FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Analyse der Netzwerkstrukturen sowie Erweiterung eines Wörterbuchs zur automatisierten Identifikation von Hass-Tweets

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Software Engineering & Internet Computing

eingereicht von

Klaus Walla, B.Sc.
Matrikelnummer 1028877

an der Fakultät für Informatik
der Technischen Universität Wien
Betreuung: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Andreas Rauber, Univ. Doz.

Wien, 8. November 2016

Klaus Walla	Andreas Rauber

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Analysis of network structures and enlargement of a dictionary for the automated identification of Hate Tweets

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Software Engineering & Internet Computing

by

Klaus Walla, B.Sc.
Registration Number 1028877

to the Faculty of Informatics

at the TU Wien

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Andreas Rauber, Univ. Doz.

Vienna, 8th November, 2016

Klaus Walla

Andreas Rauber

Erklärung zur Verfassung der Arbeit

Klaus Walla, B.Sc.
Schimmelgasse 23/3/15, 1030 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 8. November 2016

Klaus Walla

Danksagung

An dieser Stelle möchte ich mich bei allen Personen, die mich in den verschiedensten Situationen auf unterschiedlichste Weise unterstützt haben, sei es direkt im Bezug auf das Verfassen der Diplomarbeit oder durch deren indirekten Einfluss mit Hilfe von motivierenden Worten oder Taten jeglicher Art, bedanken.
Besonderer Dank gilt meinem Betreuer Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Andreas Rauber, Univ. Doz. der mit seinen Ideen und Anregungen die Diplomarbeit vorantrieb und in allen Phasen unterstützend zur Seite stand. Weiterer Dank gilt Dr. Markus Wolf der die deutsche Version des LIWC-Wörterbuchs für meine durchgeführten Experimente frei zur Verfügung stellte. Bedanken möchte ich mich auch bei Andreas Riegler, BSc einem Studienkollegen und vor allem Freund ohne dem ich das Studium nicht mit solcher Freude und in der kurzen Zeit hätte bewältigen können.
Die vorliegende Arbeit widme ich meiner gesamten Familie, welche mir dieses Studium ermöglicht hat beziehungsweise zu jeder Zeit den Rücken gestärkt und freigehalten hat, um den gesamten Fokus darauf legen zu können.

Kurzfassung

In den letzten Jahren rückte Hassrede speziell durch deren einfache und anonyme Verbreitung über soziale Netzwerke immer mehr in das Blickfeld der Gesellschaft und stellt mittlerweile ein nicht zu verachtendes Problem dar. Speziell in einem sozialen Medium, wie Twitter, können die großen Mengen an Tweets nur unzureichend auf herabwürdigende Inhalte untersucht werden, um entsprechend darauf zu reagieren.
Deshalb soll in dieser Arbeit ein Ansatz beruhend auf Supervised Machine Learning vorgestellt werden, der Hasspostings automatisch als solche identifiziert. Damit dies bewerkstelligt werden kann, wurden Features eingesetzt, die sich bereits in vorangegangen Arbeiten für die Erkennung von offensiven Äußerungen bewährten und Eigenheiten der verwendeten Sprache und des Tweet-Inhalts berücksichtigen. Zusätzlich wurde spezielles Augenmerk auf Features gelegt, die durch die Analyse der Netzwerkstruktur und durch den Einsatz eines für die Hassidentifikation angepasstes Wörterbuch gewonnen werden können. Letztendlich wird mit den resultierenden Features ein Modell eines Klassifikators trainiert, welcher den Tweet als neutral oder hasserfüllt einstuft. Zum Einsatz kamen dabei ein Support Vector Machine-, ein Naive Bayes- und ein Random Decision Forest-Klassifikator.
Zur Evaluierung der Performance des Machine Learning Algorithmus wurden verschiedene Experimente durchgeführt, die Aufschluss darüber geben sollen wie sich Features und dessen Kombinationen auf die Exaktheit der Klassifikationen auswirkt, wie gut die Ergebnisse der jeweiligen Klassifikatoren ausfallen und wie die Parameter dieser angepasst werden müssen, um die Resultate weiter zu optimieren. Auf Grundlage der kalkulierten Messwerte soll als Endergebnis dieser Arbeit jene Kombination aus Feature-Set und Klassifikator mit dessen Parametereinstellungen präsentiert werden, von der man sich die beste Identifikation von Hasspostings verspricht.

Abstract

In recent years, hate speech moved specially due to their simple and anonymous distribution through social networks more and more into the focus of the society and is now constituting a non-negligible problem. Especially in a social medium like Twitter, the large amounts of posts can only be inadequately investigated for derogatory content or offensive language in order to react accordingly.
Therefore an approach, based on supervised machine learning, is presented in this work, which identifies hate postings automatically. For this to be accomplished, features already proven in previous works for the recognition of offensive remarks, which consider characteristics of the used language and the Tweet content, were included. In addition, special attention was paid to features that can be gained by analyzing the network structure and the use of a dictionary customized for the hate identification. Finally, the model of a classifier is trained with the resulting features, which classifies a Tweet as neutral or hateful. In detail a Support Vector Machine-, Naive Bayes and Random Forest- classifier came to use.
To evaluate the performance of the machine learning algorithmus different experiments were carried out, which should give information about how features and its combinations affect the accuracy of the classifications, as well the respective classifiers perform and how the parameters of these have to be adjusted to optimize the results further. Based on the calculated values the combination of feature set and classifier with its optimal parameter settings, from which one expects the best identification of hate postings, is presented as the final result of this work.

Inhaltsverzeichnis

KAPITEL 1

Einführung

1.1 Motivation

Aufgrund des aktuellen Krieges im Nahen Osten und dem daraus resultierenden Asylstromes nach Europa, welcher viele Regierungen vor große Probleme stellt, wurde ein Großteil der Bevölkerung verunsichert, entwickelte Ängste und schürte in weiterer Folge in vielen Fällen auch Hass gegenüber den Flüchtlingen. Diesem Hass wurde gerade in sozialen Netzwerken freier Lauf gelassen und nebenbei scheinbar unzählige rechtsextreme oder allgemein im Bezug auf Asylanten negativ eingestellte Seiten gegründet, welche diesen Hass und den Unmut weiter förderten. Filterung dieser einschlägigen Seiten und Beiträge fand seitens der sozialen Netze maximal im geringen Ausmaß und manuell statt. Diese Tatsachen regten an hasserfüllte Postings automatisch zu erkennen und waren Motivation dafür einen solchen Machine Learning Algorithmus auch für die deutsche Sprache und die Eigenheiten von Tweets zu entwickeln und auf dessen Effektivität zu evaluieren.

1.2 Problemstellung

Allgemein versteht man unter Hassrede (engl. Hate Speech) die Verwendung von Ausdrücken, die zu Hass gegen Personen oder Gruppen aufstacheln, die zu Gewalt- oder Willkürmaßnahmen gegen sie auffordern oder welche die Menschenwürde anderer angreift. Sie unterscheidet sich gegenüber der Beleidigung dadurch, dass jemand nicht als Individuum sondern über die Angehörigkeit zu einer Gruppe herabgewürdigt wird[1].
Hassrede im Internet wurde mit der Entstehung sozialer Netzwerke wie Twitter, Facebook und Youtube und dem darauffolgenden Anstieg an Beliebtheit dieser allgegenwärtig,

[1] http://www.amadeu-antonio-stiftung.de/hatespeech/was-ist-ueberhaupt-hate-speech/ - Amadeu Antonio Stiftung, 2016-05-30

stieg in der Anzahl und erreicht daher eine große Menge an Personen. Speziell nach Straftaten, durchgeführt von Personen mit anderen Nationalitäten, oder Ereignisse wie dem derzeitigen Asylchaos reagieren aufgebrachte Menschen oft mit Hass, welchem oft in sozialen Netzwerken freier Lauf gelassen wird. Ein Grund dafür ist sicherlich dem Umstand geschuldet, dass von vielen das World Wide Web als rechtsfreier Raum verkannt wird und sich Hetzer deshalb unangreifbar fühlen. Es ist daher auch keine Seltenheit, dass Leute ihre negativen Gedanken auf Facebook und Twitter, anders als in kleineren Foren, unter ihrem bürgerlichen Namen (Klarnamen) verbreiten. Tatsächlich erfüllen solche virtuellen Handlungen nicht nur strafrechtliche Tatbestände, wie Verhetzung, Beleidigung oder der "Öffentlichen Aufforderung zu (terroristischen) Straftaten", sondern können auch zivilrechtliche Konsequenzen, wie den Arbeitsplatzverlust nach sich ziehen[2].

In den letzten Jahren entwickelte sich Hassrede dadurch zu einem bedeutenden Problem der Gesellschaft, vor dem weder die Regierungen, noch die Verantwortlichen der sozialen Netze die Augen verschließen konnten, da Hassbotschaften für jeden Einzelnen allgegenwärtig wurden. Österreich ging vor allen Dingen auf rechtlicher Ebene mit der Verschärfung des Verhetzungsparagraphen (§ 283 StGB), welcher seit 1.1.2016 in Kraft ist, gegen diese hasserfüllten Botschaften vor.[1] Auf EU-Ebene sind Facebook, Twitter und Google nach der E-Commerce-Richtlinie dazu verpflichtet, strafbare Inhalte auf ihren Seiten zu löschen und wurden daher angehalten ihr Löschverhalten und ihre Richtlinien dahingehend zu verbessern[3]. Tatsächlich überarbeiteten Betreiber von sozialen Netzwerken kürzlich ihre Richtlinien im Bezug auf Hassrede und verkündeten dagegen aktiver vorzugehen. Im Detail kündigte Twitter an, mehr Konten zu blockieren oder Nutzer zur Verifizierung mit einer Telefonnummer aufzufordern, wenn diese Hassbeiträge verbreiten beziehungsweise anschließend nicht entfernen[4]. Facebook geht dabei sogar einen Schritt weiter, installierte eine Löschtruppe für den deutschen Raum und investierte rund eine Millionen US-Dollar in Maßnahmen zur Förderung von Gegenrede[5,6].
Die große Menge an täglich produzierten Daten (z.B.: rund 500 Mio. Tweets täglich-Stand Nov. 2015[7]) macht es für diese manuellen Vorgehensweisen jedoch unmöglich alle Hassbeiträge in akzeptabler Zeit zu erkennen und zu entfernen. Die automatische, effiziente beziehungsweise möglichst korrekte Identifizierung von Hassbeiträgen würde daher eine enorme Aufwandsersparnis bedeuten und wäre mit darauffolgender manueller

[2] https://ggr-law.com/persoenlichkeitsrecht/faq/kommentar-falsch-social-media-internet-strafanzeige-abmahnung/ - Gulden Röttger Rechtsanwälte, 2016-05-30

[3] http://derstandard.at/2000026874533/EU-Justizminister-sagen-Hasspostings-im-Internet-den-Kampf-an - Der Standard, 2016-05-30

[4] http://www.n24.de/n24/Nachrichten/Politik/d/7837714/twitter-verspricht-haerteres-vorgehen-gegen-hass-posts.html - N24, 2016-05-30

[5] http://diepresse.com/home/techscience/internet/4907295/Facebook-praesentiert-Initiative-gegen-Hasspostings - Die Presse, 2016-05-30

[6] https://www.tagesschau.de/inland/facebook-hasspostings-loeschen-101.html - Tagesschau, 2016-05-30

[7] http://socialmedia-institute.com/uebersicht-aktueller-social-media-nutzerzahlen/ - Socialmedia Institute, 2016-05-30

Überprüfung und allfälliger Entfernung der wahrscheinlich beste Ansatz, um diesen unerwünschten Botschaften entgegenzuwirken.
Da es aber für deutschsprachige Hasspostings noch keine Forschungen im Bezug auf deren automatische Erkennung gibt, soll mit dieser Diplomarbeit eine Methode für Tweets entwickelt werden, die dies mithilfe von Techniken aus dem Bereich Machine Learning und Information Retrieval ermöglicht. Als zusätzliche Neuerung soll ein Netzwerkfeature entworfen werden, welches die Twitter-Architektur und den resultierenden Netzwerkgraphen nutzt, um so die Einstellung und Interessen von Nutzern gegenüber polarisierender Twitter-Nutzer und Seiten, welche Ausgangspunkte beziehungsweise Brennpunkte von Ablehnung, Hass und in weiterer Folge Hasspostings sind, zu bestimmen. Weiters sollen bereits bekannte Technologien und Eigenschaften von Twitter, so angepasst beziehungsweise genutzt werden, dass sie für den Bereich der Hasserkennung optimal eingesetzt werden. Unter Hassposting werden im Kontext dieser Arbeit Beiträge, in denen Hassrede und/oder Beleidigungen, also Hass gegenüber Individuen, vorkommt, verstanden.

1.3 Forschungsfragen

Das erwartete Ergebnis dieser Arbeit ist ein auf Machine Learning basierender Algorithmus, der dabei hilft Hasspostings aus Twitter automatisch zu identifizieren. Um eine möglichst hohe Fehlerfreiheit zu gewährleisten sollen dabei aktuelle State-of-the Art Features implementiert, diese in verbesserter Weise kombiniert und/oder modifiziert werden.
Es soll auch festgestellt werden, in welchem Ausmaß die einzelnen Features für die Verbesserung der Erkennung von Hasskommentaren beitragen, wie und ob sich diese bei Kombinationen gegenseitig beeinflussen und welche Merkmale für den besten Klassifizierungsalgorithmus ausgewählt werden sollten. Als Besonderheit soll hier die Realisierung des Verfahrens für die deutsche Sprache hervorgehoben werden, die es in der Form noch nicht gibt und es daher bei der Umsetzung der Features Lösungen zu finden gilt. Im Laufe der Arbeit werden auch Klassifizierer wie Naive Bayes und Random Decision Forest evaluiert, wobei das Hauptaugenmerk aufgrund der bereits erwiesenen Effizienz in vorangegangen Forschungsarbeiten am Support Vector Machine-Klassifikator (SVM) liegt.
Damit die verschiedenen Ansätze auch ordnungsgemäß getestet und überprüft werden können, muss im ersten Schritt auch ein manuell annotierter Korpus für das Training des Algorithmus erstellt werden. Die Datengrundlage für diesen Korpus liefern die von Twitter über einen gewissen Zeitraum gesammelten Beiträge, wobei einerseits explizit nach Postings mit hassassoziierenden Inhalt und andererseits nach Tweets in neutralen Kontexten gesucht werden soll.
Als weiteres Teilergebnis wird ein Lexikon mit spezifischen, negativ behafteten Ausdrücken, wie Schimpfwörtern oder Beleidigungen einbezogen, um in erster Linie damit verbundene lexikalische Features testen zu können. Zusätzlich werden die im Thesaurus enthaltenen Vokabel in den Linguistic Inquiry and Word Count (LIWC) - Wortschatz eingepflegt, um diesen an die Besonderheiten von Hassbeiträgen anzupassen. Im Detail

sollen auch zusätzliche Kategorien im Bezug auf Hass entworfen und die bereits enthaltenen und neu integrierten Hasswörter in diese eingeteilt werden. In weiterer Folge werden die Auswirkungen auf die durch das LIWC generierten Features und Klassifikations-Ergebnisse untersucht.
Zur weiteren Verbesserung der Klassifikation sollen auch bestimmte Eigenheiten von Tweets genützt werden. So zum Beispiel wird versucht, Informationen bezüglich der Anzahl an Retweets und Likes eines Beitrags oder die Art des Tweets (Antwort oder Posting) für die Einteilung in eine Klasse heranzuziehen. Auch die Vergangenheit von Usern soll analysiert werden, indem bereits verfasste Hasspostings gezählt werden. Diese Hintergrundinformation wird als eigenständiges Feature dienen und als Basis für den Schwellenwert, ab wie vielen hasserfüllten Tweets ein Nutzer als "Hassposter" erachtet wird, herangezogen. Besonders hervorzuheben ist die Umsetzung von Features, die sich aus der Analyse des Twitter-Netzwerks ergeben. Aufgrund der speziellen Follower-Eigenschaft von Twitter bietet es sich an, Vernetzungen zwischen verschiedenen User zu untersuchen, um gegebenenfalls Zusammenhänge festzustellen, welche für die Erkennung von Hasspostings hilfreich sein könnten. So zum Beispiel liegt die Hypothese nahe, dass User, welche einschlägigen Twitter-Seiten und notorischen Hasspostern folgen, eher dazu gewillt sind, negatives Gedankengut zu verbreiten.

Nachfolgend werden die sich ergebenden Forschungsfragen formuliert, die mithilfe dieser Diplomarbeit beantwortet werden sollen:

- Können Features aus dem sozialen Netzwerk Twitter gewonnen werden, welche dabei helfen die automatische Erkennung von Hasspostings zu verbessern, wobei als Vergleich dazu ein nur aus State-of-the-Art Features berechnetes Ergebnis dienen wird? Welche Dimensionen nehmen diese Verbesserungen an? Wie können diese Features am besten kombiniert werden?
  - Wie wirken sich spezielle Eigenschaften von Tweets, wie die Anzahl der enthaltenen Hashtags, Likes oder die Unterscheidung zwischen Retweet und Tweet auf die Messwerte wie F-Sore, Precision und Recall aus?
  - Können Informationen über die User (Anzahl der geposteten Tweets, Follower, Mitgliedschaften in Gruppen,...) also der Knotenpunkte im Netzwerk für eine Verbesserung der Ergebnisse eingesetzt werden?
  - Welche Features, die für eine bessere Klassifikation sorgen, können aufgrund der Analyse des durch die Follower und Freunde - Option von Twitter aufgespannten Netzwerkstruktur/Graphen extrahiert werden?
  - Inwieweit beeinflusst ein Feature, wie die aggregierte Anzahl der bereits getwitterten Hasspostings eines Users, die Resultate?
- Kann ein hassbezogenes Lexikon erstellt und in die Wortliste eines Linguistic Inquiry and Word Count (LIWC)-Dictionaries eingegliedert werden, damit die Performance bei der Identifikation von Hasspostings durch die angepassten LIWC-Features erhöht wird?

  - Wäre es im Bezug auf das Ergebnis sinnvoll, die bestehende LIWC-Taxonomie um Hass-spezifische Kategorien zu erweitern?
  - Kann der Thesaurus mit Hasswörtern auch für weitere Features, zum Beispiel für die Filterung von Bag-of-Word N-Grams angewandt werden, um die Messwerte weiter zu verbessern?

- Welche Klassifizierer eignen sich für die Erkennung von Hassrede in Twitter am besten?
  - Im Detail welcher Klassifizierer aus Naive Bayes, Random Decision Forest und Support Vektor Machine performt am effektivsten?
  - Welche Parametereinstellungen liefern ein optimales Ergebnis?

1.4 Überblick - Struktur der Arbeit

Die Struktur der zugrundeliegenden Diplomarbeit folgt dem zeitlichen Ablauf der durchgeführten Forschungsarbeiten. Zu Beginn wird in Kapitel 2 - Related Work die Definition von Hasspostings aus verschiedenen Gesichtspunkten beleuchtet, um ein gemeinsames Verständnis darüber zu gewinnen. Des weiteren werden bestehende vergleichbare Forschungsarbeiten vorgestellt, aus denen Anforderungen und Vorgehensweise an die automatisierte Erkennung von hasserfüllten Inhalten ersichtlich werden. Zusammenfassend stellt dieser Abschnitt die Grundlage für den eigenen entwickelten Algorithmus dar, dessen Design in Kapitel 3 - Methodik aufbereitet werden. Als Kernthemen fungieren dabei Punkte wie die Zielsetzung, der erstellte Trainingskorpus oder die verwendeten Features. Hauptaugenmerk wurde zusätzlich auf die asymmetrische Kostenanalyse von False Positive- und False Negative Klassifikationen gelegt. Details zur Evaluierung des umgesetzten Algorithmus kann in Kapitel 4 gefunden werden. Dabei werden Punkte wie die einzelnen Features, diese in Kombination als Feature-Sets, die eingesetzten Classifier, die qualitative und quantitative Performance untersucht, gefolgt von eine Analyse der Kosten-Nutzen und des Bias. Die Arbeit schließt in Kapitel 6 mit einer Zusammenfassung der erzielten Erkenntnisse und einem Ausblick auf mögliche zukünftige Forschungsbereiche, um die automatisierte Identifikation von Hasspostings weiter zu verbessern.

KAPITEL 2

Related Work

2.1 Einleitung

Dieses Kapitel soll einen Überblick über den derzeitigen Stand der Technik in der automatisierten Hassposterkennung und verwandten Anwendungsfeldern geben. Besonders untersucht wird dabei inwieweit LIWC-Attribute und die Analyse von Netzwerkstrukturen in diesem Kontext eingesetzt wurden und zur Identifikation von Hass beitrugen. Zusätzlich wird der Begriff Hassposting und Hassrede von verschiedenen Gesichtspunkten beleuchtet und erläutert. Auch der aktuelle Trend und die Verbreitung von Hasspostings in Österreich wird dabei genauer untersucht.

2.2 Hasspostings

2.2.1 Verbreitung von Hasspostings in Österreich

Seit längerer Zeit beschäftigen Hasspostings die Benutzer sozialer Netzwerke und dominieren immer wieder die Berichterstattung der Medien. Der Verein Zara berichtete beispielsweise von nahezu einer Verdopplung rassistischer Hetzbotschaften und Vorfälle über das Internet im Jahr 2015, im Bereich von Social Media sogar von einer Verdreifachung im Vergleich zum Vorjahr.[1]

Auch laut dem Jahresberichts des Bundesamts für Verfassungsschutz von 2014 sind Anzeigen aufgrund strafrechtlicher verhetzender Postings um 30 Prozent gestiegen (476 auf 629). Generell wurde der deutliche Anstieg von Hasspostings damals als aktueller Trend beziehungsweise Entwicklung ausgemacht.[2] Im Jahresbericht von 2015 nimmt "Hasskriminalität im Internet" bereits ein ganzes Kapitel ein, in dem unter anderem Hassrede genauer beleuchtet wird. Aus soziologischer Sicht wird angeführt, dass soziale

[1] http://www.zara.or.at/_wp/wp-content/uploads/2016/03/ZARA_Rassismus_Report_2015_web_fin.pdf

Netzwerke wie Facebook und Twitter, in denen laut Bericht seit Mitte des Jahres 2015 das Diskussionsklima von „aggressiven bedrohlichen, beleidigenden, verhetzenden, fremdenfeindlichen und besonders asylfeindlichen Kommentaren“ beherrscht wird, oft die „einzige Nachrichten- und Auskunftsquelle zum gesellschaftlichen Geschehen“ darstellt und wesentlich zur Meinungsbildung beitragen. Außerdem wird darauf hingewiesen, dass dadurch nur „ein gefilterter und eingeschränkter Blick auf die Realität zur Verfügung steht“, wodurch „insbesondere junge, ungefestigte Menschen Gefahr laufen, Radikalisierern und Hetzern Glauben zu schenken, diesen zu folgen und sich zu Hasspostings oder anderen möglicherweise gewalttätigen Handlungen hinreisen lassen“. Speziell durch die userorientierte Aufbereitung von Inhalten auf Webseiten, die auf Basis bisheriger Interessen des Nutzers funktioniert, entsteht eine gewisse „Informationsblase“, wodurch der Nutzer beispielsweise neben den vielen Falschmeldungen oder Halbwahrheiten über die Flüchtlingslage beziehungsweise über Migranten, keine widersprechenden Informationen mehr erhält. Zusätzlich wird angeführt, dass „Rechtsextremisten ihren Hass gekonnt in rationaler Argumentation verschleiern“ und das „was rational, logisch und emotionslos erklärt wird, der bewussten Aufstachlung zu Hass und entsprechenden gewalttätigen Agitationen gegen Menschen dient“. All diese Tatsachen können laut Bericht „verängstigte, von der gegenwärtigen Migrationslage überforderte Benutzer“ veranlassen, mittels Hassposting-Inhalten, möglicherweise auch nur einmalig, Dampf abzulassen. Andere wiederum sehen sich in ihren Vorurteilen bestätigt und leiten daraus eine „Aufforderung und Legitimation für diskriminierende Handlungen und Gewalt gegen Menschen ab“. Dieser „Anstieg von fremden-, asyl- und besonders islamfeindlicher Aktivitäten“ drückte sich durch eine Erhöhung der Meldungen an die Internet-Meldestelle für NS-Wiederbetätigung, bei der auch Hinweise für Hasskommentare aus anderen Bereichen behandelt werden, von 3.354(2014) auf 3.913(2015) aus (16,7%). Die davon strafrelevanten Anzeigen erfuhren sogar eine Verdopplung. Meldungen zu asyl- und islamfeindlichen Inhalten überwogen, wobei darauf hingewiesen wird, dass die „ermittelten tatverdächtigen Männer und Frauen aus allen Altersgruppen, Bevölkerungsschichten und Nationalitäten stammen“.[3]

Der Verfassungsschutzbericht des Jahres 2016 berichtete indes von einem Rückgang der Meldungen an die Internet-Meldestelle für NS-Wiederbekämpfung von 3.913 auf 3.124 (-20,2%).[4] Grund dafür könnte die Entspannung der Flüchtlingssituation sein.

2.2.2 Definition von Hasspostings/Hassrede

Ab wann ein Beitrag als Hassposting beziehungsweise der Inhalt als Hassrede bezeichnet werden kann, ist oft nicht ganz eindeutig zu definieren, wodurch ein gewisser Graubereich entsteht. Allgemein kann gesagt werden, dass es für jeden Menschen eine persönliche Trennlinie gibt und Hassrede immer vom Kontext abhängig ist und von Staat zu Staat aufgrund der unterschiedlichen Geschichte anders ausgelegt wird. Das Thema verbale Gewalt und Hass(rede) bewegt sich generell im Schnittpunkt zwischen Linguistik, Soziologie, Psychologie und Politik, wodurch auch der Standpunkt im Bezug auf Hasspostings entscheidend sein kann. Der Begriff “Hassrede“ selbst kommt nicht aus der Sprachwissenschaft sondern ist ein politischer mit mehr oder weniger starken Bezügen zu juristischen

Tatbeständen.[2] Nachfolgend werden die verschiedenen Sichtweisen im Zusammenhang mit Hasspostings/Hassrede beschrieben, um diese zu verstehen und zu erklären.

Rechtswissenschaften/Rechtsprechung
Der rechtliche Standpunkt kann dabei helfen die Beiträge besser einzuordnen und eine ungefähre Trennlinie zwischen neutralem und strafbaren Hassposting zu ziehen. Nicht jede rassistische Äußerung die von der überwiegenden Mehrheit als Hassrede wahrgenommen wird, muss jedoch zwangsweise strafbar sein. Die Rechtsprechung stellt dabei generell eine Schnittstelle zwischen Politikwissenschaft und Sprachwissenschaft dar, da zum einen bestimmte sprachliche Äußerungen juristisch bewertet werden müssen, zum anderen bestimmte politische Normen Eingang in das Strafgesetzbuch finden.[5]

Nachfolgend werden die rechtlichen Tatbestände, die bei Hasspostings in Österreich erfüllt werden können aufgelistet.[3] Allen voran steht der Tatbestand der Verhetzung:

- **§283 StGB Verhetzung:**
  "Wer öffentlich auf eine Weise, dass es vielen Menschen zugänglich wird (rund 30 Personen), zu Hass oder Gewalt gegen eine Kirche oder Religionsgesellschaft oder aufgrund vorhandener oder fehlender Kriterien der Rasse, der Hautfarbe, der Sprache, der Religion oder Weltanschauung, der Staatsangehörigkeit, der Abstammung, der nationalen oder ethnischen Herkunft, des Geschlechts, einer körperlichen oder geistigen Behinderung, des Alters oder der sexuellen Ausrichtung einer definierten Gruppe von Personen oder gegen ein Mitglied einer solchen Gruppe ausdrücklich wegen der Zugehörigkeit zu dieser Gruppe aufstachelt, beschimpft oder in der öffentlichen Meinung herabsetzt", macht sich wegen Verhetzung strafbar. „Wer Kriegsverbrechen und Völkermorde, die von einem inländischen oder einem internationalen Gericht rechtskräftig festgestellt wurden, billigt, leugnet, gröblich verharmlost oder rechtfertigt", macht sich ebenfalls wegen Verhetzung strafbar.[1]

  Das Kriterium, dass die Aussagen vor ausreichend vielen Personen getätigt werden muss, ist in sozialen Netzwerken oft schon in kleinen Gruppen innerhalb von sozialen Netzwerken erfüllt. In Absatz (4) des Strafgesetzbuches §283 ist weiters definiert, dass auch das „Befürworten, Fördern und die Weiterverbreitung von hetzerischen Material" strafbar ist.[1] Somit können auch Retweets oder das Liken von Hasspostings im Einzelfall zu Strafen führen. Ausgenommen davon ist die Weiterverbreitung im Zuge von Gegenrede und Aufklärung. Generell können Verurteilte je nach Schwere (z.B. mehr als 150 Personen zugänglich gemacht) ihrer Tat mit einer Freiheitsstrafe von bis zu 3 Jahren rechnen.

- **§ 111 StGB Üble Nachrede**
  „Wer einen anderen in einer für einen Dritten wahrnehmbaren Weise eine verächtlichen Eigenschaft oder Gesinnung vorwirft oder eines unehrenhaften Verhaltens

[2] http://www.amadeu-antonio-stiftung.de/hatespeech/was-ist-ueberhaupt-hate-speech/
[3] https://www.saferinternet.at/news/news-detail/article/hasspostings-im-internet-was-sagt-das-gesetz-577/

oder eines gegen die guten Sitten verstoßenden Verhaltens beschuldigt, das geeignet ist, ihn in der öffentlichen Meinung verächtlich zu machen oder herabzusetzen“, macht sich wegen übler Nachrede strafbar.[6]

- **§ 115 StGB Beleidigung**
  „Wer öffentlich oder vor mehreren Leuten (mehr als zwei Personen) einen anderen beschimpft, verspottet, am Körper misshandelt oder mit einer körperlichen Misshandlung bedroht“, macht sich wegen Beleidigung strafbar.[7]
  Nicht jede Beleidigung erfüllt jedoch dessen Tatbestand, da „Beschimpfungen und angedrohte Misshandlungen, zu denen sich eine Person in einer Umständen nach entschuldbaren Weise aufgrund von Entrüstung beziehungsweise im Effekt hinreisen ließ und nachvollziehbar sind“, ausgenommen sind.[7] Insbesondere rassistisch motivierte Beleidigungen gegen eine Gruppe die in § 283 Abs. 1 aufgelistet sind, werden zu einem Ermächtigungsdelikt (§ 117 Abs. 3 StGB). Das heißt, dass solche Delikte von der Staatsanwaltschaft mit der Ermächtigung des/der Betroffenen von Amts wegen zu verfolgen sind und deshalb keinerlei Prozesskosten für ihn anfallen können. Die Anzeige eines Hasspostings fällt dadurch aufgrund des verringerten finanziellen Risikos natürlich leichter. Verurteilte können mit einer dreimonatigen Freiheitsstrafe oder mit einer Geldstrafe von bis zu 180 Tagessätzen bestraft werden.

- **§ 297 StGB Verleumdung**
  „Wer einen anderen dadurch der Gefahr einer behördlichen Verfolgung aussetzt, dass er ihn einer von Amts wegen zu verfolgenden mit Strafe bedrohten Handlung oder der Verletzung einer Amts- oder Standespflicht falsch verdächtigt“, macht sich wegen Verleumdung strafbar.[8] Das Strafausmaß kann bis zu 5 Jahre betragen.

- **§ 152 StGB Kreditschädigung**
  “Wer unrichtige Tatsachen behauptet und dadurch den Kredit, den Erwerb oder das berufliche Fortkommen eines anderen schädigt oder gefährdet“, macht sich wegen Kreditschädigung strafbar und ist mit einer Freiheitsstrafe von bis zu sechs Monaten oder mit einer Geldstrafe von bis zu 360 Tagessätzen zu bestrafen.[9]

- **§ 107c StGB Cyber-Mobbing**
  „Wer im Wege einer Telekommunikation oder unter Verwendung eines Computersystems in einer Weise, die geeignet ist, eine Person in ihrer Lebensführung unzumutbar zu beeinträchtigen, eine längere Zeit hindurch fortgesetzt(im Einzelfall zu überprüfen), eine Person für eine größere Zahl von Menschen (ab 10 Personen) wahrnehmbar an der Ehre verletzt oder Tatsachen oder Bildaufnahmen des höchstpersönlichen Lebensbereiches einer Person ohne deren Zustimmung für eine größere Zahl von Menschen wahrnehmbar macht“, macht sich wegen Cyber-Mobbing strafbar.[10] Das maximale Strafausmaß für Cyber-Mobbing beträgt bis zu 3 Jahre, wenn dieses Selbstmord oder einen Selbstmordversuch nach sich zieht. Unter den höchstpersönlichen Lebensbereichen fallen unter anderem das Sexualleben, der

sensible Bereich des Familienlebens, Krankheiten, Behinderungen und religiöse Ansichten.[4]

- **§107 StGB Gefährliche Drohung**
  „Wer einen anderen gefährlich bedroht, um ihn in Furcht und Unruhe zu versetzen," macht sich wegen gefährlicher Drohung strafbar und „ist mit einer Freiheitsstrafe von bis zu einem Jahr oder mit einer Geldstrafe von bis zu 720 Tagessätzen zu bestrafen."[11] Wird mit dem Tod, einer erheblichen Verstümmelung oder auffallenden Verunstaltung, mit einer Entführung, mit einer Brandstiftung, mit einer Gefährdung durch Kernenergie, ionisierende Strahlen oder Sprengmittel oder mit der Vernichtung der wirtschaftlichen Existenz oder gesellschaftlichen Stellung gedroht, sind Freiheitsstrafen von bis zu drei Jahren möglich.[11] Dieser Tatbestand wird im Kontext von Hasspostings oft im Zuge durch die Bedrohung von Politikern erfüllt.

Das jedoch auch die Rechtsprechung nicht immer eindeutig ist, zeigt ein Fall vom Sommer 2015 bei dem ein Lehrling ein Bild eines syrischen Flüchtlingsmädchens mit den Worten "Flammenwerfer währe da die bessere Lösung." kommentierte. In vielen Augen stellt dies zweifellos ein Hassposting dar, das Verfahren gegen ihn wurde jedoch eingestellt und damit begründet, dass eine gefährliche Drohung nicht vorliege, da niemand konkret bedroht wurde und auch der Tatbestand der Verhetzung nicht erfüllt sei.[5] Ein weiterer Fall aus dem Jahr 2013 bei dem ein damals 19-jähriger abfällige Witze über Türken, in denen auch Ausdrücke wie "Gaskammer" beinhaltet waren, online tätigte, zeigt, dass der Unterschied zwischen Witz und Verhetzung verschwindend gering ist. Damals wurde der Jugendliche nur aufgrund eines beinhaltenden ";)"-Emoticons in letzter Instanz freigesprochen, da ihm dadurch keine Absicht nachgewiesen werden konnte, Türken in ihrer Menschenwürde zu verletzten.[6]

Linguistik/Sprachwissenschaft
Aus sprachwissenschaftlicher Sicht beinhalten Hasspostings gewisse sprachliche Ausdrucksmittel. Einerseits sind das abwertende Wörter, die sich direkt auf eine Bevölkerungsgruppe beziehen (z.B. Jugo, Musel, Spast) oder solche die bestimmte Gruppen auf Stereotypen reduzieren beziehungsweise ihnen solche vorurteilsbehaftete Eigenschaften nachsagen und deshalb verunglimpfen (z.B. Mongo, Schlitzauge). Andererseits geben bereits einzelne Wortsilben oder grammatische Muster Hinweise auf Hasspostings. So zum Beispiel sind "-ling" (z.B. Flüchtling), "-ler" (z.B. Assozialer) oder in Österreich "-ant"(z.B. Asylant) konkrete Indizien für herabwürdigende Inhalte, das Muster bestehend aus "Gruppe + Richtung", wie im Beispiel von "Ausländer raus!", kann ebenfalls Hinweise liefern.[12] Im Kontext dieser Arbeit sollen diese Merkmale mittels Hasswortlexikon, Character N-Grams und Typed Dependencys für grammatische Muster erkannt und verarbeitet werden.

[4] http://www.raoe.at/news/single/archive/cybermobbing-kuenftig-gerichtlich-strafbar/
[5] https://futurezone.at/digital-life/ermittlungen-nach-flammenwerfer-posting-eingestellt/146.745.408
[6] https://futurezone.at/netzpolitik/hass-postings-was-verboten-ist-und-welche-strafen-drohen/144.065.820

Weiters stellen sich Sprachwissenschaftler die Frage, was es sprachlich bedeutet Hass auszudrücken. Dabei wird einerseits angeführt, dass es auf die Intention des Verfassers ankommt, dieser mit seiner Aussage Hass erzeugen will und/oder selbst dabei Hass empfindet. Bei unabsichtlicher Hassrede durch Unwissenheit oder bei Satire kann also nicht von solcher ausgegangen werden. Dem gegenüber steht die Auffassung, dass die Betroffenensicht entscheidend ist, Hassrede also aus sprachwissenschaftlicher Perspektive gegeben ist, wenn ein wahrnehmbarer Teil der Sprachgemeinschaft die Aussage als herabwürdigend für eine Bevölkerungsgruppe empfindet.[13] Weitere Elemente, welche Hassrede aus der Sicht der Sprachwissenschaft prägen, sind laut Amadeu Antonio Stiftung folgende:[13]

- **Gleichsetzung**: Juden=Israel, Schwarzer=Afrika, Alle Moslems sind Terroristen!
- **Verschwörungstheorien**: Israel hat einen Anschlag auf die eigene Bevölkerung inszeniert, um von der Kritik an der Außenpolitik abzulenken.
- **De-realisierung** - Eine verzerrte, realitätsabgehobene Konzeptualisierung durch Ausblendung von Fakten oder in Form von Falschaussagen: Alle Politiker hassen Deutschland.
- **Gruppenbildung** - Gegenüberstellung von Wir- und Ihr-Gruppe und das Konstruieren eines Handlungszwangs: Wenn wir uns von denen weiter auf der Nase herumtanzen lassen, werden wir alle sterben.
- **Normalisierung von bestehenden Diskriminierungen**: Ist doch kein Wunder, dass die Schwarzen so behandelt werden.

Zusätzlich existieren verschiedene sprachwissenschaftliche Tests die durchgeführt werden können, um Hasspostings zu kategorisieren. So stehen beispielsweise der *BÜRGER-Test* für das Erkennen von Rassismus, der *3D-Test* für Antisemitismus und der *DON-Test* für Sexismus zur Verfügung. Jeder Buchstabe der Akronyme steht dabei für eine andere Eigenschaft der Beiträge, welche Betroffene herabwürdigt. Abbildung 2.1 veranschaulicht diese Eigenschaften rassistischer Texte laut Bürger-Test und liefert Beispiele zu jedem Merkmal.

