser Worte getestet, ob es im Lexikon enthalten ist. Die Worte, die nicht im Lexikon gefunden wurden, wurden weiter mit Expansion (Hilfe von JoBimText) getestet. In den letzten Tests wurden einige Parameter benutzt, die mit Hilfe von Trainingsdaten festgelegt worden sind. Am Ende wurden die Ergebnisse zusammengefasst und der Tweet wurde abhängig von Grenzwertpunkten klassifiziert.

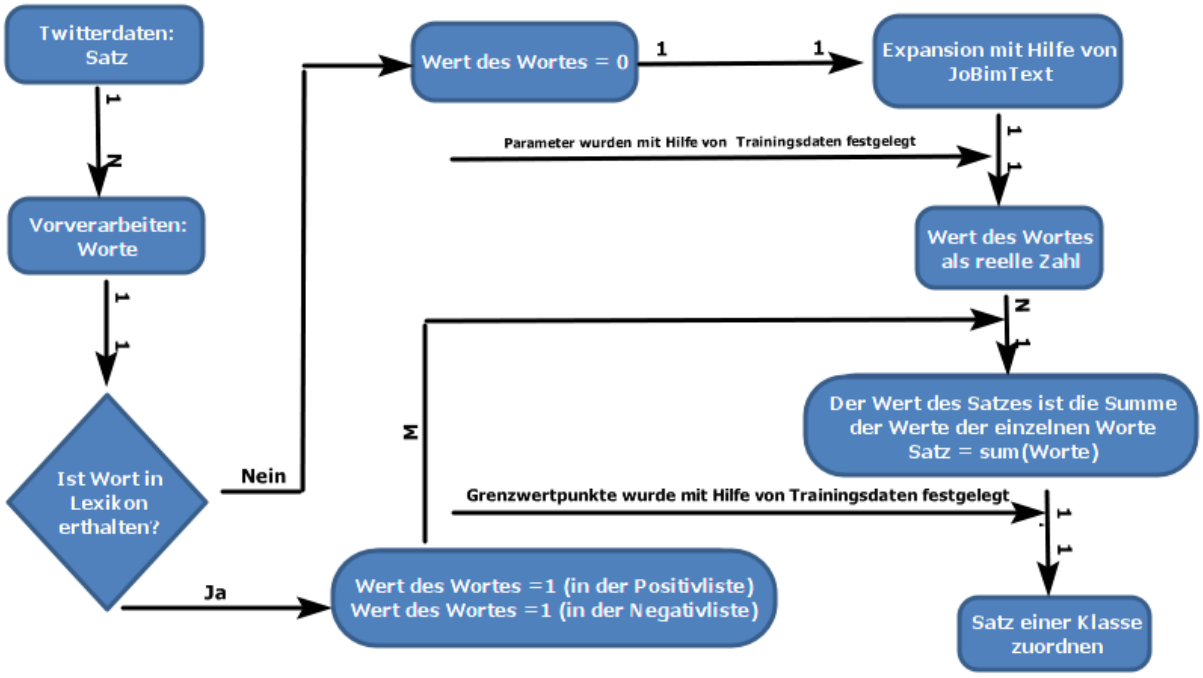

Figure 1: Klassifizieren mit Methoden

Abbildung 1 zeigt den Prozess zur Klassifikation eines Tweets.

- Klassfizieren mit Weka:
  Es wurde einige Features für jeden Tweet erzeugt, wie z.B. die Anzahl von positiven Worten (Worte, die der Positivliste enthalten sind) oder negativen Worte, usw.. Alle Worte der Twitterdaten und alle Features wurden als Attribut in Weka angewendet. Testdaten und Trainingsdaten sind separat. Die Algorithmen SVM (Support Vector Machine) und N-B (Naïve Bayesian method) wurden mit den Trainingsdaten trainiert und dann mit den Testdaten angewendet.

2 Thema und Hintergrund

Dieses Kapitel enthält eine kurze Einführung in des Thema: Sentiment Analysis, den Hintergrund: Distributionelle Semantik (DS) und Machine Learning, und die Anwendungstechniken: JoBimText und Weka.

2.1 Sentiment Analysis

Sentiment Analysis hat die Aufgabe positive und negative Meinungen und Gefühle zu identifizieren oder auszuwerten [33]. Heutzutage wird nicht nur mit Texten geforscht sondern auch mit Bildern. Auch mit Google Glass kann man Sentiment-Analyse betreiben. Mit einer speziellen App können die Emotionen von Menschen ausgewertet werden [3]. Der weitaus größte Teil der Forschung findet allerdings ist immer noch an Texten statt.

Sentiment Analysis einer großen Menge von Mikroblog Daten kann zeigen, welche Dinge bei den Leuten im Moment beliebt sind, welche Marken sie zur Zeit mögen und welche nicht. Diese Daten sind wichtig für Unternehmen und können dazu benutzt werden um Marketingstrategien anzupassen oder um Produkte zu verbessern. Es kann auch untersucht werden, ob eine bestimmte Person ein hohes Ansehen hat, oder nicht. Das könnte für die Politik nützlich sein. Die Anwendung von Sentiment Analysis ist auch für Psychologie, Finanzprognosen, Soziologie und andere Forschungsbereiche geeignet.

Textliche Sentiment Analyse ist aber nicht so einfach, denn oft haben ähnliche Sätze eine gegenteilige Bedeutung, wie z.B.:

Diese Jacke passt dir.
Passt diese Jacke dir?

Der erste Satz ist deutlich positiv, aber der Zweite dürfte negativ sein, obwohl die ”wichtigen” Worte in beiden Sätzen gleich sind:

Jacke, passt, dir

Ein weiteres Problem ist, dass die positiven Worte manche mal auch eine negative Bedeutung haben.

So ein tolles Wetter, es regnet wieder!

Das Wort *tolles* darf im diesem Satz nicht als positiv bewertet werden. Ein weiteres Problem ist, dass sich Humor nur schwer analysieren lässt.

2.2 Distributionelle Semantik (DS) und JoBimText

Distributionelle Semantik ist eine Struktur, die für die automatisierte Berechnung der Wortbedeutung verwendet wird. Die Bedeutung eines Wortes wird als Vektor repräsentiert. Vektoren stellen die statistische Verteilung der sprachlichen Kontexte zusammen (siehe Beispiel). Die Ähnlichkeit von zwei Worten ist der Abstand zwischen den beiden Vektoren, die die Worte repräsentieren.

[3] http://www.wallstreet-online.de/nachricht/6627493-sentiment-analysis-app-bestimmt-gefuehlslage

Im folgenden Beispiel wird eine ganz einfache Distributionelle Semantik gezeigt. Die Daten sind:

- *Ich esse Fisch.*
- *Ich esse Obst.*
- *Ich trage eine Brille.*
- *Ich trage eine Kette.*

Hinter *esse* kommt das Wort *Fisch* oder *Obst*, d.h. in 50% der Fälle steht das jeweilige Wort hinter *esse*. Hinter *esse* kommt weder *Brille* noch *Kette*, d.h. die Worte kommen in 0% der Fälle hinter dem Wort *esse*. Daraus ergibt sich folgende Relationstabelle:

	Fisch	Obst	Brille	Kette
esse	50	50	0	0
trage	0	0	50	50

Table 1: Es wird die Wahrscheinlichkeit eines Wortes gezeigt, hinter *esse* oder *trage* zu stehen.

Daraus wurde ein Vektorraum gebildet:

Figure 2: Vektorraum

Abbildung (2) zeigt, dass der Abstand zwischen *Fisch* und *Obst* 0 ist. In diesem Fall wird angenommen, dass *Fisch* und *Obst* ähnlich sind. *Brille* und *Kette* sind ebenfalls ähnlich, aber *Fisch* und *Brille* nicht.

Aus diesem Beispiel wird ersichtlich, dass das Wichtigste und die Basis der DS, d.h. das Wesen der DS, Korpora und Statistik sind [15]. Korpora sind die Quelle der Informationen, anhand derer die Verteilung der Worte ermittelt wird. Statistik wird verwendet, um die Worte auf einen Vektor abzubilden.

Die Abbildung der Bedeutung von Worten im sprachlichen Kontext durch Geometrie hilft bei der automatischen Analyse folgender Fragestellungen: ob zwei Worte eine ähnliche Bedeutung haben; ob zwei Worte eine analoge Bedeutung haben; und damit auch allgemein beim klassifizieren von Texten [1]. Darüber hinaus wird DS noch im Bereich Information Retrieval oder bei der Wortsinn Begriffsklärung [31] angewendet.

2.2.1 JoBimText

JoBimText ist ein Open-Source-Framework für die Anwendung des DS, das lexikalisierte Features verwendet[4]. Es bietet eine automatisierte Text Expansion und wurde von TU Darmstadt und IBM Research entwickelt.

Es kann über ein Web Demo (online Benutzung) verwendet werden. Als Eingabe sind Worte oder ein Satz erlaubt. Mit Hilfe von verschiedenen Analysemodellen werden Beziehung zwischen den Worten der Eingabe als Graphik oder als Liste dargestellt. Nach dem Anklicken der gewünschten Worte wird ein Liste von Paaren (”Jo” und ”Score”) gezeigt. ”Jo” sind die Expansions Worte, die vom Modell berechnet wurden. ”Score” sind dazugehörigen Ähnlichkeitsmaße.

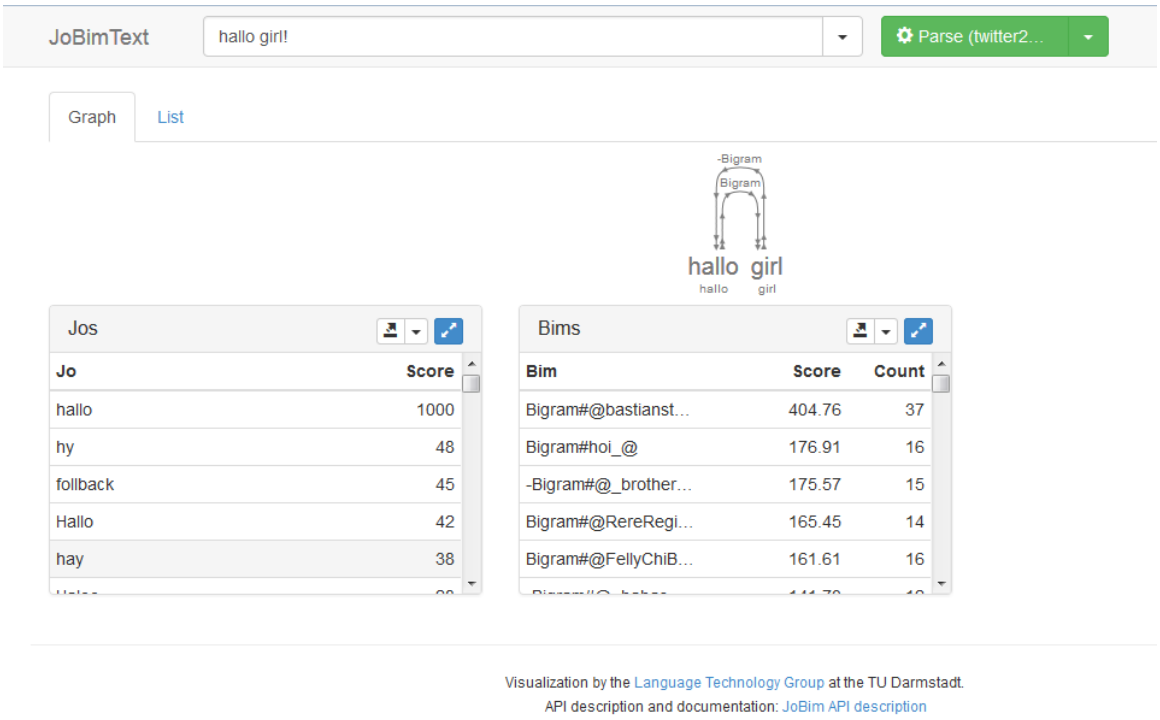

Figure 3: Web Demo Beispiel

Der theoretische Hintergrund von JoBimText wird in der Arbeit ”Text: now in 2D” [2] erklärt. Mit Hilfe einer ”Holing Operation” (sie erstellt zwei verschieden Mengen von Beobachtungen: Sprachelemente und Kontext Features) wird die Häufigkeit der Elemente von beiden Mengen erfasst. Dann wird ein relativer Wert für jedes Element berechnet. Nach Pruning und Aggregate erhält man den endgültigen Score von jedem Element. Ausgegeben werden die Elemente nach Score sortiert.

[4] http://maggie.lt.informatik.tu-darmstadt.de/jobimtext/

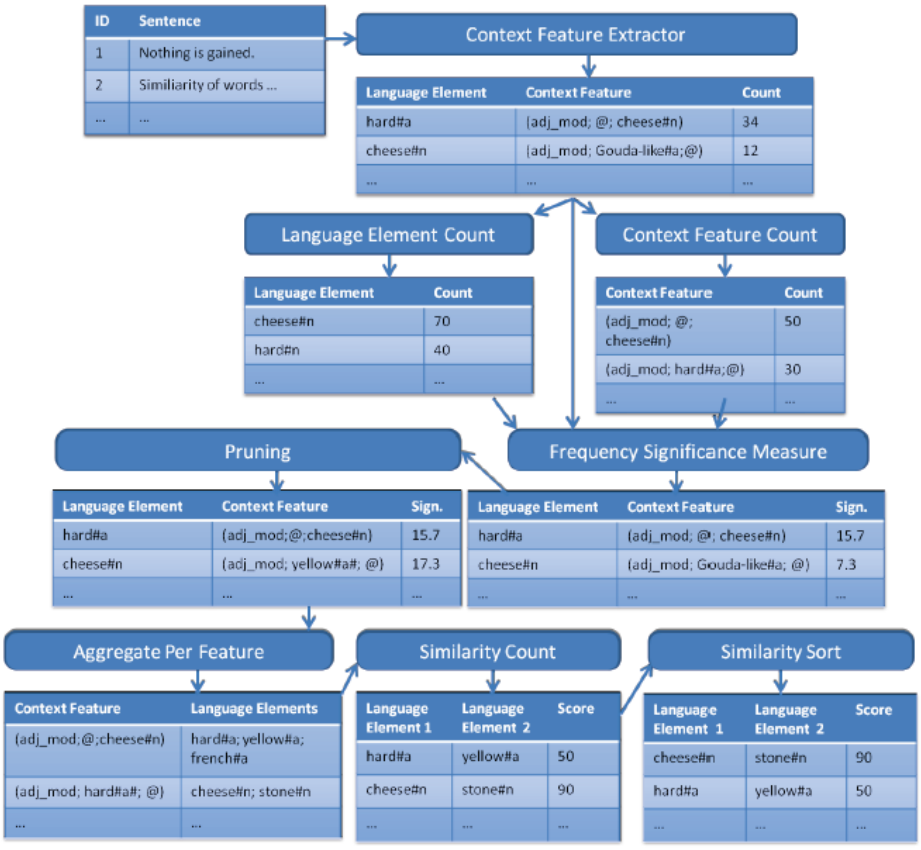

Figure 4: Arbeitsfluss Quelle: [2]

Das kontextabhängige Verfahren von JoBimText hat die Baseline gegenüber einem nicht kontextabhängigen verbessert [2]: In der Ähnlichkeitsanalyse von 1000 häufigen und 1000 seltenen Worten hat die von JoBimText verwendete Methode LMI 1.158% Verbesserung bei den besten fünf Worten gegenüber der Methode von Lin [17] erreicht. Bei der Untersuchung wurde ein Korpus der Größe 120M verwendet.

2.3 Machine Learning und Weka

Die Methoden des Machine Learning (ML), ermöglichen die automatisierte Analyse einer großen Menge an Daten. Dabei entscheidet ein Computerprogramm selbst, welche Informationen am relevantesten sind. Dieses zusammengefassten Informationen werden dann verwendet, um automatische Vorhersagen zu machen oder Menschen zu helfen, sich eine Meinung zu bilden.

2.3.1 Weka

Weka ist die Abkürzung von Waikato Environment for Knowledge Analysis. Die Hauptentwickler arbeiten an der University Waikato Neuseeland. Der Quellcode von Weka kann von der Seite [5] heruntergeladen werden.

[5] http://www.cs.waikato.ac.nz/ml/weka

WEKA als eine Data-Mining-Workbench enthält eine große Menge an Machine Learning Algorithmen, die Data-Mining-Aufgaben übernehmen können, wie z.B. Datenvorverarbeitung, Klassifikation, Clustering, usw. Weka ist heutzutage ein beliebtes und auch eines der umfassendsten Tools für Data Mining und Machine Learning. (18-jährige Geschichte der Entwicklung).

3 Ähnliche Arbeiten

Sentiments können durch Text ausgedrückt werden, entweder offensichtliche oder unauffällige. Die Textklassifikation ist im Moment ein sehr beliebtes Thema im Bereich des "Natural language processing" und die dazugehörige Sentiment-Analyse zieht eine große Aufmerksamkeit auf sich [22]. Laut der Arbeit von Saif et al. sind die Daten von Twitter eine Goldgrube, mit deren Hilfe Firmen ihre Reputation überwachen können [25]. Viele Wissenschaftler haben auch Sentiment von Tweets analysiert [13] [14] [20].

Momentan gibt es folgende Klassifizierungsansätze der Sentiment-Analyse: subjektive, objektive Binary Klassifizierung; positive, negative Binary Klassifizierung von subjektiven Informationen [36] oder multivariate Klassifizierung. Bei der multivariaten Klassifizierung gibt es drei bis sieben Klassen: Koppel und Schler [12], Kouloumpis et al. [13] und Pak und Paroubek [20] haben mit den Klassen "positive, negative und neutral" gearbeitet. Li et al. haben sogar mit den sieben Klassen "Glück, Mögen, Wut, Ekel, Angst, Traurigkeit und neutral" gearbeitet, einer Fine-Grained Sentiment Analysis [16].

Technisch gibt es zwei Arten von Sentiment-Analyse:

- Wörterbuch- (Lexikon, eine wertvolle Ressource [6]) oder Korpus-basierte Sentiment-Analyse [20].
- Machine Learning [9] [30].

Wörterbücher enthalten für jede Klasse eine Liste.

Die Wörterbuch basierte Sentiment Analyse wird häufig für die Wortebenen benutzt [14]. Und es ist auch ein wesentlicher Bestandteil für "finegrained sentiment analysis " [11]. Taboada et al. [29] haben mit Hilfe der Methode "SO-CAL-Full", einer verbesserten Version von "SO-CAL" (Semantic Orientation CALculator) [28], bis zu 78.74% Korrektheit erreicht. Weitere Ergebnisse sind 71.2% von Choi und Cardie mit Hilfe von Adapted Lexikon [6] und 84.2% von Hu und Liu [10].

Für Machine Learning ist Feature Abstraktion sehr wichtig. Geeignete Features führen zu einem besseren Ergebnis. Mit dem Feature "Unigrams Bsubjectivity" für das Machine Model ME (MaximumEntrop) wurden 87.40% Korrektheit erreicht [4]. NB (NaiveBayes) und SVM (SupportVectorMachine) sind zwei weitere beliebte Modelle für Machine Learning [14].

Es gibt mehrere "level" für Textanalyse. Word-level [34] [11] (für Mikroblogging geeignet, weil die Texte "kurz" sind. Z.B. Twitter erlaubt nur 140 Zeichen pro Text), phrase-Level [33]; sentence level [11], dokument-level [21] und noch feature-level [14].

Sentiment-Analyse kann bei der Beurteilung einer Sache ein besseres Analyse Ergebnis erreichen. Es ist eine neue Technik mit großem Anwendungswert [35]. Dazu kommen viele Werkzeuge, um eine automatische Analyse [3] zu realisieren, wie z.B. Meltwater[6], Google Analytics[7] und Tweetstats[8].

[6] http://www.meltwater.com/
[7] http://www.peoplebrowsr.com/
[8] http://www.tweetstats.com/

Eine ganz neues Forschungsergebnis wurde am 20. Jan 2015 veröffentlicht: dort, wo mehr negative Worte in Twitter benutzt werden, gibt es oft eine höhere Prävalenzrate (Krankheitshäufigkeit) von Herzerkrankungen, und Gebiete in denen mehr positive Worte in Twitter benutzt werden, haben eine niedrigere Prävalenzrate [7].

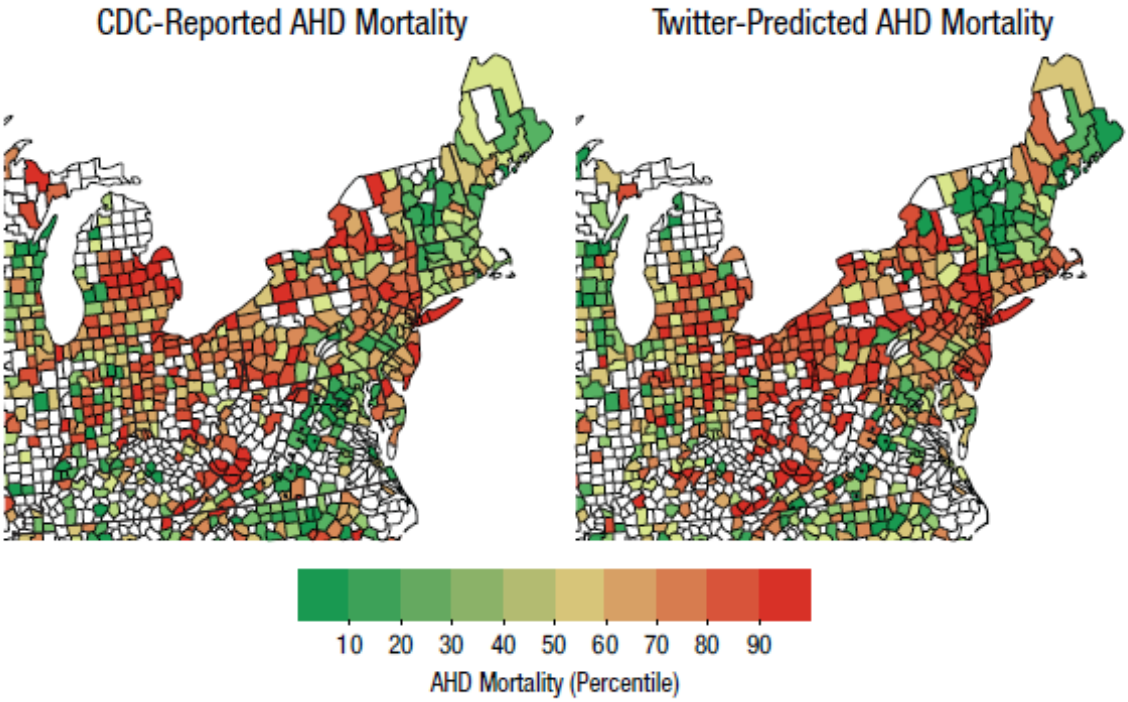

Figure 5: Quelle: [7]

SemEval Workshop ist ein Internationaler Workshop über semantische Auswertung der zum ersten Mal im Jahr 1998 stattgefunden hat. Das beste Ergebnis der Aufgabe "Sentiment Analysis in Twitter" von 2013 erreicht F_1 (siehe Kapitel 7.1) 69% auf dem Twitter-Datensatz [19]. Und im 2014 wurde bei derselben Aufgabe das beste Ergebnis für F_1 nochmal auf 70.96% verbessert [24].

4 Daten und Vorverarbeitung

Es wurde englischsprachige Tweets für das Festlegen von Parametern, Training und Testen verwendet. Es wurde zwei Listen von englischen Worten als Lexikon verwendet: Positivliste und Negativeliste. Dazu kommt noch eine Liste von Filterdaten, die ebenfalls englische Worte enthält: stopwords.

4.1 Trainings- und Testdaten

- task_B_dev_data (dev_data)
  Die dev_data sind Teil des Datensatzes "Task B Development Data" von der Aufgabe "Message Polarity Classification" von "Sentiment Analysis in Twitter" der im Rahmen des Workshops SemEval-2013 bereitgestellt wurde[9]. Die Daten wurden verwendet um die Parameter (Neg, max_count, min_score und level: siehe Kapitel 5 (In den Methoden verwendete Parameter)) für die Methoden festzulegen. Sie wurden auch verwendet um zu testen, ob mit einem geänderten Lexikon und stopwords ein besseres Klassifikationsergebnis erreicht werden kann, als mit den jeweiligen Originalen.

- task_B_train_data (train_data)
  Die train_data sind Teil des Datensatzes "Task B Training Data" von der Aufgabe "Message Polarity Classification" von "Sentiment Analysis in Twitter" der im Rahmen des Workshops SemEval-2013 bereitgestellt wurde[8] . Die Daten wurden beim Machine Learning als Trainingsdaten verwendet.

- task_B_test_data
  Die test_data_all sind Teil des Datensatzes "Task B Test Data" von der Aufgabe Task #9 von "Sentiment Analysis in Twitter" der im Rahmen des Workshops SemEval-2014 bereitgestellt wurde. Die test_data_tweets sind eine Teilmenge von test_data_all, die nur die Tweets ohne Live-Journal und Tweet sarcasm Daten enthält [24]. Die Daten werden in der "Lexical-Based" Methode und beim Machine Learning als Testdaten verwendet.

Es wurde zwei doppelte Einträge aus dev_data, 63 aus train_data und 16 aus test_data_all entfernt. In test_data_all sind 957 Datensätze nicht mehr erreichbar, d.h. die Datensätze können nicht mehr heruntergeladen werden. Die Datensets enthalten am Ende die folgenden Anzahlen von positiv, negativ und neutral markierten Datensätzen:

Datenset	Positive Labels	Negative Labels	Neutral Labels	Gesamtanzahl
dev_data	370	186	454	1010
train_data	2311	883	3120	6314
test_data_all	3051	1369	3596	8016
test_data_tweets	2134	672	1987	4793

Table 2: Gesamt positive, negative und neutrale Labels aller Tweets der Datensets

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

Es ist in ca. 10% der Tweets nicht ganz eindeutig zu bestimmen zu welcher Klasse sie gehören. In diesen Fällen hängt die Klasse von der persönlichen Meinung des Klassifizierers ab. Z.B. in "train_data" ist folgender Tweet:

I can not change yesterday. I can only make the most to day, and look with hope toward tomorrow. (*Anonymous*)

als negativ klassifiziert aber es könnte auch neutral sein.

4.2 Polarity-Lexikon

Es gibt zwei Versionen des Lexikons. Version 1 enthält nur eine Positivliste (2086 Worte mit Emoticons/Smileys) und eine Negativliste (4987 Worte mit Emoticons/Smileys). Das Lexikon stammt von Hu and Liu 2004 [10] und im Folgenden wird "pn" als Abkürzung dafür verwendet. Version 2 besteht aus einer Positivliste (2304 Worte), einer Negativliste (4153 Worte) und einer Neutralliste (430 Worte). Die drei Listen enthalten jeweils keine Emoticons/Smileys. Das Lexikon stammt von Wilson, Wiebe, & Hoffmann, 2005 [33] und im Folgenden wird "pnn" als Abkürzung dafür verwendet. Basierend auf diesen zwei Versionen, wurden zwei leicht geänderte Lexika erstellt (siehe Anhang pn_new und pnn_new), die mit "pn_new" bzw. "pnn_new" abgekürzt werden.

Mit dev_data (1010 Tweets, zwei doppelte Tweets wurden nur einmal gezählt) wurden folgende Tests durchgeführt:

- Es wurde die "Lexical-Based" Methode (siehe Kapitel 6.1) für alle vier Version mit dev_data als Input verwendet.

- Alle Datensätze mit positivem Wert wurden als positiv klassifiziert, mit negativem Wert wurden als negativ klassifiziert und der Rest als neutral klassifiziert.

Das Klassifizierungsergebnis ist in folgender Tabelle zu sehen:

datenset	correct count	Precision	Recall	F_1
pn	581	0.5449	**0.5427**	0.5438
pn_new	**599**	**0.5765**	0.5229	**0.5484**
pnn	484	0.4597	0.4457	0.4526
pnn_new	589	0.5637	0.5019	0.5310

Table 3: Ergebnis der "Lexical-Based" Methode mit verschiedenen Lexikon Versionen und Grenzwert (-1, 1).
Evaluationsmaße siehe Kapitel 7.1

Das beste Ergebnis wurde mit dem geänderten Lexikon Version 2 (pn_new) erzielt. Daher wurde dieses als Standardlexikon für alle weiteren Experimente verwendet. Für die beiden Wortlisten der geänderten Versionen pn_new und pnn_new siehe Anhang.

4.3 Filterdaten: stopwords

Die Wortliste stopwords enthält zwei Arten von Worten. Bei der einen Art handelt es sich um Worte, die sehr häufig vorkommen, aber im Vergleich zu anderen Worten, keine wirkliche Bedeutung haben, wie z.b. "it", "is", "i" usw... Bei der anderen Art handelt es sich um Worte wie "will", usw. Solche Worte werden zu oft verwendet um die Bedeutung eines Satzes zu beeinflussen.

Im dieser Arbeit wird die Liste stopwords verwendet um Untersuchungszeit und Speicher zu sparen und das Rauschen der Expansionen zu verkleinern (Von den Worten in der Liste stopwords wird nicht erwartet, dass sie eine positive/negative Bedeutung haben, aber mit Expansion können sie trotzdem eine positive/negative Bedeutung bekommen).

Z.B. das Wort "far" in stopwords hat nach der Untersuchung mit der Methode multi_expansion_count den Wert -0.0053(siehe 6.2.4).

stopwords (stopwords_en.txt[10] (571 Worte)) wird als Filterdaten im dieser Arbeit verwendet.

Es werden die Negierungsworte (ein solches Wort kann die Bedeutung eines Tweets ins Gegenteil verkehren, wie z.B. "isn't") aus stopwords entfernt, weil in den Methoden ein Negierungsparameter (siehe Kapitel 5.1) verwendet wurde. Und einige Worte, die zum positiven oder negativen tendieren könnten, wie "like, right, useful,..." wurden aus der Liste stopwords entfernt.

ain't	appreciate	appropriate	aren't	available	awfully	best
better	can't	cannot	clearly	couldn't	didn't	doesn't
don't	enough	hadn't	hasn't	haven't	isn't	like
liked	reasonably	right	sensible	shouldn't	sorry	thank
thanks	unfortunately	unlikely	useful	wasn't	welcome	well
weren't	willing	won't	wonder	wouldn't		

Table 4: Alle 40 aus den original stopwords entfernten Worte.

4.4 Expansionsdaten

Expansionsdaten wurden für alle Worte, die untersucht werden sollten, aber nicht im Lexikon oder Filterdaten (stopwords) enthalten sind, verwendet.