Anatol Stefanowitsch betont jedoch, dass auch innerhalb der Sprachgemeinschaft umstritten ist, was als Hassrede angesehen werden kann. Angehörige nicht betroffener Gruppen befinden Ausdrücke oft als nicht abwertend, da sie sich nicht gegen sie richten. Andererseits reagieren Mitglieder einer betroffenen Gruppe oft übersensibel bei bestimmten Ausdrücken, die jedoch mehrheitlich von der betreffenden Gruppe beziehungsweise von der Allgemeinheit als neutral eingestuft werden. Weiters beschreibt er die implizite Ausprägung von Hassrede, bei der keinerlei verunglimpfende Ausdrücke verwendet werden, sondern verborgene Grundannahmen über eine Bevölkerungsgruppe angenommen werden. Als Beispiel kann die politische Äußerung „Migranten sind willkommen, wenn sie sich an

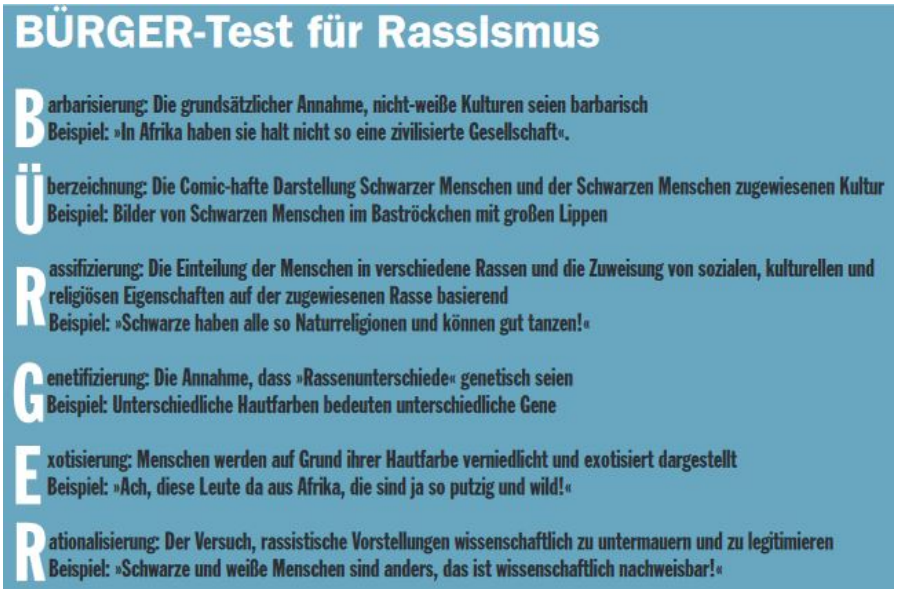


Abbildung 2.1: Bürger Test für Rassismus.[13]

die Gesetze halten.“ herangezogen werden, die zwar unbedenklich klingt, aber implizit angenommen wird, dass Migranten sich nicht an Gesetze halten.[12]

Abschließend fügt Anatol Stefanowitsch an, dass Hassrede kein sprachliches Problem ist, da Sprache nur beschreibt und nicht bewertet, sondern vielmehr ein gesellschaftliches Problem, da Sprache ein Verständnis über die allgemeingültigen gesellschaftlichen Realitäten erzeugt. „Hassrede ist nicht (nur) ein Problem des kommunikativen Umgangs“, sondern „ist zentral an der Erzeugung des Hasses und der für den Hass notwendigen Denkmodelle beteiligt“.[12]

Politwissenschaft
Neben dem Umstand, dass Hassrede eine Begleiterscheinung vieler politischer Konflikte ist, wie die aktuelle umstrittene Flüchtlingspolitik zeigt, beschäftigen sich Politologen intensiv mit der Frage inwieweit die Beschränkung der Hassrede ein Eingriff in die Redefreiheit liberaler Demokratien bedeutet. Gegner der Einschränkung von Redefreiheit betonen, dass einerseits die Sprecher Recht auf freie Meinungsäußerung haben, unabhängig wie bedenklich diese ist, andererseits die Hörer keine Informationsfreiheit mehr besitzen, da ihnen nicht mehr die volle Information zur Verfügung steht, auch wenn diese Information rassistischer oder anderer fragwürdiger Natur ist. Zusätzlich würde die Wahlfreiheit der Hörer beraubt, da man Hassrede hören muss, um sich gegen sie zu entscheiden. Dem gegenüber steht die Auffassung, dass bestimmte Bevölkerungsteile durch Hassrede sozial, physisch und psychisch beeinträchtigt werden, es also zu Ungerechtigkeiten zwischen Teilen der Bürger des Staates kommt. Der Staat hat jedoch die Aufgabe alle Einwohner gleich zu behandeln und diese zu schützen. Würde er dies nicht tun, müssten sich die Betroffenen selbst dagegen auflehnen, welches die Bestrebungen der Hassredner

begünstigen würden. Zusätzlich wären diese möglicherweise so eingeschüchtert, dass durch ihr darauffolgendes Verstummen ihre demokratische Mitbestimmung leidet. Infolgedessen muss die Einschränkung der Redefreiheit erlaubt sein, um die Demokratie selbst zu schützen.[5] Drei weitere Argumente für die Redefreiheit und gegen die Einschränkung liefert Stephen A. Smith:

1. *Symptombekämpfung*: Die Zensur von Hassrede bekämpft lediglich das Symptom und nicht die eigentlichen sozialen Probleme innerhalb der Gesellschaft.
2. *Hemmwirkung:* Dadurch, dass sich es der Staat zur Aufgabe macht Hassrede zu bekämpfen, beschäftigt sich die Bevölkerung nicht damit, welcher Umstand für die Bildung einer verantwortungsbewussten und engagierten Zivilgesellschaft hinderlich ist.
3. *Signalwirkung:* Hassrede stellt ein wichtiges Alarmsignal dar, welches die Akteure und diskriminierende Gruppierungen erkennen lässt. Damit können staatliche Gegenmaßnahmen begründeterweise gegen sie eingeleitet werden. Würde die Hassrede beschränkt, könnten solche Gruppen ihre wahren Absichten in der Öffentlichkeit verschleiern.[14]

Soziologische/Psychologische Sicht
Aus soziologischer Sicht definiert sich Hassrede nicht ausschließlich durch die Intention des Verfassers oder das Empfinden des Betroffenen sondern ist auch abhängig vom gesellschaftlichen Standpunkt. So sind die gesellschaftlichen Normalvorstellungen in einer bestimmten sozialen Situation zu einem bestimmten Zeitpunkt ebenfalls entscheidend.[15]

Generell definieren sich Sprachgemeinschaften neben linguistischen Aspekten auch über solche sozialer Art. So wird die Sprache des Einzelnen von seiner Religion, Bildung, ethnische Zugehörigkeit etc. beeinflusst.[16] Laut Lann Hornscheidt ist es neben Sprachanalysen, die zeigen wie sich diskriminierende Botschaften zusammensetzen und manifestieren, mindestens genau so wichtig Sozialanalysen anzustellen, welche untersuchen wie Hassrede „in einer konkreten gesellschaftlichen Situation behandelt wird und was überhaupt von bestimmten sozialen Gruppen als diskriminierende Äußerung aufgefasst wird und was nicht."[15]

Weiters beschreibt Karl Marker, dass Hassrede nur in liberalen Demokratien als soziales Problem betrachtet wird, es also auf die sozialen Normen und Werte beziehungsweise politische Kultur einer Gesellschaft ankommt, um Hassrede negativ gegenüber zu stehen. In solchen Staatsformen kann der „soziale Frieden innerhalb der Gesellschaft durch Hassrede nachhaltig gefährdet werden, wodurch staatliche Akteure interessiert sind, Hassrede zu unterbinden, um einerseits die öffentliche Ordnung wie auch die Flucht in die soziale Isolation betroffener Bevölkerungsgruppen zu verhindern, die sich „irgendwann nicht mehr an Wahlen, Abstimmungen und Diskussionen beteiligen, es nicht mehr wagen, öffentlich für ihre Interessen und Überzeugungen einzutreten", woraufhin „zunehmend die „falschen", nämlich die diskriminierenden Gruppen Einfluss auf politische Institutionen

und den gesellschaftlichen Diskurs“ haben.[17] Dem gegenüber setzt Bollinger, dass uneingeschränkte Hassrede notwendig ist, um die Gesellschaft zu schockieren und ihr, ähnlich wie es die funktionale Skandaltheorie beschreibt, bewusst werden zu lassen wie wichtig es ist sich gegen diese Intoleranz und für ihre Werte einzusetzen.[18] Diese These entkräftet Karl Marker, da er behauptet, dass uns Hassrede nur schockiert, wenn wir bereits gute Demokraten sind und Werte wie Toleranz längst verinnerlicht haben. Andernfalls trifft Hassrede innerhalb der Gesellschaft auf Ignoranz oder sogar Begeisterung.[17]

Aus sozialwissenschaftlicher Perspektive ist weiters die Frage interessant, woher dieser Hass in der Gesellschaft rührt. Monika Schwarz-Friesel beispielsweise streicht heraus, dass bestimmte Abneigungen auch auf einem „sozial verankerten und kontinuierlich reproduzierten Vorurteilssystem“ gründen, da erworbenes, von verschiedenen Sozialinstanzen vermitteltes, Wissen über Bevölkerungsgruppen gesellschaftlich geprägt und vorgeformt ist. Andere Perspektiven oder Wahrheiten werden oft ignoriert beziehungsweise bekämpft.[19][20] Weiters unterstreicht Doris Unger, dass Hassrede wahrscheinlich dazu beiträgt „Rassismus, Homophobie und Sexismus innerhalb einer Gesellschaft aufrechtzuerhalten oder zu verstärken“, ergänzt aber, dass eine gewisse Menge an Hasspostings und bereits eine gewisse Akzeptanz in der Gesellschaft gegeben sein muss, um soziale Auswirkungen, wie beispielsweise Benachteiligungen am Arbeitsmarkt, -platz oder im Bezug auf die Ausbildung für die betroffene Gruppe zu bewirken.[21][22]

Christoph Reinprecht, Professor am Institut für Soziologie der Uni Wien, versucht ebenfalls zu erklären, wieso gerade die Flüchtlingskrise eine derartige Welle an Hasspostings auslöste. Ein Grund dafür ist, dass seiner Meinung nach der Fremdenhass in jedem von uns verankert ist und dieser in solchen Zeiten und Situationen Resonanz findet und gedeiht. Das Potential an Ablehnung gegenüber Fremden ist laut ihm außerdem seit den sechziger Jahren unverändert und konnte auch durch den Bildungsaufstieg nicht vermindert werden, vielmehr bewirkt dieser derzeit einen Bruch in der Gesellschaft, da es mehr öffentliche Kritiker, mehr Auseinandersetzung und mehr Hinterfragung gibt, wodurch sich Gegner und Befürworter klarer positionieren, die Polarisierung also steigt. Zusätzlich führt er an, dass dieser Hass in der Gesellschaft schwer zu durchdringen ist, da auf Hass gegenüber Flüchtlingen mit Hass auf die Hassposter reagiert wird und die Antwort auf diesen wiederum Hass ist. Die Gründe, warum dieser Hass gerade in den letzten Jahren ausgebrochen ist und an Fahrt aufnimmt, sieht Reinprecht in den steigenden Abstiegsängsten, die durch Krieg, Terror, Wirtschaftskrisen und eben auch durch die Migration genährt wurden und der fehlenden Stabilität und wirtschaftlichen beziehungsweise sozialen Sicherheit, an die die österreichische Bevölkerung in den vorigen Jahrzehnten gewohnt war. Gegenüber rationalen Argumenten und positiven Fällen von Integration sind viele Leute resistent und lassen sich eher durch ihre starken Gefühle leiten, welche sich aufgrund der breiten Zustimmung und vermeintlichen Rückendeckung anderer Hassposter zusätzlich in Gewalt gegen die Hassobjekte ausdrücken.[7] Laut Nestor Kapusta, Psychoanalytiker und Professor an der Medizinischen Universität in Wien, versuchen Menschen durch Hass jene Objekte, die ihnen Angst bereiten, zu zerstören. Sie

[7] https://christinapausackl.wordpress.com/2015/11/02/der-hass-der-uns-trennt/

suchen in Krisenzeiten einfache Erklärungen und Sündenböcke beziehungsweise greifen zu primitiven Abwehrmechanismen. Dadurch, dass die Regierungen in der Flüchtlingsfrage ebenfalls hilflos wirkte, wurde der Hass weiter gefördert. All diese Umstände und Ängste machen sich andere Parteien zu nutze, legitimieren und stärken Hass beziehungsweise fördern und instrumentalisieren diese Ängste.[7]

Zusammenfassung
Um zu verstehen was ein Hassposting und Hassrede ist, sind wie beschrieben viele Sichtweisen nötig. Es ist weiters nicht leicht zu sagen, woher dieser Hass kommt, rührt jedoch oft von Ängsten oder persönlicher Unzufriedenheit, wobei Flüchtlinge oder andere Bevölkerungsteile für viele ein Ventil darstellen, um diesen Zorn auf den Alltag und ihre Situation abzulassen. Zusätzlich wird oft die Frage diskutiert, wie sehr beziehungsweise ob Hasspostings überhaupt reguliert werden sollten, da dadurch die Redefreiheit beschränkt werden würde. Beide Seiten stellen dabei nachvollziehbare Argumente auf, der richtige Weg liegt jedoch wahrscheinlich in der Mitte, indem rechtlich relevante Postings und Hetze gefiltert und behandelt werden. Es geht also nicht um Zensur, sondern darum, dass die Regeln des Miteinanders eingehalten werden.[8] Twitter, Facebook und andere dürfen jedenfalls keine Plattformen für hetzerische Äußerungen sein, die speziell in ihrer Überzeugung noch nicht gefestigte Jugendliche mit falschen Fakten oder hetzerischen Inhalten versorgen. Generell gibt es eine Grenze, wie verschiedenartige Meinungen zum Ausdruck gebracht werden, um sachlich diskutieren zu können. Insbesondere Hasspostings überschreiten diese und gefährden durch deren breiten und öffentlichen Zugang auf Social Networks den sozialen Frieden, indem eine Grundlage für Intoleranz und Hass geschaffen wird, welcher die betroffenen Gruppen in Folge dessen sozial, physisch oder psychisch treffen kann. Verfahren gegen eindeutige hetzerische Beiträge, Beleidigungen oder Bedrohungen können deshalb zusätzlich abschreckend wirken und vielleicht der Gesellschaft aufzeigen wie verroht und empathielos manche Mitmenschen mittlerweile geworden sind. Dies kann zu einem Umdenken und zur Verarbeitung des Hasses führen.[9,10]

2.3 State-of-the-Art

2.3.1 Identifikation offensiver Inhalte

Die meiste vorausgehende Arbeit im Bereich der Erkennung von offensiven Inhalten erstreckt sich über einige miteinander verwandte Felder. Da diese Themengebiete als Vorläufer für die Identifikation von Hassrede angesehen werden können und auch Beleidigungen für die Klassifikation als “Hassposting“ nicht unwesentlich sind, werden relevante Arbeiten dieser Gebiete in diesem Abschnitt gesondert untersucht. Im Detail lassen sich die Arbeiten in verschiedene Bereiche einteilen, wobei versucht wird folgende Inhalte zu erkennen:

[8] https://www.rbb-online.de/wirtschaft/beitrag/2016/03/software-aus-bernau-gegen-hass-und-hetze-im-internet.html
[9] http://derstandard.at/2000021533588/Pro-und-Kontra-Facebook-Hasspostings-loeschen
[10] https://www.wissenswertes.at/index.php?id=politik-hassposting

- Cyber-Bullying[23][24][25][26][27][28]
- Radikale Inhalte [29][30][31]
- Belästigungen beziehungsweise Beleidigungen[32][33][34][35][36]
- Filterung von Schimpfwörtern[37][38]

Auf die wichtigsten Arbeiten, die die Grundlage für die Hassrede Erkennung lieferten, wird nachfolgend genauer eingegangen.

Yin et al. waren 2009 einer der ersten die dem Problem von Beleidigungen in Onlineforen entgegenwirkten, indem sie Klassifikationen von Posts mithilfe von Supervised Machine Learning durchführten. Dabei trugen verschiedene N-Grams, selbst entworfene Patterns, welche sich aus den verschiedenen Positionen der Personalpronomen und einem in unmittelbarer Nähe im Satz befindlichen Hasswort zusammensetzen und kontextbasierte Features, welche vorangegangene negative Beiträge berücksichtigen, zur Erstellung des Machine Learning-Modells bei. Als Grund für die Erstellung solcher Patterns zur Erkennung von Beleidigungen wurde die häufige Verwendung von Personalpronomen im Zusammenhang mit einer verbalen Belästigung genannt. Dieser Umstand wird in dieser Arbeit durch die lexikalischen Features, welche auch die Anzahl der persönlichen Fürwörter in einem Tweet misst, abgedeckt. Evaluiert wurde der Algorithmus mittels den Kennzahlen Precision P, Recall R und F-Score, welcher für die Klasse der Beleidigungen in seiner besten Ausprägung die korrespondierenden Werte 0.394, 0.619 und 0.481 erzielten. [36]
Ein Jahr später entwarf Razavi et al. ein ähnliches System für die Erkennung offensiver Inhalte in jeglichen Bereichen digitaler Texte, wie beispielsweise in E-Mails, Foren, sozialen Netzwerken, auf Collaborative Writing Websites (Wikipedia) oder sogar in SMS-Nachrichten. Erstmalig wurden auch die verschiedenen Kategorien im Bezug auf Art und Ziel der herabwürdigenden Inhalte definiert, jedoch während der Klassifikation nicht zwischen ihnen unterschieden. Für die automatisierte Identifikation der Beleidigungen wurde ein auf 3-Level basierendes Klassifikationsmodell entwickelt und sich einem selbst entworfenem Lexikon mit dem Namen Insulting and Abusive Language Dictionary (IALD), welches gewichtete offensive Wörter, Phrasen, und Redewendungen enthält, bedient. Aus manchen Einträgen des Wörterbuchs wurden zusätzlich Pattern für eine bessere Term- und Phrasenerkennung erzeugt, indem allgemeine Wörter durch Wildcards ersetzt wurden. Zur Evaluierung wurde eine 10-fachen Kreuzvalidierung durchgeführt und die Anzahl der korrekt klassifizierten Texte in Prozent beziehungsweise die Precision der jeweiligen Klassen gemessen. Nach dem dritten Level der Klassifizierung konnten 96.72% korrekt klassifiziert werden und die Precision betrug 95.6% bei neutralen Texten und 100% bei offensiven Inhalten. Speziell die erfolgreiche Verwendung eines Wörterbuches ist für die zugrundeliegende Diplomarbeit interessant.[32]
Chen et al. verwendeten einen vielfach größeren Umfang an Features indem auch lexikalische, inhaltsbezogene, strukturelle und schreibstilspezifische Merkmale für die Filterung

von offensiven Inhalten für Minderjährige in Youtube herangezogen wurden. Zusätzlich zählen Chen et al. auch zu den Ersten, die in diesem Bereich von Dependency-Parser generierte Features testeten und umsetzten. Jedoch wurde auf eine strikte Definition von beleidigender Sprache verzichtet, da die Trennlinie von Eltern oder Lehrpersonen gezogen wird, welche als Zielgruppe für deren Tool angesehen wird. Im besten Setting konnte ein F-Score von rund 85% mittels einem Support Vector Machine Classifier erreicht werden. Eine Vielzahl der angewandten Features konnten für den umgesetzten Algorithmus dieser Arbeit herangezogen werden und brachte zusätzlich Erkenntnisse über die Auswirkungen der Kombination von Feature-Kategorien. [35]

2.3.2 Identifikation von Hassrede

Speziell für den Anwendungsbereich der "Hassrede-Erkennung" existieren nur vereinzelte Ansätze, welche sich jeweils nur auf eine bestimmte Zielgruppe von Hass beziehen. Zusätzlich wird der Vergleich mit bestehenden Forschungsergebnissen erschwert, da der Begriff Hassrede unterschiedlich definiert, nur auf spezielle Aspekte und Angriffsziele des Hasses eingegangen und auf verschiedene Onlineplattformen angewandt wird. Bis vor kurzem existierten in dem Bereich auch keine öffentlich zugänglichen Datenbestände zur Evaluation der Ergebnisse.[39] In älteren Arbeiten wurden deshalb keine einheitlichen, meist nicht nachvollziehbare Daten für die Auswertung herangezogen, womit der Vergleich mit diesen an Aussagekraft verliert. Für die deutsche Sprache fehlen solche Evaluationsets bisher gänzlich und auch Methoden zur Identifikation von Hassrede wurden nur für Englisch umgesetzt beziehungsweise evaluiert.

Eine Arbeit die sich erstmals umfassend und speziell mit dem Thema der Identifikation von Hassrede beschäftigt, wurde von Warner und Hirschberg publiziert. Darin wird nicht nur der Begriff Hassrede genau definiert und zu ähnlichen Begriffen wie Flaming abgegrenzt, sondern auch versucht den Hass, anders als in dieser Arbeit, in seinen unterschiedlichen Ausprägungen gegenüber verschiedenen Gruppierungen (anti-afrikanisch, anti-Immigranten, anti-muslimisch,...) mithilfe jeweils speziell für den Bereich angepassten Sprach-Modellen auszumachen. Auch hier wird wie in den beiden zuvor beschriebenen Ansätzen Supervised Machine Learning mithilfe einer Support Vector Machine(SVM) betrieben, jedoch werden zusätzlich Word Sense Disambiguation Techniken (Wortsinnunterscheidungstechniken) eingesetzt, um die positive oder negative Polarität von gewissen Wörtern zu bestimmen, welche als zusätzliches Merkmal in den Klassifikationsprozess einfließt. Anhand eines 1000 Paragraphen umfassenden Korpus, zusammengestellt aus Daten von Yahoo! und Inhalt von einschlägigen Webseiten, deren URLs der American Jewish Congress (AJC) zur Verfügung stellte, wurde eine Accuracy von 0,94, eine Precision von 0.68, ein Recall von 0.60 und ein F-Score von 0.63 evaluiert.[40]
Burnap et al. wählten einen ähnlichen Ansatz und untersuchten verschiedene Kombinationen von Classifier (SVM, Random Forest Decision Tree(RFDT), Bayesian Logistic Regression(BLR)) und lexikalische, syntaktische beziehungsweise Bag-of-Word (BoW)

Features im Zusammenhang mit der Identifikation von hasserfüllten und oder feindseligen Äußerungen gegenüber bestimmten Religionen oder ethnischen Gruppen auf Twitter als Folge von negativen Ereignissen die mit diesen Minderheiten in Zusammenhang stehen. Wie in dieser Diplomarbeit dienen als Datenbasis über einen Zeitraum von zwei Wochen gesammelte Tweets, von denen 2000 der insgesamt 450000 Postings von drei Personen annotiert wurden. Das optimale Setting resultiert dabei in einem gesamtheitlichen F-Score von 0,95. Die Identifikation von Hasspostings, unabhängig vom verwendeten Classifier, funktioniert jedoch nicht mit einer solch hohen Güte. Für diese Klasse beträgt der Wert des F-Scores nämlich lediglich 0,77.[41] Als Anstoß beziehungsweise Grundlage für das Verwenden von Typed Dependency Features und einem Lexikon mit hassbezogenen Wörtern diente die vorangegangene bereits erwähnte Forschung von Chen et al..[35] Aufbauend auf den gewonnenen Erkenntnissen wurde 2016 ein weiteres Paper von Burnap et al. veröffentlicht, in dem das Verfahren und die Features weiter verfeinert wurden, wodurch für die Klasse von Hasstweets der F-Score 0,68, jener für neutrale Tweets 0,98 und ein gesamter F-Score von 0,96 erzielt wurde.[42] Hier sei noch einmal hervorzuheben, dass Burnap et al. anders als viele vorangegangene Arbeiten Twitter als Datenursprung für seine Untersuchungen bezüglich der Hassposting-Erkennung heranzog und damit als Grundlage für diese Diplomarbeit noch interessanter wird. Ebenfalls wurde die gute Performance von Typed Dependency Features untermauert, weshalb diese auch in dieser Arbeit Anwendung finden und genauer evaluiert werden.
Zu einen der neuesten Veröffentlichungen zählt die Arbeit von Nobata et al., welche versucht die State-of-the Art Methoden und Feature-Sets der letzten Jahre zu kombinieren. Resultat ist eine Aufzählung einer großen Anzahl an Natural Language Processing(NLP) - Features, die in die Kategorien N-Gram, linguistisch, syntaktisch und distributional semantisch unterteilt wurden. Ausgehend von dieser fundierten Basis sollten weitere Verbesserungen erzielt und neue syntaktische und Comment Embedding - Features evaluiert werden. Dabei wurden auch die Schwierigkeiten in dem Bereich der Erkennung von offensiven Inhalten angeführt und die Missstände in Bezug auf die Vergleichbarkeit durch fehlende öffentliche Evaluationsets aufgezeigt. Daher wurde als zusätzliches Ziel der Arbeit definiert, einen solchen öffentlich zugänglichen Datensatz, welcher in Bezug auf Beleidigungen klassifizierte Benutzerbeiträge enthält, zur Verfügung zu stellen. Dieser stellt auch den ersten seiner Art dar und umfasst 2000 Kommentare. Als Datenbasis dafür wurden über ein Jahr lang insgesamt etwas mehr als eine Millionen Kommentare von Yahoo! Finanz und News gesammelt, wovon rund 85000 davon beleidigend waren. Nach Anwendung ihres vorgestellten Modells auf das Evaluationset konnte ein F-Score von 0.826 erreicht, Recall(0.825) und Precision(0.827) unterschieden sich davon minimal. Da der Korpus lediglich englische Texte beinhaltet, kann die Performance des im Zuge dieser Arbeit entwickelten Algorithmus jedoch nicht auf Basis dieses Sets evaluiert werden. [39] Waseem et al. erarbeiteten ebenfalls einen Korpus mit rund 16000 annotierten Tweets für die Hassrede Erkennung die aus einem Zeitraum von zwei Monaten stammten. Zusätzlich untersuchten sie die mithilfe eines Logistic Regression Classifier welche Arten von Features, die hasserfüllten Tweets ihres Korpus am besten identifizieren. Neben Bag-of-Word Features und Character N-Grams wurden Merkmale bezüglich der Wort-

und Tweetlänge, des Geschlechts des Verfassers und dessen Herkunft evaluiert. Ergebnis der Forschung war die Erkenntnis, dass Character N-Grams eine solide Basis für die Hassrede Identifikation liefern. Alleine mithilfe von Character N-Grams wurde ein F-Score von 0,7389 erzielt, weshalb diese auch in das Feature-Set dieser Arbeit aufgenommen wurden. Demographische Features, mit Ausnahme der Information über das Geschlecht, schaffen hingegen keinen Mehrwert. [43] Während der Erstellung der Diplomarbeit wurde eine weitere Forschung von Zia et al. veröffentlicht, die dem Thema dieser Arbeit äußerst ähnlich ist. Diese verfolgen ebenfalls den Ansatz mittels Supervised Machine Learning Algorithmus unter Anwendung drei verschiedener Classifier (SVM, Naive Bayes, k Nearest Neighbor -kNN) hasserfüllte Tweets mit Hilfe von Twitter-Netzwerkfeatures und Attribute durch Wortlisten, die positive oder negative Terme enthalten, zu erkennen. Im Gegensatz zu dem in dieser Diplomarbeit präsentierten Ansatz wird jedoch nicht versucht allgemein Hass zu erkennen sondern nur Hass, der auf gegensätzliche Religionen gründet. Eine weitere Ähnlichkeit besteht darin, dass Tweets über die Twitter API und einer Filterung nach gewissen Schlagwörtern beziehungsweise von bestimmten Seiten aufgezeichnet wurden, jedoch über einen Zeitraum von neun Monaten. Pro Klasse und Religion (Judentum, Christentum und Islam) wurden jeweils immer rund 2500 Tweets gesammelt. Um die Performance ihres Classifiers einzuordnen wurde eine 10-fache Kreuzvalidierung herangezogen, welche ergab, dass der SVM Classifier bei der Identifikation von religionsspezifischem Hass mit einem F-Score von durchschnittlich 0,944 am besten arbeitete. Aufgrund des ähnlichen Settings bietet sich diese Arbeit hervorragend als Referenz für die eigene Implementierung an der diese gemessen werden kann und wird deshalb im Zuge der Arbeit explizit für einen Vergleich herangezogen. Damit können mögliche Verbesserungen aufgrund der durchgeführten Erweiterungen im Bereich der Twitter Features und Anpassung eines speziellen Wörterbuchs ersichtlich gemacht werden.[44] Eine weitere aktuelle Veröffentlichung von Vigna et al. konzentriert sich auf die Erkennung von Hasspostings auf Facebook für die italienische Sprache. Verwendet wurden neben den üblichen Wort- und Character-N-Grams, Attribute durch eine Sentimentanalyse, Part-Of-Speech- beziehungsweise Dependency Features und ganze Wort-Embedding Lexikons. In diesem Zusammenhang wurde zusätzlich zu einem SVM-Classifier das rückgekoppelte(rekurrente) neurale Netz Long Short Term Memory (LSTM) für die Erkennung von Abhängigkeiten innerhalb eines Satzes eingesetzt. Vigna et al. wählte potentiell hasserfüllte Facebook-Seiten aus und sammelte so rund 17500 Kommentare von denen 6500 annotiert wurden. Interessant ist die Tatsache, dass der vorgestellte Algorithmus für eine Unterscheidung von Hass in eine schwache und starke Ausprägung nicht funktioniert. Die binäre Klassifikation liefert einen F-Score von 0,728 für Hasspostings und 0,838 für neutrale FB-Kommentare bei einem LSTM Classifier. Der SVM Klassifikator arbeitet mit F-Score Werten von 0,718 und 0,851 ähnlich.[45]

2.3.3 Feature-Generierung durch eine Netzwerkanalyse

Bezüglich der Nutzung von Erkenntnissen die durch die Netzwerkanalyse gewonnen werden, kann die Arbeit von Wadhwa et al. angeführt werden. Dabei liegt der Fokus jedoch nicht auf der Identifikation von Hasspostings sondern Mitgliedern radikaler Grup-

pen und terroristischer Organisationen innerhalb sozialer Netzwerke, die Eigenschaften von Netzwerken wie die Dynamik oder die enthaltenen Schlüsselknoten erst nach einem Clustering der Postings passiert.[29] Ting et al. verfolgen mit ihrem Einsatz von sozialen Netzwerkanalysetechniken ein ähnliches Ziel und versuchen sogenannte organisierte Hassgruppen auf Facebook, welche andere Nutzer im Bezug auf deren Eigenheiten beleidigen oder bestimmte Firmen verunglimpfen, zu erkennen. Im Detail werden aus gesammelten Daten solcher Gruppen mithilfe eines Graph-Analyse Tools (UCINet 6.0) Social Network Structure(SNS) Features extrahiert und für die Klassifikation von Facebook-Gruppen genutzt. Beispiele für solche Merkmale sind der Clustering Koeffizient, Zentralität, Dichte und die durchschnittliche Pfadlänge. Verbesserung konnte durch diese Art der Features, die für ihre Kalkulation viel Zeit in Anspruch nehmen, jedoch keine erzielt werden, weshalb eine solche Art von Netzwerkfeatures in dieser Arbeit nicht einbezogen wurden.[46]

2.3.4 Linguistic Inquiry and Word Count (LIWC) Features zur Textklassifikation

Für den Einsatz von LIWC im Bereich der Erkennung von Hasspostings oder anderen Lexika, die sich einer Taxonomie bedienen, konnten keinerlei relevanten Forschungen gefunden werden, wodurch gerade dessen Evaluierung als sehr interessant erscheint. Im Kontext von Machine Learning und speziell Textklassifikation wurden jedoch sehr wohl Features durch das LIWC-Dictionary und mögliche Erweiterungen davon untersucht. Mohtasseb et al. beispielsweise verwendet LIWC um die Knoten der ConceptNet Ontologie zu annotieren. Sie versprechen sich dadurch im Textklassifikationsprozess die LIWC-Kategorie nicht von isolierten einzelnen Wörtern, sondern von sogenannten Konzepten zu betrachten beziehungsweise je Kontext richtige LIWC-Features zu extrahieren. Als Ergebnis der Kombination aus LIWC und ConceptNet wird die Ontologie mit dem Namen PsychoNet präsentiert, die einer Erweiterung des LIWC um eine semantische Ebene entspricht. Aufgrund der Relationen innerhalb der Ontologie können Funktionen umgesetzt werden, die zusätzliche Merkmale aus dem Text generieren. Beispielsweise lassen sich Rückschlüsse auf die Kategorien der Wörter des Kontextes und der übergeordneten Terme ziehen oder die psycho-linguistische Ähnlichkeit zu anderen Konzepten feststellen. Im Rahmen ihrer Arbeit wurden Texte im Bezug auf deren Stimmungslage klassifiziert und ist daher dem Thema dieser Master-Thesis ähnlich.[47][48]

2.4 Zusammenfassung

In diesem Kapitel wurde die mittlerweile weite Verbreitung von Hasspostings in Österreich aufgezeigt und versucht Hasspostings beziehungsweise Hassrede unter Bezugnahme mehrerer Perspektiven, wie der rechtlichen, politischen, sozialen oder sprachlichen. Weiters wurde der State-of-the-Art von Kernelementen, die in dieser Diplomarbeit behandelt werden, vorgestellt. Im Detail wurde allgemein der Stand der Technik für die Erkennung offensiver Inhalte und darauf aufbauend speziell die gängigen Methoden zur Hassrede Identifikation ermittelt und aufgeschlüsselt. Aufgrund der hohen Bedeutung von LIWC

und Features auf Basis des Twitter Netzwerks, wurden für diese Themengebiete aktuelle Forschungsarbeiten aufbereitet.

KAPITEL 3

Methodik

3.1 Einleitung

Das Kapitel “Methodik“ soll einen Überblick über die Methoden, Technologien und die Schritte geben, die notwendig sind, um aus rohen Twitter-Daten ein Modell zu erstellen, welches eine optimale Klassifikation von Hasspostings ermöglicht. Nachfolgend werden im Detail die Umsetzung des zugrundeliegenden *Supervised Machine Learning Algorithmus* und dessen Teilabschnitte wie die Annotation von Tweets, das Preprocessing dieser, die Merkmale der Tweets und User, die letztendlich die Grundlage für die Klassifikation darstellen, und die ausgewählten Klassifikatoren beschrieben. Weitere Hilfsmittel, wie das Lexikon befüllt mit Hasswörtern oder das Datenmodell und die Sammlung der Twitter-Daten, die in dieser Arbeit von Nöten sind, sind ebenfalls in diesem Kapitel veranschaulicht.

3.2 Supervised Machine Learning Algorithmus

Für das Klassifizieren der Hasspostings wird ein *Supervised Machine Learning* Ansatz herangezogen. Darunter versteht man einen Algorithmus der versucht aus bereits manuell annotierten Trainingsdaten eine Funktion beziehungsweise Modell zur Klassifikation von neuen Objekten zu inferieren.[49]

Die dafür benötigten Daten werden aus Twitter extrahiert und anhand des Inhalts, dementsprechend in Klassen eingeteilt. Der Algorithmus extrahiert anschließend aus diesen Daten, die anhand mehrerer Preprocessing-Schritte aufbereitet werden, Feature-Vektoren, welche mit der Information über die Klasse, dem “Supervisor“-Element, die Trainingsobjekte darstellen. Aus diesen Trainingsdaten wird vom jeweils verwendeten Classifier eine Mapping-Funktion $\mathbf{f}$ erzeugt, die versucht aus Feature-Vektoren $\mathbf{X}$ von

neuen Tweets die korrekte Klasse **Y** zu bestimmen 3.1.

$$Y = f(X) \tag{3.1}$$

Ziel ist es diese Mapping-Funktion beziehungsweise dieses Modell für die Bestimmung der Klassen und die Auswahl der Features so zu wählen, dass der Klassifikationsprozess optimal ist. Abbildung 3.1 zeigt eine Übersicht über die zuvor beschriebenen Begriffe und den groben Prozess des Supervised Machine Algorithmus.

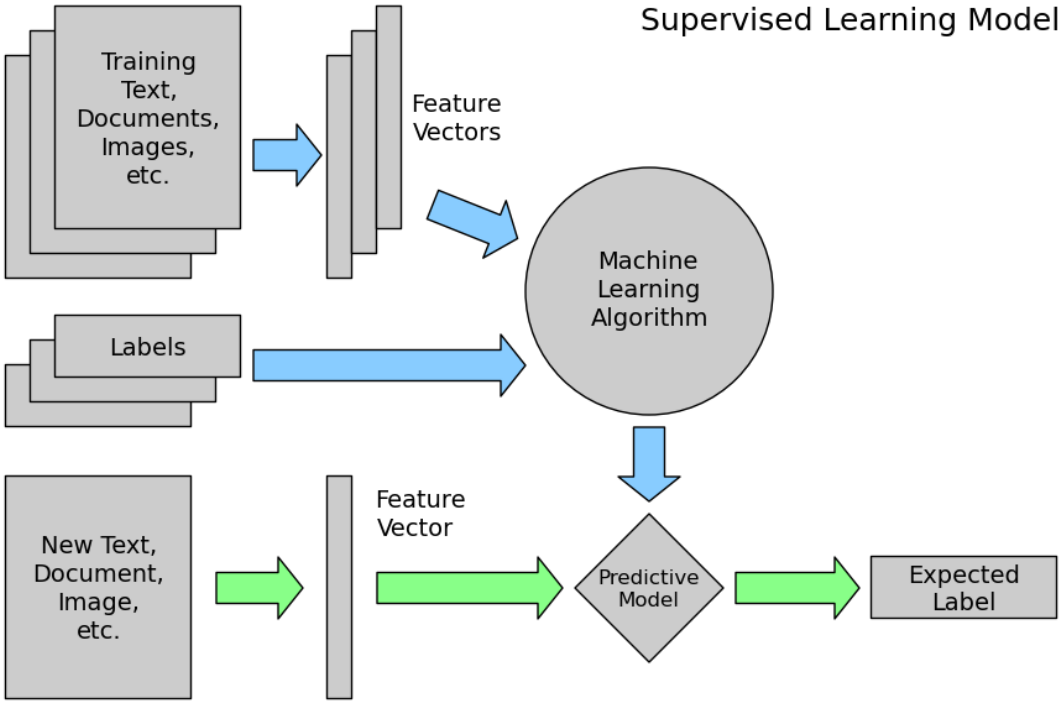


Abbildung 3.1: Supervised Machine Learning Prozess.[1]

3.3 Asymmetrische Kostenanalyse von False Positve- und False Negative Klassifikationen

Durch die Kostenanalyse soll festgestellt werden wie False Positive und False Negative Klassifikationen ins Gewicht fallen beziehungsweise welche Kosten für die jeweilige Fehlklassifikation anfallen und wie diese im Verhältnis stehen. Die Tabelle 3.1 listet die möglichen Kostenursachen, die je Fehler bei einer Filterung durch Twitter auftreten können. Dabei wird auch zwischen einem vollautomatischen und halb-automatischen Ansatz, bei dem Mitarbeiter die vorgefilterten Tweets nochmals im Bezug auf Hassinhalte überprüfen, unterschieden.

[1] http://www.astroml.org/sklearn_tutorial/general_concepts.html

Kostengründe für Falschklassifikationen	**automatisch**	**semi-automatisch**
False Negative (FN)	-manuelle Bearbeitung bei nachträglicher Meldung -psychische Kosten -schlechte Reputation	
False Positive (FP)	-Zensur -Kundenverlust -schlechte Reputation -(Wiederherstellung)	-manuelle Bearbeitung

Tabelle 3.1: Mögliche Kosten für eine falsche Klassifikation je Ansatz

Bei einer False Negative Klassifikation fallen bei beiden Methoden zur Filterung von Hasspostings auf Twitter die gleichen Kosten an. Dabei handelt es sich einerseits um Kosten, die für den Einsatz von Mitarbeitern für die nachträgliche Bearbeitung von nicht-erkannten Hasspostings, welche von User gemeldet werden, entstehen. Weiters ergeben sich auch immaterielle Kosten, wie psychische Belastungen für den Betroffenen bei Beleidigungen. Ebenfalls kann sich die von Nutzern wahrgenommene Nichtlöschung hasserfüllter Inhalte auf die Reputation von sozialen Netzwerken auswirken. Beispielsweise verpflichtete sich Facebook und Twitter zur Löschung solcher Beiträge innerhalb von 24 Stunden.[2] Die Nichteinhaltung wirft ein schlechtes Licht auf die sozialen Netze, gibt den Leuten das Gefühl, dass nur die Gewinnmaximierung im Vordergrund steht und führt in späterer Folge zu Kunden und Werbepartnerverlusten.[3] Hier ist jedoch zu betonen, dass all diese Kosten bei keiner oder einer (ungenügenden) rein manuellen Filterung, wie es derzeit der Fall ist, aufgrund der weit größeren Anzahl an vorerst veröffentlichten Hasspostings, um ein Vielfaches höher sind.

Bei False Postive Klassifikationen muss die automatische und halb-automatische Filterung differenzierter betrachtet werden. Im Fall eines vollautomatischen Löschungsverfahren könnte man sich bei vielen Fehlklassifikationen von neutralen Postings in Richtung Zensur begeben, der Ruf könnte also laut werden, dass versucht wird bestimmte Informationen, die gegen keinerlei Richtlinien verstoßen, den Nutzern vorzuenthalten. Dies könnte soweit gehen, dass eine störungsfreie Konversation oder sachliche Diskussion nicht mehr möglich wäre und User in großen Mengen mittelfristig andere Plattformen nutzen. Generell ist es für einzelne Nutzer ärgerlich wenn ihre offensichtlich neutralen Beiträge (wiederholt) gelöscht werden. Deren durchschnittliche Verweildauer auf dem sozialen Netzwerk könnte bei einem solchen Vorfall kurzfristig drastisch sinken, bei wiederholten Vorfällen zu einer bereits erwähnten Abwanderung des Users führen. Durch die angeführten Probleme ergeben sich wiederum Einbuße der Werbeeinnahemen aufgrund geringerer Nutzer

[2] https://www.tagesschau.de/inland/facebook-hasspostings-loeschen-101.html - Tagesschau
[3] http://derstandard.at/2000021533588/Pro-und-Kontra-Facebook-Hasspostings-loeschen

beziehungsweise langfristig der Verlust von Werbepartnern. Bei einer geplanten Wiederherstellungsfunktion könnten durch die notwendige manuelle Validierung zusätzliche Mitarbeiterkosten anfallen, die im Extremfall für jedes gefilterte Hasskommentar anfallen. In einem solchen Fall sollte deshalb gleich eine semi-automatische Methode eingesetzt werden. Allgemein kann dadurch durchaus angenommen werden, dass durch die "Zensur" von einem neutralen Posting höherer Schaden entsteht, als die Nichtfilterung eines Hasspostings.

Bei einem halb-automatischen Ansatz würden sich die Gewinneinbuße durch Zensur erheblich senken, da ein geschulter Mitarbeiter noch einmal den vorgefilterten Beitrag bewertet und auf jeden Fall eindeutige False Positives beseitigt. Jedoch können aufgrund der Unschärfe bei der Definition von Hassrede weiterhin falsche Löschungen durchgeführt werden. Dies könnte ebenfalls zu oben genannten Problemen führen, jedoch in einem wahrscheinlich weit geringerem (beziehungsweise vernachlässigbaren) Ausmaß. Vielmehr summieren sich die Kosten durch die zusätzlichen Gehälter von Mitarbeitern für diese Aufgabe.

Nachfolgend werden die Kosten für eine fehlerhafte Klassifikation berechnet, indem die Kosten für Mitarbeiter je Beitrag und die entgangenen Werbeeinnahmen durch kurzfristige Kundenverluste ermittelt werden. Die psychischen Kosten, die einem Betroffenen durch beleidigende Inhalte widerfahren, werden hier nicht berücksichtigt, da diese sehr schwer zu quantifizieren sind und für eine einzige FN-Klassifikation, die durch die Filterung ohnehin zu Einzelfällen werden sollten, vernachlässigbar sind. Zusätzlich sei erwähnt, dass solche psychischen Kosten durch gar keine Filterung, wie es derzeit der Fall ist, mit Sicherheit ein Vielfaches ausmachen.

Personalkosten
Als Gehalt für einen Mitarbeiter eines Lösch-/Validierungsteams wird rund 1700 Euro brutto angesetzt, welches rund 200 Euro über dem Mindestlohn von Deutschland liegt. Grund für diese Annahme war ein entsprechender Artikel der "Süddeutschen Zeitung", welcher von einem Gehalt knapp über dem Mindestlohn für Mitarbeiter des Berliner Facebook Lösch-Teams berichtet.[4] Insofern wurde diese Information auf Twitter Mitarbeiter umgelegt. Zusätzlich decken sich jene 1700 Euro dem durchschnittlichen Einkommen eines Call Center Mitarbeiters in Berlin, dessen einfache Tätigkeit dem eines Löschteam-Mitarbeiters, welcher in gewisser Weise ebenfalls Kundenanfragen bearbeitet indem er "gemeldete" Kommentare bewertet, nahe kommt.[5]

Weiters wird angenommen, dass es einer Person möglich ist innerhalb von *10* Sekunden ein Posting zu klassifizieren. Basierend auf den Informationen in den nachfolgend angeführten Berichten steht Facebook-Moderatoren auch gar keine längere Zeit für das Bewerten von Postings zur Verfügung.[4,6,7] Rechnet man eine Leerzeit von *2* Sekunden ein, schafft es

[4] http://www.sueddeutsche.de/digital/exklusive-sz-magazin-recherche-inside-facebook-1.3297138
[5] https://www.gehaltsvergleich.com/gehalt
[6] https://www.theguardian.com/news/2017/may/21/facebook-moderators-quick-guide-job-challenges
[7] http://oe3.orf.at/stories/2844796/

ein Mitarbeiter maximal rund *5* Tweets pro Minute, *300* pro Stunde, *2400* pro Tag(8 Stunden), *11550* pro Woche (38,5h) beziehungsweise *46200* pro Monat zu bewerten.