Durch die URL:

$$http://maggie.lt.informatik.tu-darmstadt.de:10080/jobim/ws/api/\mathbf{twitter2012Bigram}/jo/similar/\mathbf{Gesuchtes\ Wort}?numberOfEntries=\mathbf{Gewünschte\ Anzahl\ der\ zurückgeliferten\ Elemente}\&format=json$$

wird eine Anfrage nach einem gesuchten Wort an JoBimText gesendet und die Antwort als Datensatz im Json Format heruntergeladen. "twitter2012Bigram" ist die verwendete "Parse": Holing operation [2] und die heruntergeladene Liste darf maximal 200 Elemente enthalten. Jedes Element der Liste bestehen aus einem Wort und dem dazugehörigen "Score" Wert.

[10] https://sites.google.com/site/kevinbouge/stopwords-lists

Beispiel:

Eine Anfrage mit diesen Parametern

$$gesuchtes\ Wort\ =\ ''no''$$
$$gewünschte\ Anzahl\ =\ 50$$
$$Holing\ operation\ =\ twitter2012Bigram$$

ergibt folgende Liste:

{''error'' : null,'' method'' :'' getSimilarTerms'','' holingtype'' : {''name'' :'' twitter2012Bigram'','' isDefault'' : false},'' results'' :
[{''score'' : 1000.0,'' key'' :'' no'','' contextScore'' : null},
{''score'' : 278.0,'' key'' :'' No'','' contextScore'' : null},
...
{''score'' : 29.0,'' key'' :'' have'','' contextScore'' : null}]}

4.5 Abdeckung des Lexikons und der Expansions

In diesem Abschnitt wurde die Abdeckung des Lexikons und der Expansions untersucht.

	Durchschnitt der Abdeckungswerte
Lexikon	0.3617
Expansion	0.9940
Lexikon mit demselben Label	0.2430
Expansion mit demselben Label	0.3550

Table 5: Ergebnis der Abdeckung von dev_data mit der Methode ac_red und max_count=20, amp_fac=16 und damp_fac=0.5

Ein Abdeckungswert eines Lexikons ist die Anzahl aller positiven und negativen Worte eines Tweets, die im Lexikon enthalten sind, geteilt durch die Anzahl aller Worte des Tweets.

Ein Abdeckungswert eines Lexikons mit demselben Label ist die Anzahl aller positiven/negativen Worte eines Tweets, die im Lexikon enthalten sind, geteilt durch die Anzahl aller Worte des Tweets, falls der Tweet ein positives/negatives Label hat.

Ein Abdeckungswert einer Expansion ist die Anzahl aller positiven und negativen Worte eines Tweets, die im Lexikon enthalten sind oder einen nicht null Wert mit Hilfe von Expansion bekommen können, geteilt durch die Anzahl aller Worte des Tweets.

Ein Abdeckungswert einer Expansion mit demselben Label ist der Anzahl alle positiven/negativen Worte eines Tweets, die im Lexikon enthalten sind, oder einen nicht null Wert mit Hilfe von Expansion bekommen können, geteilt durch die Anzahl aller Worte des Tweets, falls der Tweet ein positives/negatives Label hat.

Die Durschnittswert davon ist der Durschnittwert von aller Tweets in dev_data.

Tabelle 5 zeigt, dass mit Expansion ein besseres Ergebnis als bei Lexikon mit demselben Label und Expansion mit demselben Label erreicht wird.

Beim Lexikon mit demselben Label ist der durchschnittliche Abdeckungswert 0.2430 und beim Lexikon ist der Wert 0.3617. Das Rauschen beträgt Lexikon/Expansion ohne dasselbe Label, bzw. 0.1187 von Lexikon und 0.639 von Expansion. D.h. das Rauschen wird durch die Expansion stark vergrößert.

5 In den Methoden verwendete Parameter

In diesem Kapitel werden die Parameter vorgestellt, die in den Methoden verwendete werden.

5.1 Negierungsparameter: Neg

Es gibt Wörter, die die Bedeutung eines Satzes in das Gegenteil verkehren. Diese Worte werden hier "Negierungsworte" genannt. Ein Beispiel dafür ist "isn't". Der Satz: "It is good." ist eindeutig positiv gemeint und der Satz: "It isn't good", hat eine negative Bedeutung. Solche Wörter haben eine wichtige Bedeutung in Sätzen und sollten berücksichtigt werden. Weil die Regeln für den Satzbau flexibel sind, können solche Wörter an verschiedenen Stellen in Sätzen stehen.

ain't	can't	cannot	aren't	couldn't	doesn't	cant
don't	didn't	hadn't	hasn't	haven't	isn't	never
weren't	wasn't	won't	shouldn't	wouldn't		

Table 6: Alle Worte, die den Negierungsparameter auf -1 setzen

In diesem Experiment wurden nur drei Fälle getestet.

- (Wort negieren) nur der Wert des Wortes, das direkt hinter dem Negierungswort in der vorverarbeiteten Wortliste steht, wurde negiert (*(-1)).
- (Worte negieren) Der Wert aller Worte, die hinter dem Negierungswort in der vorverarbeiteten Wortliste stehen, wurde negiert.
- (Satz negieren) Wenn ein Satz ein Negierungswort enthält, wurde sein Wert negiert.

Mit dev_data wurden Tests mit allen drei Negierungsfällen und ohne Negierungsfall durchgeführt. Die dazu verwendete Methode ist die "Lexical-Based" Methode mit Grenzwert (-1, 1).

Negierungsart	correct count	Precision	Recall	F_1
nicht negieren	599	0.5549	0.5331	0.5438
Wort negieren	**604**	**0.5829**	**0.5413**	**0.5564**
Worte negieren	586	0.5565	0.4868	0.5193
Satz negieren	454	0.4678	0.4412	0.4541

Table 7: Ergebnis der verschiedenen Negierungsarten
Evaluationsmaße siehe Kapitel 7.1

Weil "Wort negieren" das beste Ergebnis geliefert hat, wurde im Folgenden nur diese Art von Negierungsparameter verwendet.

5.2 Anzahl der verwendeten Worte aus der Expansion List: max_count

Es werden maximal 200 Worte von JoBimText zurückgeliefert, wenn ein Wort untersucht wird. Aber es ist unklar, wie viele Worte der Expansionslist verwendet werden sollen, um ein besseres

Klassifizierungsergebnis zu erreichen. Dieser Parameter beschränkt, wie viele Expansionsworte, die von JoBimText zurückgeliefert werden, in der Methode untersucht werden.

5.3 Mindest Score der verwendeten Worte aus der Expansion List: min_score

Die maximal 200 Worte, die von JoBimText zurückgeliefert werden, haben jeweils einen Score. Aber welche Worte in der Methode verwendet werden sollen, um ein besseres Klassifizierungsergebnis zu erreichen, ist unklar. Dieser Parameter beschreibt eine Untergrenze für den Score. Nur die Worte, deren Score größer ist als der eingegebene Parameter, dürfen in der Methode verwenden werden.

5.4 Amplification Factor: amp_factor

Es gibt immer viele "unbekannte" Worte (die Worte, die nicht in Lexikon und stopwords enthalten sind) in der Expansionlist, die von JoBimText zurückgeliefert wird. Z.B. wenn "gooood" nachgefragt wurde, wurde eine solche Liste von JoBimText zurückgeliefert:

gooood 1000, *goood* 114, *good* 102, *goooood* 63, *Goood* 47, *Good* 47, *goodd* 42, *gooooood* 41
goooooood 40, *gooooooood* 34, *Gooood* 33, *gooodd* 30, *Gooooood* 29, *goooddd* 29, *gooddd* 27, ...

Allein unter den ersten 15 Worten sind schon 13 falsch geschriebene Worte dabei, die nicht im Wörterbuch, dem Lexikon oder den stopwords enthalten sind. Von insgesamt 15 Worten sind nur zwei Worte "bekannt". Das führt fast immer dazu, dass eine Methode für diesem Satz den Wert null zurückliefert. Um das zu verhindern oder zu verbessern wird der "amp_factor" verwendet. In diesem Beispiel wird überhaupt erst ein Wert ungleich null zurückgeliefert, wenn mindestens die Hälfte der Worte "bekannt" sind. Dann könnte der amp_factor auf 3 gesetzt werden und die Anzahl der bekannten Wörter auf "3*2=6" erhöht werden. Das könnte dann zu einem Wert, der nicht null ist, führen. Somit erhöht ein amp_factor den Prozentsatz von "bekannten" Worten, um einen nicht null Wert zurückzuliefern.

5.5 Damp Factor: damp_factor

Der zurückgelieferte Wert, der als Expansionswert berechnet wurde, ist auch nicht immer hilfreich. Manchmal ist der Wert zu klein, dann funktioniert die Expansion kaum. Wie im Beispiel in Kapitel 5.4. kann der Wert $\frac{2}{15}$ als Expansionswert einer Methode zurückgeliefert werden. Im Vergleich dazu hat ein Wort in der Positivliste den Wert 1 und $\frac{2}{15}$ ist viel zu klein um den Gesamtwert eines Satzes zu ändern. Daher kann der Wert mit dem damp_factor (>1) multipliziert werden um den zurückgelieferten Wert zu vergrößern. Wenn der zurückgelieferte Wert allgemein zu groß ist, dann wird der damp_factor auf einen Wert zwischen 0 und 1 gesetzt um das endgültige Ergebnis zu verkleinern.

5.6 Anzahl der Expansions: level

Dieser Parameter wird nur für Multiexpansion verwendet. Mit level = N wird eine "nicht erfolgreiche" Expansion (mit Rückgabewert 0) nochmal bis zu maximal N mal erweitert. Während die meisten verwendeten Methoden (siehe nächste Kapitel: "absolut count", "score") als Breitensuche

gesehen werden können, funktioniert Multiexpansion wie eine Art Tiefensuche. Und der level ist die maximale Tiefe der Suche.

5.7 Grenzwertpunkte

Ein Grenzwertpunkt ist eine reelle Zahl, die die Grenze zwischen zwei Klassen darstellt. Alle zum Klassifizieren verwendeten Methoden bilden einen Tweet auf eine reelle Zahl ab. Bei der Entscheidung, wie die reellen Zahlen den Klassen positiv, negativ oder neutral zugeordnet werden, spielen die Grenzwertpunkte eine große Rolle. Die "lexcical Based" Methode versucht das maximal erreichbare Klassifizierungsergebnis zu verbessern und die Grenzwertpunkte entscheiden über das tatsächlich Klassfizierungsergebnis.

In dieser Arbeit werden die Grenzwertpunkte auf drei verschiedene Arten erstellt: Maximalwert Methode, Festgrenzwertpunkte Methode und Methode der ähnlichen Verteilung.

5.7.1 Maximalwert Methode

Es wurde die Maximalwert Methode [32] verwendet, um die Grenzwertpunkte festzulegen. Bei dieser Methode wird der Grenzwertpunkt zwischen den Klassen positiv und neutral in das Intervall [0, 1] gelegt und der Grenzwertpunkt zwischen den Klassen negativ und neutral in das Intervall [0, -1]. Jedes Intervall wird in 100 Teile zerlegt. Die beiden Werte, die das beste Klassifikationsergebnis erreicht haben, wurden als Grenzwertpunkte für Trainingsdaten und Testdaten verwendet.

5.7.2 Festgrenzwertpunkte Methode

Eine weitere Art ist die Festgrenzwertpunkte Methode [-0.5, 0.5]. Alle Datensätze, deren Endwert größer gleich 0.5 ist, wurden der Klasse positiv zugeordnet, deren Endwert kleiner oder gleich -0.5 ist, wurden der Klasse negativ zugeordnet und die restlichen der Klasse neutral zugeordnet. Sie wird nur für Testdaten verwendet.

5.7.3 Methode der ähnlichen Verteilung

Die letzte Methode ist die Methode der ähnlichen Verteilung. Die Methode versucht eine ähnliche Verteilung beim Klassifizierungsergebnis von Testdaten wie von Trainningsdaten zu erreichen. Die Methode wurde nur zur Klassifizierung von Testdaten: test_data_all verwendet, weil die Verteilungen der positiven, neutralen und negativen Tweets ähnlich sind. Es wurde untersucht, wie die Trainingsdaten nach dem Training mit der maximalen Anzahl korrekt klassifizierter Tweets verteilt sind.

Dann wurde versucht mit Hilfe der Maximalwert Methode eine ähnliche Verteilung der Testdaten wie bei den Trainingsdaten zu erreichen. D.h. Die die endgültigen Grenzwerte der Testdaten sind die, mit denen die minimalen Werte von

$$(|test_{pos}| - Prozent_{train\_pos} * |Testdaten|)/38 +$$
$$(|test_{neu}| - Prozent_{train\_neu} * |Testdaten|)/45 +$$
$$(|test_{neg}| - Prozent_{train\_neg} * |Testdaten|)/17$$

erreicht werden. Und die Testdaten werden auch damit klassifiziert. $|test_{pos}|$ ist die Anzahl der positiven Tweets aus den Testdaten, die mit dem aktuellen Grenzwert klassifiziert werden. $Prozent_{train\_pos}$ sind der Prozentsatz der positiven Tweets der Trainingsdaten. $|Testdaten|$ sind die Anzahl (8016) der Tweets der Testdaten. 38, 45, und 17 sind die Prozente der positiven, neutralen und negativen Labels der Testdaten.

6 Methoden

Es wurden die ”Lexical-Based” Methode und Machine Learning für die Experimente verwendet.

6.1 ”Lexical-Based” Methode: list

Die Basismethode ist die ”Lexical-Based” Methode ohne Expansion. In der Lexikon basierten Methode bildet Semt(Tweet)(1) einen Tweet auf eine Zahl ab. Und Semt(Wort) (2) ist der Wert des Wortes und |Tweet| ist die Anzahl der Worte des Tweets. Ein Wort in der Positivliste hat den Wert 1, in der Negativliste den Wert -1 und sonst 0.

$$Semt(Tweet) = \sum_{i=0}^{|Tweet|} (Semt(Wort_i)) \tag{1}$$

$$Semt(Wort) = \begin{cases} Neg * 1 & falls\ Wort\ in\ der\ Positivliste \\ Neg * -1 & falls\ Wort\ in\ der\ Negativliste \\ 0 & sonst \end{cases} \tag{2}$$

Ein Tweet mit einem positiven Wert wird immer als positiv klassifiziert, mit negativem Wert als negativ und die restlichen als neutral klassifiziert.