Dies würde pro Posting, bei dem zuvor angenommenen Gehalt, Personalkosten von rund *0,0367965* Euro (*~4* Euro Cent) ausmachen. In US-Dollar wären das bei einem Umrechnungssatz von 1,07495 (Stand 7.2.2017) *0,0395544* US-Dollar oder *~3,9554¢*.

Dabei muss aber beachtet werden, dass anders als im Fall von False Positives ein False Negative nicht zwangsweise zu Personalkosten führt, da nur ein Bruchteil davon gemeldet werden und deshalb von Mitarbeitern analysiert werden müssen. Laut Twitter Transparenzbericht wurden von staatlichen Organisationen im ersten Halbjahr 2016 5195 Löschgesuche eingebracht, 761 entfielen dabei auf Gerichte, 4434 auf Behörden und Polizei.[8] Diese Zahl erscheint auf dem ersten Blick sehr niedrig, hat sich jedoch gegenüber dem Zeitraum des Vorjahres mehr als verfünffacht, was im deutschen Raum auch auf die zu diesem Zeitpunkt aufkommende Hassrede-Debatte zurückzuführen ist.[9] Leider liegen jedoch keine Daten vor, wie viele private Personen die Option zum Melden von strafrechtlich relevanten beziehungsweise gegen die Verhaltensregeln von Twitter verstoßende Tweets in Anspruch nehmen. Es wird daher angenommen, dass persönliche Beleidigungen in einer Konversation vom betroffenen User mit hoher Wahrscheinlichkeit gemeldet werden und auch Hassrede aufgrund der erhöhten Sensibilität der Gesellschaft im Bezug auf dieses Thema vermehrt gemeldet wird.

Deshalb wird für die nachfolgende Kalkulation die optimistische Annahme getroffen, dass rund 20% aller Hasspostings von User gemeldet werden und deshalb pro nicht identifizierten Hassbeitrag auch nur 20% der zuvor berechneten Personalkosten für eine False Positive Klassifikation festgelegt. Dies würde je False Negative Klassifikation Personalkosten von *0,00791088* US-Dollar (*~0,7911¢*) verursachen.

Einnahmeentgang durch Kundenverlust
Twitter erwirtschaftet den größten Teil seines Umsatzes durch Werbung und zusätzlich durch den Verkauf von Lizenzen an der kommerziellen Nutzung und Analyse seiner Daten. Die Werbeeinnahmen erfolgen durch gesponserte Tweets, die in der Nachrichtenliste von Nutzern angezeigt werden, gesponserte Accounts, welche den User vorgeschlagen werden und durch gesponserte Trends, die in der Trendliste angezeigt werden. Die ersten beiden Optionen generieren jedoch nur Einnahmen, wenn der User mit dem Tweet oder dem Account interagiert, also entweder antwortet beziehungsweise einen Retweet erstellt oder dem Account folgt. Gesponserte Trends müssen von Kunden unabhängig von User bezahlt werden. 2013 wurde laut Medienberichten von Twitter bis zu *200000* US-Dollar pro Tag dafür in Rechnung gestellt.[10] Im dritten Quartal von 2016 wurden von einem Umsatz von 615,934 Millionen US-Dollar insgesamt 544,966 Millionen US-Dollar durch Werbung generiert und 70,968 Millionen durch den Verkauf von Daten. Im selben Zeitraum waren

[8] https://transparency.twitter.com/en/removal-requests.html
[9] https://netzpolitik.org/2016/twitter-transparenzbericht-mehr-auskuenfte-mehr-loeschgesuche-mehr-copyright-takedowns/
[10] http://blogs.faz.net/netzwirtschaft-blog/2013/10/17/einnahmequellen-von-twitter-3557/

durchschnittlich 317 Millionen User aktiv.[11]

Legt man diese Umsätze nun auf einen einzelnen Nutzer um ergeben sich *1,719* US-Dollar pro User pro Quartal, *0,573* US-Dollar pro User pro Monat und *0,019* US-Dollar pro User pro Tag an Werbeeinnahmen. Diese Zahl entspricht auch in etwa dem Werbewert vom ersten Quartal 2014 bei dem pro 10 Timeline Ansichten *0,0144* US-Dollar berechnet wurden.[12] Durch eine ungerechtfertigte Filterung(Zensur) könnte sich das Userverhalten wie bereits erwähnt negativ auswirken und Twitter beziehungsweise die User-Timeline nicht mehr (so häufig) besucht werden. Es wird daher vereinfacht angenommen, dass die betroffenen Nutzer Twitter an solchen Tagen durchschnittlich halb so oft in Anspruch nehmen als im Normalfall. Die entgangenen Werbeeinnahmen würden sich dann je Tweet auf *0,0095508* US-Dollar (*~0,9551¢*) belaufen, wobei hier die zugegebenermaßen vereinfachte Prämisse aufgestellt wird, dass pro Tag und User nur eine False Positive Filterung auftritt. Die nächste falsche Filterung für diesen User an diesem Tag würde nämlich in diesem Modell wiederum nur ein Viertel an Entgang der Tageseinnahmen verursachen, die nächste ein Achtel und so weiter.

Umsätze aus Datenlizenzen generiert Twitter *788533* US-Dollar pro Tag, wobei bei den bereits in der im Abschnitt Problemstellung 1.2 erwähnten 500 Millionen Tweets täglich gesagt werden kann, dass rund *0,0015771* US-Dollar (*~0,1577¢*) pro Tweet bezahlt werden. Pro User würden sich diese Einnahmen aus Datenverkäufen auf *0,002485* US-Dollar pro Tag belaufen. Da jedoch Datenlizenzen nicht pro Tweet beziehungsweise User gekauft werden, kann nicht so einfach ein Einnahmeentgang auf einen Tweet oder User umgelegt werden. Trotzdem kann davon ausgegangen werden, dass der Verkauf an Daten rückläufig sein wird, falls die Anzahl an False Positive Filterungen hoch ist und mögliche Analyseergebnisse der Käufer verzerrt werden, weshalb die angeführten Kosten pro Tweet bei einem zu unrecht gefiltertetn Tweet als Einnahmeentgang angenommen werden.

Insgesamt würde sich der anteilige Einnahmeentgang pro False Positive Klassifikation durch die Zensur hervorgerufene Verminderung des Werbeeinnahmepotenzials beziehungsweise durch den Verlust an Käufern für Daten auf *0,011127866* US-Dollar oder rund *1,1128¢* summieren. Für die semi-automatische Filterung wird der Einnahmeentgang vernachlässigt, da dieser aufgrund der doppelt durchgeführten Analyse verschwindend gering ist. Trifft man die Annahme, dass der Algorithmus mit eine FP-Rate von 4% und ein geübter Mitarbeiter mit einer FP-Rate von 2% arbeitet , beträgt die kombinierte FP-Rate 0,0008 und würde pro Tweet einen prozentualen Anteil von *0,000008902* US-Dollar oder circa *0,0004451¢* an Umsatzentgang ergebe, weshalb dieser in weiterer Folge in den Berechnungen nicht weiter berücksichtigt wird.

Tabelle 3.2 fasst die Kosten pro False Positive und False Negative Klassifikation nochmals zusammen. Die Werte werden dabei in gerundeten US Cents angegeben.

Aus den berechneten Kosten würde sich also im Fall einer automatischen Filterung von *0,79109¢* zu *1,1128¢* ein Verhältnis zwischen FN und FP Klassifikation von 1:1,41

[11] http://finanzmarktwelt.de/twitter-quartalszahlen-4-45295/
[12] https://investor.twitterinc.com/releasedetail.cfm?releaseid=843245

Kostensätze für die Fehlklassifikationen pro Tweet in ¢		**automatisch**		**semi-automatisch**
False Negative Error (FN)	Personalkosten	0,79109 ¢		
	Einnahmeentgang	-		
	Gesamtkosten	**0,79109 ¢**		
False Positive Error (FP)	Personalkosten	-		3,95544 ¢
	Einnahmeentgang	Werbung	0,95508 ¢	vernachlässigbar
		Datenverkauf	0,15771 ¢	
	Gesamtkosten	**1,11279 ¢**		**3,95544 ¢**

Tabelle 3.2: Kostensätze für eine falsche Klassifikation je Ansatz

beziehungsweise eine Gewichtung von 42% zu 58% ergeben. Bei der semi-automatischen Filterung sind die Personalkosten pro FP von *3,95544¢* natürlich sehr hoch, würden aber, wie bereits erwähnt, den Vorwurf der willkürlichen Zensur und den damit verbundenen mittel- bis längerfristigen Abgang von Kunden und vor allem User weiter einschränken beziehungsweise ganz zerstreuen, da eindeutige FP nicht mehr vorkommen und Twitter-Verhaltensregeln besser angewandt werden können. Bei diesem Ansatz beträgt das Kosten-Verhältnis zwischen FN und FP 1:5, die Gewichtung 17% zu 83%.

Um nun diese, zumindest bei einem halb-automatischen Ansatz, doch große Asymmetrie zwischen den Fehlertypen abzubilden, wurde der später beschriebene annotierte Tweet-Korpus so angelegt, dass Hass und neutrale Postings zu ungefähr gleichen Teilen enthalten sind. Dieser Umstand entspricht natürlich nicht den realen Verteilungsverhältnissen, bei denen Hasspostings einen viel geringeren Anteil annehmen. Dieses Oversampling der Hasspostings soll jedoch dazu führen, dass dieser Klasse im Zuge des Maschinen Learning Algorithmus höhere Bedeutung zu Teil wird.

Das Problem von Standard-Klassifikatoren ist jedoch, dass für FP und FN Klassifikationen die selben Kosten angenommen werden. Der Grenzwert (engl. Threshold), ab wann ein Tweet als Hassposting eingestuft wird, beträgt in diesem Fall 0,5. Das heißt ab einer Wahrscheinlichkeit von mehr als 50%, dass es sich bei einem Beitrag um ein Hassposting handelt, wird er vom Algorithmus als solches festgelegt. Im Kapitel Evaluierung 4.6 wird später deshalb der optimale Threshold, bei dem die Kosten minimal sind, für die Classifier und ihre besten Feature-Sets mit Hilfe einer Cost-Benefit Analyse von Weka berechnet.

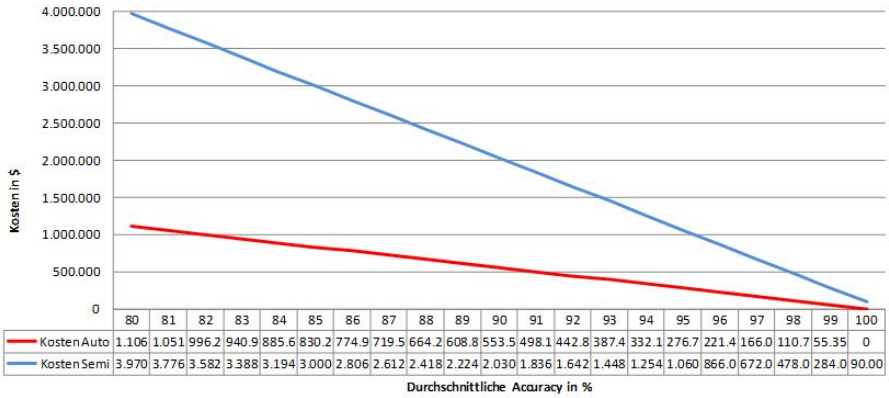
3.4 Zielsetzung

Nachdem nun die Kosten für die unterschiedlichen Fehlklassifikationen ermittelt wurden, kann auf Basis dieser Schätzung eine Zielsetzung für den resultierenden Supervised

Machine Learning Algorithmus getätigt werden, damit dieser kosteneffizient arbeitet und auch für die Praxis eingesetzt werden kann.

Da jedoch keine Informationen von sozialen Netzwerken über die tägliche Menge an verfassten Hasspostings vorliegen beziehungsweise nicht exakt abgeschätzt werden kann, wird mittels einer Hochrechnung versucht das Ausmaß von hasserfüllten Postings pro Tag einigermaßen zu evaluieren. In einer Untersuchung von 500 Postings, die mittels neutralen häufig vorkommenden Wörtern gesammelt wurden und vom selben Tag stammen, konnten 9 Hasspostings ausgemacht werden. Dies würde exakt 1,8% an Hasspostings ausmachen. Würde man wieder eine tägliche Menge von rund 500 Millionen Tweets täglich annehmen wären das 9 Millionen hasserfüllte Beiträge täglich. Bei einem Algorithmus mit einer Accuracy von 90% wären das insgesamt 50 Millionen Fehlklassifikationen(10%), wobei dies 49,1 Millionen an FP und 900000 an FN ausmachen würde, wenn der Algorithmus für beide Klassen gleich gut funktioniert. Dies würde bei den zuvor berechneten Kostensätzen 553499,7$ bei einem automatischen beziehungsweise 1949240,85$ bei einem halb-automatischen Ansatz pro Tag verursachen. Bei der semi-automatischen Methode kämen zusätzliche Personalkosten von 320390,64$ für die erneute Begutachtung gefilterter Hasspostings(8,1 Millionen) hinzu. Zugegebenermaßen ist diese Annahme für die FP-Rate ziemlich hoch und wird in der Praxis geringer ausfallen. Aufgrund der hohen Anzahl an neutralen Postings im Vergleich zu Hasspostings, würde man zusätzlich den Grenzwert für eine Hassposting-Klassifikation sehr hoch ansetzen, also einen Tweet erst dann als hasserfüllt einstufen, wenn dies mit hoher Sicherheit gegeben ist. Diese extreme Hypothese soll jedoch zeigen, dass bei einer vermeintlich guten Accuracy von 90% bei diesen FN- und FP-Raten schon sehr hohe Kosten anfallen würden und das System nicht wirtschaftlich wäre.

Das Diagramm in Abbildung 3.2 zeigt die anfallenden Kosten bei unterschiedlicher Performance des Algorithmus für beide Methoden bei gleichen FN- und FP-Raten je Klasse. In den Diagrammen 3.3, 3.4 und 3.5 werden jeweils die Verläufe der FN-



	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100
Kosten Auto	1.106	1.051	996.2	940.9	885.6	830.2	774.9	719.5	664.2	608.8	553.5	498.1	442.8	387.4	332.1	276.7	221.4	166.0	110.7	55.35	0
Kosten Semi	3.970	3.776	3.582	3.388	3.194	3.000	2.806	2.612	2.418	2.224	2.030	1.836	1.642	1.448	1.254	1.060	866.0	672.0	478.0	284.0	90.00

Abbildung 3.2: Kostenverlauf für semi- und automatischen Filterung bei gleichen FN- und FP-Raten.

Kosten, der FP-Kosten bei automatischer Filterung und der FP- und TP-Kosten bei semi-automatischer Filterung bei unterschiedlicher Accuracy dargestellt. Durch diese entkoppelte Betrachtungsweise der Accuracy für neutrale und Hasspostings können auch die Kosten bei ungleichen FN- und FP-Raten leicht ermittelt werden.

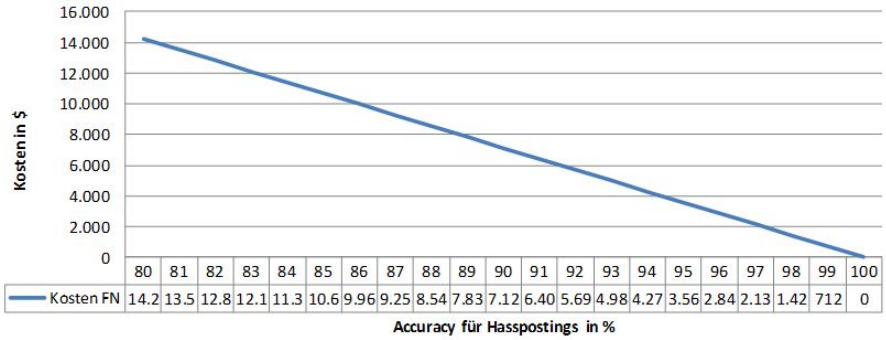


Abbildung 3.3: Kostenverlauf bei automatischer Filterung für FN-Klassifikationen.

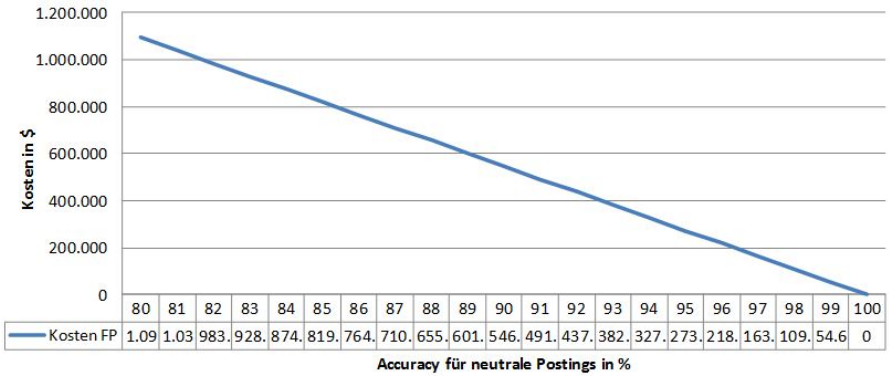


Abbildung 3.4: Kostenverlauf bei automatischer Filterung für FP-Klassifikationen.

Generell kann gefolgert werden, dass die FP-Raten sehr gering gehalten werden müssten, da tägliche Kosten von mehr als einer halben Millionen US-Dollar für die Filterung von Hassinhalten auch für umsatzstarke Unternehmen wie Twitter oder Facebook kaum in Frage kommen, unabhängig ob dies eine positivere Wahrnehmung ermöglichen würde. Bezieht man sich auf die errechneten Werte aus den Diagrammen müsste die Accuracy bei der Identifikation von neutralen Postings, ausgehend von einem automatischen Ansatz bei mindestens 91%, bei einem semi-automatischen schon bei 98% liegen, damit die Kosten nicht die halbe Millionengrenze überschreiten. Die Kosten für eine FN-Klassifikation fallen im Gegensatz dazu bei einer vergleichsweise geringeren Accuracy für die Hassposting-Klasse von beispielsweise 80% mit rund 14200$ kaum ins Gewicht.

Die Zielsetzung für den konzipierten Algorithmus muss es sein, von der Kostenperspektive aus gesehen, neutrale Beiträge mit einer Accuracy von mindestens 91% zu erkennen.

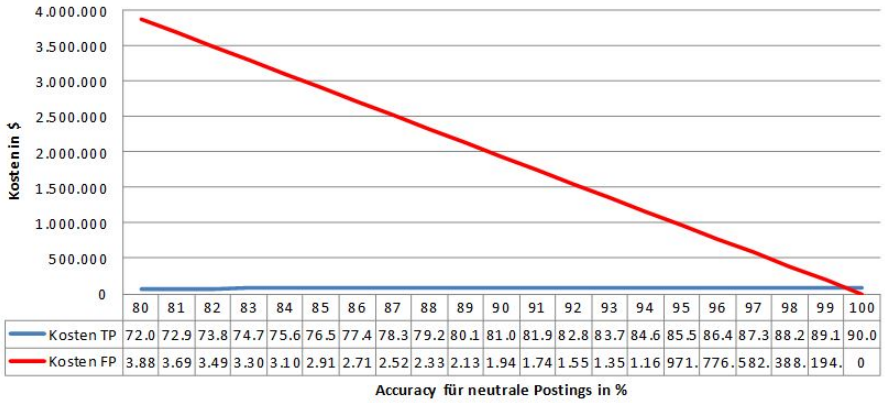


Abbildung 3.5: Kostenverlauf bei semi-automatischer Filterung für FN- und FP-Klassifikationen.

Andernfalls ist an einem wirtschaftlichen Einsatz nicht zu denken. Zusätzlich ist es natürlich Ziel und Reiz der Arbeit, dass mit dem resultierenden Algorithmus die Accuracy für die Identifikation von Hasspostings so hoch wie möglich ist, weshalb deren Zielwert auch mit mindestens 90% angesetzt wird. Bei einem Praxis Einsatz wäre eine Accuracy größer 80% aus wirtschaftlicher Sicht zwar ebenfalls tragbar, der Reputationsverlust, der aufgrund der Nichterreichung selbstauferlegter Limits (Filterung innerhalb von 24 Stunden etc.) entsteht, wäre dadurch natürlich umso höher. Beim User würde das subjektives Gefühl aufkommen, dass hasserfüllte Postings gar nicht beziehungsweise unzureichend gefiltert werden und in weiterer Folge würde das Vertrauen in das Unternehmen sinken.

3.5 Datensatz

3.5.1 Twitter-Datensammlung

Als Datenquelle für den Machine Learning Algorithmus wurden Kommentare von Twitter verwendet. Twitter eignet sich für das Sammeln von Daten dahingehend sehr gut, da nicht nur das Echtzeit-Streaming von öffentlichen Tweets über eine API mit sämtlichen Filterparametern ermöglicht wird, sondern auch einzelne Anfragen über die Twitter Rest API, um spezifische Informationen zu erhalten, möglich sind. Anders als die Streaming API unterliegt die Rest API im Bezug auf die Anzahl der übergebenen Parameter beziehungsweise der Requests jedoch genau definierten Limits, wodurch diese in dieser Arbeit nur für die nachträgliche Gewinnung von Netzwerkinformationen für ausgewählte Benutzer verwendet wird. Zur erleichterten Nutzung der Programmierschnittstellen wird die Java-Bibliothek Twitter4j[13] verwendet. Nachfolgend wird die detaillierte Verwendung der angebotenen APIs im Kontext dieser Arbeit offengelegt.

[13] http://twitter4j.org/en/index.html

Twitter Streaming API

Die Gewinnung der rohen Tweets und inkludierten Metainformationen, also dem Großteil der benötigten Daten, erfolgte über die Streaming API. Die Filteroptionen erlaubten es hier als Sprache der Tweets Deutsch auszuwählen und eine Liste an Keywörtern zu übergeben, nach denen die geposteten Beiträge gefiltert werden konnten. Der Inhalt muss also in deutscher Sprache formuliert worden sein und zumindest ein Wort aus dieser Liste in jedem Beitrag enthalten sein, um gecrawlt zu werden. Die Grundlage für die Aufzählung an Wörtern nach denen getrackt wird, bietet ein modifiziertes Hasswort-Lexikon, welches auch für die Extraktion einiger Features verwendet wurde. Dadurch konnte die Wahrscheinlichkeit, dass es sich bei dem gefilterten Tweet um ein Hassposting handelt, erhöht werden, wodurch in weiterer Folge die Dichte an Hasspostings im Datenbestand ansteigt. Zusätzlicher Vorteil durch diese Vorgehensweise ist, dass Beiträge, welche trotz einschlägiger Wörter als neutrale Postings angesehen werden können, im fertigen Korpus enthalten sind und der Algorithmus auch für solche Spezialfälle trainiert werden kann. Damit jedoch nicht nur solche Einzelfälle berücksichtigt werden, wurde der Datensatz um solche Tweets erweitert, welche vom Stream ohne die Angabe von hasserfüllten Keywörtern empfangen worden sind. Da Twitter es jedoch nicht erlaubt die Sprache als einzige Restriktion für den Streaming Prozess festzulegen, wurde eine Liste mit den 100 häufigsten deutschen Wörtern als zusätzlicher Filter verwendet[14], um ein möglichst breites Spektrum an Tweets zu erhalten.

Twitter Rest API

Mit der von Twitter zur Verfügung gestellten Rest API lassen sich gezielt Informationen rund um bestimmte User oder zu in der Vergangenheit verfasste Tweets extrahieren. Aufgrund dieser Möglichkeiten war die REST API für das Vervollständigen des Datensatzes bedeutend.
Im Zuge dieser Arbeit wurde sie im ersten Fall dazu verwendet, um die Anzahl an Retweets und Likes der Beiträge zu aktualisieren, da sich diese auch nach dem Zeitpunkt des Live-Crawlings verändern können. Im Detail wurde die *GET statuses/lookup - REST API* benutzt, welcher bis zu 100 Tweet-IDs als Parameter übergeben werden können und als Ergebnis alle assoziierten Daten pro ID liefert. Die Anzahl der Anfragen sind dabei auf 180 pro 15 Minuten limitiert[15].
Da Informationen bezüglich Follower beziehungsweise Freunden von Nutzern von der Streaming API nicht zur Verfügung gestellt werden, mussten diese Daten ebenfalls mithilfe einer geeigneten Schnittstelle nachträglich gesammelt werden. Für diese Angelegenheit bot sich die *GET followers/ids - REST API* an, welche in 5000er Schritten die Follower-IDs zu einem bestimmten Benutzer, dessen ID als Parameter übergeben wurde, zur

[14] http://www.thegermanprofessor.com/top-500-german-words/
[15] https://dev.twitter.com/rest/reference/get/statuses/lookup

Verfügung stellt. Die Anzahl der Anfragen sind hier auf 15 pro 15 Minuten beschränkt[16]. Zusätzlich wurden während dieses Prozesses auch alle Daten von neuen User über die *GET users/lookup - REST API* abgefragt und in die Datenbank eingespeist. Hier beträgt das Rate Limit 180 Requests pro 15 Minuten [17].

Ergebnis der Datensammlung

In einem Zeitraum von rund acht Tagen, genauer gesagt im Zeitraum von 26.05.2016 um 17:08 bis 04.06.2016 um 09:45, wurden 75797 Tweets, die ein hasserfülltes Wort beinhalten, gesammelt. Die Größe der Datenbank macht circa 36116 KB aus. Zusätzlich wurden dazu 3876 Bilder mit einem Speichervolumen von 173 MB extrahiert. Die Daten der 15940 Tweets die durch das Crawling von Beiträgen mittels neutralen häufig vorkommenden Wörtern hinzugefügt wurden, umfassen 42401 KB und wurden am 09.07.2016 im Zeitraum von 11:56 bis 17:59 gesammelt. Zu diesen Postings wurden 1850 Bilder mit einem Speichervolumen von 180 MB gespeichert.

3.5.2 Datenmodell

Um die von Twitter gesammelten Daten möglichst effizient für die weitere Bearbeitung zu speichern, wurde folgende Datenbankstruktur entworfen, welche aus neun Tabellen besteht, die in diesem Kapitel genauer erklärt werden. Es sei jedoch darauf hingewiesen, dass die Informationen der Tabellen *UserBlocksUser*, *UserFavoritesTweet* und *UserFollowsUser* nicht bereits mithilfe der **Twitter Streaming API** extrahiert werden können, sondern erst nachträglich pro gewünschtem User über die **Twitter Rest API**, welche jedoch bekanntlich bestimmten Ratelimits unterworfen ist, eingefügt werden. Abbildung 3.6 zeigt das Entity Relationship Diagramm der eingesetzten Datenbank.

Tweet
Die zentrale Tabelle *Tweet* beinhaltet, wie der Name schon sagt, alle Daten im Bezug auf ein getätigtes Posting inklusive der Message in Typed Dependecy Form und der Information, um welche Ausprägung es sich bei dem Tweet handelt. Im konkreten Fall sind die Kategorien *neutraler Beitrag*, *Hassrede*, *Beleidigung* und *Sonstiges* möglich, die jeweils durch eine subjektive Einschätzung festgelegt werden können.

- **tweetid:** Id des Tweets
- **creator_userid:** Id des Users der den Tweet erstellt hat (INTEGER)
- **createdate:** Datum und Uhrzeit der Erstellung
- **content:** Inhalt des Tweets

[16] https://dev.twitter.com/rest/reference/get/followers/ids
[17] https://dev.twitter.com/rest/reference/get/users/lookup

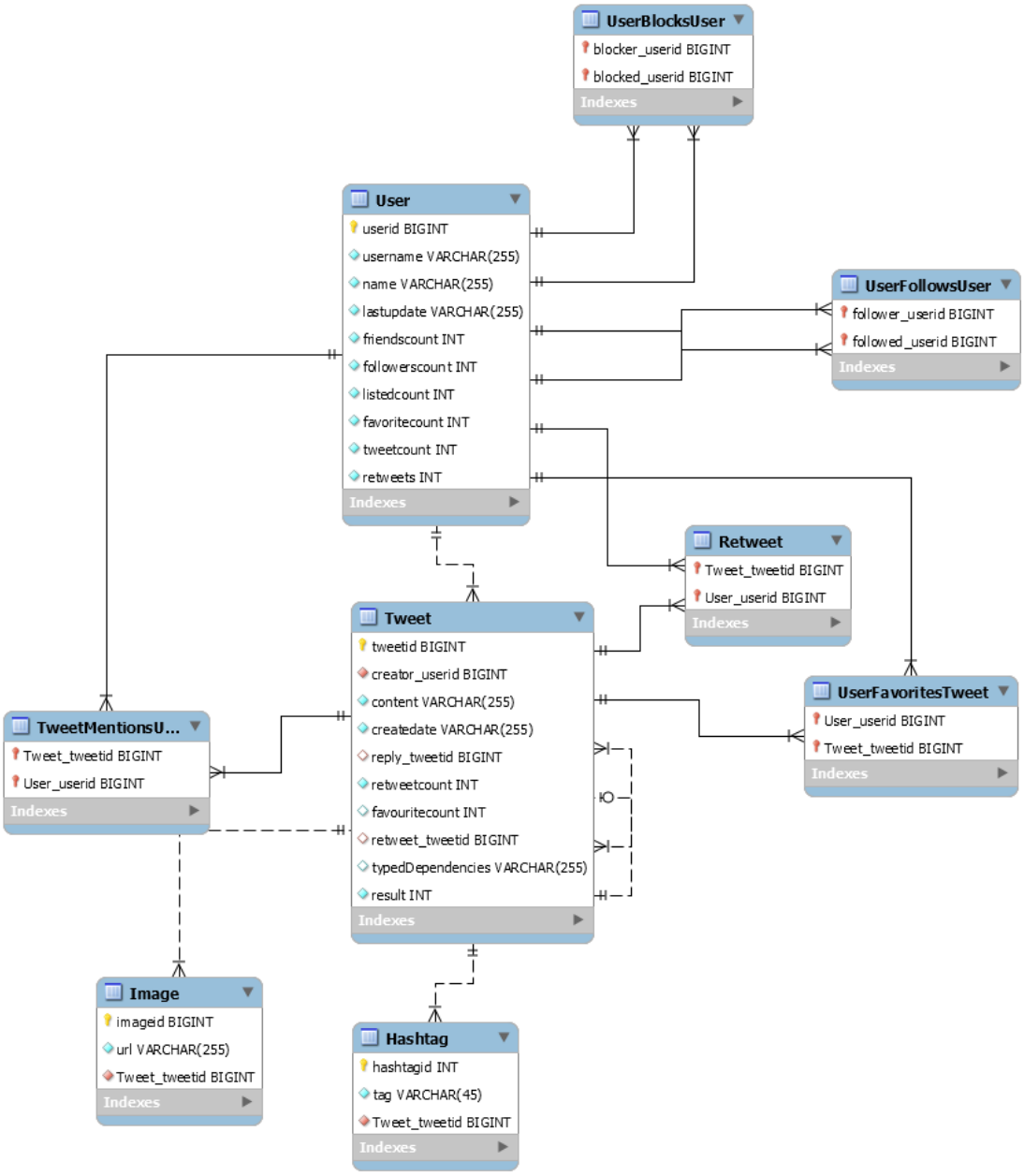


Abbildung 3.6: Entitiy Relationship Diagramm der Datenbank zur Speicherung von Daten aus Twitter.

- **reply_tweetid:** Id des Tweets auf den geantwortet wurde, falls es sich um eine Antwort auf einen Tweet handelt

- **retweetcount:** Anzahl der erfolgten Retweets

- **favouritecount:** Anzahl der Likes im Bezug auf den Tweet
- **retweet_tweetid:** Id des Tweets der retweeted wurde, falls es sich um einen Retweet handelt
- **typedDependencies:** Typed Dependency Darstellung der Tweetmessage
- **result:** Ergebnis der Klassifizierung - 0 für neutrales Posting, 1 für Hassposting, 2 für Beleidigung und 3 für Sonstiges

User
In der Tabelle *User* werden jegliche Informationen über den Twitterbenutzer und dessen Aktivität in Form von verfassten Tweets, Retweets, Freunden etc. gespeichert.

- **userid:** Id des Users
- **username:** Angezeigter Name des Users
- **name:** Name des Users
- **friendscount:** Anzahl der Twitterbenutzer denen der User folgt
- **followerscount:** Anzahl der Twitterbenutzer die dem User folgen
- **listedcount:** Anzahl der öffentlichen Listen (Gruppen) in denen der User gelistet ist
- **favoritecount:** Anzahl der Likes die der User vergeben hat
- **tweetcount:** Anzahl der vom User verfassten Tweets
- **retweets:** Anzahl der vom User getätigten Retweets

Hashtag
Aufgrund der großen Bedeutung von Hashtags in Twitter wurden diese zusätzlich separat in einer eigenen Tabelle erfasst. Da Hashtags auch redundant in Tweets vorkommen können, wurde nicht die ID des Beitrags als Primary Key gewählt, sondern diese generiert. Zu jedem Hashtag wird dazu der Inhalt des Tags und die ID des Tweets gespeichert, um diese eindeutig zuordnen zu können. Es wird bei der Erfassung also nicht unterschieden, ob der selbe Hashtag schon in anderen Tweets bereits vorgekommen ist.

- **hashtagid:** Automatisch generierte Id des Hashtags
- **tag:** Inhalt des Hashtags (ohne vorangestellte Raute)
- **Tweet_tweetid:** Id des dazugehörigen Tweets

Image
Die Daten zu den Bildern, wobei vier pro Tweet möglich sind, beinhaltet die Tabelle *Image*.

- **imageid:** Automatisch generierte Id des Bildes
- **url:** Pfad zu dem lokalen Speicherort beziehungsweise Verzeichnis des Bildes
- **Tweet_tweetid:** Id des dazugehörigen Tweets

UserBlocksUser
Um nachzuvollziehen welche User von welchem User geblockt worden sind, wurde die Tabelle *UserBlocksUser* angelegt.

- **blocker_userid:** Id des Users der einen anderen Benutzer blockt
- **blocked_userid:** Id des Users der geblockt wurde

TweetMentionsUser
Da während des Verfassens von Tweets auch mehrere User über das @-Symbol verlinkt werden können, war auch hier eine eigene Tabelle mit dem Namen *TweetMentionsUser* erforderlich.

- **Tweet_tweetid:** Id des Tweets in dem der User verlinkt wurde
- **User_userid:** Id des Users der verlinkt wurde

UserFavoritesTweet
In der Tabelle *UserFavoritesTweet* werden die gelikten Tweets der User gespeichert.

- **User_userid:** Id des Users der einen Tweet gelikt hat
- **Tweet_tweetid:** Id des Tweets den der User gelikt hat

UserFollowsUser
Die Tabelle *UserFollowsUser* ist notwendig, damit die Follower eines Users ermittelt werden können beziehungsweise um umgekehrt feststellen zu können, welchen Benutzern ein User folgt.

- **follower_userid:** Id des Users der einem User folgt
- **followed_userid:** Id des Users dem gefolgt wird

3.5.3 Annotierter Datenkorpus

Damit die von Twitter gesammelten Daten als Trainingskorpus für den Machine Learning Algorithmus genutzt werden können, müssen diese zuvor kategorisiert werden. Dazu wurde mit Hilfe eines eigens implementierten grafischen Annotierungstool, dessen Oberfläche in Abbildung 3.7 dargestellt wird, eine der bereits im Abschnitt Datenmodell erwähnten Kategorien, nämlich *neutrales Posting, Hassrede, Beleidigung* oder *Sonstiges*, den Tweets manuell zugeordnet. Die Einteilung der Beiträge basiert auf den in diesem Teil formulierten

Abbildung 3.7: Tool zur Annotierung der Tweets.

Definitionen und wurde bei Grenzfällen durch eine weitere Person, die mit dem Thema vertraut ist, begutachtet, um das Ergebnis zu verifizieren und in weiterer Folge durch die Reduktion von False Positive - beziehungsweise False Negative - Kennzeichnungen das resultierende trainierte Modell zu verbessern. Nachfolgend werden die einzelnen Label noch genau erläutert und verständlich gemacht, wann ein Posting mit welchem gekennzeichnet wird:

- **neutrales Posting:** Twitter Beitrag der weder eine Person beleidigt, noch Hass gegen einen Teil der Bevölkerung schürt oder anderweitig moralisch fragwürdigen Inhalt aufweist.

- **Hassrede:** Tweet der zu Hass oder zu Gewalt und Willkürmaßnahmen gegen eine durch bestimmte vorhandene oder abwesende Merkmale definierte Gruppe der Bevölkerung oder gegen einen Einzelnen ausdrücklich aufgrund der Mitgliedschaft zu einer solchen Gruppe auffordert. Solche Merkmale können die politische Gesinnung, Rasse, Hautfarbe, Sprache, die Religion oder Weltanschauung, Staatsangehörigkeit, Abstammung, nationale oder ethnische Herkunft, das Geschlecht, eine körperliche oder geistige Behinderung, das Alter oder die sexuelle Ausrichtung sein. Ein weiterer Grund für die Kategorisierung eines Beitrags als Hassrede ist gegeben, wenn dieser die Menschenwürde anderer dadurch verletzt, indem er Teile der Bevölkerung über die Zugehörigkeit zu einer zuvor angeführten Gruppe beschimpft, böswillig verächtlich macht, in der öffentlichen Meinung herabsetzt oder verleumdet.

- **Beleidigung:** Ein Posting fällt unter die Kategorie Beleidigung, wenn dadurch eine Person als Individuum verunglimpft oder herabgesetzt wird, also nicht über die Mitgliedschaft zu einer Gruppe oder aufgrund der Eigenheiten einer bestimmten Gruppe, der die Person angehört.
- **Sonstiges:** Beiträge, die nicht eindeutig mit einen der bereits beschriebenen Labels annotiert werden können, werden unter dem Begriff "Sonstiges" zusammengefasst. Darunter fallen zum Beispiel Tweets, welche indirekte Hassrede enthalten, in denen Beleidigung und Hassrede vermischt werden oder solche, die verwerflichen Inhalt enthalten, jedoch nicht die erforderlichen Kriterien für Hassrede oder Beleidigung zur Gänze oder gar nicht erfüllen. Unter indirekter Hassrede wird der Gebrauch von Verunglimpfungen im nicht herkömmlichen Sinn verstanden. So zum Beispiel richtet sich bei der Beschimpfung eines heterosexuellen Mannes als "Schwuchtel" die Hassrede indirekt auf die Gruppe der homosexuellen Männer, der herabgewürdigte Mann wird "lediglich" beleidigt.

Letztendlich wurden insgesamt 2837 Tweets manuell annotiert, davon entsprachen 1436 neutralen Beiträgen, 865 Hassrede und 483 einer Beleidigung. 53 verfasste Tweets wurden der Klasse "Sonstiges" zugeteilt. Die Aufteilung zwischen vorurteilslosen Postings und Tweets mit anderem Label ist daher in etwa ausgeglichen.

Da im Kontext dieser Diplomarbeit nur zwischen zwei Klassen unterschieden wird, werden die Tweets, die den Kriterien der *Hassrede* oder der *Beleidigung* entsprechen, für das Trainingsmodell als äquivalent betrachtet und mit dem Überbegriff *Hassposting* zusammengefasst. Somit werden im Endeffekt Instanzen entweder als neutrale oder als hasserfüllte Beiträge, die generell Hass verbreiten, egal ob gegenüber Gruppen oder Individuen, klassifiziert. Die genauere Unterteilung wurde jedoch deshalb getroffen, um in möglichen späteren Untersuchungen spezifischer klassifizieren zu können.

Nachfolgend werden einige interessante Beispiele von Hasspostings aus dem annotierten Korpus aufbereitet, um ein gewisses Gefühl für Hassbeiträge zu bekommen:

"Verpisst euch ihr ganzen Drecksschmarotzer und vorallem die Dreckskanaken hier!! Der nächste der mich anlabert bekommt den Schädel...."

Dieser Satz fällt eindeutig in die Kategorie der Hassrede, da in diesem Zusammenhang mit dem Wort "Dreckschmarotzer" Einwanderer beziehungsweise mit "Dreckskanaken" speziell südländische Einwanderer verunglimpft werden. Zusätzlich wird dieser Bevölkerungsgruppe Gewalt angedroht.

"Schon schlimm genug, daß Bimbos und Muselmanen in der deutschen Nationalmannschaft spielen. Schade um die Schokolade https://t.co/833enrKc2s"

Dieses Hassposting ist ein gutes Beispiel für die oft verwendete und erschreckend erscheinende Sachlichkeit beziehungsweise Emotionslosigkeit mit der Bevölkerungsgruppen herabgewürdigt werden und Hass geschürt wird.

"@minikuimoi Gib dir gleich ne kopfnuSS, Du mongO..."

Diese Beleidigung gegen einen anderen Twitter-User, wobei zusätzlich Gewalt angedroht wird, stellt auch indirekte Hassrede gegenüber der Menschen dar, die an dem Down-Syndrom erkrankt sind. Außerdem könnte die spezielle Schreibweise des Wortes "Kopfnu**SS**" Hinweise auf eine Affinität zu nationalsozialistischen Gedankengut.

"Sie tun was sie immer tun! Kämpfen und Zerstören! Die friedliche Zeit in Europa ist vorbei #krimigranten #norefugees https://t.co/pWBGKoTkPC"

Dieses Beispiel zeigt, dass nicht unbedingt eindeutige Schimpfwörter verwendet werden müssen, um als Hassposting eingestuft zu werden. Dieser Beitrag beispielsweise verallgemeinert, dass alle Flüchtlinge beziehungsweise Muslime gewaltbereit und kriminell sind, was zusätzlich durch den Neologismus "krimigranten" unterstrichen wird.

"@1000AX7 Wir müssen die Juden ausrotten" ""Wieso eig Homosexuelle ins Gefängnis? Homosexuelle UMBRINGEN Das ist die Lösung!!!!!!!!!!!!"

Diese Tweets fordern klar zur Gewalt gegen Menschen mit jüdischem Glauben oder gleichgeschlechtlicher Orientierung auf.

"Der gemeine Musel als solches ist übrigens kein gewöhnliches Haustier, man kann ihn auch kaum erziehen oder... https://t.co/g60Xlh34nh"

Durch diesen Tweet wird der muslimischen Bevölkerung sogar entmenschlicht und einem Tier gleichgesetzt.

3.6 Interrater-Reliabilität

Unter der Interrater-Realibilität wird das Ausmaß der Übereinstimmung der annotierten Daten bei verschiedenen Bewertern verstanden. In diesem Kapitel wird der Trainings-Korpus, dessen Postings von einer weiteren neutralen Person nochmals bewertet wurden, dahingehend untersucht und die Interrater-Reliabilität mithilfe des Cohens-Kappa Werts bestimmt. Dies soll zeigen, ob die Annotationen vom Rater unabhängig sind, die Tweets also objektiv nach den definierten Kriterien klassifiziert werden konnten oder ob Hasspostings in einen zu großen Graubereich fallen, wodurch eine klare binäre Einteilung nicht möglich ist.

Die Auswertung und Gegenüberstellung der Bewertungen des zweiten Raters führte zu dem in Tabelle 3.3 dargestellten Ergebnis. Daraus ergab sich bei insgesamt 2445 Übereinstimmungen ein Cohens-Kappa Wert von 0,7992 beziehungsweise ein korrigierter Cohens-Kappa Wert von 0,8174, welches einer ausgezeichneten Übereinstimmung entspricht. Bezieht man die Tatsache mit ein, dass im Machine Learning-Algorithmus dieser

		Bewerter 2 (neutral)			
		Hassrede	Bleidigung	Neutral	Zeilensumme (Randhäufigkeit Zeile)
Bewerter 1	Hassrede	724	53	88	865
	Beleidigung	53	400	30	483
	Neutral	101	14	1321	1436
	Spaltensumme (Randhäufigkeit Spalte)	878	467	1439	2784

Tabelle 3.3: Kreuztabelle der Beurteilungen der zwei Rater

Arbeit letztendlich nur zwischen Hassposting und neutralen Posting unterschieden wurde, beläuft sich der Cohens-Kappa Wert sogar auf 0,8324 und der korrigierte Cohens Kappa Wert auf 0,83261 bei 2551 Übereinstimmungen.

Die Einteilung in nur zwei Klassen ist also legitim und die Entscheidung wird durch diese Auswertung unterstützt.

3.7 Hasswort Lexikon

Für die Umsetzung und Evaluierung einzelner Features wurde ein sogenanntes Hasswort-Lexikon erstellt. Grundlage dafür war eine Liste gängiger Schimpfwörter[18], welche angepasst und um Hasswörter aus den gesammelten Tweets erweitert wurde. Der genaue Prozess wie die Liste der Uni Wien ergänzt wurde, wird im Abschnitt LIWC-Features 3.9.3 erläutert, da das Kriterium für die Vollständigkeit des LIWC-Wörterbuchs auch für das Hasswort-Lexikon herangezogen wurde. Unter dem Begriff Hasswort fallen dabei Schimpfwörter, Beleidigungen, Verunglimpfungen, herabwürdigende Ausdrücke und Terme die oft bei Hassbeiträgen negativ benutzt werden (z.B.: abschieben, erschießen, vergasen,...). Im Detail wird dieses Lexikon für die Filterung von Message- und Typed Dependency N-Grams, die Erweiterung des LIWC-Thesaurus und für die lexikalischen Features *NumberOfHatefulTermsInApostrophe*, *NumberOfHatefulTerms* und *DensityOfHatefulTerms* verwendet. Eine vollständige Liste der eingesetzten Hasswörter ist im Appendix A.1 enthalten.

3.8 Preprocessing

In diesem Kapitel werden die nötigen Vorbereitungsschritte für die Extraktion von Features, die auf der Message des Tweets beruhen, genauer beschrieben. Als erstes werden die Beiträge in einzelne Wörter aufgeteilt. Weitere Maßnahmen sind einerseits nötig da jedes unterschiedliche Wort eine eigene Dimension im Feature-Vektor repräsentiert, wodurch der Algorithmus bei einer hohen Dimensionalität langsam und ineffizient wird. Mithilfe

[18] http://www.unet.univie.ac.at/ a0201502/php/verflisk/index.php

von Stoppwort Entfernung, Stemming und Case Folding wird versucht die Dimensionen zu reduzieren. Andererseits werden die übrig gebliebenen Features so gewichtet, dass die Länge von Postings und übrig gebliebene Stoppwörter keinen beziehungsweise weniger Einfluss auf die Klassifikation haben. Eine Übersicht über die textbezogenen Featurekategorien in welchen die jeweiligen Preprocessingschritte angewandt werden, ist am Ende jedes Kapitels in den Tabellen 3.4 bis 3.10 aufgeschlüsselt.

3.8.1 Tokenisierung der Tweets

Unter Tokenisierung kann allgemein die Aufteilung einer Phrase in einzelne Wörter verstanden werden. Dies kann in der deutschen Sprache einfach erfolgen, da die einzelnen Wörter durch Leerzeichen getrennt sind. Zusätzlich werden bei allen Sonder- und Satzzeichen der Text segmentiert, wodurch diese irrelevanten Zeichen gefiltert werden. Ausgenommen von dieser Filterung ist der Text für die linguistischen Features, da bei diesen auch solche Zeichen als Indikator für die Klasseneinteilung herangezogen werden. Als Tokenizer für die BoW und N-Gram Features wurde der *NGramTokenizer* von Weka eingesetzt. Dieser bietet die Möglichkeit die Zeichen, bei denen der Satz segmentiert werden soll, und welche N-Grams generiert werden sollen, individuell anzupassen (Regex: ^0-9a-zA-ZäÄöÖüÜß). Für die übrigen Attribute, bei denen der Text in tokenisierter Form vorliegen musste, wurde der Tweet einfach nach jedem Leerzeichen getrennt und Sonder- oder Satzzeichentoken je Feature nachträglich gesondert behandelt. Grund dafür war, dass diese Zeichen wie bereits erwähnt für die Berechnung linguistischer Attribute nötig waren. Konkret wurde für diese Aufgabe der *Apache OpenNLP*-Tokenizer eingebunden.

Feature	***Anwendung***
Bag of Word 3.9.1	√
Message N-Gram 3.9.1	√
Character N-Gram 3.9.1	√
Linguistik Features 3.9.2	√
Lexikalische Features 3.9.2	√
LIWC Feature 3.9.3	√
Typed Dependency 3.9.1	√
Mistake Feature 3.9.5	-
Comment Embedding 3.9.7	√

Tabelle 3.4: Übersicht der Anwendung von Tokenisierung.

3.8.2 Case Folding

Nach der Segmentierung der Beiträge in einzelne Wörter werden alle Buchstaben der erhaltenen Token noch in Kleinbuchstaben transformiert, um ein einheitliches Format zu erhalten, das zusätzlich dabei hilft die Wörter einfacher zu vergleichen. Die unterschiedli-

che Schreibweise von “Hilfe“, “hilfe“ und “HILFE“ würde ansonsten drei verschiedene Wörter im BoW-Vektor ergeben.

Feature	***Anwendung***
Bag of Word	√
Message N-Gram	√
Character N-Gram	√
Linguistik Features	-
Lexikalische Features	√
LIWC Feature	√
Typed Dependency	√
Mistake Feature	-
Comment Embedding	√

Tabelle 3.5: Übersicht der Anwendung von Case Folding.

3.8.3 Stoppwort Filterung

Wortformen aus den geschlossenen Wortklassen, das sind Artikel, Konjunktionen und Präpositionen, können gefiltert werden, da diese keine Aussagekraft für die Klassifikation eines Tweets haben und in den meisten Texten vorkommen. Solche Wörter werden auch Stoppwörter genannt. Zur Bestimmung der Klasse werden Wörter aus den offenen Wortklassen herangezogen, welche sich aus Adjektiv, Nomen und Verb zusammensetzen. Das Filtern der irrelevanten Wörter wird mit einer sogenannten Stoppwortliste in Form einer Textdatei realisiert, welche jener des Snowball Stemmers für die deutsche Sprache entspricht.[19] Die in dieser Liste enthaltenen Stoppwörter werden mit den Wörtern des Tweets verglichen. Stimmen Wörter überein, werden diese als Feature nicht weiter berücksichtigt. In dem umgesetzten Algorithmus wird diese Liste einfach als Argument an die StringToWordVektor-Klasse von Weka übergeben, welche die Filterung vornimmt. Beispiele für Stoppwörter sind aber, allen, ob, so,.... Die Liste der Stoppwörter befindet sich im Appendix A.2.

3.8.4 Wort Stemming

Relevante Wörter eines Dokuments können dann noch mithilfe von Wortstemming auf ihren Wortstamm zurückgeführt werden. Dadurch ergibt sich der Vorteil, dass Wörter, die sich nur aufgrund der Zugehörigkeit zu unterschiedlichen Wortarten (tapfer [Adverb], tapfere [Adjektiv] → tapf) beziehungsweise innerhalb einer Wortart durch Deklination oder Konjugation voneinander unterscheiden, als gleiches Wort betrachtet werden.

Beispiel: Netzwerk → Netzwerk, Netzwerke → Netzwerk
auffälliges → auffall, auffallend → auffall

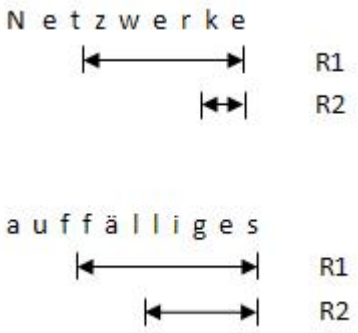
[19] http://snowball.tartarus.org/algorithms/german/stop.txt

Feature	*Anwendung*
Bag of Word	√
Message N-Gram	√
Character N-Gram	√
Linguistik Features	-
Lexikalische Features	-
LIWC Feature	-
Typed Dependency	√
Mistake Feature	-
Comment Embedding	-

Tabelle 3.6: Übersicht der Anwendung von Stoppwort Filterung.

Aus den zwei verschiedenen Wörtern, die eigentlich dieselbe Bedeutung haben, wird ein identes Wort. Dadurch wird der Inhalt der Beiträge besser erkennbar.
Diese Arbeit verwendet einen Wortstemming-Algorithmus für die deutsche Sprache von Snowball der über die StringToWordVector-Klasse von Weka gesetzt werden kann.[20] Der Algorithmus setzt sich wie folgt zusammen und soll nachfolgend anhand der zuvor beschriebenen Beispiele näher beleuchtet werden:

1. Zuerst müssen zwei Regionen definiert werden, nämlich R1 und R2:
   - R1 ist die Region nach dem ersten Konsonanten, der einem Vokal folgt, bis zum Ende des Wortes. Falls kein Vokal in einem Wort existiert oder kein Konsonant nach einem Vokal folgt, ist die Region 0. Die Region muss jedoch mindestens drei Buchstaben enthalten.
   - R2 ist die Region nach dem ersten Konsonanten, der einem Vokal in R1 folgt, bis zum Ende des Wortes. Falls keine zwei Vokale in einem Wort existieren oder kein Konsonant nach dem zweiten Vokal folgt, ist die Region 0.



Definition der Regionen

[20] http://snowball.tartarus.org/algorithms/german/stemmer.html

2. Danach ist eine Liste von Konsonanten definiert, welche nötig ist um gültige s-Endungen beziehungsweise st-Endungen zu finden. Es werden also nur s- und st-Endungen entfernt, vor welchen ein Konsonant aus dieser Liste steht. Folgende Konsonanten sind enthalten: b, d, f, g, h, k, l, m, n, r und t. Bei st-Endungen ist das r nicht enthalten.

3. Entfernen folgender Endungen, wenn sie sich innerhalb von Region 1 befinden:

   - *em*, *ern* und *er*
   - *e* , *en* und *es*
   - *s* (wenn gültige s-Endung)

   Netzwerke → Netzwerk, auffälliges → auffällig

4. Nochmaliges entfernen folgender Endungen(innerhalb R1):

   - *en*, *er* und *est*
   - *st*

5. Entfernen folgender Endungen, wenn sie sich innerhalb von Region 2 befinden:

   - *end* und *ung*
   - *ig*, *ik* und *isch*
   - *lich*, *heit* und *keit*

   auffällig → auffäll, auffallend → auffall

6. Zuletzt werden noch Umlaute durch “normale” Vokale ersetzt z.B: ä → a.

   auffäll → auffall

Daraus ergibt sich:

Netwerk ⟷ Netzwerke, auffälliges ⟷ auffallend

3.8.5 TF-IDF Wortgewichtung

Die Termgewichtung basiert auf zwei Maßeinheiten, nämlich der Term Frequency (TF) und der Inverse Document Frequency (IDF). Der kombinierte Wert TF-IDF wird dann je Wort (N-Gram) und Tweet berechnet und stellt eine gewichteten Koordinate für den Vektor des Tweets dar3.2:

$$\text{TF-IDF Maß} = TF \cdot IDF \tag{3.2}$$

Feature	***Anwendung***
Bag of Word	√
Message N-Gram	√
Character N-Gram	√
Linguistik Features	-
Lexikalische Features	-
LIWC Feature	-
Typed Dependency	√
Mistake Feature	-
Comment Embedding	-

Tabelle 3.7: Übersicht der Anwendung von Wort Stemming.

Der StringToWordVector von Weka stellt für beide Gewichtungen Methoden zur Verfügung. Nachfolgend wird näher auf diese Begriffe und deren Umsetzung in Weka eingegangen:

Term Frequency (TF)

Wie der Name schon sagt, stellt dieser Wert die Häufigkeit des Vorkommens eines Wortes in einem Tweet dar. Hier stellt sich also die Frage, welche Terme (Wörter) häufig vorkommen, denn es wird angenommen, dass oft vorkommende Wörter den Inhalt von Beiträgen am besten wiedergeben. Zuerst wird dieser Wert angeglichen indem der Logarithmus aus diesem berechnet wird. Weiters wird die Häufigkeit eines Wortes normalisiert, indem sie mit dem Wert aus der durchschnittlichen Länge von Beiträgen durch die aktuelle Länge des Beitrags in Relation gesetzt wird. Dadurch werden lange Tweets, die viele Wörter enthalten, nicht höher gewichtet als andere. Da Postings jedoch von Twitter auf eine maximale Anzahl von 140 Zeichen limitiert sind, stellt dies in diesem Fall ein geringes Problem dar. Die Gleichung 3.3 zeigt die sich dadurch ergebende Formel für die Term Frequency Gewichtung.

$$TF = \log(Frequenz\,des\,Wortes\,im\,Tweet) \cdot \frac{Durchschnittliche\ \text{Länge}\ der\ \text{Beiträge}}{Aktuelle\ \text{Länge}\ des\ Beitrags} \tag{3.3}$$

Inverse Document Frequency (IDF)

Die Inverse Dokumentenhäufigkeit wiederum gewichtet die Terme, die aussagekräftiger sind höher. Es wird angenommen, dass Wörter, die nicht oft in den Beiträgen vorkommen, aussagekräftiger sind als Wörter, die häufig in den Beiträgen der Sammlung vorhanden sind. Wörter, die in vielen verschiedenen Tweets mit unterschiedlichen Themen auftreten, sind mit hoher Wahrscheinlichkeit Stoppwörter, die noch nicht entfernt wurden. Kurz

gesagt ist die Inverse Document Frequency der logarithmisierte Wert der Anzahl der Tweets mit dem betreffenden Term in Bezug auf alle Tweets 3.4.

$$IDF(x) = \log(\frac{Gesamtanzahl\ der\ Tweets}{Anzahl\ der\ Tweets\ mit\ Term\ x}) \tag{3.4}$$

Feature	***Anwendung***
Bag of Word	√
Message N-Gram	√
Character N-Gram	√
Linguistik Features	-
Lexikalische Features	-
LIWC Feature	-
Typed Dependency	√
Mistake Feature	-
Comment Embedding	-

Tabelle 3.8: Übersicht der Anwendung von Wortgewichtung.

3.8.6 Minimum Term Frequency

Ein weiteres Mittel zur Reduktion der Feature-Anzahl ist das Setzen einer unteren Schranke bezüglich der Anzahl mit der ein Wort im Korpus vorkommen muss. Gedanke dahinter ist, dass seltene beziehungsweise fehlerhafte Wörter nicht berücksichtigt werden, da ihre Aussagekraft limitiert ist, den Aufwand und Zeit für die Klassifikation aber unverhältnismäßig in die Höhe treiben. Zur Realisation dieses Preprocessing-Schritts in dieser Arbeit kann wiederum eine Option in Wekas StringToWord-Filter gesetzt werden. Als bester Parameter erwies sich zwei für die minimale Anzahl an Vorkommnissen.

Feature	***Anwendung***
Bag of Word	√
Message N-Gram	√
Character N-Gram	√
Linguistik Features	-
Lexikalische Features	-
LIWC Feature	-
Typed Dependency	√
Mistake Feature	-
Comment Embedding	-

Tabelle 3.9: Übersicht der Anwendung von Minimum Term Frequency.

3.8.7 StringToWord Filter

Wie in den Absätzen zuvor bereits mehrfach erwähnt, bietet Weka für String Attribute einen sogenannten StringToWord Filter. Dieser bietet nicht nur zahlreiche Optionen für das Preprocessing von Textphrasen und in weiterer Folge Worttoken, sondern ist auch unbedingt nötig um Zeichenketten(Strings) in Vektoren mit nummerischen Werten zu transformieren, um von Weka verarbeitet werden zu können. Bei diesem Prozess werden je Tweet und Wort, deren Anzahl gezählt und gewichtet.

Feature	***Anwendung***
Bag of Word	√
Message N-Gram	√
Character N-Gram	√
Linguistik Features	-
Lexikalische Features	-
LIWC Feature	-
Typed Dependency	√
Mistake Feature	-
Comment Embedding	-

Tabelle 3.10: Übersicht der Anwendung des StringToWord-Filters.

3.9 Features

Unter einem Feature im Bezug auf Machine Learning versteht man eine messbare Eigenschaft eines bestimmten beobachteten Phänomens.[50] Sie beschreiben in unserem Kontext also die Merkmale der Tweets und derer Verfasser anhand denen der Classifier die Klasse bestimmt.[51] Die Ausprägungen der Features können dabei in Weka folgender Herkunft sein:[21]

- ***nummerisch:*** Das Feature wird durch eine Gleitkomma Zahl dargestellt.
- ***nominal:*** Das Feature wird durch einen Wert aus einem vordefinierten, in der Anzahl beschränkten, Set an nominalen Werten dargestellt.
- ***string:*** Das Feature wird durch einen beliebigen nominalen Wert dargestellt.
- ***date:*** Das Feature repräsentiert ein Datum, wird intern jedoch als Gleitkomma Zahl, welche die Millisekunden seit 1.1.1970 speichert, dargestellt.
- ***relational:*** Das Feature kann andere Features beinhalten. Diese Art von Features werden nötig bei der Verwendung von Multi-Instanz Daten. Bei solchen

[21] http://weka.sourceforge.net/doc.dev/weka/core/Attribute.html

Daten werden einzelne Instanzen nicht individuell einer Klasse zugeordnet sondern in einem annotierten *Bag* zusammengefasst. Als Anwendungsbeispiel kann die Image-Klassifikation genannt werden, bei dem ein Bag aus Feature-Vektoren der Teilbereiche des Bildes besteht. Aus diesen ergibt sich die resultierende Klasse für das gesamte Bild.

3.9.1 N-Gram/Bag-of-Word - Features

Basis der Feature-Sets bilden einige Variationen an N-Gram beziehungsweise Bag-of-Word(BOW) Features, welche in diesem Absatz genau aufgeschlüsselt werden. Grund für deren Umsetzung ist, dass neben den Informationen über einzelne Wörter in Postings auch ein bestimmter Kontext der Wörter gesichert werden soll. Getestet wurden dabei Unigrams, also ein Bag-of-Words Modell, bis hin zu zusätzlichen *Message N-Grams*, welche aus bis zu fünf Wörtern erzeugt wurden (5-Gram). Beispielsweise werden aus dem Satz

Lass doch mal deine Phantasie spielen.

bei einem Trigram folgende Features erzeugt:

Lass doch mal | *doch mal deine* | *mal deine Phantasie* | *deine Phantasie spielen*

Aufgrund der Arbeit von Burnap et al. wurden als Option nur solche Wörter beziehungsweise N-Grams als Features für die Klassifikation herangezogen, die als hasserfüllt interpretiert werden könnten.[41] Umgesetzt wurde diese Filterung mit dem Thesaurus, der in Kapitel 3.7 beschrieben wurde. Im Zuge der Term-Erkennung reicht bereits eine Teilübereinstimmung der Wörter aus, damit diese als ident betrachtet werden.
Typed Dependency N-Grams, welche diesen Kontext noch besser abbilden, indem im Zusammenhang stehende, im Kommentar jedoch weit entfernt liegende, Wörter inklusive ihrer Art der Verbindung zusammengefasst werden, wurden ebenfalls umgesetzt. Als Variation wurden in einigen Testdurchläufen nur solche Typed Dependency Triple berücksichtigt, die zumindest ein Hasswort beinhalten. In Abschnitt 3.9.4 wird die Funktionsweise und Implementierung zum Erhalt von Typed Dependencys näher erläutert.
Zur Steigerung der Robustheit gegenüber gewollten oder ungewollten Rechtschreib- oder Grammatikfehlern in Wörtern kamen *Character N-Grams* zum Einsatz. Die Character-Trigrams ***Pha, han, ant, nta, tas, asi, sie*** für das Wort *"Phantasie"* wären also gegenüber anderen Schreibweisen wie *"Fantasie"* robuster, da nur zwei der generierten Trigrams unterschiedlich wären.
Damit diese Textfeatures in Vektoren mit nummerischen Werten transformiert werden, welche vom Machine Learning Algorithmus verarbeitet werden können, wurde bei dieser Art der Features die bereits erwähnte StringToWordVector-Klasse von Weka genutzt. Diese transformiert nicht nur die vorkommenden Wörter in Wortvektoren, sondern bietet auch die Option, beliebige Tokenizer einzusetzen, welche es wiederum erlauben, beliebige N-Grams zu erzeugen.

- *Message N-Grams (1-5)*
- *Hateful Message N-Grams (1-5)*
- *Typed Dependency N-Grams (1-3)*
- *Hateful Typed Dependency N-Grams (1-3)*
- *Character N-Grams (4-7)*

3.9.2 Linguistik Features

Linguistik Features wurden für die Extraktion von Informationen über die Spracheigenheiten des Verfassers angewandt. Beispiele dafür sind die Anzahl und Art der verwendeten Satzzeichen, die durchschnittliche Wortlänge oder die Länge des Postings.

- *LengthInTokens:* Anzahl der Wörter im Text des Tweets.
- *AvgLengthOfWord:* Durchschnittliche Wortlänge der Terme im Text des Tweets.
- *NumberOfSentences:* Anzahl der Sätze im Text des Tweets
- *AvgSentenceLength:* Durchschnittliche Satzlänge der Sätze im Text des Tweets.
- *NumberOfCharacters:* Anzahl der Buchstaben im Text des Tweets.
- *NumberOfPunctuation:* Anzahl der Satzzeichen .!?" im Text des Tweets.
- *NumberOfSpecialPunctuation:* Anzahl der speziellen Zeichen !"#$%&'()*+,-./:;<=> ?@[]_‘{|}\~^ im Text des Tweets.
- *NumberOfOneLetterTokens:* Anzahl der Wörter bestehend aus einem Buchstaben im Text des Tweets.
- *NumberOfCapitalizedLetters:* Anzahl der Großbuchstaben im Text des Tweets.
- *NumberOfURLs:* Anzahl der URLs im Text des Tweets.
- *NumberOfNonAlphaCharInMiddleOfWord:* Anzahl der Wörter im Text des Tweets, die in ihrer Mitte nicht nur Buchstaben enthalten.

Lexikalische Features

Features, wie die Anzahl an Modalverben oder hasserfüllten Wörtern, bei denen eine Wortliste (Lexikon) für deren Erhalt eingesetzt wurde, werden in die Unterkategorie “Lexikalische Features“ eingeteilt.

- *NumberOfHatefulTerms:* Anzahl von Hasswörtern im Text des Tweets. Beispiele: “Neger“, “Asylantenpack“, “Idioten“,... Insgesamt enthält die Liste für den Abgleich 229 hasserfüllte Wörter. A.1

- *NumberOfHatefulTermsInApostrophe:* Anzahl von Hasswörtern unter Anführungszeichen im Text des Tweets. Diese Art der Schreibweise wird oft in Postings verwendet, die keine hasserfüllte Botschaft mit der Verwendung solcher Wörter beabsichtigen.

- *DensityOfHatefulTerms:* Dichte von Hasswörtern im Text des Tweets. Diese ergibt sich durch NumberOfHatefulTerms/LengthInTokens.

- *NumberOfDiscourseConnectives:* Anzahl der Konnektivpartikel(engl. discourse connective), auch Konnektivadverben genannt, im Text des Tweets. Diese fungieren wie Konjunktoren sind jedoch im Nebensatz integriert und stellen spezifische inhaltliche Relationen, seien es beispielsweise kausale oder temporale, zwischen zwei Sachverhalten her. Beispiele: aber, jedoch, kaum, daraufhin,... [52] Die Liste für die Übereinstimmung an Konnektivpartikel beinhaltet 134 Wörter. A.3

- *NumberOfDiscourseParticels:* Anzahl der Abtönungspartikel (engl. discourse particels) im Text des Tweets. Diese bringen Erwartungen und Einstellungen des Verfassers hinsichtlich spezifischer Sachverhalte zum Ausdruck und sind Kennzeichen konzeptionell mündlicher Sprache, in unserem Fall Umgangssprache. Beispiele: eh, eigentlich, überhaupt,... [52] Die Liste der Abtönungspartikel enthält 21 Terme.A.4

- *NumberOfModalVerbs:* Anzahl der Modalverben im Text des Tweets. Dadurch sollen Überzeugung und Selbstvertrauen des Verfassers gemessen werden. Beispiele: dürfen, können, müssen,... [52] Die Liste an Modalverben umfasst 32 Wörter.A.5

- *NumberOfFirstPersonPronouns:* Anzahl der Personalpronomen der ersten Person im Text des Tweets. Diese werden von Verfassern von Kommentaren eingesetzt, um zum Beispiel allgemeine Aussagen oder eigene Meinungen zu vertreten, wie es auch bei Hasspostings der Fall sein kann. Deshalb werden solche Pronomen auch als Sprecher-Pronomen bezeichnet, da der Autor(Sprecher) im Vordergrund steht. Beispiele: ich, wir, meiner, uns,... [52] Die Liste für Personalpronomen der ersten Person enthält sieben Pronomen. A.6

- *NumberOfSecondPersonPronouns:* Anzahl der Personalpronomen der zweiten Person im Text des Tweets. Diese werden gebraucht, um auf eine andere Person Bezug nehmen zu können. In direkten Beleidigungen anderer Personen sind diese daher nicht wegzudenken. Derjenige dem das Kommentar gilt, steht deshalb im Vordergrund, weswegen auch von Hörer-Pronomen die Rede ist. Beispiele: du, ihr, deiner, euer,... [52] Die Liste an Personalpronomen der zweiten Person umfasst ebenfalls sieben Wörter.A.7

- *NumberOfThirdPersonPronouns:* Anzahl der Personalpronomen der dritten Person im Text des Tweets. Mithilfe dieser Sprachelemente kann auf bereits eingeführte beziehungsweise im Kontext vorhandene Personen, Gegenstände und/oder Sachverhalte, die weder die Rolle des Sprechers noch des Hörers innehaben, Bezug genommen werden. Eingesetzt werden Personalpronomen der 3. Form auch, um

sich von anderen abzugrenzen.[52] Im Kontext von Hassrede gegenüber anderen ethnischen, politischen, religiösen Gesinnungsgruppen oder einfach andersdenkenden Einzelpersonen.[53] Beispiele: er, sie, ihm, es,... [52] Die Liste an Personalpronomen der dritten Person umfasst neun Wörter.A.8

- *NumberOfDemonstrativPronouns:* Anzahl der Demonstrativpronomen im Text des Tweets. Mittels Demonstrativpronomen lässt sich auf Personen, Sachverhalte oder Gegenstände im Umfeld des Verfassers und Empfängers der Botschaft hinweisen.[52] Im Bezug auf Hasspostings kann es sich dabei auch um aktuelle negative Ereignisse handeln. Beispiele: diese, derjenige, jener, solcher,... [52] Die Demonstrativpronomen-Liste enthält 50 Terme. A.9

- *NumberOfIndefinitePronouns:* Anzahl der Indefinitpronomen im Text des Tweets. Indefinitpronomen helfen dabei auf Personen, Sachverhalte oder Gegenstände, welche nicht näher charakterisiert oder identifizierbar sind, hinzuweisen. Beispiele: niemand, jemand, sämtliche, etliche,... [52] Die Liste an Indefinitpronomen beinhaltet 129 Wörter für den Abgleich. A.10

- *NumberOfInterrogativPronouns:* Anzahl der Interrogativpronomen im Text des Tweets. Zur Bildung von Ergänzungs- und Nachfragen werden Interrogativpronomen benötigt. Beispiele: wer, was, wem, welche,... [52] Die Interrogativpronomen-Liste umfasst 25 Wörter.A.11

- *NumberOfHappyEmoticons:* Anzahl der Emoticons im Text des Tweets, die glückliche Gefühle widerspiegeln und sich durch den definierten Regex [:=xX][-co]?[)D]>]|<3|;D bilden lassen. Beispiele: :D, <3, xD, :-D,... [22]

- *NumberOfSadEmoticons:* Anzahl der Emoticons im Text des Tweets, die traurige Gefühle widerspiegeln und sich durch den Regex [:=xX;][-']?[(<[C/] bilden lassen. Beispiele: :(, :[, :<(, :/,... [9]

- *NumberOfCheekyEmoticons:* Anzahl der Emoticons im Text des Tweets, die als frech assoziiert werden können und sich durch den Regex [:=xX][-o]?[Pbp)]D]|;[-o]?[)]] bilden lassen. Beispiele: ;), :P, :-b, =P,... [9]

- *NumberOfAmazedEmoticons:* Anzahl der Emoticons im Text des Tweets, die mit Erstaunen assoziiert werden können und sich durch den Regex [:=][-]?[oO0] bilden lassen. Beispiele: :o, :0, :-O, =o,... [9]

- *NumberOfAngryEmoticons:* Anzahl der Emoticons im Text des Tweets, die mit Zorn assoziiert werden können und sich durch den Regex [:>][-:]?[@(|][|] bilden lassen. Beispiele: :@, >:(, :-||,... [9]

[22] https://de.wikipedia.org/wiki/Emoticon

3.9.3 LIWC - Features

Linguistic Inquiry and Word Count(LIWC) ist ein vom Psychologen James W. Pennebaker in den 90er Jahren entwickeltes Programm zur Textanalyse mit welchem sich verschiedene Emotionen, die Persönlichkeit des Autors, Beweggründe, Gedankenmuster, soziale Aspekte und Wortarten durch das Zählen der Wörter eines Textes extrahieren lassen.[54] Basis dieses Programms ist ein Lexikon von Wörtern, welchen 68 verschiedene Kategorien zugewiesen sein können, die definiert wurden, um den sozialen und psychologischen Zustand des Verfassers von Texten zu erfassen. Im Hinblick auf Hasspostings soll genauer gesagt die Emotionalität der verwendeten Sprache in Tweets kalkuliert werden. Bereits Pennebaker selbst entdeckte bei der Analyse von verfassten Schriften des damaligen Anführers der Terrororganisation "Al-Kaida" Osama Bin Laden und seinem Stellvertreter Aiman Al-Zawahiris, also Hasspredigern schlechthin, dass in diesen signifikant mehr emotionale Wörter, davon überdurchschnittlich viele mit feindseliger Botschaft, als in üblichem Textmaterial vorkommen.[53] Nachteil dieser Methode ist, dass Wörter ohne deren Zusammenhänge betrachtet werden. Dies soll in dieser Arbeit jedoch durch den Einsatz von Message 2-5 Gram (3.9.1), Typed Dependency- (3.9.4) und Comment Embedding - Features (3.9.7) übernommen werden. Für die deutsche Sprache wurde unter der Leitung von Dr. Markus Wolf an der Universität Heidelberg - Forschungsstelle für Psychotherapie(FOST) in Kooperation mit einer Arbeitsgruppe von James Pennebaker eine Übersetzung des LIWC Wörterbuchs von 2001 (LIWC 2001) angefertigt, welche rund 7600 Einträge in 68 verschiedenen Kategorien umfasst.[55] Dieser Thesaurus wurde als Grundlage für die Berechnung der LIWC-Features herangezogen und um Wörter und Kategorien erweitert und angepasst, damit er den Eigenheiten von Hasspostings gerecht wird. Die eingepflegten Wörter entstammen dem selben Hasswort-Lexikon, das ebenfalls für das lexikalische Feature *NumberOfHatefulTerms* herangezogen wird und deshalb diesem Attribut ähnelt. Durch die Einteilung in Kategorien im Rahmen des LIWC wird dieses Feature sozusagen verfeinert und bietet in weiterer Folge einen Mehrwert für die Klassifizierung, da nicht jedes hasserfüllte Wort gleich gewichtet wird. Jede Kategorie entspricht nämlich einem Feature für den Machine Learning Algorithmus. Innerhalb des LIWC-Thesaurus sind auch einige Wörter der Form *Beispielwort** enthalten bei denen keine exakte Übereinstimmung mit Termen in Texten nötig ist, um für die jeweilige LIWC-Kategorie gezählt zu werden (z.B.: auslösch*). Nachfolgend werden als Beispiele die im Kontext der Hassposting-Identifizierung am wichtigsten erscheinenden Wortkategorien aus dem LIWC 2001 und dazugehörige Beispielwörter aufgelistet:

- **Affect** (dt.Affekt)
  - *Positive_emotion (dt. positive Emotion):* aktiv, spannend, Spaß,...
    - Optimism (dt. Optimismus): selbstsicher, Sieg, sorgenfrei,...
  - *Negative_emotion (dt. negative Emotion):* ausnutzen, beklagen, beklagen,...
    - Anxiety (dt. Besorgnis/Angst): beunruhigend, demütigend, erschreckend,...
    - Anger (dt. Zorn): erzürnt, Feind, fies,...

    - Sad (dt. Trauer): gekränkt, Heimweh, hilflos,...

- **Social** (dt.Sozial)
  - *Communication (dt. Kommunikation):* abgefragt, äußern, Unterhaltung,...
  - *Other_reference: (dt. Bezugnahme)* dein, dir, euch,...
  - *Friends (dt. Freunde):* Exfreund, Gast, Gesellschaft,...
  - *Family (dt. Familie):* Großvater, Mama, Nichte,...
  - *Humans (dt. Menschen):* Säugling, Völker, Ehrenmann,...
- **Informal Speech** (dt. Informelle Sprache)
  - *Swear (dt. Schimpfwörter):* Abschaum, bescheuert, Idiot,...
  - *Non-fluency (dt. nicht fließend):* ähm, hm, tss,...
  - *Fillers (dt. Lückenfüller):* naja, sozusagen, weißnich,...
- **Relativity** (dt. Relativität)
  - *Space (dt. Raum):* nach, Osten, umgeben, weg,...
- **Personal Concerns** (dt. persönliche Interessen)
  - *Religion (dt.Religion:* Moslem, verflucht, unsterblich,...
  - *Death (dt. Tod):* Hinrichtung, vergasen, Kopfschuss...
- **Physical** (dt. körperlich)
  - *Body (dt. Körper):* Arsch, durstig, Fresse,...
  - *Sex:* abtreiben, homosexuell, vergewaltigen,...

Die händisch hinzugefügten Kategorien, welche gesammelte hasserfüllte Wörter besser beschreiben sollen, sind die folgenden:

- **Hateful Term Related**
  - *Race (dt. Rasse):* Affe, Genozid, Rassenschande,...
  - *Skin_Colour (dt. Hautfarbe):* Bimbo, Schwarzafrikaner, Neger,...
  - *Nationality (dt. Staatsangehörigkeit):* Drecksvolk, Invasoren, Kümmeltürke,
  - *National_Origin (dt. nationale Herkunft):* Kanacke, Tschusch, Asylantenpack,...
  - *Ancestry (dt. Abstammung):* Dreckshaufen, Gesindel, Schwarzafrikaner,...
  - *Ethnic_Background (dt. ethnischer Hintergrund):* Drecksvolk, Genozid, Schweinevolk,...

- *Gender:* Flittchen, Weib, Trulla,...
- *Disability (dt. Behinderung):* bescheuert, degeneriert, Missgeburt,...

Bei der Erweiterung des LIWC-Wörterbuchs wurden die Kategorien entsprechend unserer Definition der Hassrede in Absatz 3.5.3 beziehungsweise aufgrund der Kriterien und Merkmale der Verhetzung und der zu schützenden Personengruppen laut Strafgesetzbuch § 283(1) ausgewählt. Die Wörter des Hasswortlexikons A.1 wurden dann händisch den bestehenden Kategorien und den neuen Hass-Kategorien zugeordnet. Hasswörter können dabei mehreren Kategorien zugeordnet werden, da eine eindeutige Zuordnung nicht immer möglich beziehungsweise sinnvoll ist. Das Wort "Schwarzafrikaner" findet sich beispielsweise in den Kategorien "Schimpfwort", "Rasse" und "Abstammung" wieder. Im LIWC-Thesaurus bereits enthaltene Wörter wurden wenn möglich ebenfalls durch die zusätzliche Zuordnung zu Hass-Kategorien verfeinert. Am häufigsten wurde diese zusätzliche Kategorisierung bei Termen aus der Wortkategorie "Swear/Schimpfwort" vorgenommen. Insgesamt wurden die ursprünglichen 68 Kategorien um 14 Hass-bezogene erweitert und zusätzlich 204 Hasswörter in das LIWC-Wörterbuch aufgenommen, wodurch das endgültige Dictionary exakt 7802 Einträge umfasst. Wie sich die Änderungen im Bezug auf die Klassifikation von Hasspostings auswirken, wird durch das Diagramm 3.8 veranschaulicht bei dem die Werte aus Tabelle 3.11 visualisiert werden. Als Classifier werden die Support Vector Machine (SVM), Naive Bayes (NB) und Random Decision Forest (RDF) eingesetzt. Wie der Verlauf des Liniendiagramms zeigt flacht die F-Score

LIWC - Dictionary Evaluierung	F-Score			∅△
	SVM	NB	RDF	
Standard (1)	0,643	0,663	0,674	-
(1) + HW Kategorien (2)	0,649	0,666	0,679	0,0046
(2) + HW Liste Basis (3)	0,818	0,81	0,83	0,1546
(3) + HW Spezifisch (4)	0,859	0,856	0,883	0,0466
(4) + HW Related	0,859	0,854	0,884	-0,0003

Tabelle 3.11: F-Score Resultate der einzelnen LIWC-Dictionary Erweiterungen

Kurve immer weiter ab und stagniert, womit das Lexikon als umfassend angesehen wurde. Weitere Hasswörter, die überwiegend exotischer Natur wären, würden das Resultat wenn überhaupt nur noch minimal verbessern. Das Liniendiagramm zeigt auch die Vorgehensweise für die Erstellung des endgültigen Wörterbuchs. Zuerst wurden bestehende Wörter zusätzlich den neuen Hasswortkategorien zugeordnet. Überwiegend handelte es sich dabei wie bereits erwähnt um sehr gebräuchliche Schimpfwörter, deren Kategorien verfeinert wurden. Diese Maßnahme erzielte bereits eine erste kleinere Verbesserung des F-Scores über alle Classifier hinweg. Im nächsten Schritt wurden weitere rund 150 ausgewählte

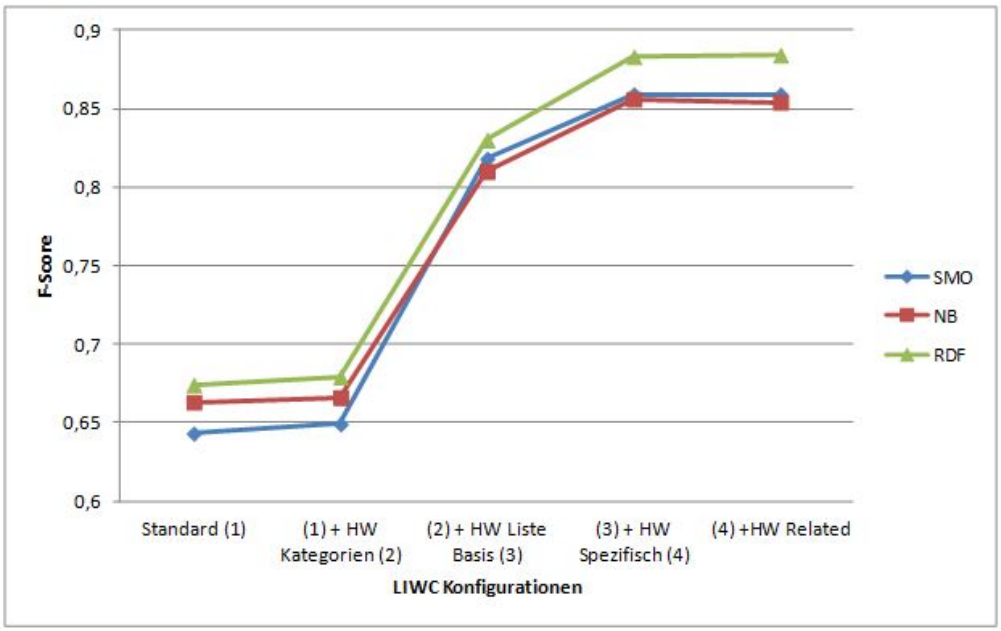


Abbildung 3.8: Liniendiagramm aus den F-Score Resultaten der einzelnen LIWC-Dictionary Erweiterungen

gängige Schimpfwörter aus einer bestehenden Schimpfwortsammlung der Uni Wien hinzugefügt, die im originalen LIWC-Lexikon nicht vorhanden waren.[23] Extrem unübliche Schimpfwörter, Wörter (z.B. driesch in d locka), die allgemein nicht als Schimpfwort angesehen werden (z.B. Chorknabe), wurden dabei nicht berücksichtigt. Diese Maßnahme stellte sich als effektiv heraus und verbesserte das Ergebnis enorm (rund 15,5%). Als nächstes wurde der Thesaurus um noch weitere rund 50 nicht existente Schimpfwörter, die sich auf eine ethnische Herkunft, Abstammung, Religion, Rasse, Behinderung, etc. beziehen erweitert. Damit soll sichergestellt werden, dass Hassrede in Verbindung mit einem anderen Glauben, Herkunft und der vorherrschenden Flüchtlingswelle beziehungsweise die im Allgemeinen dem Tatbestand der Verhetzung entspricht, erkannt wird. Ursprung für diese Wörter waren einerseits die eigenen gesammelten Hass-Tweets, andererseits Facebook Postings, die auf Webseiten[24,25,26] gesammelt wurden, um den Bias nicht zu erhöhen. Dadurch konnte die Performance durchschnittlich um $0,046\dot{}$ gesteigert werden. Zuletzt wurden Terme ins LIWC-Dictionary aufgenommen, die zwar keine Schimpfwörter sind aber immer wieder in Hasspostings verwendet werden. Überwiegend sind dies Verben und Adjektive die bei Aufrufen zur Gewalt eingesetzt werden, aber auch Wörter wie "Genozid" oder "Gaskammer", welche oft in Beiträgen vorkommen, die einen gerichtlich anerkannten Völkermord verleugnen. Diese Erweiterung des LIWC-Wörterbuchs stellt

[23] http://www.unet.univie.ac.at/ a0201502/php/verflisk/index.php
[24] https://perlen-aus-freital.tumblr.com/
[25] https://www.eaudestrache.at/
[26] http://hasshilft.de/

ebenfalls das bereits in Kapitel 3.7 erwähnte eigenständige Hasswort-Lexikon für die Umsetzung weiterer Features dar.