6.2 ”Lexical-Based” Methode und Expansion mit Hilfe von JoBimText

In der Lexikon basierten Methode mit Expansion wurde mit Hilfe von JoBimText versucht, den Wert eines Wortes (Semt(Wort)), das weder in der Positivliste noch in der Negativliste enthalten ist, zu erweitern. Das bedeutet, dass der Wert eines Wortes, der vorher 0 ist, Chancen hat, nicht mehr 0 oder sogar eine reelle Zahl zu sein. Damit wird ein abgestufter Wortwert durch die Expansion erzeugt.

$$Semt(Wort) = \begin{cases} Neg * 1 & falls\ Wort\ in\ der\ Positivliste \\ Neg * -1 & falls\ Wort\ in\ der\ Negativliste \\ Neg * ExpansionSemt(Wort) & sonst \end{cases} \tag{3}$$

Beispiel 1(Tweet):

Homework catch up day and then dinner with the ″Big Brother″ before the hurricane. I kinda need school not to happen on Monday

Nach dem Vorverarbeiten (Url-Filter, Twitter-Benutzernamen-Filter, Tokenisierung und stopwords-Filter) erhält man folgende Wortliste:

Homework, catch, day, dinner, Big, Brother, hurricane, kinda, school, happen, Monday

Ohne Expansion ist der Wert des Tweets 0:

$$Semt(Homework,\ catch,\ day,\ dinner,\ Big,\ Brother,\ hurricane,\ kinda,\ school,\ happen,\ Monday) = 0$$

Das liegt daran, dass keines der Worte in der Positiv- oder Negativliste enthalten ist. Mit der Hilfe von Expansion (Methode ac_red amp_fac =32 damp_fac = 0.5 max_count = 100) hat es den Wert:

$$Semt(Homework,\ catch,\ day,\ dinner,\ Big,\ Brother,\ hurricane,\ kinda,\ school,\ happen,\ Monday) = -0.2$$

Weil

$$Homework : -0.035353534;\ catch : -0.05050505;\ Big : 0.05050505;$$
$$hurricane : -0.07070707;\ kinda : -0.04040404;\ happen : -0.055555556;$$

Beispiel 2 (Tweets aus dev-data):

1: Homework catch up day and then dinner with the ”Big Brother” before the hurricane. I kinda need school not to happen on Monday (wurde von einem menschlichen Experten negativ bewertet)

2: I googled ”coffee & its immediate response, mechanisms” the first page that appeared ”September 11 attacks, Wiki”.. I dont see the link here (wurde von einem menschlichen Experten negativ bewertet)

3: It’s 3 ferry Friday between Torpoint and Plymouth with waiting times upto 10 minutes. [PC] (wurde von einem menschlichen Experten neutral bewertet)

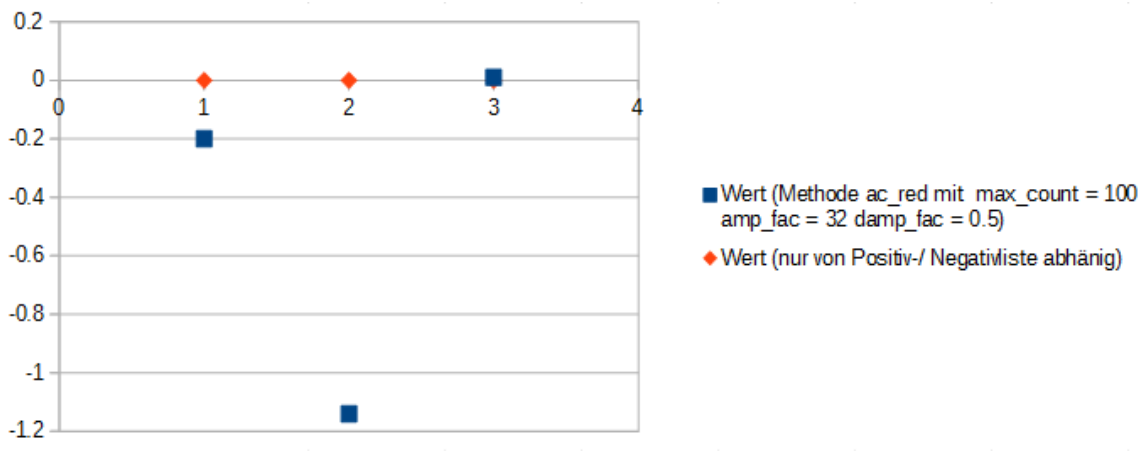

Figure 6: x: Nummer des Tweets, y: Wert des Tweets (Semt(Tweet))

In der Abbildung wird gezeigt, dass die drei gelben Punkte auf einer Linie liegen, weil alle drei Tweets den Wert 0 haben. Das heißt, egal wie man es klassifiziert, liegen die drei Tweets in einer Klasse und mindestens ein Tweet wurde falsch klassifiziert (z.B. alle Tweets, deren Wert kleiner als 1 ist, wurden als negativ klassifiziert). Und die blauen Punkte, die nicht mehr auf einer Linie liegen, haben schon viel mehr Klassifikationsmöglichkeiten als die gelben. Beispielsweise erhält

man ein korrektes Klassifikationsergebnis, wenn man alle Tweets, deren Wert kleiner als 0.5 und größer als 0.1 ist als neutral klassifiziert und alle Tweets, deren Wert kleiner gleich 0.1 ist, als negativ klassifiziert.

Die Methoden, die in diesem Kapitel beschrieben werden, versuchen mit Hilf von Expansion (JoBimText) eine besseres Klassifizierungsergebnis zu erreichen.

6.2.1 Verwendete Begriffe

Die folgende Tabelle erklärt einige Begriffe, die in den kommenden Methoden verwendet werden, mit ihren Abkürzungen:

	Abkürzung	Bedeutung
JoBimText(Wort)	JoBimText	Liste aller zu untersuchenden Wörter aus der Expansion
#JoBimText(Wort)	#JoBimText	die Anzahl der zu untersuchenden Wörter in der Expansionsliste
#pos(Wort)	#pos	die Anzahl der zu untersuchenden Wörter aus der Expansion, die in der Positivliste liegen
#neg(Wort)	#neg	die Anzahl der zu untersuchenden Wörter aus der Expansion, die in der Negativliste liegen
#sum(Wort)	#sum	#pos + #neg
score(pos(Wort))	score(pos)	Summe der Scores aller zu untersuchenden Wörter aus JoBimText(Wort), falls das Wort in der Positivliste enthalten ist.
score(neg(Wort))	score(neg)	Summe der Scores aller zu untersuchenden Wörter aus JoBimText(Wort), falls das Wort in der Negativliste enthalten ist.
score(sum(Worte))	score(sum)	score(pos)+ score(neg)
score(Wort)	–	Summe der Scores aller zu untersuchenden Wörter aus JoBimText(Wort).

Table 8: Verwendete Begriffe der Methoden

6.2.2 Absolute Count

Es wurde ein Wert erzeugt, der davon abhängt, ob positive oder negative Worte am häufigsten in der Expansionwordlist vorkommen. In diesen Methoden wurde untersucht, wie sich das Verhältnis der Anzahl der positiven Worte zur Anzahl der negativen Worte aus der Expansion auswirkt.

Eine Word-Expansion wurde durch den eingegebenen Parameter max_count beschränkt. Es wurden nur die ersten N (N=max_count) Worte untersucht, die von JoBimText zurückgeliefert wurden.

$$JoBimText(Wort) = \begin{cases} (w_0, \ldots, w_{mac\_count-1}) & falls\ \#JoBimText > mac\_count \\ (w_0, \ldots, w_{\#JoBimText-1}) & sonst \end{cases} \quad (4)$$

- **ac_nred** : Absolute Count ohne Rückgabewertverkleinerung

  Die Expansion wurde als ein erweitertes Lexikon verwendet, und der Wert des Wortes, der aus der Expansion gebildet wird, wird so behandelt, als wäre das Wort im Lexikon enthalten gewesen. Ein Expansionswert ist eins, wenn die Anzahl von positiven Worten größer ist als die Anzahl von negativen Worten, und minus eins, wenn umgekehrt. Sonst ist der wert 0.

$$ExpansionSemt(Wort) = \begin{cases} 1 & falls\ \#pos > \#neg \\ -1 & falls\ \#pos < \#neg \\ 0 & sonst \end{cases} \quad (5)$$

- **ac_red :** "absolute count" mit Rückgabewertverkleinerung

  Die Methode hier reduziert den Expansionswert ($\frac{(\#pos \mid \#neg) * damp\_fac}{\#JoBimText(wort)}$) und untersucht den Einfluss auf die Klassifikation. Eingabeparameter sind max_count, damp_fac und amp_fac. Ein Expansionswert ist positiv, wenn die Anzahl von positiven Worten größer ist als die Anzahl von negativen Worten, und negativ, wenn umgekehrt. Sonst ist der wert 0.

  Eine zusätzliche Bedingung ist:

$$\#JoBimText(Wort) < amp\_fac * \#sum \tag{6}$$

  Wenn die Bedingung erfüllt ist, dann ist der zurückgeliefert Wert:

$$ExpansionSemt(Wort) = \begin{cases} \frac{\#pos * damp\_fac}{\#JoBimText} & falls\ \#pos >= \#neg \\ -\frac{\#neg * damp\_fac}{\#JoBimText} & falls\ \#pos < \#neg \end{cases} \tag{7}$$

  und sonst wird 0 zurückgeliefert.

  Mit den Trainingsdaten (dcv_data) wurden zuerst die Parameter max_count mit amp_fac=16 und damp_fac=0.5 untersucht. Die Kandidaten für max_count sind: 200, 100, 50, 20, 10, 5, 2.

max_count	available correct count	Precision	Recall	F_1
200	617	0.6074	0.5017	0.5495
100	617	0.6070	0.4956	0.5457
50	618	0.5998	0.5206	0.5574
20	**626**	**0.6319**	0.5046	0.5611
10	622	0.6233	0.5056	0.5583
5	608	0.5848	0.5299	0.5560
2	614	0.5797	**0.5579**	**0.5686**

Table 9: Ergebnis der Parameteruntersuchung max_count

  Tabelle 9 zeigt, dass mit dem Parameter max_count=20 die maximale Anzahl korrekt klassifizierter Trainingsdaten erreicht wird. Für Testdaten wird dann max_count auf 20 gesetzt.

  Dann wird der Parameter damp_fac mit max_count=20 und amp_fac=16 untersucht. Die Kandidaten für damp_fac sind: 0.1, 0.2, 0.5, 1, 2.

  Tabelle 10 zeigt, dass mit dem Parameter damp_fac=0.5 die maximale Anzahl korrekt klassifizierter Trainingsdaten erreicht wird.

  Dann wird den Parameter amp_fac mit max_count=20 und damp_fac=0.5 untersucht. Die Kandidaten für damp_fac sind: 2, 4, 8, 16, 32, 64, 128.

  Tabelle 11 zeigt, dass mit dem Parameter amp_fac=16 die maximale Anzahl korrekt klassifizierter Trainingsdaten erreicht wird.

damp_fac	available correct count	Precision	Recall	F_1
0.1	626	0.6319	0.5046	0.5611
0.2	626	0.6319	0.5046	0.5611
0.5	**626**	**0.6319**	0.5046	0.5611
1	623	0.6211	0.5098	0.56
2	610	0.6046	**0.5257**	**0.5624**

Table 10: Ergebnis der Parameteruntersuchung damp_fac

amp_fac	available correct count	Precision	Recall	F_1
2	611	0.5773	**0.5613**	**0.5692**
4	613	0.5923	0.5379	0.5638
8	619	0.6083	0.5166	0.5587
16	**626**	**0.6319**	0.5046	0.5611
32	623	0.6277	0.4847	0.547
64	623	0.6277	0.4847	0.547
128	623	0.6277	0.4847	0.547

Table 11: Ergebnis der Parameteruntersuchung amp_fac

amp_fac=16, max_count=20 und damp_fac=0.5 werden für Testdaten verwendet. Und mit diesen drei Parametern wird die Maximalwert Methode [32] verwendet, und daraus wird (-0.99, 0.31) als Grenzwertpunkt bestimmt. Die dazugehörigen neuen positiven Tweets sind 341, neue neutrale Tweets sind 543 und neues negative Tweets sind 126.

Für Testdaten wird dann (-0.99, 0.31) für die Maximalwert Methode [32] verwendet und $\frac{341}{1010}, \frac{543}{1010}, \frac{126}{1010}$ wird für die Methode der ähnlichen Verteilung verwendet.

6.2.3 Ähnlichkeitsscore

Es wird ein Wert zu erzeugt, der von dem Score der positiven oder negativen Worte in der Expansionwordlist abhängig ist. In dieser Methode wird Word-Expansion nicht mehr durch die maximale Anzahl der zu untersuchenden Worte aus der Expansion List sondern durch den minimalen Score (Ähnlichkeitswert) beschränkt. Es wurde untersucht, wie sich der Score von positiven und negativen Worten aus der Expansion auf die Klassifizierung auswirkt.

Wie schon weiter oben beschrieben, liefert JoBimText eine Liste zurück und jedes Element dieser Liste besteht aus einem Wort und seinem Ähnlichkeitswert. Es wurden nur die Worte, deren Ähnlichkeitswert größer als der eingegebene Wert des Parameter min_score ist, verwendet.

$$
\begin{aligned}
score(pos(Wort)) &= \sum_{Wort_i \in JoBimText(Wort)} score(Wort_i) \quad falls\ Wort_i\ in\ der\ Positivliste\ enthalten\ ist \\
score(neg(Wort)) &= \sum_{Wort_i \in JoBimText(Wort)} score(Wort_i) \quad falls\ Wort_i\ in\ der\ Negativliste\ enthalten\ ist \\
score(Wort) &= \sum_{Wort_i \in JoBimText(Wort)} score(Wort_i) \\
score(sum) &= score(pos) + score(neg)
\end{aligned}
\tag{8}
$$

- **sl_red :** linear Score mit Rückgabewertverkleinerung

In dieser Methode wurde untersucht, wie sich das Verhältnis der Summe aller Scores der positiven Worte zu der Summe aller Scores der negativen Worte in der Expansion auf das Klassifikationsergebnis auswirkt.

Die dazugehörige zusätzliche Bedingung ist:

$$score(Wort) < amp\_fac * score(sum) \tag{9}$$

Es liefert ein float Zahl zurück:

$$ExpansionSemt(Wort) = \begin{cases} \frac{score(pos) * damp\_fac}{score(Wort)} & falls\ score(pos) > score(neg) \\ -\frac{score(neg) * damp\_fac}{score(Wort)} & falls\ score(neg) > score(pos) \\ 0 & sonst \end{cases} \tag{10}$$

Mit den Trainingsdaten (dcv_data) wurden zuerst die Parameter min_score mit amp_fac=16 und damp_fac=0.2 untersucht. Die Kandidaten von min_score sind: 500, 200, 100, 50, 20, 10, 5.

min_score	available correct count	Precision	Recall	F_1
500	604	0.5800	0.5346	0.5564
200	608	0.5884	0.5245	0.5546
100	612	**0.5966**	0.5134	0.5519
50	613	0.5933	0.5326	0.5613
20	612	0.5908	0.5197	0.553
10	**614**	0.5897	**0.5406**	**0.5641**
5	610	0.5905	0.5109	0.5478

Table 12: Ergebnis der Parameteruntersuchung min_score

Tabelle 12 zeigt, dass mit dem Parameter min_score=10 die maximale Anzahl korrekt klassifizierter Trainingsdaten erreicht wird. Für Testdaten wird dann min_score auf 20 gesetzt.

Dann wird der Parameter damp_fac mit min_score=10 und amp_fac= 16 untersucht. Die Kandidaten für damp_fac sind: 0.1, 0.2, 0.5, 1, 2.

Tabelle 13 zeigt, dass mit dem Parameter damp_fac=0.5 die maximale Anzahl korrekt klassifizierter Trainingsdaten erreicht wird. Für Testdaten wird dann damp_fac auf 0.5 gesetzt.

Dann wird der Parameter amp_fac mit min_score=10 und damp_fac=0.5 untersucht. Die Kandidaten für amp_fac sind: 2, 4, 8, 16, 32, 64, 128.

Tabelle 14 zeigt, dass mit dem Parameter amp_fac=64 die maximale Anzahl korrekt klassifizierter Trainingsdaten erreicht wird. Für Testdaten wird dann damp_fac auf 64 gesetzt.

damp_fac	available correct count	Precision	Recall	F_1
0.1	613	0.5868	0.5392	0.5620
0.2	614	0.5897	0.5406	0.5641
0.5	**616**	0.5912	**0.5432**	**0.5662**
1	614	0.5892	0.542	0.5646
2	609	**0.5915**	0.5415	0.5654

Table 13: Ergebnis der Parameteruntersuchung damp_fac

amp_fac	available correct count	Precision	Recall	F_1
2	610	0.5839	0.5467	0.5647
4	617	0.5941	**0.5474**	**0.5698**
8	616	0.5917	0.5419	0.5657
16	616	0.5912	0.5432	0.5662
32	619	0.608	0.5097	0.5545
64	**624**	**0.6182**	0.5024	0.5543
128	621	0.6116	0.497	0.5484

Table 14: Ergebnis der Parameteruntersuchung damp_fac

amp_fac=64, min_score=10 und damp_fac=0.5 wurden für die Testdaten verwendet. Und mit diesen drei Parametern wird die Maximalwert Methode [32] verwendet, und daraus wird (-1.0,0.16) als Grenzwertpunkt bestimmt. Die dazugehörigen neuen positiven Tweets sind 343, neue neutrale Tweets sind 538 und neues negative Tweets sind 129.

Für Testdaten wird dann (-1.0,0.16) für die Maximalwert Methode [32] verwendet und $\frac{343}{1010}, \frac{538}{1010}, \frac{129}{1010}$ wird für die Methode der ähnlichen Verteilung verwendet.

- **slog_red** : Logarithmus von Score mit Rückgabewertverkleinerung

  Statt dem normalen Score wurde in dieser Methode einer 10er Logarithmus von Score verwendet. Damit wird die Wirkung des exponentiell verkleinerten Gewichts von Score untersucht.

  Anstatt des normalen score wird log_score verwendet:

$$
\begin{aligned}
log\_score(pos(Wort)) &= \sum_{Wort_i \in JoBimText(Wort)} \log(score(Wort_i)) \quad falls\ Wort_i\ in\ der\ Positivliste\ enthalten\ ist \\
log\_score(neg(Wort)) &= \sum_{Wort_i \in JoBimText(Wort)} \log(score(Wort_i)) \quad falls\ Wort_i\ in\ der\ Negativliste\ enthalten\ ist \\
log\_score(Wort) &= \sum_{Wort_i \in JoBimText(Wort)} \log(score(Wort_i))
\end{aligned}
\tag{11}
$$

  Die dazugehörige zusätzliche Bedingung ist:

$$
log\_score(Wort) < amp\_fac * (log\_score(pos(Wort)) + log\_score(neg(Wort))) \tag{12}
$$

Es liefert ein float Zahl zurück:

$$ExpansionSemt(Wort) = \begin{cases} \frac{log\_score(pos(Wort)) * damp\_fac}{log\_score(Wort)} & falls\ log\_score(pos(Wort)) \geq log\_score(neg(Wort)) \\ -\frac{log\_score(neg(Wort)) * damp\_fac}{log\_score(Wort)} & falls\ log\_score(neg(Wort)) > log\_score(pos(Wort)) \\ 0 & sonst \end{cases} \quad (13)$$

Es wird der Parameter damp_fac mit min_score=10 und amp_fac=64 untersucht. Die Kandidaten für damp_fac sind: 0.1, 0.2, 0.5, 1, 2.

damp_fac	available correct count	Precision	Recall	F_1
0.1	620	0.6106	0.5057	0.5532
0.2	622	**0.6155**	0.5057	0.554
0.5	**623**	0.5990	**0.536**	**0.5657**
1	623	0.5990	0.536	0.5657
2	613	0.6132	0.4934	0.5468

Table 15: Ergebnis der Parameteruntersuchung damp_fac

Tabelle 15 zeigt, dass mit dem Parameter damp_fac=0.5 die maximale Anzahl korrekt klassifizierter Trainingsdaten erreicht wird. Für Testdaten wird dann damp_fac auf 0.5 gesetzt.

Dann wird der Parameter amp_fac mit min_score=10 und damp_fac=0.5 untersucht. Die Kandidaten für amp_fac sind: 2, 4, 8, 16, 32, 64, 128.

amp_fac	available correct count	Precision	Recall	F_1
2	610	0.5839	0.5467	0.5647
4	614	0.5838	**0.5515**	**0.5672**
8	617	0.5942	0.542	0.5669
16	616	0.5986	0.5194	0.5562
32	620	**0.6052**	0.5233	0.5613
64	**623**	0.5990	0.5359	0.5657
128	622	0.5958	0.53	0.5610

Table 16: Ergebnis der Parameteruntersuchung amp_fac

Tabelle 16 zeigt, dass mit dem Parameter amp_fac=64 die maximale Anzahl korrekt klassifizierter Trainingsdaten erreicht wird. Für Testdaten wird dann amp_fac auf 64 gesetzt.

amp_fac=64, min_score=10 und damp_fac=0.5 wurden für die Testdaten verwendet. Und mit diesen drei Parametern wird die Maximalwert Methode [32] verwendet, und daraus wird (-0.99,0.07) als Grenzwertpunkt bestimmt. Die dazugehörigen neuen positiven Tweets sind 389, neue neutrale Tweets sind 483 und neues negative Tweets sind 138.

Für Testdaten wird dann (-0.99,0.07) für die Maximalwert Methode [32] verwendet und $\frac{389}{1010}, \frac{483}{1010}, \frac{138}{1010}$ wird für die Methode der ähnlichen Verteilung verwendet.

- **sk_red :** Hybrid-Methode "konstant"

Diese Methode ist von der Anzahl und dem Score der Expansionen abhängig. Die Expansionslist ist durch minimal Score beschränkt, wie bei allen anderen Score Methoden. In der Methode wurden zusätzlich noch #pos, #neg verwendet. Es wurde nach der Beschränkung von minimal Score untersucht, ob das Rauschen der Expansion weniger wird im vergleichen zur "Absolute Count" Methode.

Die dazugehörige zusätzliche Bedingung ist:

$$\#JoBimText(Wort) < amp\_fac * \#sum \tag{14}$$

Es liefert ein float Zahl zurück:

$$ExpansionSemt(Wort) = \begin{cases} \frac{\#pos * damp\_fac}{\#JoBimText} & falls\ \#pos >= \#nega \\ -\frac{\#neg * damp\_fac}{\#JoBimText} & falls\ \#pos < \#nega \end{cases} \tag{15}$$

Es wird der Parameter damp_fac mit min_score=10 und amp_fac= 64 untersucht. Die Kandidaten für damp_fac sind: 0.1, 0.2, 0.5, 1, 2.

damp_fac	available correct count	Precision	Recall	F_1
0.1	616	0.6112	0.484	0.5402
0.2	616	**0.6112**	0.484	0.5402
0.5	**616**	0.6094	0.4993	0.5489
1	615	0.6035	**0.5037**	**0.5491**
2	606	0.5958	0.498	0.5425

Table 17: Ergebnis der Parameteruntersuchung amp_fac

Tabelle 17 zeigt, dass mit dem Parameter damp_fac=0.5 die maximale Anzahl korrekt klassifizierter Trainingsdaten erreicht wird. Für Testdaten wird dann damp_fac auf 0.5 gesetzt.

Dann wird der Parameter amp_fac mit min_score=10 und damp_fac=0.5 untersucht. Die Kandidaten für damp_fac sind: 2, 4, 8, 16, 32, 64, 128.

amp_fac	available correct count	Precision	Recall	F_1
2	604	0.5810	0.5347	0.5569
4	610	0.5835	0.5445	**0.5633**
8	608	0.5726	**0.5501**	0.5611
16	616	0.5978	0.5179	0.555
32	616	**0.6139**	0.5039	6 0.5535
64	616	0.6094	0.4993	0.5489
128	**619**	0.6057	0.5104	0.554

Table 18: Ergebnis der Parameteruntersuchung amp_fac

Tabelle 18 zeigt, dass mit dem Parameter amp_fac=128 die maximale Anzahl korrekt klassifizierter Trainingsdaten erreicht wird. Für Testdaten wird dann damp_fac auf 128 gesetzt.

amp_fac=128, min_score=10 und damp_fac=0.5 wurden für Testdaten verwendet. Und mit diesen drei Parametern wird die Maximalwert Methode [32] verwendet, und daraus wird

(-0.98,0.16) als Grenzwertpunkt bestimmt. Die dazugehörigen neuen positiven Tweets sind 355, neue neutrale Tweets sind 521 und neues negative Tweets sind 134.

Für Testdaten wird (-0.98,0.16) für die Maximalwert Methode [32] verwendet und $\frac{355}{1010}, \frac{521}{1010}, \frac{134}{1010}$ wird für die Methode der ähnlichen Verteilung verwendet.

- **savg_red :** ein Durchschnittsscore mit Rückgabewertverkleinerung

  Es wurde die Wirkung der Durchschnittswerte von Score Werten der Expansionen untersucht, um zwischen den gleichen Werten bei der Scoresumme aber einer unterschiedlichen Anzahl positiver/negativer Expansionworte zu unterscheiden.

  Es verwendet die Durchschnittswerteformeln:

$$a : \frac{score(pos)}{\#pos}, \quad b : \frac{score(neg)}{\#neg}, \quad avg\_sum : \frac{score(Wort)}{\#JoBimText(Wort)}$$

  Die dazugehörige zusätzliche Bedingung ist:

$$score(Wort)) < amp\_fac * (score(posi) + score(nega)) \tag{16}$$

  Der dazugehörige zurückgelierferte Wert ist

$$ExpansionSemt(Wort) = \begin{cases} \frac{a * damp\_fac}{avg\_sum} & falls\ a >= b \\ -\frac{b * damp\_fac}{avg\_sum} & falls\ b > a \\ 0 & sonst \end{cases} \tag{17}$$

  Es wird der Parameter damp_fac mit min_score=10 und amp_fac=32 untersucht. Die Kandidaten für damp_fac sind: 0.1, 0.2, 0.5, 1, 2.

damp_fac	available correct count	Precision	Recall	F_1
0.1	**623**	0.6191	0.5024	**0.5547**
0.2	610	**0.6202**	0.4873	0.5458
0.5	534	0.5081	0.4628	0.4844
1	464	0.4132	0.488	0.4475
2	395	0.3471	**0.5275**	0.4187

Table 19: Ergebnis der Parameteruntersuchung damp_fac

Tabelle 19 zeigt, dass mit dem Parameter damp_fac=0.1 die maximale Anzahl korrekt klassifizierter Trainingsdaten erreicht wird.

Dann wird der Parameter amp_fac mit min_score=10 und damp_fac=0.1 untersucht. Die Kandidaten für damp_fac sind: 2, 4, 8, 16, 32, 64, 128.

Tabelle 20 zeigt, dass mit dem Parameter amp_fac=32 die maximale Anzahl korrekt klassifizierter Trainingsdaten erreicht wird.

amp_fac	available correct count	Precision	Recall	F_1
2	610	0.5829	**0.5429**	**0.5622**
4	610	0.5933	0.5324	0.5612
8	616	0.6093	0.5126	0.5568
16	618	0.6169	0.5079	0.5571
32	**623**	0.6191	0.5024	0.5547
64	617	0.624	0.475	0.5394
128	618	**0.6499**	0.4328	0.5196

Table 20: Ergebnis der Parameteruntersuchung amp_fac

Die Parameter amp_fac=32, min_score=10 und damp_fac=0.1 wurden für Testdaten verwendet. Und mit diesen drei Parametern wird die Maximalwert Methode [32] verwendet, und daraus wird (-0.9, 0.55) als Grenzwertpunkt bestimmt. Die dazugehörigen neuen positiven Tweets sind 335, neue neutrale Tweets sind 543 und neues negative Tweets sind 132.

Für Testdaten wird (-0.9, 0.55) für die Maximalwert Methode [32] verwendet und $\frac{335}{1010}, \frac{543}{1010}, \frac{132}{1010}$ wird für die Methode der ähnlichen Verteilung verwendet.

- **savg_u1_red** : eine Update des Durchschnittsscores mit Rückgabewertverkleinerung

  In dieser Methode wird die Auswirkung auf das Klassifikationsergebnis untersucht, wenn das Gewicht der einzelnen Scores im Durchschnittsscore vergrößert wird. In dieser Methode wird der Einfluss untersucht, wenn die Durchschnittswerte der Scores gleich sind, aber jeder einzelne Score doch unterschiedlich ist Wie z.B. Durchschnittswerte von (3, 3, 3) und (1, 2, 6) sind jeweils 3, aber $\frac{\prod}{\#}$ davon sind 9 und 4.
  Weil der nach der Multiplikation der Scores das Ergebnis zu groß sein kann, wird hier statt des normalen Scores der 10er Logarithmus der Scores benutzt. Die verwendeten Formeln sind:

$$a : \frac{\prod \log(score(pos))}{\#pos}, \quad b : \frac{\prod \log(score(neg))}{\#neg}, \quad sum : \frac{\prod \log(score(Wort))}{\#JoBimText(Wort)},$$

  Die Bedingung ist

$$a * b * amp\_fac > \log(sum) \tag{18}$$

  und der dazugehörige zurückgelieferte Wert ist

$$ExpansionSemt(Wort) = \begin{cases} \frac{a * damp\_fac}{sum} & falls\ a >= b \\ -\frac{b * damp\_fac}{sum} & falls\ b > a \\ 0 & sonst \end{cases} \tag{19}$$

  Es wird der Parameter damp_fac mit min_score=10 und amp_fac=32 untersucht. Die Kandidaten für damp_fac sind: 0.1, 0.2, 0.5, 1, 2.

  Tabelle 21 zigt, dass mit dem Parameter damp_fac=0.1 die maximale Anzahl korrekt klassifizierter Trainingsdaten erreicht wird.

damp_fac	available correct count	Precision	Recall	F_1
0.1	**608**	**0.5972**	0.4836	**0.5344**
0.2	604	0.5907	0.4777	0.5282
0.5	588	0.5388	0.4901	0.5133
1	572	0.4958	**0.5041**	0.4999
2	559	0.4969	0.4584	0.4769

Table 21: Ergebnis der Parameteruntersuchung damp_fac

amp_fac	available correct count	Precision	Recall	F_1
2	603	0.5823	**0.5163**	**0.5473**
4	604	0.5838	0.4994	0.5383
8	604	0.5856	0.4876	0.5321
16	607	0.5952	0.4837	0.5337
32	**608**	**0.5972**	0.4836	0.5344
64	607	0.5954	0.4808	0.532
128	607	0.5954	0.4808	0.532

Table 22: Ergebnis der Parameteruntersuchung amp_fac

Dann wird der Parameter amp_fac mit min_score=10 und damp_fac=0.1 untersucht. Die Kandidaten für damp_fac sind: 2, 4, 8, 16, 32, 64, 128.

Tabelle 22 zeigt, dass mit dem Parameter amp_fac=32 die maximale Anzahl korrekt klassifizierter Trainingsdaten erreicht wird.

amp_fac=32, min_score=10 und damp_fac=0.1 werden dann für Testdaten verwendet. Und diesen drei Parametere werden für die Maximalwert Methode [32] verwendet, und daraus wird (-1, 0.53) als Grenzwertpunkt bestimmt. Die dazugehörigen neuen positiven Tweets sind 334, neue neutrale Tweets sind 545 und neues negative Tweets sind 131.

Für Testdaten wird (-1, 0.53) für die Maximalwert Methode [32] verwendet und $\frac{334}{1010}, \frac{545}{1010}, \frac{131}{1010}$ für die Methode der ähnlichen Verteilung verwendet.

- **savg_u2_red :** eine weiteres Update des Durchschnittsscores mit Rückgabewertverkleinerung

  In dieser Methode wird der Einfluss untersucht, wenn die Methoden savg_red und savg_up1_red immer dasselbe Ergebnis liefern, aber die Anzahl der positiven/negativen Worte in der Expansionslist unterschiedlich ist.