3.9.4 Typed Dependency Features

Wie bereits in Abschnitt 3.9.1 erläutert wurden Typed Dependency Relationen als Features an den Machine Learning Algorithmus übergeben, um auch syntaktische Zusammenhänge zwischen Wörtern, die im Satz weit auseinanderliegen, zu erfassen. Ein weiterer Vorteil liegt darin, dass der gebildete Kontext im Gegensatz zu den Part-of-Speech (PoS) Tags von Satzteilen eindeutig ist. Beispielsweise ist die Sequenz an PoS-Tags für die Phrasen “Schickt sie nach Hause!“ und “Lasst sie in Frieden!“ ident, wodurch es zu Verwechslungen bei der Klassifikation kommen kann. Mittels Typed Dependency Parsing ist es möglich, ebenfalls solche Muster in hasserfüllten Postings zu erkennen, da über die Bezeichnung der Beziehung teils Rückschlüsse auf diese getroffen werden können. Eine Dependency-Struktur stellt dabei die Beziehungen zwischen zwei einzelnen Wörtern dar, wobei eines der beiden als Kopf(engl. head) und das andere als Abhängiger(engl. dependant) bezeichnet wird. Indem in Zusammenhang stehende Wörter über gerichtete Kanten verknüpft werden, entsteht ein gerichteter azyklischer Graph, ein sogenannter Dependency Tree. Bei Typed Dependency-Strukturen werden zusätzlich auch der Name der grammatikalischen Beziehung, die zwei Wörter miteinander verbindet, ermittelt. Zusätzlich können auch Informationen über die Prädikat-Argument Beziehung in Erfahrung gebracht werden.
In dieser Arbeit wird zur Erstellung der Typed Dependencies das *Mate Tools - Toolkit*[27] verwendet, da es auch ein Modell für die deutsche Sprache mit sich bringt. Der dabei verwendete Second Order Maximum Spanning Tree (MST) Dependency Parsing Algorithmus wird in [56] genauer beschrieben. Die Namen der Kanten werden entsprechend dem Tiger Annotationsschema gelabelt.[28] Für die Extraktion der semantischen Beziehungen bestehend aus Prädikat und Argument setzt Mate Tools einen Machine Learning basierten Ansatz ein.[57] In vier Schritten wird dabei durch mehrere Classifier ermittelt, ob es sich bei einem Wort um ein Prädikat oder ein Argument handelt, welchem Prädikat oder Argument es im Detail entspricht und zuletzt die entsprechende Rolle der Beziehung identifiziert. Diese semantischen Beziehungen werden in dieser Arbeit jedoch nicht als Feature eingesetzt und lassen eine Option für die Zukunft offen.

In Abbildung 3.9 wird der gesamte Prozess zur Bildung der Typed Dependencies dargestellt. Darin ist ersichtlich, dass für das Preprocessing ein Tokenizer, Part-of-Speech Tagger und ein Lemmatizer angewendet werden. Durch den Lemmatizer wird mithilfe der Information des Part-of-Speech Taggers die im Kontext korrekte gebeugte Form des Wortes feststellt, bei der es sich um die Wörterbuchform, oder auch bekannt als Lemma, handelt und das Wort auf diese reduziert.

[27] http://www.ims.uni-stuttgart.de/forschung/ressourcen/werkzeuge/matetools.html
[28] http://www.ims.uni-stuttgart.de/forschung/ressourcen/korpora/TIGERCorpus/annotation/tiger_scheme-syntax.pdf
[29] http://de.sempar.ims.uni-stuttgart.de/parse

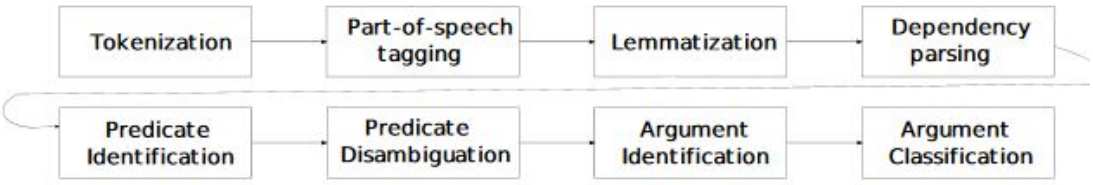


Abbildung 3.9: Mate Tools Dependency Parser Architektur[58]

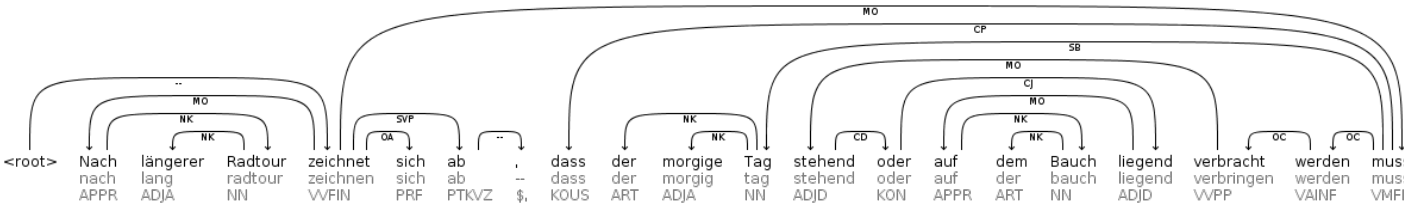


Abbildung 3.10: Typed Dependency Graph des neutralen Beispielsatzes.[29]

Um das Konstrukt eines Typed Dependency Trees weiter zu verdeutlichen zeigt Abbildung 3.10 anhand eines Beispielsatz die dazugehörigen syntaktischer Abhängigkeiten inklusive grammatikalischer Beschriftung. Der angeführte Satz

Nach längerer Radtour zeichnet sich ab, dass der morgige Tag stehend oder auf dem Bauch liegend verbracht werden muss.

ist in leicht modifizierter Form aus dem Twitter-Datensatz entnommen und wird nach dem Dependency Parsing in folgender Form als Feature an den Algorithmus übergeben:

MO(zeichnen,nach) NK(radtour,lang) NK(nach,radtour) root(ROOT,zeichnen)
OA(zeichnen,sich) SVP(zeichnen,ab) CP(muss,dass) NK(tag,der) NK(tag,morgig)
SB(muss,tag) MO(verbringen,stehend) CD(stehend,oder) MO(liegend,auf)
NK(bauch,der) NK(auf,bauch) CJ(oder,liegend) OC(werden,verbringen)
OC(muss,werden) MO(zeichnen,muss)

Die Darstellung des Typed Dependency Tripels setzt sich dabei aus dem Namen der Beziehung, dem Kopf und zuletzt aus dem abhängigen Wort zusammen. Auch die Wortreihenfolge bleibt erhalten, da die Typed Dependencies nach dem abhängigen Wort und deren Position im Satz gereiht werden, wodurch die Aussage des Satzes nachvollzogen werden kann. Nachfolgend werden die hier vorliegenden grammatikalischen Beziehungen näher beschrieben. Der allgemeine Beispielsatz wurde so gewählt, dass die wichtigsten Phrasen und Beziehungen dieser enthalten sind:

MO(zeichnen,nach): Die Präpositionalphrase (PP) “nach längerer Radtour“ fungiert

als Modifikator (MO) für das Verb “zeichnen“, womit dieses mit dem Kopf der Präpositionalphrase nämlich “nach“ über das Label “MO“ miteinander verbunden wurde.

NK(radtour,lang): Die Phrase “längerer Radtour“ ist ein Teilelement der zuvor beschriebenen Präpositionalphrase. “Längerer“ fungiert dabei als attributives Adjektiv für den Kopf “Radtour“ und ist von diesem abhängig. Dies wird mit der Beschriftung Noun Kernel Element (NK), in diesem Fall adjektivisches Kernelement, verdeutlicht.

NK(nach,radtour): Durch die Präposition “nach“ wird die Präpositionalphrase gebildet. “Radtour“ ist dabei ein von “nach“ abhängiges Nomen und steht ebenfalls als ein Noun Kernel Element in Beziehung.

root(ROOT,zeichnen): Das Wort “zeichnen“ ist unabhängig beziehungsweise “kopflos“, also von keinem anderen Wort abhängig und bildet daher die Wurzel des Baumes. Dadurch, dass es sich beim Kopf des Satzes um ein Verb handelt, kann man bei diesem Beispielsatz auch von einer Verbphrase (VP) sprechen.

OA(zeichnen,sich): Das Wort “sich“ ist Teil der Verbphrase. Es handelt sich dabei um ein reflexives Personalpronomen im vierten Fall das von dem Term “zeichnen“, dem Kopf, abhängig ist. Dies ist durch die Bezeichnung der Beziehung als Akkusativobjekt (OA) verdeutlicht.

SVP(zeichnen,ab): Auch das Wort “ab“ ist Teil der Verbphrase und ist als abgetrennter Verbzusatz (engl. Separable Verb Prefix) von “zeichnen“, dem Kopf der Beziehung abhängig.

CP(muss,dass): Satzeinleitende Konjunktionen wie in diesem Fall “dass“ werden als Komplementierer(CP) annotiert. Das finite Modalverb “muss“ bildet gleichzeitig den Kopf des Nebensatzes und der Beziehung. “dass“ ist die davon abhängige Konjunktion, die auch die Verbletztstellung von “muss“ auslöst.

NK(tag,der): “der Tag“ bildet einen Teil der Nominal Phrase (NP) “der morgige Tag“. Der Artikel “der“ ist dabei vom Wort “Tag“ abhängig, das den Kopf der Beziehung bildet und ist ein Element der Kern Nominalphrase (NK).

NK(tag,morgig): So wie zuvor ist auch “morgige Tag“ Bestandteil der Nominalphrase. “Morgiger“ fungiert dabei als attributives Adjektiv für den Kopf “Tag“ und ist von diesem abhängig. Somit handelt es sich um ein adjektivisches Element der Kern Nominalphrase (NK).

SB(muss,tag): Die Nominalphrase “der morgige Tag“ ist das Subjekt des Nebensatzes und ist vom Verb “muss“, dem Kopf des Dependency-Tripels, abhängig.

MO(verbringen,stehend): Das Wort “stehend“ bildet den Kopf der konjugierten Adjektive “stehend oder auf dem Bauch liegend“. Diese Phrase wird auch Adjektivphrasen - Koordination (CAP) genannt. Diese Adjektive bestimmen den Kopf “verbringen“ dieser Dependency näher, fungieren also als Modifikator (MO) und sind von diesem abhängig.

CD(stehend,oder): Die koordinierende Konjunktion (CD) “oder“ ist vom Kopf der Beziehung, der durch das Wort “stehend“ gebildet wird, abhängig. Die adverbialen Adjektive “stehend“ und “liegend“ bilden die Konjunkte (CJ). Konjunkte und Konjunktion bilden in diesem Fall wie zuvor erwähnt eine Adjektivphrasen- Koordination (CAP).

MO(liegend,auf): Die Präpositionalphrase “auf dem Bauch“ bestimmt das Adjektiv “liegend“ näher, sind daher über eine Modifikator Beziehung miteinander verbunden, wobei “liegend“ den Kopf bildet und “auf“ die von diesem abhängige Präposition ist.

NK(bauch,der): Die Phrase “der Bauch“ ist ein Teilelement der zuvor beschriebenen Präpositionalphrase. “der“ fungiert dabei als Artikel für den Kopf “Bauch“ und ist von diesem abhängig. Dies wird mit der Beschriftung Noun Kernel Element (NK) verdeutlicht.

NK(auf,bauch): Durch die Präposition “auf“ wird die Präpositionalphrase gebildet. “Bauch“ ist dabei ein von “auf“ abhängiges Nomen und steht ebenfalls als ein Noun Kernel Element in Beziehung.

CJ(oder,liegend): Das Wort “liegend“ ist ein Konjunkt (CJ) der Konjunktion “stehend oder liegend“ und ist deshalb vom Bindewort “oder“ abhängig.

OC(werden,verbringen): Der Infinitiv “werden“ stellt den Kopf der Dependency dar und dient dabei als Hilfsverb(Auxiliarverb), um gemeinsam mit dem Vollverb “verbringen“, dem Partizip, das Partizip Perfekt “verbracht werden“ zu konstruieren. “werden“ wird als Komplement bezeichnet, die resultierende Beziehung und “verbringen“ wird Kausalobjekt (OC) genannt.

OC(muss,werden): Das finite Modalverb “muss“ ist das Komplement des Kausalobjekts “werden“.

MO(zeichnen,muss): Der Nebensatz dessen Kopf “muss“ darstellt, wird als Modifikator in den Hauptsatz, dessen Kopf “zeichnen“ bildet, eingebettet. Durch die Beziehung wird klar, dass der Nebensatz dabei vom Hauptsatz abhängig ist.

Um nun die im Beginn des Abschnitts aufgezählten Vorteile von Typed Dependencies im Bezug auf Hasspostings zu erläutern soll folgendes typisches Hassposting dazu dienen:

Dieses Pack sollte man ohne Umwege heimschicken!

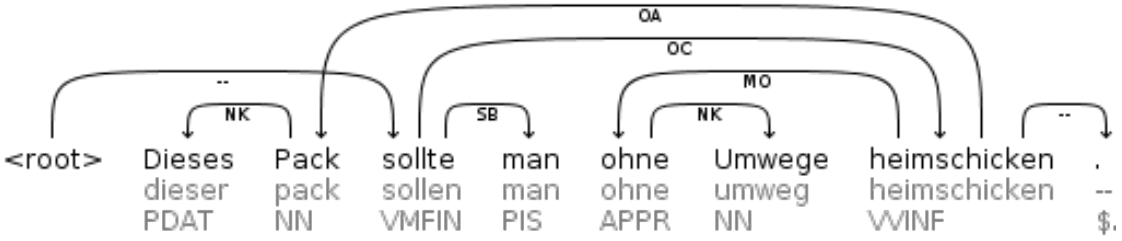


Abbildung 3.11: Typed Dependency Graph des hasserfüllten Beispielsatzes.[30]

Der Output des Dependency Parsers lautet wie folgt und wird in Abbildung 3.11 grafisch dargestellt:

NK(pack,dieser) SB(sollen,pack) root(ROOT,sollen) SB(sollen,man)
MO(heimschicken,ohne) NK(ohne,umweg) OC(sollen,heimschicken)

Dieser Satz zeigt gut, dass auch Wörter die weit auseinander liegen in Relation stehen. Hier würde nicht einmal ein 5-Gram Feature dabei helfen den Zusammenhang zwischen "Pack" und "heimschicken" herzustellen. Die Beziehung OA(heimschicken,pack), bei dem "heimschicken" den Kopf bildet und "pack" als Akkusativobjekt (OA) der Nominalphrase davon abhängig ist, lässt zusätzlich auf ein Muster der Ausgrenzung schließen, welches in der Form öfter auftreten kann.
Da nicht alle Typed Dependency Relationen gleich viel Aussagekraft haben, werden in manchen Experimenten nur solche berücksichtigt, welche am meisten Information für die Klassifizierung von Hasspostings liefern. Chen et al. erarbeiten sich diesbezüglich eine Auflistung an geeigneten Relationen, die auch für diese Arbeit in den Experimenten übernommen wurde.[35] Tabelle 3.12 zeigt die korrespondierenden Typed Dependencies aus dem Tiger Annotationsschema nach denen gefiltert wurde. Da die Wortreihenfolge beibehalten wird, kommen auch durch Typed Dependencies gebildete N-Grams zum Einsatz, wodurch man sich einen weiteren Einblick in den typischen Kontext von Hasspostings verspricht. Als weitere Variation werden nur solche Typed Dependencies herangezogen die in ihrem Triple ein hasserfülltes Wort enthalten. Dieses Feature ist in Anlehnung an die Idee von Burnap et al. entstanden, nur solche N-Grams zu berücksichtigen, welche Hasswörter beinhalten.[41]

3.9.5 Mistake-Feature

Das Mistake-Feature nimmt die Anzahl an Grammatik- und Rechtschreibfehlern in den Lernprozess des Algorithmus auf. Grund dafür ist, dass Hasspostings oft im Affekt und mit viel Emotion verfasst werden und deshalb angenommen wird, dass bei den Autoren die Fehlerfreiheit ihrer Beiträge in den Hintergrund rückt. Typografische und Fehler bezüglich der Groß- und Kleinschreibung fließen aufgrund der generellen Vernachlässigung

[30] http://de.sempar.ims.uni-stuttgart.de/parse

Kantenname	***Abhängigkeit***
AG	Genitivattribut
APP	Apposition
CC	Vergleichskomplement
CD	Konjunkt
CJ	Konjunktion
CM	Vergleichskonjunktion
DA	Dativ
MO	Modifikator
NG	Negation
NK	Element der Kern-Nominalphrase(NP)
OA	Akkusativobjekt
OA2	Zweites Akkusativobjekt
OC	Klausales Objekt
PD	Prädikativ
PNC	Eigenname
SB	Subjekt
SBP	Logisches Subjekt im Passivsatz

Tabelle 3.12: Berücksichtigte Dependency Beziehungen.

dieser in sozialen Netzen nicht in den Feature-Wert ein. Für die Erfassung der Fehler wird der Spell-Checker *Language Tool* für die deutsche Sprache verwendet, welcher eine API für Java zur Verfügung stellt.[31]

3.9.6 Twitter-Netzwerk Features

Bei den Twitter-Netzwerk-Features handelt sich um extrahierte Eigenheiten, die speziell aus dem sozialen Netzwerk Twitter gewonnen werden können. Beispiele dafür sind die Anzahl an Retweets, Hashtags und Favourites pro Tweet oder die Menge an Freunden und Follower pro User. Diese Art der Features sollen Aufschluss über die Verbreitung des Tweets und die Vernetzung des Users geben und in weiterer Folge auch dazu beitragen, zwischen Hass- und neutralen Posting zu unterscheiden.

Tweetbezogene Features

In die Unterkategorie “Tweetbezogene Features“ werden solche Features aufgenommen, die mittels Informationen über den Tweet, Aussagen zur Klasse des Postings treffen können. Diese Merkmale sind speziell für Twitter und können nur zum Teil in anderen sozialen Netzwerken gefunden werden.

[31] https://languagetool.org/

- *FavouriteCount:* Anzahl an Favourites/“Gefällt mir“-Angaben des Tweets, das heißt wie oft der Beitrag mit einem Herz von anderen User markiert wurde.
- *RetweetCount:* Anzahl an Retweets des Tweets, das heißt wie oft der Beitrag von anderen User geteilt wurde.
- *NumberOfMentionedUser:* Anzahl an User, die im Tweet erwähnt beziehungsweise verlinkt wurden.
- *IsReply:* Boolean-Wert der angibt, ob es sich bei dem Beitrag um eine Antwort auf einen Tweet handelt.
- *IsRetweet:* Boolean-Wert der angibt, ob es sich bei dem Beitrag um einen Retweet handelt.
- *NumberOfHashtags:* Anzahl an benutzten Hashtags im Tweet. Hashtags sind Schlagwörter, welche auf bestimmte aktuelle Themen Bezug nehmen, den Tweet kategorisieren und deshalb besondere Aussagekraft aufweisen.

Userbezogene Features

In die Unterkategorie “Userbezogene Features“ werden solche Features aufgenommen, die mittels Informationen über den User, welcher einen Knoten im Netzwerk darstellt, Aussagen zur Klasse des Postings treffen können. Es folgt eine Liste der verwendeten Features die sich aus den Knotenattributen ergeben, wobei die ersten vier Merkmale über die Twitter Rest API abgefragt werden können und die restlichen aus den vorhanden Daten über den User berechnet werden. Besonders hervorzuheben ist das Feature *NumberOfHateposts*, welches Informationen über die Klasse von älteren Beiträgen des Nutzers aggregiert, um so eine gewisse “Vorgeschichte“ des Users zu modellieren. Grund dafür ist die Annahme, dass hasserfüllte Beiträge bei vielen User keine Ausreißer sind, sondern diese generell einen Hang zum Twittern von Hasspostings haben.

- *NumberOfFriends:* Anzahl der Twitter-Nutzer denen der User folgt.
- *NumberOfFollower:* Anzahl der Twitter-Nutzer die dem User folgen.
- *NumberOfTweets:* Anzahl an verfassten Tweets des Users.
- *ListedCount:* Anzahl an beigetretenen Gruppen des Users.
- *LengthOfUsername:* Länge des Usernamens.
- *LengthOfName:* Länge des Namens des Users.
- *NumberOfWordsInName:* Anzahl der Wörter aus denen der Name besteht.
- *NumberOfHateposts:* Anzahl der bereits verfassten Hasspostings des Users.

Follower/Graph Features

Graph-Features, die mithilfe eines Graph-Analysetools erzeugt werden können (clustering coefficient, closeness centrality, betweenness centrality,...), werden im Rahmen dieser Diplomarbeit aus mehreren Gründen nicht im Detail untersucht. Erstens wäre die Extraktion der Features ohnehin bei großen Netzwerken sehr zeitintensiv und im Fall einer Live-Klassifikation müsste der Graph ständig zusätzlich modifiziert und angepasst werden, um diese Art von Features für neue User zu berechnen. Zweitens zeigt die vorangegangene Arbeit von Ting et al., die solche Features generieren und analysieren, dass deren Einsatz keine Performance-Steigerung bringt.[59] Damit aus der Twitter-Netzwerkstruktur trotzdem Nutzen gezogen wird, werden Graph-Features speziell für die Hasspostingerkennung entworfen, die im weiteren Verlauf als Follower-Features bezeichnet und als erfolgversprechender angesehen werden.

Unter dem Begriff "Follower Features" werden jene Merkmale zusammengefasst, welche als Wert die Anzahl an einschlägigen Seiten (*FollowerSiteFeatures*) beziehungsweise der Hassposter (*FollowerUserFeature*), denen der Verfasser des jeweiligen Tweets folgt, repräsentieren. Durch diese Analyse der Netzwerkstruktur soll die Information gewonnen

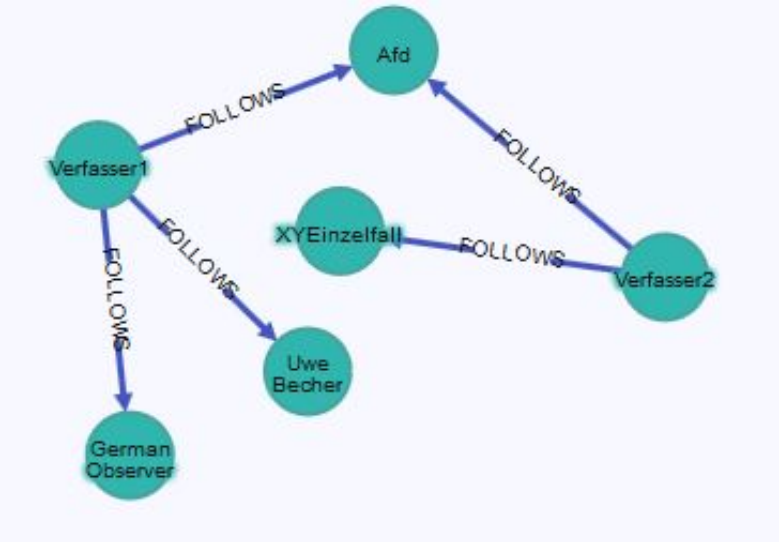


Abbildung 3.12: Symbolische Darstellung der Vernetzung von User mit einschlägigen Seiten.

werden, ob der Beitragsverfasser generell in hassschürende Postings von bekannten rechtsradikalen, ausländerfeindlichen, etc. Seiten und User interessiert ist. Auch Seiten von in der Politskala rechtsstehenden Parteien wurden in dieses Set aufgenommen. Für die Evaluierung des Features wurden insgesamt 19 Seiten dieser Art händisch ausgewählt und deren Follower in die Datenbank gespeichert. Um nun den nummerischen Wert des Features zu erhalten, wurde gezählt wie oft sich der Verfasser des jeweiligen Tweets unter den Followern der Seiten befindet. Abbildung 3.12 zeigt eine symbolische Darstellung

dieses Netzwerk-Features, bei dem Verfasser1 drei und Verfasser2 zwei entsprechenden Seiten folgt. Des weiteren wurden notorische Hassposter, welche in dieser Arbeit als

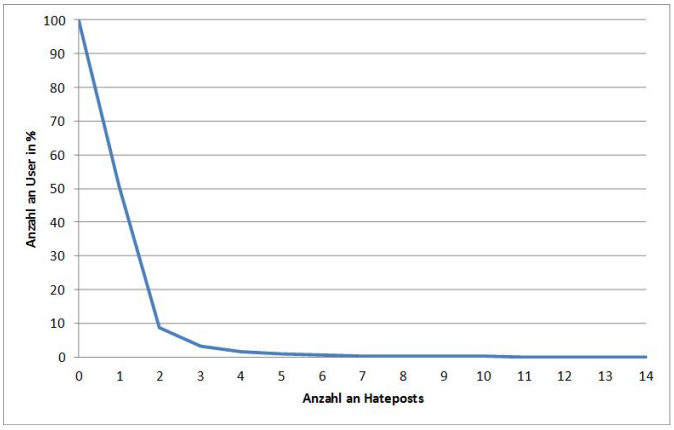


Abbildung 3.13: Verteilung der User im Bezug auf die Anzahl an verfassten Hasspostings.

solche definiert wurden, wenn sie mindestens sechs als Hassposting klassifizierte Tweets verfassten, aus dem Datensatz extrahiert. Wiederum wurde eruiert wie vielen Hasspostern der jeweilige Verfasser des zu klassifizierenden Beitrags folgt. Bei einem möglichen Einsatz dieses Algorithmus für eine Live-Klassifikation von Tweets, würden identifizierte Hasspostings und deren User natürlich gespeichert werden, wodurch sich die Menge der Hassposter und die Werte des Features kontinuierlich verändern würden.

Abbildung 3.13 stellt die Verteilung an Hassposter in unserem Datenkorpus dar. User mit mehr als sechs hasserfüllten Beiträgen machen dabei weniger als 1% (9 Hateposter) aus. Insgesamt kann gesagt werden, dass rund 50% der gesammelten Nutzer schon mindestens ein Hassposting getweetet haben, 50% lediglich neutrale Postings.

3.9.7 Distributional Semantic Features - Comment Embedding

Eine verteilte Darstellung (distributed representation) mittels Vektoren von Wörtern, ganzen Sätzen oder Absätzen finden in vielen Anwendungen der natürlichen Sprachverarbeitung (natural language processing - NLP) Einsatz, da dadurch im Gegensatz zu einem Bag-of-Word Vektor die semantische Bedeutung der Wörter erhalten bleibt und diese nicht aus deren Kontext gerissen werden.[60][61] Abbildung 3.14 zeigt zum Beispiel die engsten Nachbarn des Wortes “fck“ im resultierenden Vektorraum. Im Bezug auf die Erkennung von hasserfüllten Postings versuchen die Arbeiten von [39] und [62] dadurch eine Verbesserung zu erzielen.

In den nachfolgenden Abschnitten wird der Prozess der Erstellung von Paragraph-Vektoren mit fixer Länge aus variabel langen Textelementen genauer erklärt, wie diese formal erzeugt werden und deren Umsetzung für diese Diplomarbeit beschrieben.

Abbildung 3.14: Wortwolke von semantisch ähnlichen Wörtern des Schimpfwortes “fck“.[62]

Allgemeine Beschreibung

Ausgehend von Vektordarstellungen einzelner Wörter durch die Verwendung von neuralen Netzen[60], auch neurale Wort Embeddings genannt, wurde diese Methode auf die verteilte Darstellung ganzer Sätze oder gar Dokumente, sogenannte Paragraph-Vektoren oder Comment Embeddings, erweitert.[61] Dabei werden die Wörter und Paragraphen gemeinsam als dichte, niedrig-dimensionale Vektoren modelliert, welche mit der Hilfe eines Distributed Memory Modells[61] so erlernt beziehungsweise trainiert werden, um Wörter eines Kommentars zu prognostizieren. Im Detail wird dies wie folgt erreicht: So wie beim Word2Vec - Algorithmus[60] die einzelnen Wortvektoren verknüpft oder gemittelt werden und so trainiert werden, dass sie die nächsten Wörter in einem gegebenen Kontext (Wortfenster) bestimmen können, so wird für den Erhalt der Comment Embeddings zusätzlich auch der Paragraph-Vektor mit den Wortvektoren verknüpft und liefert dadurch semantische Zusammenhänge im Bezug auf gesamte Kommentare, die die Hassposting Identifikation weiter verbessern sollen. Genauer gesagt basiert die Wahrscheinlichkeitsverteilung für die Wortbestimmung nicht nur auf den umschließenden Wörtern sondern ist zugleich abhängig vom spezifischen Kommentar. Resultat sind niedrig-dimensionale Comment Embeddings (Pragraph-Vektoren), wobei sich semantisch ähnliche Beiträge im selben Teil des Vektorraums befinden. Abbildung 3.15 zeigt die grobe Vorgehensweise

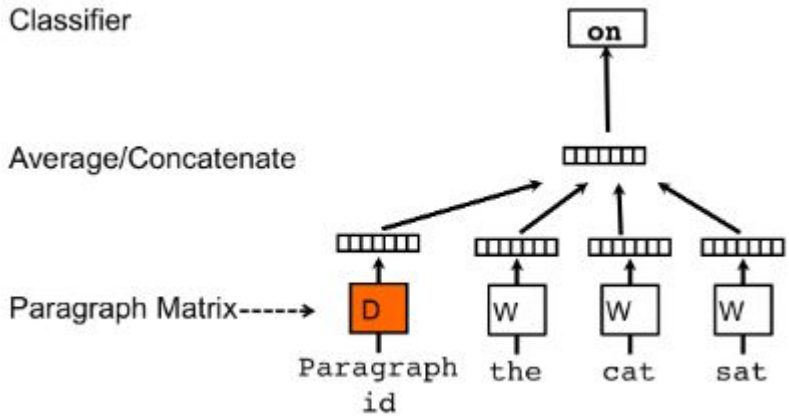


Abbildung 3.15: Framework zum Erlernen von Comment Embeddings (Distributed Memory Modell)[61]

beim Erlernen von Comment Embeddings. Der Paragraph, in unserem Fall der Tweet, wird als eindeutiger Vektor, repräsentiert durch eine Spalte der Matrix D (D: Anzahl der Tweets x Dimension des Comment Embeddings(=Paragraph-Vektor)), abgebildet. Als Dimension eines Vektorraums wird die Anzahl an Vektoren seiner Basen verstanden. Jedes Wort wird ebenfalls als eindeutiger Vektor abgebildet und stellt eine Spalte der Matrix W (W: Anzahl der Wörter x Dimension des Wort-Vektors) dar. Im Gegensatz zum Paragraphen-Vektor, der je Tweet eindeutig ist, werden die Wort-Vektoren mehrfach genutzt. So ist beispielsweise der Vektor des Wortes *Haus* für alle Beiträge gleich. In diesem Distributed Memory Modell werden der Paragraph-Vektor, der die fehlende Information des aktuellen Kontextes repräsentiert beziehungsweise als Memory für den Inhalt des Paragraphen dient und die drei Wort-Vektoren von *the*, *cat* und *sat* des Kontexts verknüpft oder gemittelt, um das vierte Wort *on* zu prognostizieren. Der Paragraph- und die Wortvektoren werden so lange angepasst, bis sie das folgende(vierte) Wort, auch genannt Teacher der regulierende Fehlersignale zurücksendet, akkurat vorhersagen können. Comment Embeddings die nahe nebeneinander liegen, also annähernd gleiche Werte im Vektor aufweisen, werden durch ähnliche Wörter beurteilt.

Vorteil dieser Methode ist, dass die Schwächen, die bei einer alleinigen Verwendung von Wort Embeddings auftreten, bewältigt werden können, da neben den semantischen Zusammenhänge der Kommentare, auch die Wortreihung durch die Hinzuziehung des Paragraphen-Vektors erhalten bleibt. Des weiteren können aufgrund der fixen Länge von Paragraphen-Vektoren diese direkt als Feature für Machine Learning Classifier benutzt werden. Auch die zusätzliche Verwendung zu einem BoW - Feature ist möglich.[61]

Formale Funktionsweise

Formell kann der Vorgang zur Erzeugung eines Comment Embeddings wie folgt definiert werden. Ausgehend von einem Datensatz K von N Kommentaren, in diesem Fall Tweets, $K = (k_1, k_2, ..., k_N)$, wobei jeder Kommentar k_n, $k_n = (w_{n1}, w_{n2}, ..., w_{n,On})$, aus einer Sequenz von O_n Wörtern besteht wird versucht low-dimensionale Vektoren von Wörtern und Kommentaren zu erlernen, die in weiterer Folge als Feature-Vektor mit einer festgelegten Dimension D dienen. Ziel dabei ist es die durchschnittliche log-likelihood zu maximieren:

$$\mathcal{L} = \sum_{k_n \in K} log\ p(k_n | w_{n1}, w_{n2}, ..., w_{n,On}) + \sum_{kn \in K} \sum_{wnt \in kn} log\ p(w_{nt} | w_{nt-c}, ..., w_{m,t+c}, k_n) \tag{3.5}$$

Das c steht hierbei für die Länge des Kontextes (Wortfenster). Die Wahrscheinlichkeit der Wörter und des Kommentars wird mittels *Softmax*-Funktion, einem Multiclass Classifier, um die Vektoren in Klassenwahrscheinlichkeiten zu konvertieren, bestimmt. Die Wahrscheinlichkeit des zentralen Wortes wird daher wie folgt definiert:

$$p(w_{nt} | w_{nt-c}, ..., w_{m,t+c}, k_n) = \frac{e^{\overline{v}^T v'_{w_{nt}}}}{\sum_{w=1}^{W} e^{\overline{v}^T v'_w}} \tag{3.6}$$

wobei W dem Wortschatz, $v'_{w_{nt}}$ dem Output-Vektor von w_{nt} beziehungsweise v'_w dem Output-Vektor des jeweiligen Wortes w und $\overline{v}$ dem gemittelten Vektor des Kontextes inklusive des Kommentars entspricht:

$$\overline{v} = \frac{1}{2c+1}(v_{k_n} + \sum_{-c \leq i \leq c, i \neq 0} v_{w_{m,t+i}}) \tag{3.7}$$

Die Wahrscheinlichkeit eines Kommentars, $p(k_n | w_{n1}, w_{n2}, ..., w_{n,On})$, wird ähnlich definiert, indem lediglich die entsprechenden Variablen aus (3.6) und (3.7) ersetzt werden. Zum Trainieren der Kommentar- und Wort-Vektoren wird stochastic gradient descent verwendet, um (3.5) zu maximieren. Um effizienter zu trainieren wird im Detail hierarchical softmax verwendet, welcher eine erhebliche Zeitersparnis liefert. [61]

Implementierung

Konkret wurde im Zuge der Diplomarbeit die Java Bibliothek *deeplearning4j* verwendet. Diese implementiert den Paragraph2Vec-Algorithmus[61] in ähnlicher Form und stellt einige Optionen, auch bezüglich des zu verwendeten Modells, zur Verfügung. Als Standard für das Trainieren der Wort-Vektoren verwendet deeplearning4j hierarchical softmax und das Skipgram - Modell, da es für größere Datensätze bessere Ergebnisse als Continuous - BOW (CBOW) liefert und allgemeinere Kontexte erzeugt werden.[60] CBOW verwendet einen Kontext für die Bestimmung des Zielworts, Skipgram ein einzelnes Wort zur Prognose des Zielkontextes. In Abbildung 3.16 wird der grobe Unterschied zwischen CBOW und Skipgram nochmals dargestellt. Diese Standardeinstellungen wurden auch in

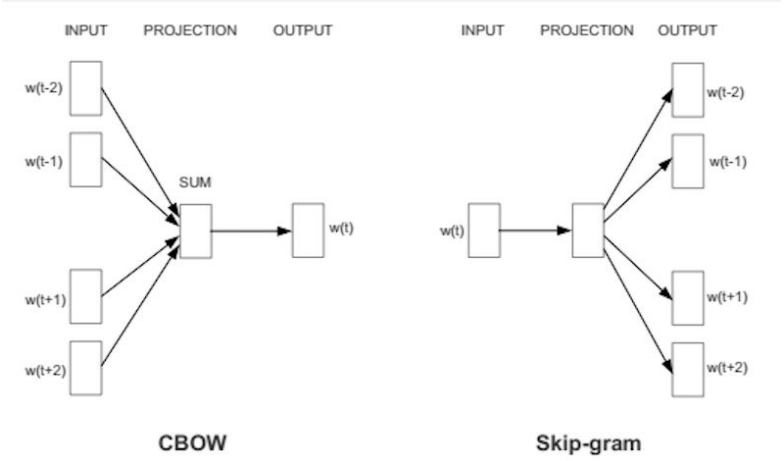


Abbildung 3.16: CBOW versus Skipgram.

unserer Implementierung für das Training der Wort Embeddings so übernommen und als einzubeziehender Kontext ein Wortfenster der Größe 10 festgelegt. Für das Erlernen des Comment Embeddings wird mittels eines Parameters das schon zuvor beschriebene Distributed Memory Modell verwendet. Der Output Vektor, welcher letztlich als Feature für den Klassifikationsprozess dient, wurde auf 100 Dimensionen(Features) begrenzt. Für die effiziente Anpassung der Parameter des neuralen Netzes wurden 10 Iterationen zugelassen. Jedes Wort, das mindestens einmal vorkam, wurde berücksichtigt. Zuletzt wurde noch die Lernrate, die Schrittweite für das Update der Koeffizienten des neuralen Netzes, damit Wörter im Feature-Raum neu positioniert werden, auf 0,025 festgelegt. Alle zuvor beschriebenen Werte der Parameter wurden kontinuierlich optimiert, indem die Ergebnisse der Klassifikation mithilfe von ausschließlich Comment Embeddings im Rahmen einer 10-fachen Kreuzvalidierung herangezogen wurden. Schließlich wurde neben den besagten Klassifikationsresultaten auch der Zeitaufwand für das Erlernen der Comment Embeddings berücksichtigt, um die in diesem Kontext optimalen Parameterwerte, welche in diesem Abschnitt zuvor ausgewiesenen wurden, festzulegen, da bei einer zu langen Dauer für das Erlernen von Comment Embeddings diese für den Praxiseinsatz unbrauchbar werden. Nachdem das Modell mit Daten trainiert wurde, können Vektoren von neuen Postings approximiert werden, indem die Distanz zu den vorhandenen Kommentaren über die im Tweet enthaltenen Wörter geschätzt wird. Nachteil ist, dass die Wörter in den Kommentaren des trainierten Modells bereits vorgekommen sein müssen. Ein großer Vorteil besteht jedoch darin, dass das Modell nicht bei jedem neuen Kommentar neu trainiert werden muss, es sich also auch für Live-Klassifikationen eignet.[39]

3.10 Attribut Auswahl

Zur Reduktion der Dimensionen werden Methoden zur Attribut-Auswahl angewendet. Diese sollen gewährleisten, dass nur Features mit ausreichend Informationsgehalt für die

Klassifikation herangezogen werden. Diese Reduktion ist speziell in diesem Algorithmus mit einer extrem großen Anzahl an Features notwendig, da sich einerseits der Vorteil der Effizienzsteigerung beim Erlernen des Modells ergibt und andererseits der Überanpassung (engl. Overfitting) vorgebeugt wird. Der Begriff Overfitting beschreibt dabei die Besonderheit, dass der Algorithmus nur für die trainierten Objekte und nicht für neue Objekte, gut funktioniert. Dies resultiert daraus, dass aufgrund der hohen Anzahl an Features nur die speziellen Eigenschaften der Trainingsdaten und nicht die Merkmale der jeweiligen Klasse erlernt werden, das Modell also zu spezifisch ist. Im Kontext dieser Arbeit wurden folgende Evaluierungsmethoden ausgewählt:

- ChiSquaredAttributeEval: Basiert auf der Berechnung des χ^2-Wertes für ein Attribut
- InfoGainAttributeEval: Basiert auf der Berechnung des Information Gain-Wertes für ein Attribut

Je größer diese Werte sind, desto wahrscheinlicher ist es, dass die Feature-Werte von der jeweiligen Klasse abhängig sind und nicht zufällig entstehen. Die genaue Berechnung und Bedeutung der Werte werden im Kapitel 4.2.2 beschrieben. Anschließend werden durch einen Ranker von Weka die aussagekräftigsten Attribute, die einen bestimmten Grenzwert übersteigen, ausgewählt.

3.11 Verwendete Classifier

In diesem Kapitel werden die im Machine Learning Algorithmus eingesetzten Classifier in ihrer grundlegenden Funktionsweise näher erläutert und welche Implementierung im Detail für die Klassifikation herangezogen wurde. Grund für die konkrete Auswahl der nachfolgend beschriebenen Classifier ist der, damit aus den populären Weka-Kategorien Funktionsbasiert, Bayes und Baum je ein Vertreter vorhanden ist.