  Wie z.B. Durchschnittswerte von (2, 2) und (2) sind jeweils 2, und $\frac{\prod}{\#}$ davon sind auch 2, aber mit der Formel: $\prod * \#$ erhält man die unterschiedlichen Ergebnisse 2 und 8.

  Die verwendeten Formeln sind:

$$a : \prod log\_score(posi) * \#posi, \quad b : \prod log\_score(nega) * \#nega, \quad sum : \prod log\_score * \#Expansion$$

Die Bedingung ist

$$a * b * amp\_fac > sum \tag{20}$$

und der dazugehörige zurückgelieferte Wert ist

$$ExpansionSemt(Wort) = \begin{cases} \frac{a * damp\_fac}{sum} & falls\ a >= b \\ -\frac{b * damp\_fac}{sum} & falls\ b > a \\ 0 & sonst \end{cases} \tag{21}$$

Es wird der Parameter damp_fac mit min_score=10 und amp_fac=32 untersucht. Die Kandidaten für damp_fac sind: 0.1, 0.2, 0.5, 1, 2.

damp_fac	available correct count	Precision	Recall	F_1
0.1	612	0.5888	0.5432	0.5651
0.2	612	**0.5888**	0.5432	**0.5651**
0.5	**614**	0.5849	0.5458	0.5647
1	613	0.5821	0.5472	0.5641
2	613	0.5834	**0.5473**	0.5648

Table 23: Ergebnis der Parameteruntersuchung damp_fac

Tabelle 23 zeigt, dass mit dem Parameter damp_fac=0.5 die maximale Anzahl korrekt klassifizierter Trainingsdaten erreicht wird.

Dann wird der Parameter amp_fac mit min_score=10 und damp_fac=0.5 untersucht. Die Kandidaten für damp_fac sind: 2, 4, 8, 16, 32, 64, 128.

amp_fac	available correct count	Precision	Recall	F_1
2	605	0.5805	0.5361	0.5574
4	606	0.5810	0.5401	0.5598
8	608	0.5837	0.5376	0.5597
16	612	**0.5874**	0.5461	0.5644
32	**614**	0.5849	**0.5458**	**0.5647**
64	614	0.5849	**0.5458**	0.5647
128	614	0.5849	**0.5458**	0.5647

Table 24: Ergebnis der Parameteruntersuchung amp_fac

Tabelle 24 zeigt, dass mit dem Parameter amp_fac=32 die maximale Anzahl korrekt klassifizierter Trainingsdaten erreicht wird.

amp_fac=32, min_score=10 und damp_fac=0.5 werden für Testdaten verwendet. Und mit diesen drei Parametern wird die Maximalwert Methode [32] verwendet, und daraus wird (-0.98, 0.02) als Grenzwertpunkt bestimmt. Die dazugehörigen neuen positiven Tweets sind 339, neue neutrale Tweets sind 493 und neues negative Tweets sind 178.

Für Testdaten wird (-0.98, 0.02) für die Maximalwert Methode [32] verwendet und $\frac{339}{1010}, \frac{493}{1010}, \frac{178}{1010}$ wird für die Methode der ähnlichen Verteilung verwendet.

6.2.4 MultiExpansion

Hier wurden weitere Expansionen untersucht, wenn von der aktuellen Expansion der Wert 0 zurückgegeben wurde. Die Idee ist: viele Worte aus dem hashtag oder viele falsch geschriebene Worte (wie z.B. *goooooood* statt *good*), haben schon eine deutlich positive oder negative Bedeutung. Wie z.B. *#DFB* ist laut Twitter[11] (das erste Suchergebnis): Die Nationalmannschaft des Deutschen Fußball-Bundes (DFB). Es ist in ende 2014 eindeutig positive gemeint, weil Deutschmanschafft der Fußballweltmeister ist. Kouloumpis et al. haben in einer Untersuchung herausgefunden, dass 14% der Worte aus dem HASH eine positive Bedeutung und 29% eine negative Bedeutung haben [13]. Es wurde auch oft gesehen, dass *gooooood*, *cooooool* und in ähnlicher Weise falsch geschriebene Worte in Tweets benutzt werden. Obwohl die Worte unbekannt sind, wurden sie von den meisten Leuten als Betonung von *good* oder *cool* verstanden, und daher sind die Worte auch positiv. Weil die Positiv- und Negativlisten nicht vollständig sind, sind auch viel bekanntes Worte "unbekannt" (d.h. sie sind in keiner der Listen enthalten). Daher wurde eine weitere Expansion verwendet.

- **multi_expansion_count**

  Eine mehrmalige Expansion (wenn nötig) mit Beschränkung max_count=N. D.h. wenn die ersten N untersuchten Worte aus der Expansion weder in der Positiv- noch in der Negativliste enthalten sind, wurde das erste Wort der Expansion weiter untersucht. Es konnte bis zu M Mal untersucht werden, wenn der Level-Parameter den Wert level=M hat. Das erste Wort aus einer der Expansionen, das im Lexikon enthalten ist (positiv oder negativ) wird auf eine reelle Zahl abgebildet und diese als Wert zurückgeliefert.

$$ExpansionSemt(Wort, level) =$$

$$\begin{cases} \frac{Score(gefundenes\ Wort)}{1000} & falls\ das\ gefundene\ Wort\ in\ Positivliste\ ist \\ -\frac{Score(gefundenes\ Wort)}{1000} & falls\ das\ gefundene\ Wort\ in\ Negativlisteist\ ist \\ ExpansionSemt(Wort_0,\ level-1) & falls\ kein\ Wort\ in\ pos/neg\ gefunden\ und\ level>0 \\ 0 & sonst \end{cases} \quad (22)$$

  Weil Score eine Zahl zwischen 1 und 1000 ist, wurde sie durch 1000 geteilt, damit das endgültige Ergebnis im Intervall [0,1] liegt.

  Es wird der Parameter level mit max_count=5 untersucht. Die Kandidaten für level sind: 1, 2, 3.

  Tabelle 28 zeigt, dass mit dem Parameter level=2 die maximale Anzahl der korrekt klassifizierter Tweets von Trainingsdaten erreicht wird.

  Dann wird der Parameter max_count mit level=2 untersucht. Die Kandidaten für max_count sind: 1, 2, 3, 5, 10, 20.

[11] https://twitter.com/search?q=%23dfb&mode=users

level	available correct count	Precision	Recall	F_1
1	618	0.6127	**0.494**	0.547
2	**620**	0.6224	0.4919	**0.5495**
3	617	**0.6228**	0.482	0.5434

Table 25: Ergebnis der Parameteruntersuchung level

max_count	available correct count	Precision	Recall	F_1
1	612	0.5798	**0.5565**	**0.5679**
2	613	0.5930	0.5224	0.5566
3	613	0.5946	0.5224	0.5573
5	**620**	**0.6224**	0.4919	0.5495
10	618	0.5998	0.5324	0.5641
20	616	0.6127	0.4718	0.5331

Table 26: Ergebnis der Parameteruntersuchung max_count

Tabelle 26 zeigt, dass mit den Parameter max_count=5 die maximale Anzahl korrekt klassifizierter Tweets von Trainingsdaten erreicht wird.

max_count=5 und level=2 werden für Testdaten verwendet. Und mit diesen drei Parametern wird die Maximalwert Methode [32] verwendet, und daraus wird (-1, 0.38) als Grenzwertpunkt bestimmt. Die dazugehörigen neuen positiven Tweets sind 340, neue neutrale Tweets sind 547 und neues negative Tweets sind 123.

Für Testdaten wird (-1, 0.38) für die Maximalwert Methode [32] verwendet und $\frac{340}{1010}, \frac{547}{1010}, \frac{123}{1010}$ wird für die Methode der ähnlichen Verteilung verwendet.

- **multi_expansion_score**

Eine mehrmalige Expansion (wenn nötig) mit Beschränkung des minimalen Scores des Wortes der Expansionslist. Nur die Worte, denen Score größer ist als der Eingabeparameter min_score, werden untersucht. Wenn alle untersuchten Worte der Expansion weder in der Positiv- noch in der Negativliste enthalten sind, wurde das erste gefundene Wort weiter untersucht. Der Parameter level wurde auch hier verwendet. Der Formel für den zurückgelieferten Wert ist:

$$ExpansionSemt(Wort, level) =$$

$$\begin{cases} \frac{Score(gefundenes\ Wort)}{1000} & falls\ das\ gefundene\ Wort\ in\ Positivliste\ ist \\ -\frac{Score(gefundenes\ Wort)}{1000} & falls\ das\ gefundene\ Wort\ in\ Negativlisteist\ ist \\ ExpansionSemt(Wort_0,\ level-1) & falls\ kein\ Wort\ in\ pos/neg\ gefunden\ und\ level > 0 \\ 0 & sonst \end{cases} \quad (22)$$

Es wird der Parameter min_score mit level=3 untersucht. Die Kandidaten für min_score sind: 200, 100, 50, 20, 10, 5.

min_score	available correct count	Precision	Recall	F_1
200	609	0.5904	**0.5257**	**0.5562**
100	613	0.5951	0.5147	0.552
50	614	0.6028	0.5147	0.5553
20	612	0.6025	0.4922	0.5418
10	**618**	**0.6212**	0.4622	0.53
5	618	0.6197	0.4604	0.5283

Table 27: Ergebnis der Parameteruntersuchung min_score

Tabelle 35 zeigt, dass mit dem Parameter min_score=10 die maximale Anzahl der korrekt klassifizierter Tweets von Trainingsdaten erreicht wird. Dann wird der Parameter level mit min_score=10 untersucht. Die Kandidaten für level sind: 1, 2, 3.

level	available correct count	Precision	Recall	F_1
1	616	0.6104	**0.4765**	**0.5352**
2	617	0.6186	0.4625	0.5293
3	**618**	**0.6212**	0.4622	0.53

Table 28: Ergebnis der Parameteruntersuchung level

Tabelle 28 zeigt, dass mit dem Parameter level=3 die maximale Anzahl korrekt klassifizierter Testdaten erreicht wird.

min_score=10 und level=3 werden für Testdaten verwendet. Und mit diesen drei Parametern wird die Maximalwert Methode [32] verwendet, und daraus wird (-1, 0.54) als Grenzwertpunkt bestimmt. Die dazugehörigen neuen positiven Tweets sind 333, neue neutrale Tweets sind 567 und neues negative Tweets sind 110.

Für Testdaten dann wird (-1, 0.54) für die Maximalwert Methode [32] verwendet und $\frac{333}{1010}, \frac{567}{1010}, \frac{110}{1010}$ wird für die Methode der ähnlichen Verteilung verwendet.

6.3 Machine Learning

In diesem Kapitel werden die Algorithmen und die Faeaturesets, die in Weka verwendet wurden, beschrieben.

6.3.1 Methode: SVM und NB

Die Algorithmen SVM und NB wurden mit den Trainingsdaten trainiert. Die dadurch erhaltenen Modelle wurde für Testdaten verwendet.

- SVM (Support Vector Machine)

  SVM ist einer der beliebtesten Algorithmen für die Klassifikation. Er versucht eine Minimierung der empirischen Fehler und Maximierung der Klassengrenzen.

- NB (Naïve Bayesian method)

  NB ordnet jedes Objekt der Klasse zu, zu der es mit der größten Wahrscheinlichkeit gehört. Die Vorteil ist, dass nur eine kleine Menge von Trainingsdaten erforderlich ist, um den Parameter abzuschätzen.

6.3.2 Features

Für die Anwendung von ML werden Features erzeugt, die im Folgenden beschrieben werden. Für jede Expansion wurden die ersten 20 Elemente untersucht, weil die Untersuchung in Kapitel 6.2.2 ergeben hat, dass mit dieser Anzahl von Elementen das beste Ergebnis erreicht wird.

Featureset 1: Die Anzahl von positiven/negativen Worten (Worte, die in der positiven/negativen Liste enthalten sind) eines Tweets

Diese beiden Features fassen die Anzahl der positiven/negativen Worte eines Tweets zusammen. Es wird erwartet, dass die beides Features auch wie ein Lexikon funktionieren.

Featureset 2: Das Verhältnis der Anzahl von positiven/negativen Worten zu der Anzahl der Worte eines Tweets

Diese beiden Features fassen die Durchschnittswerte der positiven/negativen Worte eines Tweets zusammen. Der Wert ist die Anzahl von positiven/negativen Worten geteilt durch die Gesamtanzahl der Worte des Tweets.

Featureset 3: Das Verhältnis von positiven zu negativen Worten eines Tweets

Der Wert des Features ist positiv, wenn die Anzahl von positiven Worten eines Tweets größer ist als die der negativen Worte; negativ, wenn umgekehrt und sonst 0. Wenn die Anzahl der

negativen Worte des Tweets 0 ist, ist der Wert des Features N mal 2. Dabei ist N die Anzahl der positiven Worte des Tweets. Umgekehrt, wenn die Anzahl der positiven Worte 0 ist, ist der Wert des Features N mal -2. Dabei ist N die Anzahl der negativen Worte des Tweets.

Featureset 4: Die Anzahl von Zeichen/Worten eines Tweets

Es wird die Anzahl von Zeichen/Worten eines Tweets aufgelistet.

Featureset 5: Die Anzahl der Sonderzeichen (Fragezeichen, Ausrufezeichen und Punkte) eines Tweets

Es wird die Anzahl von Fragezeichen, Ausrufezeichen und Punkten eines Tweets aufgelistet.

Featureset 6: Das Verhältnis der Gesamtanzahl positiver/negativer Worte aus der Expansion zu allen zu untersuchenden Worten

Der Wert des Features für positive Worte ist die Anzahl von allen positiven Worten aus der Expansion geteilt durch die Anzahl aller Worte des Tweets, der mit der Expansion untersucht wird. Und der Wert des Features für negative Worte ist die Anzahl der Worte des Tweets, der mit der Expansion untersucht wird, mal -1.

Featureset 7: Das Verhältnis der Gesamtanzahl der positiven/negativen Worte aus der Expansion zu allen Worten eines Tweets

Der Wert des Features für positive Worte ist die Anzahl von allen positiven Worten aus der Expansion geteilt durch die Anzahl aller Worte des Tweets. Und der Wert des Features für negative Worte ist die Anzahl aller negativen Worte aus der Expansion geteilt durch die Anzahl aller Worte des Tweets mal -1.

Featureset 8: Das Verhältnis der Summe aller positiven/negativen Scores aus der Expansion zu allen zu untersuchenden Worten

Der Wert des Features für positive Worte ist die Summe der Scores aller positiver Worte aus der Expansion geteilt durch die Anzahl der Worte eines Tweets, welcher mit der Expansion untersucht wird. Und der Wert des Features für negative Worte ist die Summe der Scores aller positiver Worte aus der Expansion geteilt durch die Anzahl der Worte des Tweets, der mit der Expansion untersucht wird, mal -1.

Featureset 9: Das Verhältnis der Summe aller positiven/negativen Scores aus der Expansion zu der Anzahl aller Worte eines Tweets

Der Wert des Features für positive Worte ist die Summe der Scores aller positiven Worte aus der Expansion geteilt durch die Anzahl aller Worte des Tweets. Und der Wert des Features für negative Worte ist die Summe der Scores aller negativen Worte aus der Expansion geteilt durch die Anzahl aller Worte des Tweets mal -1.

Featureset 10: Das Verhältnis der Gesamtanzahl der positiven und negativen Worte aus der Expansion

Der Wert des Features ist positiv, wenn der Gesamtscore aller positiven Worte von allen Expansionen größer ist als der der negativen Worte; negativ, wenn umgekehrt und sonst 0. Es wird immer die größere Zahl durch die kleinere Zahl geteilt. Wenn der Gesamtscore aller negativen Worte von allen Expansionen des Tweets 0 ist, ist der Wert des Features 2 mal der Gesamtscore der positiven Worte von allen Expansionen des Tweets. Umgekehrt, wenn der Gesamtscore der positiven Worte von allen Expansionen des Tweets 0 ist, ist der Wert des Features -2 mal der Gesamtscore aller negativen Worte von allen Expansionen des Tweets.

Featureset 11: Das Verhältnis des gesamten Scores aller positiven und negativen Worte aus allen Expansionen

Der Wert des Features ist positiv, wenn der Gesamtscore aller positiven Worte von allen Expansionen größer ist als der der negativen Worte; negativ, wenn umgekehrt und sonst 0. Es wird immer die größere Zahl durch die kleinere Zahl geteilt. Wenn der Gesamtscore der negativen Worte von allen Expansionen des Tweets 0 ist, ist der Wert des Features 2 mal der Gesamtscore der positiven Worte von allen Expansionen des Tweets. Umgekehrt, wenn der Gesamtscore der positiven Worte von allen Expansionen des Tweets 0 ist, ist der Wert des Features -2 mal der Gesamtscore der negativen Worte von allen Expansionen des Tweets.

Featureset 12: Die top 5 Scores der positiven/negativen Worte aller Expansionen des Tweets

Es werden die fünf größten Scores von positiven/negativen Worten aus allen Expansionen des Tweets aufgelistet.

7 Experiment

In diesem Kapitel werden die Ergebnisse der Experimente zusammengefasst und eine kurze Analyse der Ergebnisse dargestellt.

7.1 Evaluationsmaße

Es werden vier verschiedene Evaluationsmaße für die Methoden und ML verwendet.

- Anzahl der korrekt klassifizierten Tweets

  Es wird die Anzahl der Tweets gezählt, die von der Methode genauso klassifiziert wurden wie in den original Daten.

- Precision

  Eine positive Precision P_{pos} ist die Anzahl der Datensätze, die von der Methode/ML korrekt als positiv klassifiziert wurden, geteilt durch die Gesamtanzahl der positiven Datensätze, die von der Methode klassifiziert wurden.

  Eine negative Precision: P_{neg} ist die Anzahl der Datensätze, die von der Methode/ML korrekt als negativ klassifiziert wurden, geteilt durch die Gesamtanzahl der negativen Datensätze, die von der Methode klassifiziert wurden.

  Der Durchschnitt der Precision für die positive und negative Klassen: $P = (P_{pos} + P_{neg})/2$

- Recall

  Ein positiver Recall R_{pos} ist die Anzahl der Datensätze, die von der Methode/ML korrekt als positiv klassifiziert wurden, geteilt durch die Gesamtanzahl der positiven Datensätze in den original Daten.

  Ein negativer Recall R_{neg} ist die Anzahl der Datensätze, die von der Methode/ML korrekt als negativ klassifiziert wurden, geteilt durch die Gesamtanzahl der negativen Datensätze in den original Daten.

  Der Durchschnitt des Recalls für die positive und negative Klassen: $R = (R_{pos} + R_{neg})/2$

- F_1 (F1-score)

  Ein positiver F1-score F_{pos} ist: $F_{pos} = 2\frac{P_{pos}R_{pos}}{P_{pos}+R_{pos}}$

  Ein negativer F1-score F_{neg} ist: $F_{neg} = 2\frac{P_{neg}R_{neg}}{P_{neg}+R_{neg}}$

  Der Durchschnitt der F1-score für die positive und negative Klassen: $F_1 = (F_{pos} + F_{neg})/2$

Die oben beschriebenen Maße Precision, Recall und F1-score sind die Evaluationsmaße wie sie in den Arbeiten "SemEval-2013 Task 2: Sentiment Analysis in Twitter" [19] und "SemEval-2014 Task 9: Sentiment Analysis in Twitter" [24] verwendet wurden.

7.2 Ergebnis und Analyse der "Lexical-Based" Methode:

Methode	Maximalwert Methode				Festgrenzwertpunkte Methode				Methode der ähnlichen Verteilung			
	c. c.	Pre.	Recall	F_1	c. c.	Pre.	Recall	F_1	c. c.	Pre.	Recall	F_1
list	5071	0.6295	0.4885	0.5501	5071	0.6295	0.4885	0.5501	5071	0.6295	0.4885	0.5501
ac_nred	-	-	-	-	3562	0.4021	**0.5818**	0.4675	-	-	-	-
ac_red	5073	**0.6556**	0.4882	0.5324	5106	0.6284	0.5117	0.5641	**5096**	0.6285	0.5094	0.5627
sl_red	5088	0.6474	0.4773	0.5495	**5108**	**0.6298**	0.5113	0.5644	5080	0.6172	0.5232	0.5663
slog_red	5011	0.6216	0.4980	0.553	5107	0.6297	0.5112	0.5643	5070	0.6177	0.5206	0.565
sk_red	**5098**	0.6457	0.4853	0.5541	**5108**	0.6297	0.5117	**0.5646**	5049	0.6136	**0.5277**	**0.5674**
savg_red	5049	0.6376	0.4664	0.5387	5071	0.6193	0.5121	0.5606	5057	0.6177	0.5085	0.5578
savg_up1_red	5044	0.6348	0.4599	0.5334	5070	0.6188	0.5097	0.559	5031	0.6104	0.5053	0.5529
savg_up2_red	5072	0.6215	**0.5127**	**0.5619**	5106	0.6286	0.5116	0.5641	5077	0.6207	0.518	0.5647
m_count	5072	0.6426	0.4652	0.5397	5103	0.6277	0.5129	0.5645	5065	0.6199	0.506	0.5572
m_score	5065	0.6491	0.4485	0.5305	5105	0.6276	0.5126	0.5643	5025	**0.6329**	0.4578	0.5313

Table 29: Ergebnis der "Lexical-Based" Methoden der Testdaten test_data_all

Methode	Festgrenzwertpunkte Methode			
	c. c.	Pre.	Recall	F_1
list	**2939**	0.593	0.5182	0.5523
ac_nred	2150	0.3986	0.5817	0.4635
ac_red	2936	0.5915	0.5190	0.552
sl_red	**2939**	**0.5933**	0.5198	0.5532
slog_red	2938	0.5932	0.5195	0.5530
sk_red	2938	0.5930	0.52	0.5531
savg_red	2912	0.5809	0.517	0.547
savg_up1_red	2909	0.5813	0.5173	0.5463
savg_up2_red	2936	0.5919	0.52	0.5527
m_count	2937	0.5923	**0.5216**	**0.5539**
m_score	**2939**	0.5923	0.5214	0.5537

Table 30: Ergebnis der "Lexical-Based" Methoden der Testdaten test_data_tweets

Die Verteilung der positiven, negativen und neutralen Tweets in test_data_tweets ist im Vergleich zu train_data sehr unterschiedlich. Daher wurden die Maximalwert Methode und die Methode der ähnlichen Verteilung auf test_data_tweets nicht angewendet.

Die Tabellen 29 und 30 fassen alle Ergebnisse der "Lexical-Based" Methoden zusammen. c.c. ist das Evaluationsmaß der Anzahl der korrekt klassifizierten Tweets; Pre. ist Precsion. Die fettgedruckten Zahlen sind die besten Ergebnisse aller Methoden mit denselben Grenzwertparametern.

Tabelle 29 zeigt, dass mit der Festgrenzwertpunkte Methode eine allgemein höhere Anzahl korrekt klassifizierter Tweets erreicht wird als mit den anderen beiden Methoden. Die beiden Tabellen (29 und 30) zeigen, dass mit den meisten Methoden (mit Ausnahme von $savg\_red$ und $savg\_up1\_red$) mit Expansion, die einen verkleinerten Wert zurückliefern, eine Verbesserung F_1 gegenüber den Methoden ohne Expansion erreicht wird. Von allen Methoden wurde mit sk_red und der Methode der ähnlichen Verteilung der beste Wert für F_1 erreicht. Daraus wird ersichtlich, dass Anzahl und Score der positiven/negativen Worte aus der Expansion wichtige Werte für die untersuchten Worte sind.

Die Methoden mit Expansion liefern gute Ergebnisse für Tweets, die falsch geschriebene Worte enthalten, aber schlechte Ergebnisse für Tweets, in denen ein hashtag eine wichtige Rolle für die Bedeutung spielt.

Beispiel:

In allen Methoden wird dem Wort *goooooood* ein positiver Wert zugewiesen, aber das Wort *#hi* bekommt immer null zugewiesen, obwohl das Wort eine tendenziell positive Bedeutung hat.

Methode	pos->neu	neu->pos	pos->neg	neg->pos	neu->neg	neg->neu
list	1137	541	187	167	639	274
ac_red	1107	522	200	161	589	331
sl_red	1109	521	200	157	593	328
slog_red	1110	521	200	157	593	328
sk_red	1110	520	201	157	591	329
savg_red	1077	**584**	196	**175**	590	323
savg_up1_red	1097	536	**211**	163	592	347
savg_up2_red	1108	522	200	159	590	331
multi_expansion_count	1105	530	200	161	586	331
multi_expansion_score	1105	527	202	162	585	330

Table 31: Ergebnis der Anzahl der falsch klassifizierten Tweets der Testdaten test_data_all

Methode	pos->neu	neu->pos	pos->neg	neg->pos	neu->neg	neg->neu
list	742	397	136	93	278	208
ac_red	741	399	137	95	275	210
sl_red	742	397	136	92	277	210
slog_red	743	397	136	92	277	210
sk_red	743	397	137	93	275	210
savg_red	715	**450**	130	**110**	273	203
savg_up1_red	735	409	**144**	95	277	224
savg_up2_red	741	398	136	93	276	213
multi_expansion_count	738	404	135	94	274	221
multi_expansion_score	737	402	137	94	274	210

Table 32: Ergebnis der Anzahl der falsch klassifizierten Tweets der Testdaten test_data_tweets

In den Tabellen 31 und 32 sind "a->b" die Gesamtanzahl der als b klassifizierten Tweets, die aber das Label a haben. Beispiel: "neu->pos" ist die Gesamtanzahl der als positv klassifizierten Tweets, die aber das Label neutral haben.

Tabelle 31 und 32 zeigen, dass die meisten Methoden mit Expansion "pos->neu" und "neu->neg" besser klassifiziert haben. Wegen des zu großen Rauschens bei den Methoden *savg_red* (zu viel "neu->pos" und "neg->pos") und *savg_up1_red* (zu viel "pos->neg") wurde nur ein geringfügig besseres Klassifizierungsergebnis erreicht. Dadurch wird auch gezeigt, dass die Anzahl der positiven/negativen Worte aus Expansion ein große Rolle spielt. Deswegen kann mit den Methoden *savg_red* und *savg_up1_red* kaum ein verbessertes Klassifizierungsergebnis erreicht werden. Trotz Rauschen wurde bei den Methoden mit Expansion ein besseres Klassifizierungsergebnis erreicht.

Im Folgenden werden einige Beispiele für schwierig zu klassifizierende Tweets gezeigt:

- Der Tweet könnte richtig klassifiziert werden, wenn die Methode besser wäre.

  I do not know if I can, I have to work on Sunday ...

  Der Tweet ist negativ gelabelt, aber wird von der Methode `ac_red` als positiv klassifiziert, weil das Wort "work" in der Positivliste enthalten ist. Wenn die Methode nicht nur einzelne Worte sonderen auch Wortpaare bearbeiten könnte, würde der Tweet wahrscheinlich negativ klassifiziert werden. Z.B. könnte "have to" dann als Negierungswort verwendet werden.

- Der Tweet kann richtig klassifiziert werden, aber Balance ist schwer.

  hey u wake up late then u leave ?

  Der Tweet ist negativ gelabelt, aber wird von der Methode `ac_red` als neutral klassifiziert, weil Sem(Tweet) den Wert -0.28 hat. Wenn der Grenzwertpunkt zwischen den Klassen negativ und neutral statt -0.5 den Wert -0.2 hat, wird der Tweet richtig klassifiziert werden. Aber dann würden viele andere Tweets falsch klassifiziert.

- Der Tweet kann nicht richtig klassifiziert werden.

  Next time i \u2019ll get a king size bed... Haha

  Der Tweet ist positiv gelabelt, aber wird von der Methode `ac_red` als neutral klassifiziert, weil Sem(Tweet) den Wert -0.22 hat. Durch Expansion wird für das Wort "Haha" ein negativer Wert ermittelt, weil aus der Expansion "Haha" ähnlich zu "Funny" ist, und "Funny" in der Negativliste enthalten ist.

- Der Tweet wird richtig klassifiziert.

  Daylight Savings Time is Sunday folks.. and we 'fall' back.. maybe that will give some of you lethargic losers a reason to be early to work. ?

  Der Tweet ist negativ gelabelt, und wird von der Methode `ac_red` als negativ klassifiziert, weil Sem(Tweet) den Wert -0.52 hat.

  Es ist zwar keines der Worte des Tweets im Lexikon enthalten, aber mit Hilfe von Expansion wird durch die Summe von vielen kleinen negativen Werten der Gesamtwert von -0.52 erreicht.

- gold nicht sicher

  i will give u a call! ok? :)

  Der Tweet ist neutral gelabelt, und wird von der Methode `ac_red` als positiv klassifiziert, weil Sem(Tweet) den Wert 0.85 hat. Der Tweet ist zwar falsch klassifiziert, kann aber meiner Meinung nach auch eine positive Bedeutung haben.