3.11.1 Support Vector Machine (SVM)

Erste Ideen für die Support Vector Machine wurden bereits in den 60er Jahren von Wapnik und Tschervonenkis veröffentlicht.[63] In den nächsten Jahren wurde der Algorithmus kontinuierlich verbessert und überarbeitet, bis 1992 der SVM-Algorithmus in einer ähnlichen Form wie wir ihn heute kennen vorgestellt wurde.[64] Vereinfacht gesagt versucht die Support Vector Machine die Klassengrenzen der Trainings-Objekte so zu ziehen, dass ein maximaler Abstand (engl. Margin) zwischen diesen entsteht, welcher frei von jeglichen Objekten ist. Im Detail stellt jedes Trainingsobjekt einen Punkt im Vektorraum dar, der durch seinen Vektor definiert ist. Um nun die Klassen linear zu trennen wird durch die SVM eine sogenannte *Hyperebene* eingefügt. Diese Trennfläche soll dabei so verlaufen, dass der Abstand zwischen den jeweils nächsten Objekten maximal ist. Diese speziellen Objekte werden auch *Support Vektoren* genannt und dienen als Namensgeber für diesen Algorithmus. Vektoren die weiter entfernt angesiedelt sind werden für

die Berechnung nicht herangezogen. Der Abstand zwischen den Support Vektoren stellt den optimalen Spielraum für in der Zukunft zu klassifizierende Objekte, die zwischen den Klassengrenzen liegen, dar. Ist das Klassifikationsproblem nicht linear lösbar, behilft sich der SVM mit einem sogenannten "Kernel Trick", welcher die Objekte in einen höherdimensionalen Raum transformiert. Bei ausreichend hoher Dimension, in welcher die Hyperebene bestimmt wird, sind dann auch solche Probleme beziehungsweise die Vektoren linear trennbar.[65] In Abbildung 3.17 werden die zuvor präsentierten Begriffe

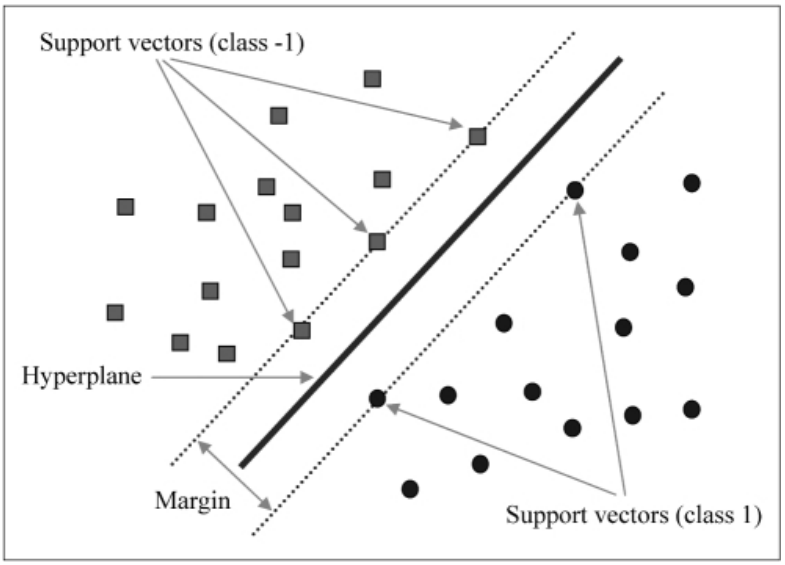


Abbildung 3.17: Optimale SVM Hyperebene im Vektorraum. [66]

und deren Zusammenhang anschaulich dargestellt. Die konkrete SVM-Implementierung, die im Kontext dieser Arbeit eingesetzt wurde, trägt den Namen *Sequential Minimal Optimization(SMO)* [67] und repräsentiert den Standard SVM-Algorithmus in Weka. Vorteil bei der Verwendung von SVM im Bezug auf diesen Machine Learning Algorithmus ist einerseits die schnelle Klassifikation, da die Berechnung nur auf wenigen Supportvektoren basiert und nicht am gesamten Trainingsset und andererseits die gute Performance bei einem hochdimensionalen Feature-Set, da SVM robust gegenüber Überanpassung ist.

Anschließend werden die verschiedenen Parameter, die im Zuge der Parameteroptimierung angepasst werden, vorgestellt.

Parameter C
Der Parameter C wirkt sich auf den Trade-off zwischen der Modell-Komplexität und dem Trainings-Error aus, das heißt je höher der Wert desto mehr Support Vektoren werden eingesetzt, um die Anzahl an nicht separierbaren Instanzen (Training-Errors) zu verhindern. Dadurch wird das Modell jedoch komplexer und es besteht die Gefahr des Overfittings.

ε-Parameter
Ähnlich wie beim C-Parameter verhält es sich mit dem Epsilon-Parameter der ebenfalls, aber in einer anderen Weise, die Anzahl an Support Vektoren durch dessen Wert beeinflusst. Im Detail wird dadurch der Grad der Genauigkeit der Approximation des Modells festgelegt. Je höher der Epsilon-Wert desto weniger Support-Vektoren beziehungsweise geringere Anforderungen an den Optimierungsprozess. Dieser Umstand resultiert wiederum in einem allgemeineren Modell.

Kernel
Als Kernel-Funktion wurde im Zuge dieser Arbeit der Polynomial-Kernel der Form $K(x, y) = (< x, y > +1)^p$ gewählt, der sich in einzelnen Tests als der erfolgversprechendste Kernel herausgestellt hat und speziell im Rahmen von Natural Language Processing gut funktioniert. X und Y stellen die Vektoren von Features da, p den Grad des Polynoms. Wie bereits erwähnt ist diese Kernel-Funktion notwendig, um die Daten leichter in einen höherdimensionalen Raum zu transformieren. Leichter daher, da statt der aufwendigen, schwierigen Berechnung des Skalarprodukts die Kernel-Funktion verwendet wird, die sich wie ein Skalarprodukt verhält.

Kalibrator
Mithilfe eines Kalibrators lassen sich die Ausgaben des Klassifikationsmodell in eine Wahrscheinlichkeitsverteilung je Klasse transformieren. Das heißt es wird je Instanz die Wahrscheinlichkeit je Klasse geschätzt, da diese Information für manche Anwendungen vorteilhaft ist.

3.11.2 Naive Bayes

Der *Naive Bayes*-Classifier basiert auf dem Satz von Bayes, der nach Thomas Bayes benannt und von diesem im 18.Jahrhundert entscheidend mitbegründet wurde. Jedes Objekt, beschrieben durch seinen Feature-Vektor, wird derjenigen Klasse zugeordnet, der es mit der größten Wahrscheinlichkeit angehört. Im Detail kann der Naive Bayes als ein Spezialfall eines Bayesschen Netzes angesehen werden, bei dem von der naiven, extrem vereinfachten Annahme ausgegangen wird, dass Features nur vom Klassenattribut abhängig sind. Untereinander sind die Features also bedingt unabhängig bei gegebenem Klassennamen. Eine weitere simplifizierende Annahme des Naive Bayes ist, dass der Klassifikationsprozess von keinen versteckten beziehungsweise verborgenen Attributen beeinflusst wird.[68] Aus diesen Hypothesen heraus resultiert die Bayessches Netz-Darstellung eines Naive Bayes Classifier, welche in Abbildung 3.18 gezeigt wird, wobei C das Klassenlabel und X_i die davon abhängigen Features darstellt.

Durch die Ableitung des Naive Bayes-Classifier von dem zuvor erwähnten Satz von Bayes und den vereinfachten Annahmen im Bezug auf die Korrelation der Features untereinander ergibt sich folgende Gleichung, die je Klasse berechnet wird[68]:

$$p(C_k|x_1, ..., x_n) = p(C_k) \prod_{i=1}^{n} p(x_i|C_k) \tag{3.8}$$

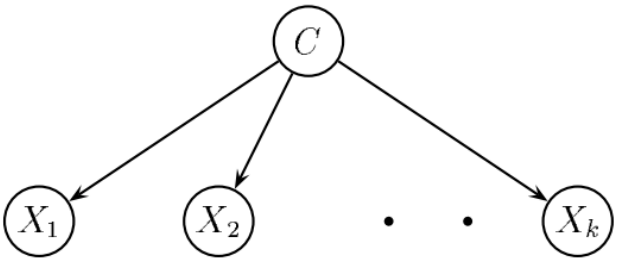


Abbildung 3.18: Bayessches Netz des Naive Bayes Classifiers. [68]

Für das Label des zu klassifizierenden Objekts wird jene Klasse ausgewählt, welche die größte *A Posteriori*-Wahrscheinlichkeit aufweist, also dem Modus der A-posteriori-Verteilung entspricht (Maximum A posteriori - MAP). Ein Vorteil aus dieses naiven Ansatzes stellt die performante Berechnung der Klassenwahrscheinlichkeiten dar. Dadurch und aufgrund der Tatsache des sehr hochdimensionalen Feature-Vektors dieser Arbeit, wird der Naive Bayes dem komplexeren, etwas besser klassifizierenden *Bayes Net*-Classifier vorgezogen. Die genaue Implementation in Weka beruht auf der Arbeit von John et al.[68], welche ebenfalls die Stärken dieses Klassifikators erkannten und weiter verbesserten, indem für die Schätzung der stetigen Verteilung der kontinuierlichen Werte von nummerischen Attributen nicht parametrisierte Kerndichteschätzer eingesetzt wurden, anstatt von einer einfachen Normalverteilung auszugehen. Somit kann für ein nummerisches Feature die richtige Dichtefunktion zur Berechnung der Klassenwahrscheinlichkeit korrekt angewendet werden.

3.11.3 Random Decision Forest

Zuletzt wurde ein Baum-Modell für die Klassifikation getestet. Im Detail handelt es sich dabei um den *Random Decision Forest*-Classifier, welcher aus einem Vielfachen an *Decision Trees* besteht, deren Modell durch eine bestimmte Art der Randomisierung der Features und Trainingsobjekte gewachsen ist. Jeder dieser Bäume erhält eine Stimme für die Entscheidung über die Klasse eines Objekts, wobei jene Klasse mit den meisten Stimmen der Instanz zugewiesen wird (engl. Majority Voting). Der Random Decision Forest wurde dem einfachen Random Decision Tree vorgezogen, da er gegenüber Overfitting resistenter ist. Tin Kam Ho war der erste der einen Random Decision Forest Algorithmus implementierte und dabei die sogenannte *Random Subspace Method* (auch Feature Bagging), mit welcher anstatt des gesamten Feature-Sets nur zufällig ausgewählte Merkmale für das Training des Forests herangezogen werden, umsetzte.[69] [70] Breiman erweiterte den Algorithmus von Ho indem er seine Methodik des Bootstrap Aggregating (auch Bagging), bei dem die Trainingsobjekte zufällig ausgewählt werden, wobei diese pro generiertem Trainingsset auch mehrmals vorkommen können, für den Prozess der Baummodellierung ableitete.[71] Dieser Algorithmus stellt auch die Basis

für die Implementation des *Random Decision Forest*-Classifiers in Weka dar. Durch die Bagging Varianten wird die Korrelation zwischen den Bäumen vermindert, da jeder mittels verschiedenen Features und Trainingsobjekten wächst und in weiterer Folge das Problem des Overfitting durch die Reduktion der Varianz bei nahezu gleichbleibenden Bias verhindert.

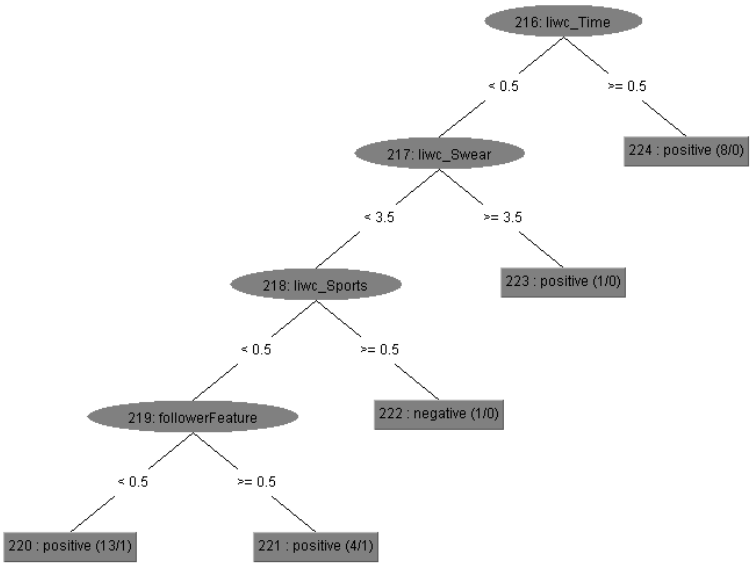


Abbildung 3.19: Ausschnitt eines Eintscheidungsbaumes.

Die Bäume die für die Vorhersage der Klassen generiert werden, bestehen aus internen Knoten und Blättern. Die internen Knoten stellen dabei die Features da, von denen für jeden möglichen Wert des Features Kanten zu Nachfolgeknoten führen, die Blätter enthalten eine der möglichen Klasse. Durch die Feature-Werte der jeweiligen Trainingsinstanz wird der Weg durch den Baum bestimmt und diese je nach Wert des resultierenden Blattes klassifiziert. Abbildung 3.19 zeigt einen Ausschnitt des Entscheidungsbaums, der basierend auf den Features *liwc_Swear*, *liwc_Sports*,*liwc_Time*, *liwc_Tentative* und *followerFeature* modelliert wurde.

Vorteil des Decision Forest Algorithmus ist, dass er sehr effizient bei großen Datenmengen arbeitet und die Tatsache, dass die einzelnen Bäume schnell trainiert werden können. Aufgrund der hohen Anzahl an Bäumen und Features benötigt es aber im Endeffekt um ein Vielfaches länger für die Erstellung des Klassifikationsmodells, als die zuvor vorgestellten Classifier.

Die Parameter für den Random Decision Forest sind die folgenden:

Anzahl an Iterationen

Bei der Anzahl an Iterationen handelt es sich um die zu erstellenden Bäume, die jeweils eine Stimme pro Tweet im Klassifikationsprozess abgeben dürfen.

Anzahl an Features
Diese Anzahl gibt an wie viele Attribute zufällig für das Feature-Bagging der einzelnen Bäume ausgewählt werden. Der Standardwert beträgt log_2 von der Gesamtmenge an Features.

Maximale Baumtiefe
Der Parameter für die Baumtiefe regelt die maximale Tiefe der Bäume.

Minimale Anzahl an Instanzen
Dieser Parameter gibt die minimale Menge an Instanzen in den Blättern der Bäume an.

Minimaler Information Gain für Split
Dabei handelt es sich um den minimalen Information Gain der Features in den Knoten, nach denen die Instanzen aufgeteilt werden.

3.12 Zusammenfassung

In diesem Kapitel wurde die Funktionsweise des implementierten Supervised Machine Learning Algorithmus und die wichtigsten Teilgebiete, die für dessen Umsetzung notwendig sind, vorgestellt. Dies beinhaltet den Korpus aus Twitter-Daten, welche mithilfe der *Twitter Streaming-* und *Twitter Rest-API* gesammelt wurden und das Datenmodell das für die Speicherung dieser entworfen wurde. Des weiteren wurde das Annotations-Tool, die verschiedenen Klassen und die Kriterien, nach welchen die manuelle Klasseneinteilung der Tweets erfolgt ist, vorgestellt. Dazu wurde die Interrater-Reliabilität bestimmt, welche so gut ausfiel, dass die strikte Unterscheidung von nur zwei Klassen beziehungsweise die Verwendung des Klassifikationsmodell möglich ist. Als nächstes beschrieben wurden die Preprocessing-Schritte die unternommen wurden, damit die Terme der Tweets für gewisse Features optimal modifiziert sind. Der interessanteste Teil des Methodik-Kapitels ist möglicherweise die große Anzahl an verschiedensten Features, die im Detail vorgestellt wurden. Besonders vielversprechende beziehungsweise komplexe Feature-Kategorien wie Merkmale aus dem Twitter-Netzwerk, Typed Dependencies oder Comment Embeddings wurden noch detaillierter behandelt. Auch das Hasswortlexikon, welches für manche Features notwendig ist, wurde ein Abschnitt gewidmet. Die Grundlagen für die Reduktion der Dimension des Feature-Vektors, die bei der großen Menge an Attributen sehr hoch ist, und die für spätere Experimente eingesetzt werden, wurden zusätzlich kurz charakterisiert. Zuletzt konnten Details über die verwendeten Klassifikatoren *Support Vector Machine*, *Naive Bayes* und *Random Decision Forest* kennengelernt werden.

KAPITEL 4

Evaluierung

4.1 Einleitung

Das Kapitel “Evaluierung“ soll grundsätzlich die verschiedenen Experimente, die einzelnen Features, Feature-Sets und Klassifikatoren aufschlüsseln, analysieren und die korrespondierenden Messwerte dieser präsentieren. Dies dient dazu die definierten Forschungsfragen zu beantworten und einen möglichst optimalen Klassifikations-Algorithmus zu finden. Natürlich werden auch die verschiedenen Methoden zur Evaluierung, Parametereinstellungen der Classifier und Kennzahlen, die als Indikatoren der Güte dienen, vorgestellt.

4.2 Evaluierungsmethoden

In diesem Kapitel werden die eingesetzten Methoden zur Evaluierung der einzelnen Features und Feature-Sets in Kombination mit den verschiedenen Classifiern beschrieben.

4.2.1 Evaluierungsmethoden Features

Für die Evaluierung der Features werden verschiedene Methoden von Weka herangezogen, welche auch für die Attribut Auswahl genutzt werden. So zum Beispiel werden Referenzwerte wie der Information-Gain oder der Chi-Quadrat-Wert je Attribut berechnet. Zusätzlich wird auch eine Subset-Evaluierung durch die Methode “CfsSubsetEval“ vorgenommen, die eine Auswahl der besten Features extrahiert und so weitere Informationen, was die Abhängigkeiten der Features untereinander betrifft, zur Verfügung stellt. Visualisierungen (Streudiagramme), bei denen die Attributwerte der Instanzen im Bezug auf die jeweils annotierte Klasse gegenübergestellt werden, sollen weitere Erkenntnisse liefern.

Information Gain
Unter dem Information Gain versteht man den Informationsgewinn durch ein bestimmtes

Feature. Dieser stellt also die Änderung der Entropie, in diesem Fall den Messwert für die Unsicherheit im Bezug auf eine Klasse, vor und nach Einbeziehung des Features dar. In Weka wird der Information Gain pro Klasse berechnet, wodurch sich folgende Formel ergibt:

$$IG(K, A) = H(K) - H(K|A) = p_K \cdot log_2\, p_K - (-p_{K|A} \cdot log_2\, p_{K|A}) \tag{4.1}$$

Die Variable K repräsentiert dabei die jeweilige Klasse, die Variable A steht für das ausgewählte Attribut, wodurch p_K die Wahrscheinlichkeit, dass ein Objekt in die Klasse K fällt, und $p_{K|A}$ die Wahrscheinlichkeit, dass ein Objekt in die Klasse K nach Einbezug des Attributs A fällt, darstellt.

Chi Quadrat Test
Durch den χ^2-Test soll auf stochastische Unabhängigkeit von Merkmalen überprüft werden. Im Detail und Bezug auf ein Klassifikationsproblem, wie es in dieser Arbeit der Fall ist, wird getestet, ob die jeweiligen Werte der Features von einer Klasse abhängig sind. Formal wird als Nullhypothese H_0 die Unabhängigkeit der Features von den Klassen angenommen, wodurch als Alternativhypothese H_1 die Abhängigkeit des Feature-Wertes von der Klasse aufgestellt wird. Ist der χ^2-Referenzwert, der bei ausreichend großen erwarteten Häufigkeiten annähernd χ^2-verteilt ist, klein, wird die Nullhypothese angenommen, bei einem großen Wert, der ein gewisses Signifikanzniveau übersteigt, verworfen. Für die Evaluierung der Features ist also ein möglichst großer χ^2-Wert interessant, da dadurch die Abhängigkeit von Feature und Klasse mit hoher Wahrscheinlichkeit signifikant ist und daher solche Merkmale am besten geeignet sind, um die jeweilige Klasse zu charakterisieren und sie von der anderen abzugrenzen. Für die Berechnung des χ^2-Wertes wird folgende Formel herangezogen:[72]

$$\chi^2 \;=\; \sum_{i=1}^{m}\sum_{j=1}^{k} \frac{(A_{ij} - E_{ij})^2}{E_{ij}} \tag{4.2}$$

Dabei stellt k die Anzahl der Klassen,
m die Anzahl der Intervalle des nummerischen oder nominellen Features,
A_{ij} die Anzahl an Objekte im i-ten Intervall und der j-ten Klasse und E_{ij} die erwartete Anzahl an Objekten im i-ten Intervall und der j-ten Klasse darstellen. E_{ij} berechnet sich aus

$$E_{ij} \;=\; \frac{R_i \cdot C_j}{N} \tag{4.3}$$

wobei R_i die Anzahl an Objekten im i-ten Intervall ($\sum_{j=1}^{k} A_{ij}$),
C_j die Anzahl an Objekten in der j-ten Klasse ($\sum_{i=1}^{m} A_{ij}$) und
N die Gesamtheit an Objekten ($\sum_{i=1}^{m} R_i$) repräsentiert.

CFS Subset Evaluation
Die Correlation-based Feature Selection (CFS) ist eine Methode, welche es ermöglicht ganzen Feature-Subsets Prüfwerte zuzuordnen. Dieser Wert ist größer, wenn das Subset Features enthält die im hohen Maße mit dem Klassenattribut korrelieren, nicht jedoch

mit den anderen Attributen des Subsets. Zur Berechnung dieses Wertes steht folgende Formel zur Verfügung:[73]

$$G_s = \frac{k \cdot r_{cf}}{\sqrt{k + k \cdot (k-1) \cdot r_{ff}}} \tag{4.4}$$

s stellt dabei das Attributeset mit k Attributen dar,
r_{cf} modelliert die Korrelation der Attribute zur Klasse und
r_{ff} die Inter-Korrelation zwischen den Attributen.

4.2.2 Evaluierungsmethoden Klassifizierung

Für die Evaluierung der von uns ausgewählten Feature-Sets und Classifier werden die Kennzahlen, die Weka bei einer k-fachen Kreuz-Validierung (engl.Cross-Validation) liefert, herangezogen. Konkret wird eine 10-fache Cross-Validation verwendet, die das annotierte Trainingset in zehn gleich große Teile aufteilt, wobei in den darauffolgenden zehn Validierungs-Runden pro Durchlauf jeweils neun Sets in unterschiedlicher Kombination für das Training und ein Set zur Validierung dienen. Die Evaluierung durch Kreuzvalidierung bringt den Vorteil, dass die Ergebnisse stabiler sind als bei der Verwendung von nur einem Testsplit, da die Kennzahlen von der Aufteilung der Daten abhängig sind und daher geringfügig variieren.

Confusion Matrix
Durch eine Confiusion Matrix kann in einem Supervised Machine Learning Algorithmus die Performance des Classifiers leicht visualisiert werden. Die vom Classifier bestimmten Klassen und die tatsächlichen Klassen der Instanzen werden in einer Matrix gegenüber gestellt. Tabelle 4.1 zeigt eine symbolische Darstellung der Confusion Matrix im Bezug

		Predicted	
		Hass	Neutral
Actual	Hass	TP	FN
	Neutral	FP	TN

Tabelle 4.1: Darstellung einer Confusion Matrix.

auf diese Arbeit. Auf die in der Matrix gegenübergestellten Variablen wird nachfolgend eingegangen:

- **True Postitives(TP):** Anzahl der korrekt klassifizierten Hasspostings
- **False Postitives(FP):** Anzahl der fälschlicherweise als Hasspostings klassifizierten neutrale Postings
- **True Negatives(TN):** Anzahl der korrekt klassifizierten neutralen Postings

- **False Negatives(FN):** Anzahl der fälschlicherweise als neutrale Postings klassifizierten Hasspostings

Weka stellt die Zahlen zusätzlich mit der Gesamtanzahl an Instanzen gegenüber, wodurch sich relative Klassifizierungs-Raten ergeben.

Accuracy
Eine der am häufigsten genutzte Kennzahl ist die sogenannte *Accuracy*. Diese Prozentzahl gibt an, wie viele Objekte korrekt klassifiziert worden sind. Für die Ermittlung ergibt sich aufgrund des in dieser Arbeit vorliegenden "2-Klassen" Problems nachfolgende Formel:

$$A = \frac{TP + TN}{N} \tag{4.5}$$

Diese stellt also im Detail das Verhältnis von der Anzahl an True Positive(TP) und True Negative(TN) klassifizierten Instanzen, also den beiden Fällen bei denen der Classifier die richtige Klasse für das Posting bestimmt hat, zur Gesamtheit an Objekten im Trainingskorpus dar.

Precision
Die *Precision* gibt Auskunft über den relativen Anteil an richtigen Klassifikationen in der jeweiligen Klasse:

$$P_{Hass} = \frac{TP}{TP + FP} \qquad P_{Neutral} = \frac{TN}{TN + FN} \tag{4.6}$$

Recall
Der *Recall* gibt Auskunft über den relativen Anteil an gefundenen True Positives beziehungsweise False Positives:

$$R_{Hass} = \frac{TP}{TP + FN} \qquad R_{Neutral} = \frac{TN}{TN + FP} \tag{4.7}$$

F-Score
Ein aus Precision und Recall kombinierter Performance Messwert ist der sogenannte *F-Score*. F_1, die traditionelle Berechnung des F-Measures, beschreibt dabei das harmonische Mittel aus Precision und Recall:

$$F_1 = \frac{2 \cdot R^M \cdot P^M}{R^M + P^M} \tag{4.8}$$

P^M und R^M stellen dabei die gemittelte Precision beziehungsweise den gemittelten Recall dar.

4.3 Feature-Evaluierung

Im nachfolgenden Kapitel werden ausgewählte Features einzeln evaluiert. Auch dafür wird eine 10-fache Cross-Validation benutzt. Je Feature-Kategorie werden die im Kapitel 4.2.1

beschriebenen Methoden angewandt und die Ergebnisse mit selektierten Visualisierungen von Features in Form von Histo- oder Streudiagrammen unterstrichen. Genauer gesagt sollen die Diagramme die Abhängigkeiten zwischen Klassen und Attributwerten darstellen. Die Featurewerte werden dabei auf der X-Achse und die manuell annotierten Klassen auf der Y-Achse abgebildet. Die Farbe Rot in den Diagrammen symbolisiert dabei die Klasse Hassposting, Blau ein neutrales Posting. Diese Vorgehensweise und das Farbschema wird für die nachfolgenden Histogramme und Streudiagramme, wenn nicht anders ausdrücklich erwähnt beibehalten. Auf die detailierte Einzel-Evaluierung von allen Message-N-Grams, Typed Dependency N-Grams, Character-N-Grams und Comment Embeddings wurde aufgrund der großen Anzahl an resultierenden Attributen beziehungsweise im Fall von Character-N-Grams und Comment Embeddings der fehlenden Aussagekraft verzichtet. Als Beispiel für Message- und Typed Dependency-Features sollen deshalb nur deren Unigrams detailliert ausgewertet werden.

4.3.1 Message/BoW-Unigram Evaluierung

Bei der Evaluierung der Message-Unigrams soll offenbart werden, welche Wörter in dem annotierten Datensatz am meisten über die entsprechenden Klassen aussagen. Um dies zu erreichen, wurden in Tabelle 4.2 und Tabelle 4.3 die ersten 15 Wörter nach ihrem Informationsgewinn beziehungsweise χ^2-Wert gereiht. Als Unterschiede in den

Avg. IG	Avg. Rank	Attributename
0.072 +- 0.003	1 +- 0	neger
0.042 +- 0.001	2.3 +- 0.46	gesindel
0.04 +- 0.001	2.7 +- 0.46	hurensohn
0.025 +- 0.001	4.4 +- 0.66	du
0.023 +- 0.001	5.2 +- 0.6	https
0.022 +- 0.001	5.7 +- 1.1	arschloch
0.017 +- 0.001	8.3 +- 0.46	dreckspack
0.017 +- 0.001	8.7 +- 0.46	musel
0.013 +- 0	10.3 +- 0.46	bastard
0.012 +- 0.001	10.7 +- 0.46	terroristen
0.01 +- 0.001	13.5 +- 1.2	vollpfosten
0.01 +- 0.001	13.7 +- 1.68	untermensch
0.01 +- 0	13.8 +- 1.54	wichser
0.009 +- 0.001	14.6 +- 1.62	ein
0.009 +- 0	15.2 +- 1.47	krimigranten

Tabelle 4.2: Information Gain Ranking der BoW Unigrams.

Avg. χ^2-Merit	Avg. Rank	Attributename
191.55 +- 6.048	1 +- 0	neger
107.009 +- 2.937	2.5 +- 0.5	hurensohn
107.728 +- 3.203	2.5 +- 0.5	gesindel
81.468 +- 4.465	4.4 +- 0.49	du
79.511 +- 4.698	4.6 +- 0.49	https
57.644 +- 3.426	7 +- 0	arschloch
47.096 +- 2.535	8.3 +- 0.46	dreckspack
46.104 +- 2.377	8.7 +- 0.46	musel
37.717 +- 2.831	10.2 +- 0.6	terroristen
33.985 +- 1.231	11.3 +- 0.64	bastard
32.227 +- 3.034	11.6 +- 0.66	ein
27.994 +- 2.074	13.5 +- 1.12	in
25.163 +- 1.205	15.5 +- 1.2	wichser
25.165 +- 2.529	15.5 +- 2.42	untermensch
25.164 +- 1.768	15.7 +- 1.19	vollpfosten

Tabelle 4.3: Ranking der BoW Unigrams nach χ^2-Wert.

zwei verschiedenen Ranglisten können mehrere geringfügige Positionstäusche zwischen nebeneinander liegenden Features ausgemacht werden. Zusätzlich fällt auf, dass das Wort *"Vollpfosten"* im Ranking nach Information Gain um vier Plätze weiter oben zu finden ist, das Wort *"in"* dafür nur im Ranking nach χ^2-Wert einen Platz unter den Top 15 findet. Durch die Streudiagramme des Wortes *"Gesindel"* 4.1 und *"du"* 4.2 wird ein gewisser Zusammenhang zwischen Hasspostings und diesen Wörtern erkennbar. Ist dieser bei dem

Wort “Gesindel“ noch wenig verwunderlich, so ist es bei dem Wort “du“ nicht selbstverständlich. Grund dafür ist jedoch, dass “du“ in vielen Beleidigungen als Anredeform verwendet wird. Generell kann ausgemacht werden, dass bekannte Schimpfwörter und abwertende Begriffe bezüglich ganzer Gruppen, wie beispielsweise “Dreckspack“ oder “Krimigranten“, den höchsten Mehrwert liefern.

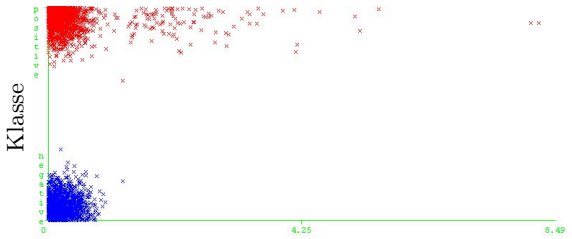


Abbildung 4.1: Streudiagramm des Unigrams “Gesindel“.

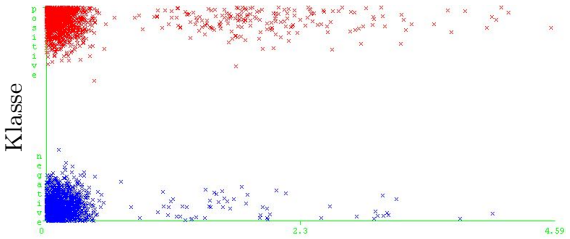


Abbildung 4.2: Streudiagramm des Unigrams “du“.

4.3.2 Typed Dependency Evaluierung

Durch die Einzel-Evaluierung der Typed Dependency Features sollen die Beziehungstypen mit dem meisten Einfluss gefunden werden. Weiters wird in diesem Kapitel das Ranking der Message Bigrams nach Information Gain mit dem IG-Ranking der Typed Dependencies gegenübergestellt und verglichen. Grund dafür ist die ähnliche Erkenntnis die diese Features liefern sollen, nämlich über den Kontext der Wörter im Tweet. Tabelle 4.4 und 4.5 dienen vorerst jedoch wie bereits bekannt für die Gegenüberstellung beider Top 15 Reihungen, die sich nach Information Gain beziehungsweise χ^2-Wert der Features ergeben.

Avg. IG	Avg. Rank	Attributename
0.015 +- 0.001	1 +- 0	pd(sein,hurensohn)
0.013 +- 0.001	2.3 +- 0.46	nk(hurensohn,ein)
0.012 +- 0.001	3 +- 0.77	nk(neger,ein)
0.011 +- 0	3.7 +- 0.46	nk(gesindel,der)
0.009 +- 0.001	5 +- 0	nk(neger,der)
0.007 +- 0	6.1 +- 0.3	nk(musel,der)
0.005 +- 0	8.4 +- 1.28	nk(gesindel,dieser)
0.005 +- 0	9.5 +- 2.16	mo(ekelhaft,so)
0.005 +- 0.001	11.6 +- 5.2	nk(arschloch,ein)
0.005 +- 0.001	12.8 +- 4.58	pd(sein,ekelhaft)
0.005 +- 0.001	14.2 +- 5.74	sb(sein,du)
0.004 +- 0	16.5 +- 5.92	ng(gehen,nicht)
0.004 +- 0	16.5 +- 4.43	pd(https,//t)
0.004 +- 0.001	16.8 +- 6.26	pd(sein,neger)
0.004 +- 0	16.9 +- 3.81	nk(an,armenier)

Tabelle 4.4: Information Gain Ranking des Typed Dependency Features

Avg. χ^2-Merit	Avg. Rank	Attributename
38.91 +- 1.959	1 +- 0	pd(sein,hurensohn)
32.02 +- 2.068	2.4 +- 0.66	nk(hurensohn,ein)
30.058 +- 2.034	3 +- 0.77	nk(neger,ein)
29.076 +- 1.077	3.6 +- 0.49	nk(gesindel,der)
24.186 +- 1.619	5 +- 0	nk(neger,der)
18.342 +- 1.229	6.2 +- 0.4	nk(musel,der)
15.409 +- 2.396	8.9 +- 3.59	nk(arschloch,ein)
15.125 +- 2.546	9.4 +- 3.53	sb(sein,du)
14.057 +- 2.118	10.1 +- 2.74	pd(sein,ekelhaft)
13.491 +- 0.984	10.3 +- 2.19	nk(gesindel,dieser)
12.742 +- 0.973	11.5 +- 2.01	mo(ekelhaft,so)
12.509 +- 1.685	13.2 +- 4.24	sb(sein,er)
11.179 +- 1.594	16.6 +- 5.52	nk(terrorist,der)
10.989 +- 1.218	17.6 +- 4.69	nk(hurensohn,der)
11.18 +- 1.75	17.6 +- 6.25	nk(genozid,der)

Tabelle 4.5: Ranking des Typed Dependency Features nach deren χ^2-Wert.

Dabei fällt auf, dass sich die ersten sechs Typed Dependency Triple decken. Danach

ist die Reihung der Attribute komplett unterschiedlich, da diese nachfolgenden Feature - Evaluationswerte alle innerhalb eines sehr kleinen Wertebereichs (0,004-0,005 bei IG beziehungsweise 11,18-15,125 bei den χ^2-Werten) liegen, also sie im ungefähr gleichen Ausmaß für die Klassifizierung beitragen. Durch diesen Umstand ergibt sich keine eindeutige Reihung wie in anderen Featurekategorien. Betrachtet man die Beziehungstypen der Wörter die es in die jeweilige Top 15 geschafft haben ergibt sich folgende Statistik 4.6:

	Anzahl in den Top 15		
Abhängigkeit	***IG***	χ^2	***Gesamt***
NK	8	10	18
PD	4	2	6
SB	1	2	3
MO	1	1	2
NG	1	0	1

Tabelle 4.6: Übersicht über die Vorkommnisse der Abhängigkeitstypen in den Top 15 Rankings nach IG und χ^2-Wert.

Klarer Spitzenreiter sind die Typed Dependencies deren Abhängigkeit mittels **NK**-Tag ausgewiesen sind. Auffällig ist, dass ein solches Tripel, welches sich aus Artikel und Hauptwort zusammensetzt, in den Top 15 als Nomen immer ein hasserfülltes Wort beinhaltet, wodurch deren hohes Ranking leicht zu erklären ist. Viel interessanter ist die Betrachtung der Tripel mit den Beziehungstypen **SB** und **NG**, welche keinerlei Hasswörter enthalten. Auf den ersten Blick weisen die Typed Dependencies *sb(sein,du)* und *sb(sein,er)* keinerlei aufsehenerregende Umstände auf, die eine Zuordnung der Instanzen in eine der Klassen erleichtern. Bei genauerer Betrachtung der zugehörigen Tweets lässt sich jedoch feststellen, dass diese Phrasen häufig bei direkten oder indirekten Beleidigungen zur Anwendung kommen. Ähnlich verhält es sich mit dem Triple *ng(gehen,nicht)*. Diese Phrase wird oft für den Satzbau benutzt, wenn Ängste über die Zukunft zum Ausdruck gebracht werden sollen, wie beispielsweise *"Das kann doch nicht gut gehen!"*. Einerseits wurde im Zusammenhang mit der Flüchtlingswelle davon Gebrauch gemacht und sollte zusätzlich zu anderen hasserfüllten Botschaften die Ablehnung beschreiben. Andererseits zeigt der Tweet *"Ich weiß nie, was Menschen davon haben, so viel Hass zu verbreiten. Gut gehen kann es einem damit doch auch nicht."* gut, dass auch in neutralen Postings solche Phrasen im positiven Sinn Anwendung finden. Für die Typed Dependency Typen **CJ**, **CD**, **OA**, **OC** und **PNC** die es zwar nicht in die Top 15 der beiden Ranglisten geschafft haben, existieren dennoch einige interessante Triple die einen IG-Wert von 0,001 bis 0,004 generieren. Die Abhängigkeiten **CJ** und **CD** fungieren beispielsweise als Label für Aufzählungen von gehässigen Bezeichnungen für Gruppen oder Einzelpersonen, veranschaulicht durch die Beispiele *cd(neger,und)* (0,003 IG) und *cj(und,gesindel)* (0,003 IG). Auch durch die Bezeichnung für Akkusativobjekte **OA** lassen sich Phrasen, die als Indikator für eine Klasse dienen, extrahieren. Als Beispiele eignen sich die Typed Dependencies *oa(planen,anschlag)* und *oa(umbringen,mich)* die einen IG von 0,001 beziehungsweise

0,002 aufweisen. Für kausale Objekte **OC** können die Beispiele *oc(können,abschneiden)* (0,002 IG), *oc(wollen,umbringen)* (0,001 IG) und *oc(werden,versenken)* (0,001 IG) für die Identifikation von Hasspostings aufgelistet werden. Durch das Tag **PNC** werden Eigennamen gekennzeichnet, unter die auch Hashtags fallen und diese Beziehung gerade deshalb für Tweets relevant ist. So generieren die Typed Dependencies *pnc(#pegida,#afd)* und *pnc(#fdp,#npd)* einen Informationsgewinn von jeweils 0,002. Die restlichen Beziehungstypen **AG**, **APP**, **CC**, **CM**, **DA**, **OA2** und **SBP** nach denen gefiltert wird, liefern weder einen Information Gain noch einen χ^2-Wert größer null, in weiterer Folge also auch keinen Mehrwert für die Klassifizierung.

Tabelle 4.7 listet nun die Top 20 Message-Bigrams nach Information Gain auf. Beim Vergleich mit der Tabelle der Typed Dependency Evaluierung können sofort viele übereinstimmende Hasswörter erkannt werden, die sowohl in den Top 15 der Typed dependency Triple als auch in den Top 20 der Message Bigrams enthalten sind. Insbesondere der Wortkontext aus NK-Beziehungen wird ebenfalls durch Message-Bigrams dargestellt, da in diesen Beziehungen oft nebeneinanderliegende Terme das Type Dependency Triple bilden. So finden sich beispielsweise *“ein hurensohn“* und *“ein neger“*, welche in Tabelle 4.7 in gestemmter Form vorliegen, sowohl in den besten Typed Dependencies und Message Bigrams wieder. Auch die Wortphrase *“so ekelhaft“* ist bereits in der Beziehung *mo(ekelhaft,so)* beinhaltet. Trotzdem stellen Message Bigrams zusätzliche Informationen zur Verfügung und haben dadurch ihre Daseinsberechtigung. Betrachtet man nämlich die Wortphrasen *“was für“*, *“so ein“* oder das mit einem Information Gain von 0,003 und daher nicht gelistete Bigram *“weg mit“*, so lassen sich eindeutig Kontexte extrahieren, die zwar nicht syntaktisch in Abhängigkeit stehen, aber dennoch bei Beleidigungen und hasserfüllten Postings des öfteren für die Satzbildung Anwendung finden.

Zuletzt soll noch die Klasse ausfindig gemacht werden, für die die Features *“https t“* und *pd(https,//t)* Hinweise der Zugehörigkeit geben. Diese werden offensichtlich aus gekürzten Urls, die in Twitter aufgrund der begrenzten Zeichenanzahl meistens genutzt werden, gebildet und sind deshalb interessant, da dieses Merkmal sowohl in den Typed Dependency Features unter den Top 15 positioniert ist und innerhalb der Message Bigram Rangliste sogar auf Platz eins rangiert. Zur Untersuchung des Umstandes dient das Histogramm 4.3 und das Streudiagramm 4.4 des Bigrams *“https t“*.

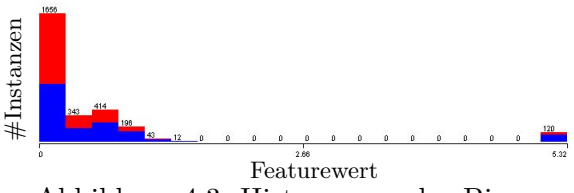


Abbildung 4.3: Histogramm des Bigrams “https t“.

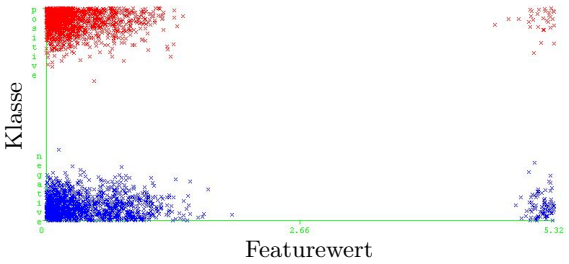


Abbildung 4.4: Streudiagramm des Bigrams “https t“.

Avg. IG	Avg. Rank	Attributename
0.026 +- 0.002	1 +- 0	https t
0.025 +- 0.002	2 +- 0	t co
0.011 +- 0.001	3 +- 0	ein hurensohn
0.008 +- 0	4.1 +- 0.3	du hurensohn
0.007 +- 0.001	5.4 +- 1.28	was fur
0.007 +- 0.001	6 +- 0.63	is terrorist
0.006 +- 0.001	8.4 +- 2.42	so ein
0.006 +- 0	8.8 +- 1.89	ein neg
0.006 +- 0	10.8 +- 2.36	so ekelhaft
0.005 +- 0	12 +- 2	gesindel https
0.005 +- 0	13.8 +- 2.93	neger und
0.005 +- 0	13.8 +- 2.56	du bastard
0.005 +- 0.001	14 +- 5.73	dreckspack https
0.005 +- 0	14.3 +- 2.61	die neg
0.005 +- 0	14.7 +- 4.03	neger in
0.004 +- 0.001	18.8 +- 5.06	genozid an
0.004 +- 0	19 +- 1.84	das gesindel
0.004 +- 0.001	19.3 +- 5.04	das sind
0.004 +- 0	19.5 +- 2.77	der neg
0.004 +- 0	23.3 +- 3.98	du untermensch

Tabelle 4.7: Information Gain Ranking des Message-Bigram Features

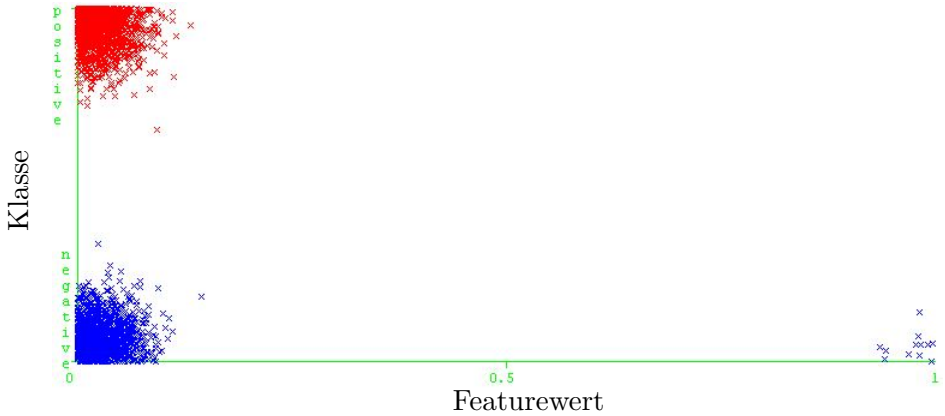


Abbildung 4.5: Streudiagramm des Typed Dependency Features “pd(https,//t)“.

Diese beiden Diagramme zeigen deutlich, dass solche Urls, die die verlinkten Ressourcen über das sichere Hypertext-Protokoll *HTTPS* verschlüsselt übertragen, häufiger in neutralen Postings wiedergefunden werden können. Auch bei der manuellen Klassifikation konnte deren überwiegende Existenz in News-Tweets subjektiv festgestellt werden. Dieser Umstand kann auch durch das Streudiagramm des Typed Dpendency Features *pd(https,//t)* bestätigt werden. Dieses sagt aus, dass solche Abhängigkeiten ausschließlich

in neutralen Tweets zu finden sind, erkennbar durch den Cluster der im Diagramm rechts unten angesiedelt ist. Im Vergleich dazu existiert keine Instanz aus der Klasse der Hasspostings mit einem solchen Feature-Wert.

4.3.3 Linguistik und Lexikon Feature Evaluierung

In diesem Kapitel werden die Features aus der Kategorie *Linguistik* und deren Unterkategorie *Lexikalisch* evaluiert. Tabelle 4.8 liefert einen Blick auf die berechneten Inforamtion Gains und die nach diesen gereihten Attributen. Dabei ist ersichtlich, dass

Avg. IG	Avg. Rank	Attributename
0.222 +- 0.004	1 +- 0	lexDensityOfHatefulTerms
0.207 +- 0.004	2 +- 0	lexNumberOfHatefulTerms
0.028 +- 0.001	3 +- 0	lexNumberOfSadEmoticons
0.027 +- 0.001	4 +- 0	lingNumberOfURLs
0.024 +- 0.001	5 +- 0	lingNumberOfNonAlphaCharInMiddleOfWord
0.02 +- 0.001	6.4 +- 0.49	lingNumberOfPunctuation
0.02 +- 0.003	7 +- 1.1	lingNumberOfSpecialPunctuation
0.019 +- 0.001	7.8 +- 0.4	lingAvgSentenceLength
0.016 +- 0.001	9.3 +- 0.9	lexNumberOfSecondPersonPronouns
0.015 +- 0.001	10.4 +- 0.66	lingNumberOfSentences
0.015 +- 0.001	10.5 +- 1.12	lingNumberOfCapitalizedLetters
0.013 +- 0.001	11.6 +- 0.8	lingLengthInTokens
0.009 +- 0.001	13 +- 0	lexNumberOfDemonstrativPronouns
0.004 +- 0.001	15.4 +- 0.92	lexNumberOfCheekyEmoticons
0.004 +- 0.001	15.4 +- 0.92	lingNumberOfOneLetterTokens
0.004 +- 0	16 +- 0.89	lexNumberOfFirstPersonPronouns
0 +- 0	18.8 +- 0.75	lingAvgLengthOfWord
0 +- 0	19.4 +- 0.8	lexNumberOfAngryEmoticons
0.003 +- 0.003	19.6 +- 6.86	lingNumberOfCharacters
0 +- 0.001	20.6 +- 2.11	lexNumberOfHappyEmoticons
0 +- 0	21.2 +- 0.75	lexNumberOfModalVerbs
0 +- 0	21.7 +- 1	lexNumberOfDiscourseConnectives
0 +- 0.001	22.8 +- 2.64	lexNumberOfAmazedEmoticons
0 +- 0.001	23.6 +- 2.65	lexNumberOfHatefulTermsInApostrophe
0 +- 0	23.8 +- 5.15	lexNumberOfThirdPersonPronouns
0 +- 0	24.1 +- 0.7	lexNumberOfInterrogativPronouns
0 +- 0	26.2 +- 0.75	lexNumberOfDiscourseParticels
0 +- 0	26.4 +- 0.49	lexNumberOfIndefinitePronouns

Tabelle 4.8: Information Gain Ranking der Linguistik- und Lexikon-Features

die lexikalischen Features *DensityOfHatefulTerms* und *NumberOfHatefulTerms* den meisten Informationsgewinn liefern. Die Dichte an hasserfüllten Wörtern im Tweet, realisiert durch *DensityOfHatefulTerms*, ermöglicht es sogar noch besser als *NumberOfHatefulTerms* Aussagen über die Klasse zu treffen, da der Wert im Bezug auf die Tweetlänge normalisiert wird. Abgeschlagen auf Rang 3 liegt das *NumberOfSadEmoticons*-Feature. Generell kann daraus gefolgert werden, dass der überwiegende Teil an Features bei denen Wörterbücher verwendet wurden, kaum für die Korrektheit der Klassifikation beitragen und den linguistischen Features etwas unterlegen sind. So tragen im Vergleich die Features *lexNumberOfIndefinitePronouns*, *lexNumberOfDiscourseParticels*, *lexNumberOfInterro-*

gativPronouns, *lexNumberOfThirdPersonPronouns*, *lexNumberOfDiscourseConnectives*, *lexNumberOfModalVerbs*, *lexNumberOfAngryEmoticons* auf der Seite der lexikalischen Attribute und nur *lingAvgLengthOfWord* auf der Seite der linguistischen Attribute nicht dazu bei, die Postings in die korrekte Klasse einzuordnen. Um den Unterschied zwischen einem aussagekräftigen Feature im Bezug auf die Hassposting-Klasse und einem von den Klassen unabhängigen Feature, bei dem die Werte bezüglich der Klassen gleichmäßig verteilt sind, sollen die Visualisierungen der Verteilungen der Featurewerte von *NumberOfHatefulTerms* in Abbildung 4.6 und *NumberOfIndefinitePronouns* in Abbildung 4.7 im Kontext der Klassen zeigen.

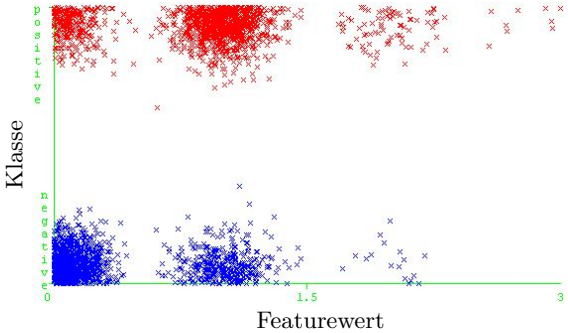


Abbildung 4.6: Streudiagramme des NumberOfHatefulTerms-Features.

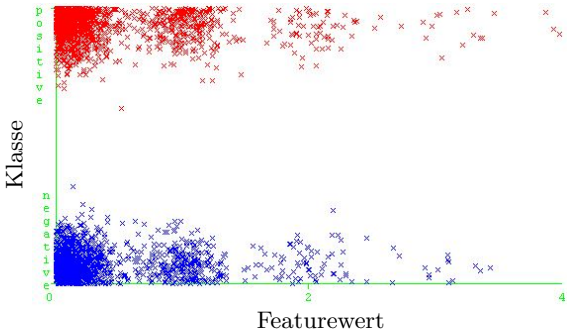


Abbildung 4.7: Streudiagramme des NumberOfIndefinitePronouns-Features.

Abbildung 4.6 zeigt klar, dass Objekte mit hasserfüllten Wörtern viel häufiger der Hassposting-Klasse zugeordnet wurden. Objekte die kein einziges Hasswort enthalten entstammen wiederum großteils der neutralen Posting-Klasse. Eine solche Abhängigkeit lässt sich bei Abbildung 4.7 mit freiem Auge nicht feststellen. Die Lücken zwischen den Clustern entlang der X-Achse ergeben sich lediglich, da diese beiden Features nur diskrete Werte annehmen können. In Abbildung 4.6 ist zusätzlich die Diversität der Featurewerte geringer, weshalb eine dadurch entstehende granularere Skalierung diese auffälligen Lücken entstehen lässt. Zuletzt soll durch die Tabelle 4.9 die Rangordnung der Attribute, die sich in dieser aufgrund der χ^2-Werte der Features ergibt, wie gewohnt bestätigt werden. Neben der fast identen Reihenfolge fällt als einziger Unterschied die vertauschten Features *NumberOfOneLetterToken* und *NumberOfCheekyEmoticons* im Hinblick auf die IG-Reihung auf.

Als Abschluss dieses Unterkapitels wird noch das am höchsten gelistete, aus zwei anderen linguistischen Features berechnete Attribut, nämlich *lingAvgSentenceLength* analysiert. Dies geschieht indem die Featurewerte(FW) von *lingNumberOfSentences* in der X-Achse und von *lingLengthInTokens* in der Y-Achse im Streudiagramm 4.8 gegenübergestellt werden. Zu erkennen ist, dass Hassposting Instanzen bei Tweets ohne jeglichem Satzzeichen, wodurch die Anzahl der Sätze im Beitrag null beträgt, und bei geringer Wortanzahl (Featurewert 1-15) überwiegen. Dabei handelt es sich oft um Beiträge, die lediglich aus einzelnen Schimpfwörtern oder kurzen Beleidigungen gebildet werden. Das Histogramm

Avg. χ^2-Merit	Avg. Rank	Attributename
720.688 +-12.86	1 +- 0	lexDensityOfHatefulTerms
683.221 +-11.856	2 +- 0	lexNumberOfHatefulTerms
91.586 +- 3.008	3.1 +- 0.3	lexNumberOfSadEmoticons
88.915 +- 2.878	3.9 +- 0.3	lingNumberOfURLs
80.037 +- 4.118	5.1 +- 0.3	lingNumberOfNonAlphaCharInMiddleOfWord
69.661 +- 3.338	6.4 +- 0.49	lingNumberOfPunctuation
66.798 +- 8.482	6.9 +- 1.22	lingNumberOfSpecialPunctuation
64.602 +- 3.604	7.8 +- 0.4	lingAvgSentenceLength
56.216 +- 2.56	9.2 +- 0.87	lexNumberOfSecondPersonPronouns
52.942 +- 4.197	10.1 +- 0.7	lingNumberOfSentences
50.35 +- 3.963	10.6 +- 0.8	lingNumberOfCapitalizedLetters
38.271 +- 3.99	11.9 +- 0.3	lingLengthInTokens
31.241 +- 3.696	13 +- 0	lexNumberOfDemonstrativPronouns
14.755 +- 2.147	15.3 +- 1	lingNumberOfOneLetterTokens
14.075 +- 1.708	15.4 +- 0.92	lexNumberOfCheekyEmoticons
12.76 +- 1.548	16.1 +- 0.7	lexNumberOfFirstPersonPronouns
0 +- 0	18.8 +- 0.75	lingAvgLengthOfWord
0 +- 0	19.4 +- 0.8	lexNumberOfAngryEmoticons
11.637 +- 9.541	19.6 +- 6.86	lingNumberOfCharacters
0.915 +- 2.745	20.6 +- 2.11	lexNumberOfHappyEmoticons
0 +- 0	21.2 +- 0.75	lexNumberOfModalVerbs
0 +- 0	21.7 +- 1	lexNumberOfDiscourseConnectives
1.219 +- 2.445	22.8 +- 2.64	lexNumberOfAmazedEmoticons
0.686 +- 2.058	23.6 +- 2.65	lexNumberOfHatefulTermsInApostrophe
0 +- 0	23.8 +- 5.15	lexNumberOfThirdPersonPronouns
0 +- 0	24.1 +- 0.7	lexNumberOfInterrogativPronouns
0 +- 0	26.2 +- 0.75	lexNumberOfDiscourseParticels
0 +- 0	26.4 +- 0.49	lexNumberOfIndefinitePronouns

Tabelle 4.9: Ranking der Linguistik- und Lexikon-Features nach deren χ^2-Wert.

4.9 des Features *lingAvgSentenceLength* bestätigt diese Annahme, betrachtet man den ersten Balken davon.

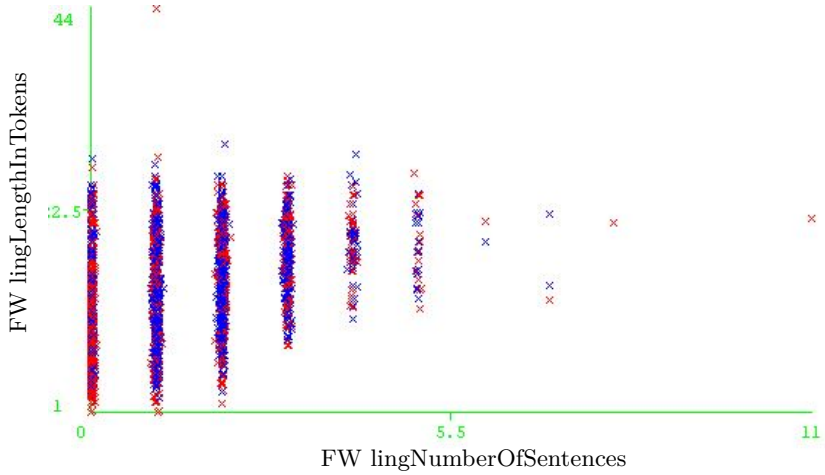


Abbildung 4.8: Streudiagramm der linguistischen Features *NumberOfSentences* und *LengthInTokens*.

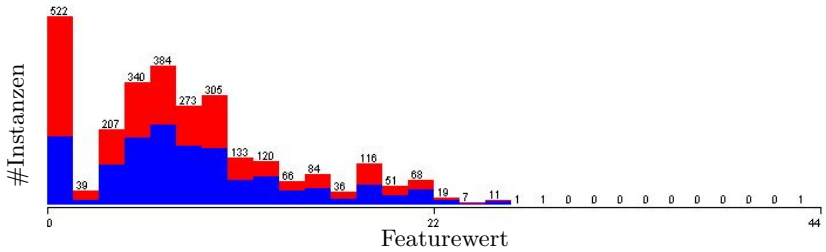


Abbildung 4.9: Histogramm des *AvgSentenceLength*-Features.

4.3.4 LIWC-Feature Evaluierung

In diesem Kapitel sollen die LIWC-Features, die sich aus den Kategorien des LIWC 2001 und selbst hinzugefügten Kategorien, die speziell für Wörter entworfen wurden, die direkt Hass verbreiten (z.B. Dreckspack) oder mit ihm in Verbindung gebracht werden können (z.B. verbrennen), zusammensetzen. Tabelle 4.10 zeigt die Rangliste nach Information Gain, Tabelle 4.11 die Features nach dem χ^2-Wert gereiht.

Aufgrund der hohen Anzahl an LIWC-Features werden in den Tabellen nur solche aufgelistet, die einen IG oder χ^2-Wert größer null aufweisen. Bei dem Vergleich der Tabellen fällt auf, dass die Kategorie *Nationality* im Ranking nach χ^2-Wert drei Plätze verliert und auch die Kategorien *Other_reference* und *Time* die Positionen tauschen. Weiters schafft es die LIWC-Kategorie *Sad* nicht einmal in die Auflistung der χ^2-Wert-Tabelle. Alle anderen Features verharren auf den selben Rängen. In den Abbildungen 4.10 und 4.11 werden die Kategorien *Swear* (dt. Schimpfwort) und *Ancestry* (dt. Herkunft/Abstammung) visualisiert, da sich hier gut die Abhängigkeit ihrer Werte von der Hassposting-Klasse erkennen lässt.

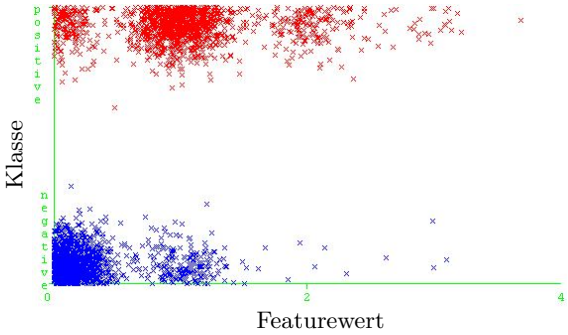


Abbildung 4.10: Streudiagramm der LIWC Swear Kategorie.

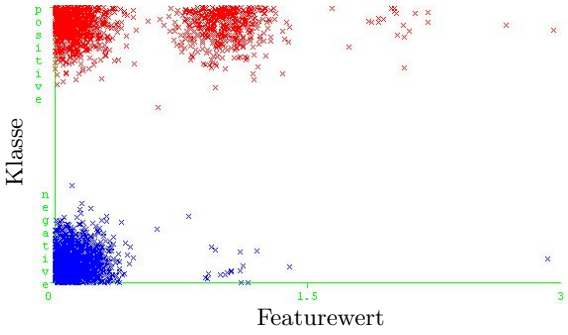


Abbildung 4.11: Streudiagramm der LIWC Ancestry Kategorie.

Das Attribut, das mit Abstand den größten Beitrag für die korrekte Klassifikation liefert, ist das *liwc_Swear* Feature. In diese LIWC Kategorie fallen alle Schimpfwörter, die sich einerseits im Standard LIWC 2001 Wörterbuch befinden und umfasst zusätzlich die Nomen aus dem Hasswortlexikon, wodurch es sich vom ähnlichen lexikalischen Attribut

Avg. IG	Avg. Rank	Attributename
0.418 +- 0.007	1 +- 0	liwc_Swear
0.155 +- 0.004	2 +- 0	liwc_Ancestry
0.084 +- 0.003	3 +- 0	liwc_Skin_Colour
0.049 +- 0.002	4 +- 0	liwc_Race
0.039 +- 0.002	5.1 +- 0.3	liwc_Sex
0.036 +- 0.002	5.9 +- 0.3	liwc_National_Origin
0.029 +- 0.002	7 +- 0	liwc_Family
0.017 +- 0.002	8.4 +- 0.66	liwc_Anxiety
0.016 +- 0.001	9.3 +- 0.9	liwc_Nationality
0.015 +- 0.001	10.5 +- 0.92	liwc_Relig
0.015 +- 0.001	10.8 +- 0.98	liwc_Time
0.015 +- 0.002	11.3 +- 1.68	liwc_Other_reference
0.012 +- 0.001	13.1 +- 0.83	liwc_Humans
0.012 +- 0.001	13.6 +- 0.49	liwc_Gender
0.009 +- 0	15.4 +- 0.49	liwc_Ethnic_Background
0.007 +- 0.001	16.3 +- 1.1	liwc_Space
0.007 +- 0.001	17.4 +- 1.11	liwc_Tentative
0.007 +- 0.001	17.5 +- 1.36	liwc_Social
0.005 +- 0.001	19.2 +- 1.78	liwc_Leisure
0.005 +- 0	20.4 +- 1.5	liwc_Sports
0.004 +- 0.001	22.2 +- 1.72	liwc_Affect
0.004 +- 0.001	23.1 +- 2.26	liwc_Disability
0.004 +- 0	24 +- 1.55	liwc_School
0.003 +- 0.001	25.1 +- 5.11	liwc_Money
0.003 +- 0.001	25.4 +- 3.93	liwc_Death
0.003 +- 0.001	26.1 +- 4.44	liwc_Certain
0.002 +- 0.002	31.2 +- 6.55	liwc_Job
0.002 +- 0.001	32.8 +- 5.69	liwc_Sad
0.002 +- 0.002	33.2 +-11.34	liwc_Occup

Tabelle 4.10: Information Gain Ranking der LIWC-Features.

NumberOfHatefulTerms, welches zusätzlich zu Schimpfwörtern jegliche andere Art von hasserfüllten Wörtern zählt, unterscheidet und dadurch einen Mehrwert liefern soll.

4.3.5 Mistake-Feature Evaluierung

Bei der Evaluierung des *Mistake-Features* ergeben sich ein Information Gain von 0.016 +- 0.001 und ein χ^2-Wert von 51.681 +- 4.575. Im Vergleich zu der Linguistik-Feature Evaluierung würde das Mistake-Merkmal im guten Mittelfeld liegen. Die Diagramme 4.12 und 4.13 zeigen die Verteilungen der Objekte anhand der Anzahl an Fehlern im Kontext der Klassen.

Avg. χ^2-Merit	Avg. Rank	Attributename
1298.494 +-18.637	1 +- 0	liwc_Swear
452.477 +- 9.074	2 +- 0	liwc_Ancestry
233.051 +- 7.617	3 +- 0	liwc_Skin_Colour
161.48 +- 5.717	4 +- 0	liwc_Race
128.113 +- 5.766	5 +- 0	liwc_Sex
102.103 +- 4.208	6.3 +- 0.46	liwc_National_Origin
96.53 +- 6.591	6.7 +- 0.46	liwc_Family
56.254 +- 4.586	8.4 +- 0.49	liwc_Anxiety
51.074 +- 3.33	9.5 +- 1.12	liwc_Relig
51.315 +- 5.511	10 +- 1.26	liwc_Other_reference
49.093 +- 4.082	10.4 +- 0.92	liwc_Time
44.121 +- 2.024	12 +- 0.63	liwc_Nationality
41.167 +- 3.5	12.7 +- 0.64	liwc_Humans
30.056 +- 1.324	14.5 +- 0.67	liwc_Gender
27.085 +- 1.385	15.5 +- 1.12	liwc_Ethnic_Background
25.939 +- 4.721	16 +- 1.34	liwc_Space
22.919 +- 3.693	17.2 +- 1.25	liwc_Tentative
23.804 +- 3.727	17.4 +- 1.5	liwc_Social
18.449 +- 3.174	19.2 +- 1.78	liwc_Leisure
16.435 +- 1.495	20.4 +- 1.5	liwc_Sports
14.892 +- 2.32	21.9 +- 1.64	liwc_Affect
13.224 +- 1.642	23.3 +- 2.15	liwc_Disability
12.628 +- 1.019	23.9 +- 1.64	liwc_School
11.422 +- 4.2	25 +- 5.16	liwc_Money
11.21 +- 4.079	25.4 +- 3.93	liwc_Death
10.806 +- 4.091	26.4 +- 4.25	liwc_Certain
6.878 +- 5.832	31.2 +- 6.57	liwc_Job
7.472 +- 6.246	33.2 +-11.31	liwc_Occup

Tabelle 4.11: Ranking der LIWC-Features nach deren χ^2-Wert.

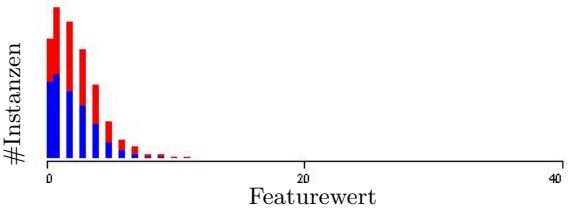


Abbildung 4.12: Histogramm des Mistake Features.

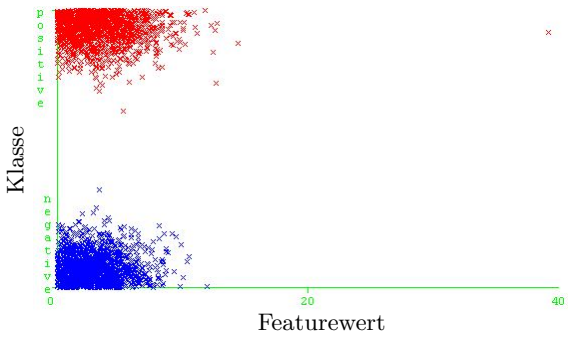


Abbildung 4.13: Streudiagramm des Mistake Features.

Grundgedanke für die Einführung dieses Features war die Annahme, dass im Affekt und Zorn verfasste Beiträge eine größere Anzahl an Fehlern enthalten. Durch das Histogramm wird diese Annahme nur bedingt bestätigt. Zwar überwiegt die Menge an neutralen Postings bei einem Fehlerwert von 0-1 leicht gegenüber den Hasspostings und im Gegenzug

verhält es sich auch genau umgekehrt bei höherem Fehlerwert. Hier ist der Anteil an Hasspostings höher. Bei gesamtheitlicher Betrachtung des Histogramms und des Streudiagramms wird jedoch der Eindruck gewonnen, dass die Verteilung ausgeglichen ist und das Auftreten von Fehlern in beiden Klassen in gleichem Ausmaß möglich ist. Würde man wiederum den Wert des IG mit allen anderen Features in Kontext setzen, wäre dieser dennoch im ersten Drittel der Rangliste wiederzufinden, wodurch gesagt werden kann, dass auch dieses Feature passabel zur Klassifizierung beiträgt, den anfänglichen Erwartungen aber nicht gerecht wird.

4.3.6 Twitter-Netzwerk-Feature Evaluierung

Abbildung 4.12 zeigt den berechneten Information Gain und die nach diesem gereihten Features. Zur Verifikation wird auch der χ^2-Wert je Netzwerk-Attribut berechnet und in Abbildung 4.13 in Form einer Rangliste abgebildet.

Avg. IG	Avg. Rank	Attributename
0.15 +- 0.003	1 +- 0.0	numberOfHateposts
0.037 +- 0.002	2.2 +- 0.4	followerSiteFeature
0.036 +- 0.002	2.8 +- 0.4	followerUserFeature
0.021 +- 0.003	4.4 +- 0.49	listedCount
0.018 +- 0.002	4.7 +- 0.64	numberOfTweets
0.014 +- 0.001	4.9 +- 0.3	retweetCount
0.01 +- 0.001	5.9 +- 0.3	isReply
0.007 +- 0.001	7 +- 0	numberOfMentionedUser
0.007 +- 0.001	8.6 +- 0.66	numberOfFollower
0.006 +- 0.001	9.7 +- 0.78	isRetweet
0.004 +- 0.002	11.1 +- 0.54	lengthOfName
0.005 +- 0.003	12.5 +- 1.28	numberOfFriends
0 +- 0	12.8 +- 0.4	favouriteCount
0 +- 0	14 +- 0.63	lengthOfUsername
0 +- 0.001	14.5 +- 0.92	numberOfHashtags
0 +- 0	16 +- 0	numberOfWordsInName

Tabelle 4.12: Information Gain Ranking der Network-Features.

Die Top 5 bilden dabei das aus mehreren Tweets aggregierte Feature *numberOfHateposts*, die Follower-Features *FollowerSite-* und *FollowerUser-Feature* auf dem geteilten zweiten Rang, die gewisse Eigenheiten von Verfassern von Hasspostings zum Vorschein bringen, gefolgt von den userbezogenen Network Features *ListedCount* und *NumberOfTweets*, die jedoch eher für die Klasse der neutralen Postings Informationsgewinn erzeugen. Diese Tatsachen sollen mit den Diagrammen 4.14 4.15 4.16 4.17 über die Verteilung der Instanzen im Bezug auf ihren jeweiligen Feature-Wert (X-Achse) und Klasse (Y-Achse) unterstrichen werden. Weiters ist ersichtlich, dass die letzten vier Features der Rangliste überhaupt keinen Informationsgewinn liefern und daher mögliche Kandidaten für eine Nichtberücksichtigung im Klassifikationsprozess sind.

Avg. χ^2-Merit	Avg. Rank	Attributename
452.477 +- 9.33	1 +- 0	numberOfHateposts
126.908 +- 4.172	2.2 +- 0.4	followerSiteFeature
121.95 +- 6.025	2.8 +- 0.4	followerUserFeature
70.116 +- 8.829	4.1 +- 0.3	listedCount
53.106 +- 5.037	5.2 +- 0.6	numberOfTweets
46.305 +- 3.863	5.7 +- 0.46	retweetCount
34.28 +- 3.813	7 +- 0	isReply
25.038 +- 2.424	8.5 +- 0.67	numberOfMentionedUser
23.36 +- 2.871	9.3 +- 1	numberOfFollower
21.066 +- 3.959	9.5 +- 0.81	isRetweet
14.18 +- 5.257	11.3 +- 1.19	lengthOfName
18.198 +- 9.287	12.6 +- 1.5	numberOfFriends
0 +- 0	12.8 +- 0.4	favouriteCount
0 +- 0	13.7 +- 1.42	lengthOfUsername
0.641 +- 1.923	14.3 +- 0.78	numberOfHashtags
0 +- 0	16 +- 0	numberOfWordsInName

Tabelle 4.13: Ranking der Netzwerk Features nach deren χ^2-Wert.

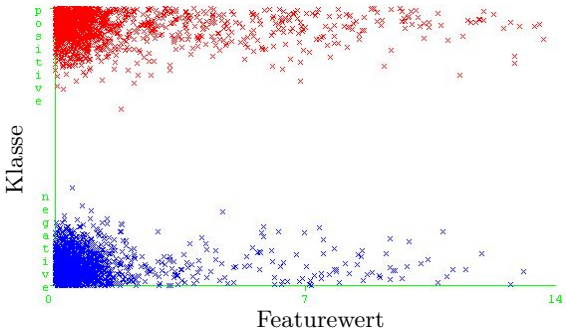


Abbildung 4.14: Visualisierung des FollowerSite- Features.

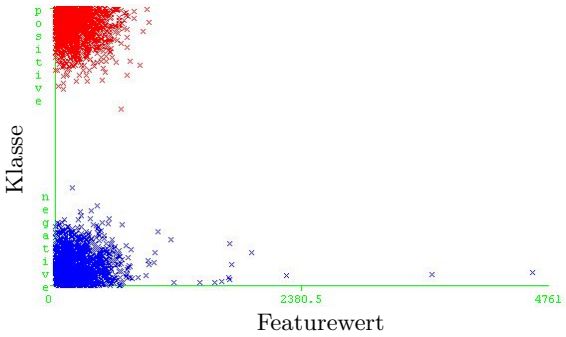


Abbildung 4.15: Visualisierung des ListedCount Features.

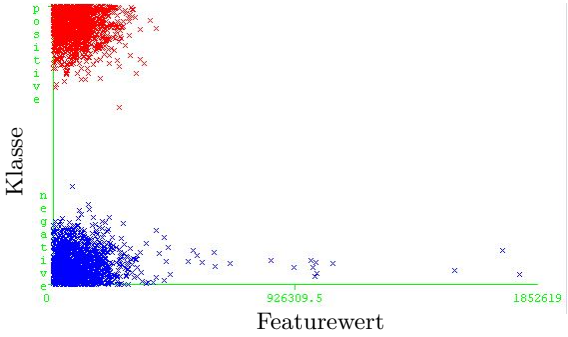


Abbildung 4.16: Visualisierung des NumberOfTweets Features.

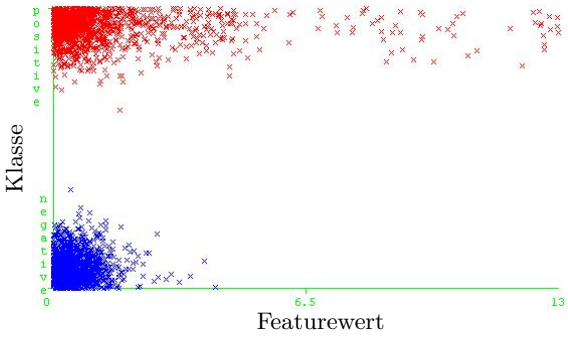


Abbildung 4.17: Visualisierung des NumberOfHateposts Features.

4.3.7 Comment Embedding und Character-N-Gram Evaluierung

Die Tabellen 4.14 4.15 und 4.16 4.17 listen den Information Gain und χ^2-Wert der Top 15 Character 4-Grams beziehungsweise Top 10 der Comment Embeddings auf. Auf die detaillierte Untersuchung wird aufgrund der fehlenden Aussagekraft dieser Features verzichtet, wodurch die Tabellen lediglich dazu dienen die Werte mit anderen Features in Kontext zu stellen. Aus den Namen der Character-Features lassen sich jedoch die üblichen Hasswörter, die schon in der Tabelle der Top-BoW Features ganz oben zu finden waren, erahnen.

Avg. IG	Avg. Rank	Attr.Name
0.074 +- 0.003	1 +- 0	n e g e
0.068 +- 0.003	2 +- 0	e g e r
0.04 +- 0.002	3.7 +- 0.64	h u r e
0.04 +- 0.002	4.1 +- 0.7	s o h n
0.038 +- 0.002	4.8 +- 1.47	g e s i
0.038 +- 0.002	5.5 +- 0.67	n s o h
0.034 +- 0.002	7.4 +- 0.8	u s e l
0.033 +- 0.001	8.3 +- 0.78	u r e n
0.033 +- 0.002	8.4 +- 0.92	m u s e
0.031 +- 0.001	10.3 +- 1.19	r e n s
0.029 +- 0.001	11.1 +- 0.7	t t p s
0.028 +- 0.001	12.6 +- 0.92	h t t p
0.028 +- 0.002	12.9 +- 2.12	n d e l
0.027 +- 0.002	14.8 +- 1.6	e s i n
0.027 +- 0.001	15.1 +- 1.87	n n e g

Tabelle 4.14: Information Gain Ranking der Character-Quad-Gram-Features.

Avg. χ^2-Merit	Avg. Rank	Attr.Name
211.894 +- 8.156	1 +- 0	n e g e
198.647 +- 7.82	2 +- 0	e g e r
110.956 +- 3.929	3.6 +- 0.66	s o h n
109.473 +- 3.559	4.3 +- 0.64	h u r e
110.008 +- 4.696	4.5 +- 1.36	g e s i
105.316 +- 3.374	5.7 +- 0.64	n s o h
98.241 +- 1.874	7.5 +- 0.92	t t p s
96.94 +- 2.378	8.2 +- 1.33	u r e n
95.235 +- 2.442	9.3 +- 1.27	h t t p
92.924 +- 3.972	10.6 +- 1.5	u s e l
92.168 +- 4.041	11.3 +- 1.68	m u s e
89.543 +- 4.132	12.7 +- 2.65	r e n s
87.545 +- 2.393	13 +- 1.73	t p s t
86.481 +- 4.204	14.2 +- 2.18	n d e l
84.975 +- 4.849	15 +- 2.24	e s i n

Tabelle 4.15: Ranking der Character-Quad-Gram Features nach deren χ^2-Wert.

Avg. IG	Avg. Rank	Attr.Name
0.01+- 0.001	1.3 +- 0.46	Attr_50
0.009 +- 0.001	1.7 +- 0.46	Attr_25
0.001 +- 0.002	6.7 +- 3.03	Attr_33
0.002 +- 0.003	6.9 +- 2.26	Attr_31
0.006 +- 0.002	10.3 +-19.27	Attr_94
0.001 +- 0.003	22.6 +- 8.8	Attr_23
0.001 +- 0.002	32.9 +- 9.31	Attr_1
0.004 +- 0.003	38.5 +-42.71	Attr_62
0.001 +- 0.003	71.7 +-33.13	Attr_61
0.001 +- 0.002	89.2 +-28.74	Attr_66

Tabelle 4.16: Information Gain Ranking der Comment Embedding-Features.

Avg. χ^2-Merit	Avg. Rank	Attr.Name
35.469 +- 4.252	1.3 +- 0.46	Attr_50
31.865 +- 2.598	1.7 +- 0.46	Attr_25
2.598 +- 7.794	6.7 +- 3.03	Attr_33
6.754 +-10.321	7 +- 2.14	Attr_31
21.842 +- 7.511	10.1 +-19.32	Attr_94
4.656 +- 9.313	22.6 +- 8.8	Attr_23
2.227 +- 6.681	32.9 +- 9.31	Attr_1
14.572 +-11.927	38.5 +-42.71	Attr_62
4.494 +- 8.989	71.8 +-32.94	Attr_61
2.79 +- 8.37	89.2 +-28.74	Attr_66

Tabelle 4.17: Ranking der Comment Embedding Features nach deren χ^2-Wert.

4.3.8 CFS Subset Evaluation

Da bis hier hin ausgewählte Features nur einzeln und ohne deren Abhängigkeiten untereinander evaluiert worden sind, sollen nun mit Hilfe der CFS Subset Evaluation 4.2.1 Implementation von Weka solche Korrelationen innerhalb der Attribute in die Bewertung einbezogen werden. Für die Bildung eines optimalen Subsets standen exklusive der BoW-Unigrams und Typed Dependencies nur die Features zur Verfügung, welche auch detailiert einzeln untersucht wurden (LIWC, Linguistik, Lexikon, Mistake, Twitter-Netzwerk). Tabelle 4.18 zeigt die Liste jener Attribute, die für ein solches Subset

Number of Selections(%)		Attributename
1	(10 %)	mistakes
1	(10 %)	liwc_Certain
5	(50 %)	liwc_Sports
3	(30 %)	liwc_School
2	(20 %)	liwc_Ancestry
9	(90 %)	liwc_Time
4	(40 %)	liwc_Other_reference
2	(20 %)	liwc_Money
1	(10 %)	liwc_Numbers
10	(100 %)	liwc_Swear
1	(10 %)	liwc_Leisure
8	(80 %)	liwc_Tentative
3	(30 %)	liwc_Anxiety
8	(80 %)	liwc_Sex
1	(10 %)	liwc_TV
8	(80 %)	liwc_Skin_Colour
2	(20 %)	liwc_Eat
1	(10 %)	liwc_Positive_Emotion
3	(30 %)	liwc_Sad
10	(100 %)	liwc_Gender
9	(90 %)	retweetCount
10	(100 %)	numberOfMentionedUser
2	(20 %)	listedCount
4	(40 %)	numberOfTweets
2	(20 %)	followerSiteFeature
1	(10 %)	followerUserFeature
10	(100 %)	numberOfHateposts
1	(10 %)	lingLengthInTokens
2	(20 %)	lingAvgSentenceLength
1	(10 %)	lingNumberOfSpecialPunctuation
1	(10 %)	lingNumberOfOneLetterTokens
1	(10 %)	lexNumberOfSecondPersonPronouns
9	(90 %)	lexNumberOfDemonstrativPronouns
5	(50 %)	lexNumberOfSadEmoticons

Tabelle 4.18: Attribut-Subset basierend auf der Correlaten-based Feature Selection(CFS)

während einer 10-fachen Kreuz-Validierung zumindest einmal ausgewählt wurden. Desto öfter diese Bestandteil sind, desto größer ist deren Wichtigkeit innerhalb des Subsets und in weiterer Folge für den Klassifikationsprozess einzuschätzen. Als Suchmethode wurde der *Greedy-Stepwise* Algorithmus gewählt. Bei der Betrachtung der Tabelle fällt

auf, dass aus der Kategorie der Twitter-Netzwerk Features alle Attribute der Top 6 der Einzelevaluierung in das CFS Subset aufgenommen wurden. Anders sieht dies bei den LIWC-Features aus, bei denen zwar ebenfalls überwiegend jene Attribute Teil des Subsets sind, welche in der oberen Hälfte der LIWC-Feature Rangliste angesiedelt sind, jedoch auch einige übernommen wurden, welche weder IG noch einen χ^2-Wert aufweisen. Grund dafür ist, dass die Kategorien des LIWC selbst ebenfalls nochmals gruppiert sind, also eine Oberkategorie besitzen und deshalb voneinander abhängig sind, wodurch der CFS Wert natürlich verschlechtert wird und es deshalb nicht für die Auswahl in das Subset reicht. Ähnlich der Fall bei den Linguistik Features, bei denen es nicht einmal die Top 2 aus der Rangliste nach IG in die Auswahl schafften. Hierbei ist jedoch offensichtlich, dass *DensityOfHatefulTerms* von *NumberOfHatefulTerms* abhängig ist, welches wiederum dem Feature *LIWC_Swear* sehr ähnlich ist und dieses bereits einen Platz im Subset einnimmt. Da viele Features innerhalb der Linguistik und Lexikon Kategorie ebenfalls ähnlich oder sogar mittels eines anderen linguistischen Feature berechnet werden, ist auch hier die Inter-Korrelation hoch, wodurch die im berechneten Subset enthaltenen Linguistik-Attribute aus den verschiedensten Bereichen ihrer Ranglisten aus IG und χ^2-Wert entstammen.