7.3 Ergebnis der Analyse von Machine Learning :

Featureset	c.c. (svm)	P (svm)	Recall	F_1 (svm)	c.c. (nb)	P (nb)	Recall	F_1 (nb)
Wortfeature	4561	0.554	0.3428	0.4235	4014	0.4555	0.2074	0.285
set 1	5059	0.6355	0.448	0.5255	4637	0.603	0.3366	0.432
set 1+2	5122	0.641	0.4725	0.544	4974	0.6105	**0.4706**	0.5315
set 1+3	5063	0.6365	0.4489	0.5265	4727	**0.6305**	0.3565	0.4555
set 1+4	5043	0.6305	0.4468	0.523	4637	0.597	0.3415	0.4345
set 1+5	5108	0.6380	0.4511	0.5285	4561	0.5725	0.3653	0.446
set 1+6	5052	0.639	0.4477	0.5265	4652	0.5898	0.3657	0.4515
set 1+7	5107	**0.6485**	0.4619	0.5395	4669	0.578	0.3793	0.458
set 1+8	5057	0.6365	0.4453	0.524	4580	0.5785	0.3466	0.4335
set 1+9	5054	0.630	0.4537	0.5275	4594	0.5675	0.366	0.445
set 1+10+11	5060	0.638	0.4496	0.5275	4565	0.5805	0.3365	0.426
set 1+12	5071	0.638	0.4482	0.5265	4421	0.552	0.3221	0.406
set 1+2+3 +5+6+7+10+11+12	5185	0.6435	0.4848	0.553	4772	0.5935	0.4223	0.4935
set 1+2+3 +5+7+12	**5189**	0.643	0.4874	0.5545	**4827**	0.5985	0.4286	**0.4995**
alle Featuresets	5184	0.643	**0.489**	**0.555**	4709	0.5845	0.4365	0.4825

Table 33: Ergebnis von Weka mit verschiedenen Featuresets der Testdaten test_data_all

Featureset	c.c. (svm)	P (svm)	Recall	F_1 (svm)	c.c. (nb)	P (nb)	Recall	F_1 (nb)
Wortfeature	2714	0.5445	0.3635	0.436	2355	0.455	0.2455	0.3155
set 1	3003	0.6085	0.492	0.5435	2703	0.5775	0.3795	0.4565
set 1+2	3017	0.610	0.495	0.5465	2756	**0.6055**	0.4005	0.478
set 1+3	3003	0.604	0.4985	0.549	2847	0.584	**0.48**	**0.519**
set 1+4	3010	0.609	0.4935	0.545	2717	0.577	0.399	0.468
set 1+5	3015	0.6145	0.497	0.55	2740	0.574	0.4045	0.47
set 1+6	3005	0.613	0.4955	0.548	2729	0.58	0.4055	0.4745
set 1+7	3013	**0.615**	0.489	0.545	2734	0.569	0.3955	0.463
set 1+8	3004	0.6085	0.493	0.5445	2673	0.5595	0.378	0.4485
set 1+9	3010	0.6125	0.4925	0.546	2704	0.567	0.382	0.4525
set 1+10+11	3010	0.613	0.4945	0.547	2688	0.5675	0.3735	0.4485
set 1+12	3006	0.6105	0.495	0.5465	2564	0.5295	0.3525	0.4095
set 1+2+3 +5+6+7+10+12	3023	0.6115	0.4995	0.55	2748	0.5745	0.4165	0.4695
set 1+2+3 +5+7+12	**3034**	0.6135	**0.5025**	**0.552**	**2776**	0.578	0.4215	0.4745
alle Featuresets	3028	0.6095	**0.5025**	0.5505	2710	0.569	0.4025	0.4575

Table 34: Ergebnis von Weka mit verschiedenen Featuresets der Testdaten test_data_tweets

Die Tabellen 33 und 34 zeigen, dass Featureset 1 wie die "Lexical-Based" Methode funktioniert und die Featuresets 6-12, die aus einer Expansion erzeugt werden, wie die Expansionsmethode funktionieren. Allgemein ist das Ergebnis der Klassifizierung mit der Festgrenzwert Methode besser als das der anderen beiden. Wie Tabelle 33 zeigt, wird mit der Kombination der Featuresets (1+2+3+5+7+12) das beste Klassifizierungsergebnis erreicht. Mit allen anderen Featuresets wird ein besseres Klassifizierungsergebnis erreicht als nur mit dem Featureset Wordfeature. Mit den Kombinationen von Featuresets 1+6, 1+7, 1+9, ..., 1+12 wird ein besseres Klassifizierungsergebnis erreicht. Daraus wird ersichtlich, dass mit Hilfe einer Expansion der F_1 im Allegemeinen erhöht werden kann. Mit den Kombinationen von Featuresets 1+7, 1+8 kann das Klassifizierungsergebnis nicht viel gegenüber dem Featureset 1 verbessert werden. Daraus wird ersichtlich, dass Anzahl und Score der positiven/negativen Worte aus der Expansion wichtige Werte für die untersuchten Worte sind.

Die ”Lexical-Based” Methode verwendet nur dev_data als Trainingsdaten und diese enthalten lediglich 1010 Tweets, während ML train_data als Trainingsdaten verwendet, welche 6314 Tweets enthalten. Wenn man das berücksichtigt, hat die ”Lexical-Based” Methode eine bessere Performance als ML.

7.4 Ergebnis im Vergleich zu existierenden Arbeiten

	F_1
bestes Ergebnis aus dieser Arbeit	56.46%
Durchschnitt der Ergebnisse aus SemEval2013 [19]	53.7%
Majority Baseline	29.19%
bestes Ergebnis aus SemEval2013 [19]	69.02%

Table 35: Ergebnis im Vergleich zu existierenden Arbeiten

Im Vergleich zu den Ergebnissen des Workshops SemEval 2013 ist der beste F_1 Wert dieser Arbeit besser als der Durchschnitt der Ergebnisse und liegt etwa bei Platz 16.

Die ”Majority Baseline” [18] ist das Ergebnis eines Mehrheits-Klassifizierers. Das beste Ergebnis von SemEval 2013 wurde von der Gruppe *Nrc Canada* erzielt. Sie verwendeten ein eigenes Lexikon *NRC Hashtag Sentiment Lexicon* [18] mit Emoticons und hashtags. Sie benutzten eine Vielzahl von ”surface-form”, ”semantic” und ”sentiment” Features. Die Klassifikation erfolgte mit Hilfe von SVM.

8 Ausblick

Für zukünftige Arbeiten wäre es hilfreich, wenn JoBimText einen dynamische Score anbieten könnte. D.h. Abhängig von der Beziehung des Wortes und der Position des Wortes wird eine jeweils unterschiedliche Expansionslist für dasselbe gesuchte Worte zurückgegeben.

Beispiel: Die Expansionlist für das Wort *cool* aus dem Datensatz *it's cool outside* soll sich von der Expansionlist für das Wort *cool* aus dem Datensatz *he is cool* unterscheiden.

Es wäre auch hilfreich, wenn JoBimText nicht nur Worte sondern Wortpaare untersuchen könnte. Dann könnte man mit JoBimText auch ein N-gramm für die Expansion verwenden.

Es könnten das Thema einer zukünftigen Arbeit sein, wie der Einfluss des Rauschens auf die Sentimentanalyse verringert, bzw. der Einfluss von "wichtigen" Worten vergrößert werden kann. Unter Rauschen versteht man ein Wort, das keine Rolle für die Bedeutung des Datensatzes spielen sollte aber trotzdem einen Einfluss hat. Dabei sind "wichtige" Worte solche Worte, die einen grossen Einfluss auf die Bedeutung eines Satzes haben.

Beispiel: Die Expansionlist des untersuchten Wortes *goooooood* enthält den Score "45" für Jo *good*. Es wäre sinnvoll, wenn das Wort *good* einen viel höheren Score erhalten würde und nicht nur denselben Score wie für Jo *goooooood* aus der Expansionlist des untersuchten Wortes *good*.

References

[1] Marco Baroni and Alessandro Lenci. Distributional memory: A general framework for corpus-based semantics. Computational Linguistics, 36(4):673–721, 2010.

[2] Chris Biemann and Martin Riedl. Text: Now in 2d! a framework for lexical expansion with contextual similarity. Journal of Language Modelling, 1(1):55–95, 2013.

[3] Erik Boiy, Pieter Hens, Koen Deschacht, and Marie-Francine Moens. Automatic sentiment analysis in on-line text. In ELPUB, pages 349–360, 2007.

[4] Erik Boiy and Marie-Francine Moens. A machine learning approach to sentiment analysis in multilingual web texts. Information retrieval, 12(5):526–558, 2009.

[5] Marc Cheong and Vincent Lee. Dissecting twitter: A review on current microblogging research and lessons from related fields. From Sociology to Computing in Social Networks, 1:343–362, 2010.

[6] Yejin Choi and Claire Cardie. Adapting a polarity lexicon using integer linear programming for domain-specific sentiment classification. Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, 2:590–598, 2009.

[7] Johannes C Eichstaedt, Hansen Andrew Schwartz, Margaret L Kern, Gregory Park, Darwin R Labarthe, Raina M Merchant, Sneha Jha, Megha Agrawal, Lukasz A Dziurzynski, Maarten Sap, et al. Psychological language on twitter predicts county-level heart disease mortality. Psychological Science, 1:1–11, 2015.

[8] Jeffrey Ellen. All about microtext-a working definition and a survey of current microtext research within artificial intelligence and natural language processing. ICAART (1), 11:329–336, 2011.

[9] Peter Gehler and Sebastian Nowozin. On feature combination for multiclass object classification. In Computer Vision, 2009 IEEE 12th International Conference on, pages 221–228. IEEE, 2009.

[10] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, pages 168–177, New York, USA, 2004. ACM.

[11] Soo-Min Kim and Eduard Hovy. Determining the sentiment of opinions. In Proceedings of the 20th international conference on Computational Linguistics, pages 1367–1373, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics.

[12] Moshe Koppel and Jonathan Schler. The importance of neutral examples for learning sentiment. Computational Intelligence, 22(2):100–109, 2006.

[13] Efthymios Kouloumpis, Theresa Wilson, and Johanna Moore. Twitter sentiment analysis: The good the bad and the omg! ICWSM, 11:538–541, 2011.

[14] Akshi Kumar and Teeja Mary Sebastian. Sentiment analysis: A perspective on its past, present and future. International Journal of Intelligent Systems and Applications (IJISA), 4(10):1–14, 2012.

[15] Alessandro Lenci. Distributional semantics in linguistic and cognitive research. Rivista di lingüística, 20(1):1–32, 2008.

[16] Yuqing Li, Xin Li, Fan Li, and Xiaofeng Zhang. A lexicon-based multi-class semantic orientation analysis for microblogs. In Web Technologies and Applications, pages 81–92. Springer, 2014.

[17] Dekang Lin. Automatic retrieval and clustering of similar words. In Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, pages 768–774, Montreal,Canada, 1998. Association for Computational Linguistics.

[18] Saif M Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Seventh International Workshop on Semantic Evaluation (SemEval 2013), pages 312–320, 2013.

[19] Preslav Nakov, Zornitsa Kozareva, Alan Ritter, Sara Rosenthal, Veselin Stoyanov, and Theresa Wilson. Semeval-2013 task 9: Sentiment analysis in twitter. Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Seventh International Workshop on Semantic Evaluation (SemEval 2013), pages 312–320, 2013.

[20] Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In LREC, pages 1320–1326, 2010.

[21] Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In Proceedings of the 42nd annual meeting on Association for Computational Linguistics, page 271, Ithaca, New York, 2004. Association for Computational Linguistics.

[22] Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. Foundations and trends in information retrieval, 2(1-2):1–135, 2008.

[23] Anthony Ralston and Chester L Meek. Encyclopedia of computer science. New York: Van Nostrand Reinhold Company, 1976, edited by Ralston, Anthony; Meek, Chester L., 1, 1976.

[24] Sara Rosenthal, Preslav Nakov, Alan Ritter, and Veselin Stoyanov. Semeval-2014 task 9: Sentiment analysis in twitter. SemEval 2014, pages 73–80, 2014.

[25] Hassan Saif, Yulan He, and Harith Alani. Semantic sentiment analysis of twitter. In The Semantic Web–ISWC 2012, pages 508–524. Springer, 2012.

[26] Joel N Shurkin. Engines of the mind: the evolution of the computer from mainframes to microprocessors. WW Norton & Company, 1996.

[27] Bharath Sriram, Dave Fuhry, Engin Demir, Hakan Ferhatosmanoglu, and Murat Demirbas. Short text classification in twitter to improve information filtering. In Proceedings of the 33rd

international ACM SIGIR conference on Research and development in information retrieval, pages 841–842. ACM, 2010.

[28] Maite Taboada, Caroline Anthony, and Kimberly Voll. Methods for creating semantic orientation dictionaries. In Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC), pages 427–432, Genova, Italy, 2006.

[29] Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. Lexicon-based methods for sentiment analysis. Computational linguistics, 37(2):267–307, 2011.

[30] Jiliang Tang, Salem Alelyani, and Huan Liu. Feature selection for classification: A review. Data Classification: Algorithms and Applications. Editor: Charu Aggarwal, CRC Press, 2013.

[31] Peter D Turney, Patrick Pantel, et al. From frequency to meaning: Vector space models of semantics. Journal of artificial intelligence research, 37(1):141–188, 2010.

[32] Hai-peng WANG, Lin SHANG, Xin-yu DAI, and Yang-sheng JI. Getting turning point for text sentiment. Journal of Jiangnan University (Natural Science Edition), 5:001, 2009.

[33] Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In Proceedings of the conference on human language technology and empirical methods in natural language processing, pages 347–354, Vancouver, B.C., Canada, 2005. Association for Computational Linguistics.

[34] Jianxin Yao, Gengfeng Wu, Jian Liu, and Yu Zheng. Using bilingual lexicon to judge sentiment orientation of chinese words. In Computer and Information Technology, 2006. CIT'06. The Sixth IEEE International Conference on, pages 38–38, Seoul, Korea, 2006. IEEE.

[35] Tianfang Yao, Xiwen Cheng, Feiyu Xu, Hans Uszkoreit, and Rui Wang. A survey of opinion mining for texts. Journal of Chinese information processing, 22(3):71–80, 2008.

[36] Yan-Yan ZHAO, Bing Qin, and Ting Liu. Sentiment analysis. Journal of Software, 21(8):1834–1848, 2010.

A Geändertes Lexikon Version 1: pn_new

Alle Worte, die neu in die Positivliste von pn (Lexikon Version 1) eingefügt wurden:

lol	:')	:p

Alle Worte, die aus der Positivliste von pn (Lexikon Version 1) gelöscht wurden:

advanced	celebration	champion	classic	clear	comfortable	divine
fair	famous	free	freedom	friendly	golden	grace
hero	holy	like	pride	ready	rich	right
safe	strong	top	well	willing	winners	wonder

Alle Worte, die neu in die Negativliste von pn (Lexikon Version 1) eingefügt wurden:

:'(

Alle Worte, die aus der Negativliste von pn (Lexikon Version 1) gelöscht wurden:

absence	abuse	adamant	anarchy	bomb	break	broke
careless	chill	clash	cloud	cold	crazy	criminal
crisis	death	demon	desert	desperate	desperation	din
dirty	drunk	dungeon	elimination	fall	fat	fell
filthy	fist	foul	gross	hard	harden	injury
mar	miss	murder	mysterious	odd	rampage	retreat
rocky	rumor	rumors	rumours	sag	savage	slow
smash	stern	tentatively	trick	vengeance	vice	wild
hole						

B Geändertes Lexikon Version 2: pnn_new

Die Worte, die neu in die Positivliste von pnn (Lexikon Version 2) eingefügt wurden, sind alle Emoticons aus der Positivliste von pn (Lexikon Version 1) sowie folgende Worte:

lol	loves	win	xd

Alle Worte, die aus der Positivliste von pnn (Lexikon Version 2) gelöscht wurden:

advanced	back	brook	bull	celebration	champion	charity
classic	clear	comfortable	court	dance	deal	deserve
dig	divine	drive	fair	famous	frank	free
freedom	friend	friendly	friends	golden	grace	haven
herald	hero	holy	hope	hopes	independence	independent
joke	kind	light	like	live	minister	moving
normal	open	original	press	pride	pro	prospect
ready	real	reason	repair	revolution	rich	right
safe	sense	sound	special	spirit	star	stars
straight	strong	top	treat	tribute	true	ultimate
vow	well	white	wide	willing	winners	wonder

Die Worte, die neu in die Negativliste von pnn (Lexikon Version 2) eingefügt wurden, sind alle Emoticons aus der Negativliste von pn (Lexikon Version 1) sowie folgende Worte:

fuck	fucking	shit

Alle Worte, die aus der Negativliste von pnn (Lexikon Version 2) gelöscht wurden:

absence	abuse	adamant	against	airs	anarchy	avoid	bar
battle	black	block	blood	bomb	break	broke	busy
careless	challenge	chill	clash	close	cloud	cold	comedy
crazy	criminal	crisis	cross	cut	death	deep	defeat
demon	desert	desperate	desperation	dig	din	dirty	disease
down	drunk	dungeon	elimination	fall	fat	fell	fight
filthy	fist	flash	force	forget	foul	gaga	gamble
game	gross	hard	harden	hole	horror	ill	injury
kick	lax	lecture	little	long	mar	miss	moon
murder	mysterious	odd	rampage	retreat	rocky	rumor	rumors
rumours	sag	savage	slow	smash	spot	stake	stern
storm	tentatively	trial	trick	vengeance	vice	war	wild