4.4 Feature-Set und Classifier Evaluierung

4.4.1 Initiales Parameter Setting der Classifier

Die nachfolgenden Experimente wurden mit den in diesem Kapitel angeführten Parametereinstellungen des jeweiligen Classifiers durchgeführt. Auf Grundlage der Feature-Sets, aus denen die besten Ergebnisse resultieren, wurde umfangreiches Parametertuning betrieben.

Support Vector Machine(SVM) - SMO
Als *Kalibrator* wurde die Logistic Regression Funktion verwendet, für den Komplexitätsparameter *C* 1.0 gewählt und *Epsilon* auf 1.0 E-12 gesetzt. Bei der Kernelwahl wurde auf den in Natural Language Processing Bereichen gut funktionierenden *PolyKernel* gesetzt.

Naive Bayes(NB)
Wie bereits in Abschnitt 3.11.2 erwähnt sorgt der Einsatz von Kerndichteschätzern bei nummerischen Attributen, deren Werte oft nicht normalverteilt sind, für eine bessere Performance. Daher wurde im Zuge der Arbeit in Weka diese Option *(useKernelEstimator)* genutzt und anders als in der Standardeinstellung aktiviert. Einzelne Tests ergaben dadurch eine oftmals enorme Verbesserung (z.B. Message N-Grams) und es kam nur in den seltensten Fällen zu minimalen Verschlechterungen (Character 4-Gram und Twitter User Feature). Nachteil ist jedoch die damit verbundene längere Evaluationszeit.

Random Decision Forest(RDF)
Die Anzahl der zu erstellenden Bäume, die jeweils eine Stimme pro Tweet im Klassifikationsprozess abgeben dürfen, wurde auf 100 gesetzt und die Anzahl an zufällig ausgewählten Attributen für das Feature-Bagging der Bäume beträgt log_2 von der Gesamtmenge an

Features. Die maximale Baumtiefe ist unbegrenzt. Durch Weka wurde die minimale Anzahl an Instanzen in den Blättern auf 1 und der minimale IG der Features in den Knoten, nach denen die Instanzen aufgeteilt werden, auf 0,001 gesetzt.

4.4.2 Signifikanztests

Zur Überprüfung der Aussagekraft der präsentierten Ergebnisse im Zuge der Feature-Set Evaluierung wurden Signifikanztests durchgeführt, bei denen als Nullhypothese angenommen wird, dass kein signifikanter Unterschied zwischen den zu vergleichenden Messwerten besteht. Mit Hilfe eines geeigneten Verfahrens ist zu ermitteln, ob die Nullhypothese zutrifft oder sie verworfen wird und die Resultate daher signifikant unterschiedlich sind. Im Kontext dieser Arbeit wurden *Paired T-Tests*, bei einem Signifikanzniveau von 0,05 durchgeführt. Als Vergleichsbasis wurde der gemittelte und daher Klassen-unabhängige F-Messwert herangezogen. Die Ergebnisse der Signifikanztests werden in Tabellen gegenübergestellt, wobei diese immer dem selben Schema entsprechen. Der linke Teil der Tabelle beinhaltet den Namen der Feature-Kategorie beziehungsweise -Set, dessen Klassifikationsergebnis jeweils mit den Resultaten, die durch jene Feature-Sets und Classifier, welche im oberen Teil abgebildet sind, gewonnen werden, verglichen wird. Das Sternzeichen **“*“** markiert ein signifikant schlechteres F-Maß, der Buchstabe **“v“** ein signifikant besseres.

4.4.3 Performancetests

Als weitere Kennzahlen für die Evaluierung sollen gemessene Laufzeiten der Feature-Extraktion und Modellbildung einschließlich Klassifizierung im Rahmen einer 10-fachen Kreuz-Validierung dienen. Die Messungen, implementiert im umgesetzten Machine-Learning Java-Programm unter Bezugnahme der aktuellen Systemzeit(Windows), erfolgen dabei mehrfach und unter möglichst ähnlichen Umständen, können jedoch trotzdem nur bedingt für wissenschaftliche Aussagen und Vergleiche herangezogen werden und sollen deshalb nur als zusätzliche Information dienen. Alle gemessenen Zeiten entsprechen Sekunden. Im Detail setzte sich die Versuchsumgebung für die Messung der Laufzeiten aus einem Samsung-Rechner (Modellnummer NP530U4C) mit Windows 10 64-Bit-Betriebssystem, einem Intel Core i7-3517U Prozessor und 8GB Arbeitsspeicher zusammen. Einzelheiten über den annotierte Datensatz, der im Zuge dessen für die Extraktion, Modellbildung und Evaluierung durch die 10-fache Kreuz-Validierung verwendet wurde, ist in Abschnitt 3.5.3 ersichtlich, besteht aber im Grunde aus insgesamt 2837 Tweets.

4.4.4 Evaluation von Featurekategorien

Nachdem die einzelnen Features im Bezug auf den Informationsgewinn untersucht worden sind, werden in diesem Kapitel die detaillierten Resultate der drei bereits vorgestellten Klassifikatoren Support Vektor Machine(SMO), Naive Bayes(NB) und Random Decision Forest(RDF) für die eingesetzten Feature-Kategorien präsentiert. Begonnen wird mit der Evaluierung der Features, die den Inhalt des Tweets modellieren und sich als mögliche Baseline für eine spätere Kombination mit zusätzlichen speziell ausgewählten

Attributen eignet. Konkret handelt es sich dabei um Message-, Typed Dependency- und Character N-Grams. Da bei dieser Art an Merkmalen eine große Menge an möglichen N-Gram-Kombinationen auftreten, wird jeweils eine logische Teilmenge davon untersucht, um die beste zu extrahieren. Für die Entscheidungsfindung werden zusätzlich Signifikanz- und Performancetests durchgeführt.

Baseline Evaluierung

Message N-Gram Evaluation
Die Resultate der Klassifikation werden unter Verwendung der Standard Klassifikationsmaße Precision, Recall und F-Maß bereitgestellt. Fett gedruckte Kennzahlen repräsentieren die besten Ergebnisse für das jeweilige Maß pro Klassifikator. Grau unterlegte Zeilen wiederum stellen das für den Classifier beste Feature N-Gram, im Bezug auf Performance, Signifikanz und vor allem dem Klassifikationsergebnis, dar, welches in späteren Feature-Kombinationen eingebunden wird. Tabelle 4.19 zeigt eine Übersicht der erzielten

Message N-Gram		**SMO**			**NB**			**RDF**		
		P	R	F	P	R	F	P	R	F
1-Gram	AW	0,861	0,86	0,86	0,802	0,802	0,801	0,88	0,87	0,869
	HW	0,885	0,876	0,875	**0,887**	**0,868**	**0,866**	0,889	0,88	0,879
1-2 Gram	AW	0,876	0,876	0,876	0,821	0,821	0,821	0,866	0,856	0,855
	HW	**0,894**	**0,889**	**0,888**	0,864	0,835	0,831	**0,899**	**0,892**	**0,891**
1-3 Gram	AW	0,875	0,875	0,875	0,814	0,813	0,813	0,865	0,853	0,851
	HW	0,892	0,887	0,887	0,857	0,823	0,817	0,897	0,89	0,89
1-4 Gram	AW	0,874	0,874	0,874	0,814	0,813	0,813	0,863	0,853	0,851
	HW	0,892	0,887	0,887	0,854	0,819	0,813	0,895	0,889	0,888
1-5 Gram	AW	0,874	0,874	0,874	0,814	0,814	0,814	0,864	0,853	0,852
	HW	0,891	0,886	0,885	0,853	0,818	0,812	0,897	0,891	0,89

Tabelle 4.19: Message N-Gram Klassifikationsresultate

Messwerte je Classifier für verschiedene Message N-Grams, welche entweder alle Wörter (AW) einbeziehen oder nur jene die den Kontext um Hasswörter (HW) abbilden. Das beste Resultat über die drei Classifier hinweg liefert der Random Decision Forest mit Hilfe von hasserfüllten Message Unigrams und Bigrams. Ähnlich gut klassifiziert der SMO-Classifier durch Einbezug der selben Features. Für spätere Experimente wurden für den SMO jedoch zusätzlich Trigrams für die Modellbildung genutzt, da Tabelle 4.21 erkennen lässt, dass 1-3 Grams im Gegensatz zu 1-2 Grams signifikant besser sind als die Unigrams aus Hasswörter (Unigram HW). Für den Naive Bayes sind auf Basis der gemessenen Werte die Hasswort-Unigrams am geeignetsten. Grund dafür könnte

zunehmendes Overfitting durch die stark erhöhte Anzahl an Features bei den restlichen getesteten Message N-Gram Feature-Kombinationen sein.

Message N-Gram	1-Gram HW			1-2 Gram HW			1-3 Gram HW			1-4 Gram HW			1-5 Gram HW		
	SMO	NB	RDF	SMO	NB	RDF	SMO	NB	RDF	SMO	NB	RDF	SMO	NB	RDF
1-Gram AW	-	*	-	*	-	*	*	-	*	*	-	*	*	-	*
1-2 Gram AW	-	*	*	-	-	*	-	-	*	-	-	*	-	-	*
1-3 Gram AW	-	*	*	-	-	*	-	-	*	-	-	*	-	-	*
1-4 Gram AW	-	*	*	-	-	*	-	-	*	-	-	*	-	-	*
1-5 Gram AW	-	*	*	-	-	*	-	-	*	-	-	*	-	-	*

Tabelle 4.20: Message N-Gram aus Hasswörter vs. allen Wörtern Signifikanztabelle

Message N-Gram	1-Gram HW			1-2 Gram HW			1-3 Gram HW			1-4 Gram HW			1-5 Gram HW		
	SMO	NB	RDF	SMO	NB	RDF	SMO	NB	RDF	SMO	NB	RDF	SMO	NB	RDF
1-Gram HW				-	v	-	*	v	-	*	v	-	-	v	-
1-2 Gram HW	-	*	-				-	v	-	-	v	-	-	v	-
1-3 Gram HW	v	*	-	-	*	-				-	-	-	-	-	-
1-4 Gram HW	v	*	-	-	*	-	-	-	-				-	-	-
1-5 Gram HW	-	*	-	-	*	-	-	-	-	-	-	-			

Tabelle 4.21: Message N-Gram aus Hasswörter Signifikanztabelle

Ob die Resultate unter Einbezug der jeweils beschriebenen Features nun signifikant besser oder schlechter sind und nicht nur auf dem Zufall basieren, wird durch die Tabellen 4.20 und 4.21 aufgelöst. Zuvor werden in Tabelle 4.20 die jeweiligen N-Grams aus allen Wörtern und die mit mindestens einem Hasswort gegenübergestellt. Es sticht sofort ins Auge, dass Message N-Grams aus allen Wörtern über alle Classifier hinweg entweder signifikant schlechter sind oder keine Signifikanz in eine Richtung aufweisen, wodurch sich in weiterer Folge nur mehr auf die Message N-Grams aus Hasswörtern konzentriert wird, welche in Tabelle 4.21 betrachtet werden. Die Analyse der darin abgebildeten Testergebnisse bestätigt beispielsweise, dass beim Einsatz von Naive Bayes die Klassifikation mithilfe der Hasswort-Unigram Features signifikant besser ist als jede andere untersuchte Feature-Kombination. Anders verhält es sich, wie bereits zuvor erwähnt, bei der Untersuchung der SMO-Resultate im Bezug auf Signifikanz. Hier können erst die HW-Message 1-3 Grams und 1-4 Grams einen statistisch signifikanten “Win“ gegenüber HW-Unigrams aufweisen. Dieser kommt zustande, da trotz minimal schlechteren Messwerten deren Varianz geringer ist. Durch die etwas schnellere Feature-Extraktionszeit und die geringere Anzahl an Features wurden letztendlich Hasswort Message 1-3 Grams gegenüber den 1-4 Grams für die Support Vector Machine Baseline bevorzugt.

Generell wird in den Diagrammen 4.18 und 4.19 die Zeit in Sekunden für die Extraktion der verschiedenen Features visualisiert. Bei Gegenüberstellung wird ersichtlich, dass, trotz der geringeren Anzahl an resultierenden Hasswort-Message N-Gram Features, deren Erhalt um ein Vielfaches länger dauert als bei vergleichbaren normalen Message N-Grams und die Zeit linear für jedes zusätzlich zu extrahierende N-Gram steigt. Ursache ist die

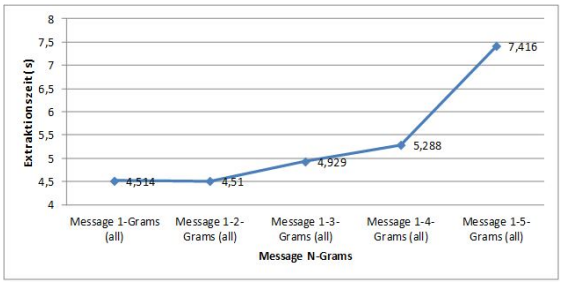


Abbildung 4.18: Extraktionszeit(s) für Message N-Grams aus allen Wörtern.

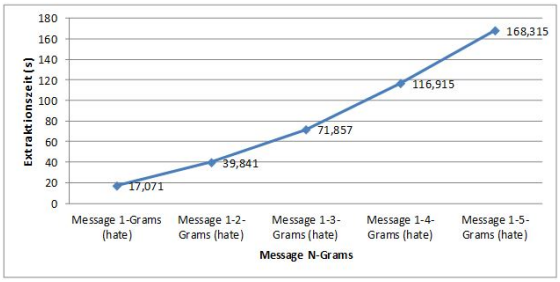


Abbildung 4.19: Extraktionszeit(s) für Message N-Grams mit mindestens einem Hasswort.

zusätzliche Untersuchung der Tweets auf Hasswörter, die mit dem Hasswort-Lexikon und einem Regex-Pattern durchgeführt wird, damit die zu vergleichenden Wörter nicht exakt übereinstimmen müssen. Bei Message N-Grams mit allen Wörtern ist erst der Anstieg der Feature-Kalkulationszeit von 1-4 Grams auf 1-5 Grams verhältnismäßig hoch, der Unterschied von rund 2 Sekunden aber zu vernachlässigen.

Typed Dependency N-Gram Evaluierung
Aufgrund ähnlicher Beschaffenheit der Typed Dependencies sollen auch für sie die bereits zuvor eingesetzten Konfigurationsmöglichkeiten angewandt werden, um somit die besten Messwerte und die damit verbundenen Features zu eruieren. Die gesammelten

Typed Dependency		**SMO**			**NB**			**RDF**		
		P	R	F	P	R	F	P	R	F
1-Gram	AW	0,752	0,735	0,729	0,695	0,695	0,695	0,728	0,712	0,704
	HW	**0,8**	0,693	0,659	**0,785**	0,668	0,624	**0,788**	0,689	0,655
1-2 Gram		0,752	**0,736**	**0,73**	0,695	0,695	0,695	0,737	**0,721**	**0,714**
1-3 Gram		0,752	**0,736**	**0,73**	0,703	**0,701**	**0,699**	0,737	0,719	0,712

Tabelle 4.22: Typed Dependency N-Gram Klassifikationsresultate

Resultate der verschiedenen Feature-Kombinationen und Classifier werden in Tabelle 4.22 zur Verfügung gestellt. Anders als bei Message N-Grams fallen beim Einsatz von lediglich Typed Dependencies, die ein Hasswort enthalten, die Resultate für jeden Classifier weit schlechter aus als bei der Berücksichtigung aller generierten Typed Dependencies. Dieser Umstand wird auch mithilfe durchgeführter Signifikanztests verdeutlicht, deren Ergebnisse in Tabelle 4.23 dargestellt sind. Auch der Einbezug von Typed Dependency Bi- und Trigrams trägt kaum zur Verbesserung der Ergebnisse bei, weshalb es im Fall der Klassifikatoren SMO und Naive Bayes eine Überlegung wert war, aus dem Typed Dependency Bereich trotz minimal schlechterer Messwerte nur Unigram-Features für die nächsten Evaluierungen heranzuziehen, da sich dadurch möglicherweise nicht zu

Typed Dependency	1-Gram			1-Gram HW			1-2 Gram			1-3 Gram		
	SMO	NB	RDF	SMO	NB	RDF	SMO	NB	RDF	SMO	NB	RDF
1-Gram				v	v	v	-	-	-	-	-	-
1-Gram HW	*	*	*				*	*	*	*	*	*
1-2 Gram	-	-	-	v	v	v				-	-	-
1-3 Gram	-	-	-	v	v	v	-	-	-			

Tabelle 4.23: Typed Dependency N-Gram Signifikanztabelle

vernachlässigende Performance-Erhöhungen im Bereich der Feature-Extraktion und Klassifikation ergeben. Im Fall von Naive Bayes könnte sich die geringe Anzahl an Typed Dependency Features in nachfolgenden Experimenten mit zusätzlichen Attributen aus anderen Feature-Kategorien sogar positiv auf das Klassifikationsergebnis auswirken, da die Overfitting Gefahr verringert wird. Die Diagramme 4.20 und 4.21 zeigen jedoch, dass die Extraktionszeit beziehungsweise Klassifikationszeit des SMO und Naive Bayes für die Feature-Sets annähernd gleich sind und diese Schwankungen mit hoher Wahrscheinlichkeit auf dem Zufall basieren, wodurch trotz fehlender Signifikanz in erster Linie das Feature-Set mit den besten Klassifikationsmesswerten für spätere Untersuchungen eingebunden wurde. Für die Support Vector Machine und den Random Decision Tree sind das Typed Dependency Uni- und Bigrams, für den Naive Bayes Uni- bis Trigrams.

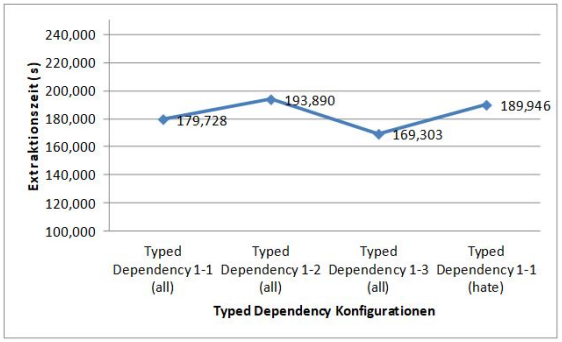


Abbildung 4.20: Extraktionszeit(s) Typed Dependency N-Grams.

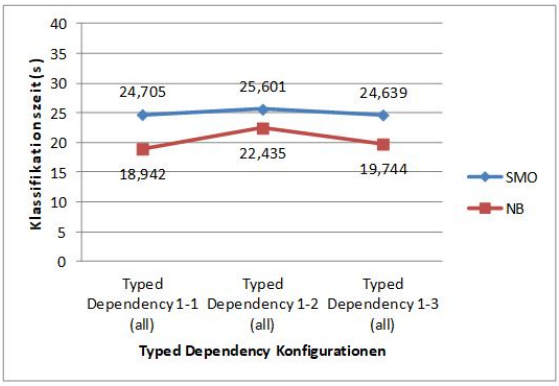


Abbildung 4.21: SMO und Naive Bayes Klassifikationszeit(s) für Typed Dependency N-Grams.

Character N-Gram Evaluierung

Als letzte Baseline-Option werden die verschiedenen Resultate der Character N-Grams untersucht. Dabei werden die einzelnen N-Grams und Kombinationen aus diesen evaluiert. Aufgrund der extrem hohen Anzahl, welche sich auf rund 20000 Features pro Set beläuft, wird auf Kombinationen von drei N-Grams verzichtet. Im Detail werden in Tabelle 4.24 nur 4- bis 7-Grams evaluiert, die aufgrund der F-Messwerte und deren auf und wieder

absteigenden Verläufe am interessantesten erscheinen. Zusätzlich werden diese N-Grams miteinander kombiniert und die dadurch resultierenden Klassifikationsergebnisse aufbereitet. Im Detail entstehen dadurch die zusätzlichen drei Feature-Sets aus 4-5, 5-6 und 6-7 Grams. Mit einem F-Messwert von 0,912 (91,2%) erzielt der SMO mithilfe von

Character N-Gram	**SMO**			**NB**			**RDF**		
	P	R	F	P	R	F	P	R	F
4-Gram	0,867	0,867	0,867	0,77	0,751	0,746	0,861	0,852	0,851
5-Gram	0,903	0,903	0,903	0,822	0,82	0,82	**0,874**	**0,861**	**0,859**
6-Gram	0,907	0,907	0,907	**0,839**	**0,839**	**0,839**	0,868	0,855	0,853
7-Gram	0,896	0,895	0,895	0,83	0,83	0,83	0,861	0,846	0,843
4-5-Gram	0,893	0,893	0,893	0,782	0,765	0,76	0,872	0,86	**0,859**
5-6-Gram	**0,912**	**0,912**	**0,912**	0,801	0,798	0,797	**0,874**	0,859	0,857
6-7-Gram	0,906	0,906	0,905	0,829	0,829	0,829	0,86	0,844	0,841

Tabelle 4.24: Character N-Gram Klassifikationsresultate

Character 5 und 6 Grams sein bestes Ergebnis, dem Naive Bayes Classifier reichen dafür lediglich Character 6-Grams, bei einem maximalen F-Messwert von 0,839 (83,9%). Für den Random Decision Forest liegen bei 5-Gram und 4-5 Gram Character-Features nahezu idente Werte vor (0,859 F-Maß), die sich nur in der Precision und im Recall unterscheiden. Signifikanztests, dessen Auswertungen in Tabelle 4.25 aufgeschlüsselt sind, belegen jedoch, dass bei Verwendung von 4-5 Gram Features ein weiterer signifikanter "Win", nämlich gegen das Character 6-7 Gram Feature-Set, erzielt werden kann. Aufgrund der höheren

Character N-Gram	**4-Gram**			**5-Gram**			**6-Gram**			**7-Gram**			**4-5-Gram**			**5-6-Gram**			**6-7-Gram**		
	SMO	NB	RDF	SMO	NB	RDF	SMO	NB	RDF	SMO	NB	RDF	SMO	NB	RDF	SMO	NB	RDF	SMO	NB	RDF
4-Gram				*	*	-	*	*	-	*	*	-	*	*	-	*	*	-	*	*	-
5-Gram	v	v	-				-	-	-	-	-	v	-	v	-	*	v	-	-	-	-
6-Gram	v	v	-	-	-	-				-	-	-	-	v	-	-	v	-	-	-	-
7-Gram	v	v	-	-	-	*	-	-	-				-	v	*	-	v	-	-	-	-
4-5-Gram	v	v	-	-	*	-	-	*	-	-	*	v				*	*	-	-	*	v
5-6-Gram	v	v	-	v	*	-	-	*	-	-	*	-	v	v	-				-	*	-
6-7-Gram	v	v	-	-	-	-	-	-	-	-	-	-	-	v	*	-	v	-			

Tabelle 4.25: Character N-Gram Signifikanztabelle

Effizienz bei Extraktion und Klassifizierung beziehungsweise wegen der bedeutend niedrigeren Menge an Features wäre in möglichen späteren Anwendungsfällen dennoch das ausschließlich aus 5-Gram bestehende Feature-Set zu bevorzugen. Die genauen Zeiten in Sekunden für die Erstellung der verschiedenen Character Feature-Sets sind aus dem Diagramm 4.22 zu entnehmen. Diagramm 4.23 zeigt die Klassifikationszeit des Random Decision Forest Classifiers je Character Attribut-Set, wobei auffällt, dass Kombinationen aus N-Grams sogar doppelt so lang dauern können. Bei kaum unterschiedlichen Klassifi-

kationsresultaten zu einfachen N-Grams, wie es in dieser Arbeit der Fall ist, kann dies als Argument gegen die Verwendung von Kombinationen aus verschiedenen N-Grams ausgelegt werden. So zum Beispiel benötigt die Modellerstellung und Evaluierung von 5-Grams rund 21 Minuten, während es bei 4-5 Grams geschlagene 43 Minuten dauert.

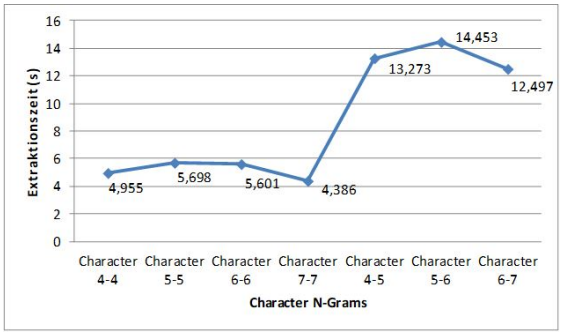


Abbildung 4.22: Extraktionszeit(s) Character N-Grams.

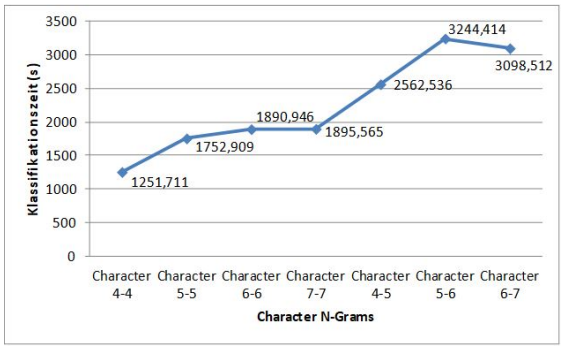


Abbildung 4.23: Random Decision Forest Klassifikationszeit(s) für Character N-Grams.

Zusammenfassung der Baseline Evaluierung

Nach der Analyse der Resultate der Feature-Kategorien die als Baseline in Frage kommen, kann festgestellt werden, dass Typed Dependency Attribute aufgrund zu schlechter Performance dafür ungeeignet sind und deshalb nur mehr Message- und Character N-Grams als Grundlage für Feature-Kombinationen im nächsten Abschnitt untersucht werden. Die endgültige Selektion an Features (N-Grams) je Classifier und Kategorie werden zusammenfassend in Tabelle 4.26 mit zusätzlichen Messwerten für die Kennzahlen True-Positive-Rate (TP) und False-Positive-Rate (FP) und für die einzelnen Klassen (Neutral/Hass) präsentiert.

Baseline-Eval. Resultat		**Message Feature**						**Character Feature**					
	Klasse	Konf.	TP	FP	P	R	F	Konf.	TP	FP	P	R	F
SMO	Neutral	HW 1-3 Gram	0,946	0,175	0,852	0,946	0,896	5-6 Gram	0,932	0,11	0,9	0,932	0,916
	Hass		0,825	0,054	0,934	0,825	0,876		0,89	0,068	0,924	0,89	0,907
	Avg.		0,887	0,117	0,892	0,887	0,887		0,912	0,09	0,912	0,912	0,912
NB	Neutral	HW 1-Gram	0,98	0,251	0,806	0,98	0,885	6-Gram	0,838	0,161	0,847	0,838	0,843
	Hass		0,749	0,02	0,972	0,749	0,846		0,839	0,162	0,83	0,839	0,834
	Avg.		0,868	0,139	0,887	0,868	0,866		0,839	0,161	0,839	0,839	0,839
RDF	Neutral	HW 1-2 Gram	0,96	0,18	0,85	0,96	0,902	4-5 Gram	0,951	0,237	0,811	0,951	0,875
	Hass		0,82	0,04	0,95	0,82	0,88		0,763	0,049	0,936	0,763	0,841
	Avg.		0,892	0,113	0,899	0,892	0,891		0,86	0,146	0,872	0,86	0,859

Tabelle 4.26: Resultat der Baseline Evaluierung

Evaluierung weiterer Feature-Kategorien

In diesem Abschnitt werden die restlichen Feature-Kategorien, dessen Attribute im Bezug auf deren Kalkulation nicht so umfangreiche Konfigurationsmöglichkeiten bieten, evaluiert. Die Resultate werden in Tabelle 4.27 aufbereitet. Besonders herausragende F-Maß Resultate sind mit der Farbe grün hervorgehoben, außerordentlich schlechte mit rot gekennzeichnet. Bei diesen Feature-Arten wurden die Ergebnisse pro Klasse inkludiert, um sie besser beurteilen zu können. So ist ersichtlich, ob ein Classifier unter Einbezug der jeweiligen Feature-Kategorie für eine bestimmte Klasse extreme Werte liefert.

Besonders interessant ist der Vergleich der Messwerte durch die Features aus dem Standard LIWC Wörterbuch und aus dem erweiterten, angepassten LIWC Korpus. Dabei ist eine signifikante Verbesserung der Messwerte mit den Features aus dem modifizierten LIWC-Korpus um rund 20% bei allen Kennzahlen und Classifieren festzustellen, wodurch sich bereits nur mithilfe der LIWC-Features ein respektables Klassifikations-Ergebnis erreichen lässt. Zusätzlich zu erkennen ist, dass sich im Fall von SMO die Follower-Features negativ beeinflussen, da deren gemeinsamer Einsatz schlechter klassifiziert als der getrennte. Bei der Klassifikation von Naive Bayes können durch die Fusion anscheinend ebenfalls keine zusätzlichen Informationen gewonnen werden. Anders verhält es sich bei den Twitter Merkmalen bei denen durch eine Kombination von user- und tweetbezogenen Twitter-Features nur der Naive Bayes- Classifier keine signifikante Verbesserung erzielen kann. Gesondert wurde auch die Performance bei zusätzlicher Berücksichtigung der bereits veröffentlichten Hasspostings pro User betrachtet. Dabei stellte sich heraus, dass im Fall von SMO (+9,2%) und Naive Bayes(+1,6%) diese signifikant gesteigert werden kann. Daher wird, trotz der für die Extraktion nötigen Datenbankzugriffe und der damit verbundenen relativ langen Berechnungszeit, dieses Feature für spätere Experimente beibehalten. Tabelle 4.28 fasst die zugehörigen Signifikanztests zwischen den einzelnen Twitter-Features zusammen.

Die in diesem Kontext besten Resultate werden durch lexikalische und angepasste LIWC Feature-Sets erreicht, Comment Embedding Features überzeugen an Hand der vorliegenden Ergebnisse am wenigsten. Als effektivster Klassifikator für diese Feature-Kategorie avanciert der Random Decision Forest (0,604 F-Score).

In den Diagrammen 4.24 und 4.25 werden die benötigten Zeiten für die Kalkulation der einzelnen Features beziehungsweise die Zeiten, die eine Klassifikation auf Basis dieser durch die verwendeten Classifier in Anspruch nimmt, aufgeschlüsselt. Die Erkenntnis, die bei einer Betrachtung von Balkendiagramm 4.24 gewonnen werden kann, lässt darauf schließen, dass für die Extraktion von Features, die sich auf spezielle Eigenschaften des Tweet-Textes beziehen, ein weit höherer Anteil an Rechenzeit aufgewandt wird. So zum Beispiel dauert die Berechnung der Follower und Twitter Attribute, trotz zusätzlicher Datenbankabfragen, jeweils maximal zwei Sekunden, während sich diese für die Mistake-Kategorie, welche gerade mal ein Feature umfasst, über eine Minute hinzieht. Ausnahme dabei sind die Features aus der LIWC-Kategorie. Das Bemerkenswerte daran ist, dass, obgleich der hohen Anzahl an Features dieser Art(82 LIWC-Features), nur 1,5 Sekunden

Feature-Familien			SMO			NB			RDF		
		Klasse	P	R	F	P	R	F	P	R	F
Mistake		Neutral	0,532	0,902	0,669	0,535	0,868	0,662	0,593	0,467	0,522
		Hass	0,598	0,156	0,247	0,582	0,197	0,294	0,537	0,659	0,592
		Avg.	0,564	0,541	0,465	0,558	0,543	0,484	0,566	0,56	0,556
LIWC	Standard	Neutral	0,626	0,873	0,729	0,657	0,734	0,693	0,665	0,749	0,704
		Hass	0,767	0,445	0,563	0,676	0,592	0,631	0,691	0,599	0,641
		Avg.	0,695	0,666	0,649	0,666	0,665	0,663	0,678	0,676	0,674
	Erweitert	Neutral	0,869	0,855	0,862	0,832	0,899	0,864	0,88	0,898	0,889
		Hass	0,848	0,863	0,855	0,882	0,807	0,843	0,889	0,869	0,879
		Avg.	0,859	0,859	0,859	0,857	0,855	0,854	0,884	0,884	0,884
Twitter (TW)	Tweet	Neutral	0,541	0,886	0,672	0,619	0,152	0,244	0,563	0,559	0,561
		Hass	0,62	0,199	0,301	0,499	0,901	0,642	0,534	0,538	0,536
		Avg.	0,579	0,553	0,492	0,561	0,514	0,437	0,549	0,549	0,549
	User o. #HP	Neutral	0,554	0,595	0,574	0,668	0,112	0,192	0,653	0,689	0,671
		Hass	0,532	0,49	0,51	0,499	0,941	0,652	0,648	0,609	0,628
		Avg.	0,544	0,545	0,543	0,586	0,513	0,415	0,65	0,651	0,65
	User m. #HP	Neutral	0,615	0,972	0,753	0,726	0,127	0,217	0,643	0,768	0,7
		Hass	0,922	0,351	0,508	0,505	0,949	0,659	0,688	0,545	0,608
		Avg.	0,764	0,671	0,635	0,619	0,525	0,431	0,665	0,66	0,656
	Tweet + User m. #HP	Neutral	0,638	0,866	0,735	0,749	0,118	0,204	0,66	0,797	0,722
		Hass	0,769	0,478	0,589	0,505	0,958	0,661	0,722	0,563	0,633
		Avg.	0,702	0,678	0,664	0,631	0,525	0,426	0,69	0,684	0,679
Linguistik (LING)		Neutral	0,606	0,513	0,556	0,577	0,712	0,637	0,59	0,612	0,601
		Hass	0,554	0,645	0,596	0,591	0,444	0,507	0,57	0,547	0,558
		Avg.	0,581	0,577	0,575	0,584	0,582	0,574	0,58	0,58	0,58
Lexikalisch (LEX)		Neutral	0,79	0,73	0,759	0,786	0,746	0,766	0,737	0,769	0,753
		Hass	0,734	0,794	0,763	0,743	0,784	0,763	0,742	0,708	0,725
		Avg.	0,763	0,761	0,761	0,766	0,764	0,764	0,74	0,74	0,739
Follower (FO)	Site Follower	Neutral	0,559	0,891	0,687	0,581	0,838	0,686	0,58	0,84	0,686
		Hass	0,685	0,251	0,368	0,673	0,355	0,465	0,674	0,352	0,463
		Avg.	0,62	0,582	0,533	0,625	0,604	0,579	0,625	0,604	0,578
	User Follower	Neutral	0,555	0,902	0,687	0,573	0,877	0,693	0,573	0,877	0,693
		Hass	0,687	0,23	0,345	0,697	0,303	0,422	0,697	0,303	0,422
		Avg.	0,619	0,577	0,521	0,633	0,599	0,562	0,633	0,599	0,562
	User + Site	Neutral	0,551	0,906	0,685	0,581	0,855	0,692	0,587	0,838	0,691
		Hass	0,68	0,213	0,324	0,69	0,343	0,458	0,683	0,373	0,483
		Avg.	0,613	0,57	0,51	0,634	0,607	0,579	0,634	0,613	0,59
Comment Embedding (COMM)		Neutral	0,523	0,969	0,679	0,522	0,885	0,657	0,602	0,711	0,652
		Hass	0,637	0,059	0,107	0,527	0,136	0,217	0,619	0,5	0,553
		Avg.	0,578	0,528	0,402	0,525	0,523	0,444	0,61	0,609	0,604

Tabelle 4.27: Feature-Kategorien Klassifikationsresultate

Twitter Feature	Tweet			User o. #HP			User m. #HP			Kombination		
	SMO	NB	RDF	SMO	NB	RDF	SMO	NB	RDF	SMO	NB	RDF
Tweet				*	-	*	*	-	*	*	-	*
User o. #HP	v	-	v				*	*	-	*	*	*
User m. #HP	v	-	v	v	v	-				*	-	*
Kombination	v	-	v	v	v	v	v	-	v			

Tabelle 4.28: Twitter Feature-Sets Signifikanztabelle

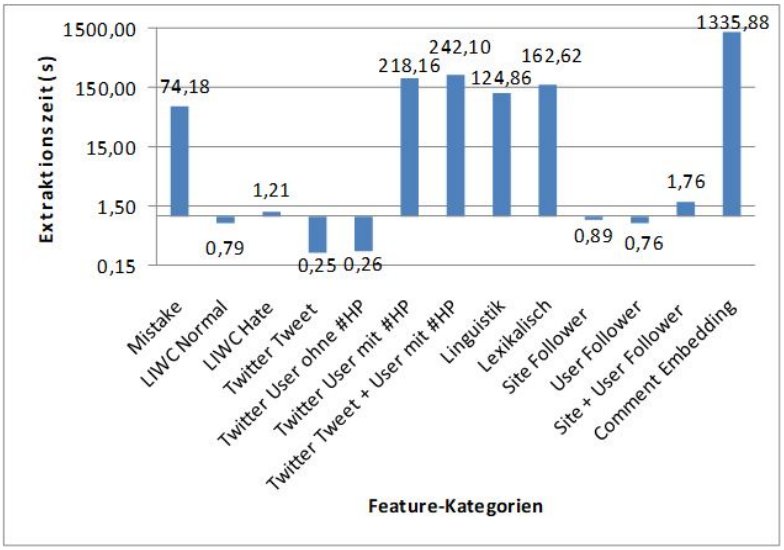
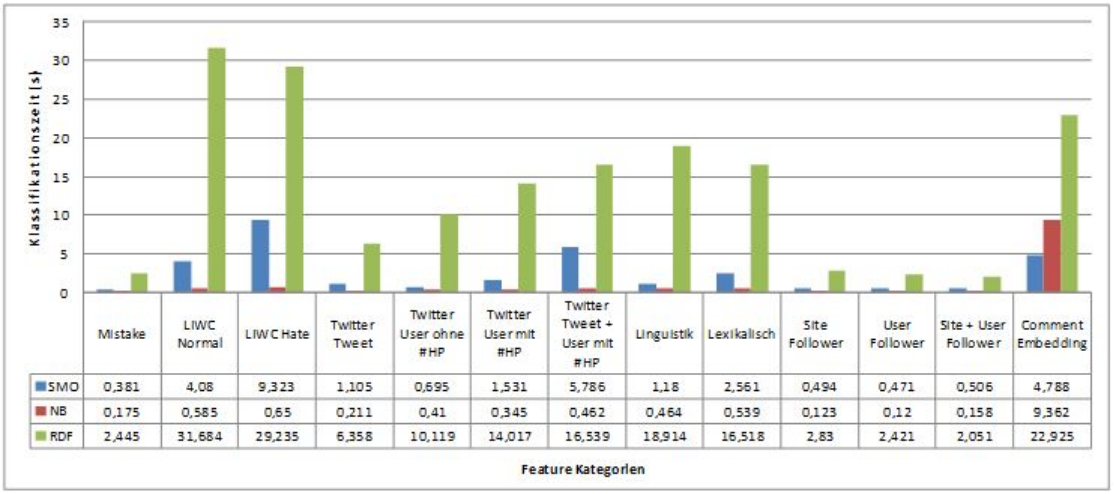


Abbildung 4.24: Extraktionszeiten(s) der Feature-Kategorien.

für deren Erzeugung benötigt wird. Grund dafür ist, dass weder Berechnungen nötig sind, wie es bei linguistischen Merkmalen vorkommt, noch wie im Fall von lexikalischen Features für jedes einzelne Feature eine Wortliste geladen werden muss. Auch der Abgleich der Terme des Tweets mit dem Inhalt der Listen ist zeitaufwändiger als jener im Rahmen der LIWC-Wortzählung, da dieser effizienter implementiert werden konnte. Für das Erlernen der Comment Embedding Vektoren aus den vorklassifizierten Daten wird mit großem Abstand am meisten Rechenzeit aufgewendet (~22min). Das Balkendiagramm 4.25, welches die Klassifikationszeiten von SMO, Naive Bayes und Random Decision Forest für die in diesem Abschnitt behandelten Features gegenüberstellt, lässt sofort erkennen, dass die Klassifikation durch den Random Decision Forest Algorithmus am längsten dauert. Ursache sind die Berechnungen und Auswertungen der vielen einzelnen Bäume, die einen Großteil der gemessenen Zeit ausmachen. Im Vergleich zu den anderen Classifiern dauert diese sogar mindestens doppelt so lang. Am effizientesten arbeitet der Naive Bayes Algorithmus.



	Mistake	LIWC Normal	LIWC Hate	Twitter Tweet	Twitter User ohne #HP	Twitter User mit #HP	Twitter Tweet + User mit #HP	Linguistik	Lexikalisch	Site Follower	User Follower	Site + User Follower	Comment Embedding
SMO	0,381	4,08	9,323	1,105	0,695	1,531	5,786	1,18	2,561	0,494	0,471	0,506	4,788
NB	0,175	0,585	0,65	0,211	0,41	0,345	0,462	0,464	0,539	0,123	0,12	0,158	9,362
RDF	2,445	31,684	29,235	6,358	10,119	14,017	16,539	18,914	16,518	2,83	2,421	2,051	22,925

Abbildung 4.25: Klassifikationszeiten(s) der Feature-Kategorien.

4.4.5 Evaluierung von kombinierten Feature-Kategorien

In diesem Kapitel werden nun die zuvor untersuchten Feature-Arten miteinander verknüpft und die entstehenden Kombinationen genauer analysiert. Dabei werden verschiedene Herangehensweisen gewählt. Zum einen werden die Baseline Feature-Sets *Message N-Gram* und *Character N-Gram* jeweils nacheinander mit jenen Feature-Kategorien erweitert, von denen man sich die meiste Verbesserung verspricht. Die Typed Dependency (TD) und Comment Embedding (COMM) Attribute werden aufgrund der zu schlechten Messwerte nicht als Baseline getestet, sondern nur als weitere Merkmale abwechselnd in das Set aufgenommen. Zum anderen werden jene Features miteinander kombiniert, die sich aufgrund der Resultate der Einzelevaluierung als vielversprechend erwiesen haben. Das heißt es werden Feature-Sets entstehen, die ausschließlich Merkmale enthalten, die einen bestimmten Information Gain Schwellenwert übersteigen oder die in der CFS-Subset Evaluierung vorkommen.

Feature-Sets auf Basis der Klassifikationsresultate der Kategorien

Tabelle 4.29 beinhaltet die Klassifikationsresultate der verschiedenen Classifier, auf Basis von Feature-Sets bestehend aus Message N-Grams, die als Baseline fungieren, und zusätzlichen Feature-Kategorien, die aufgrund der Ergebnisse in Tabelle 4.27 aus Abschnitt 4.4.4 nacheinander in das Set aufgenommen wurden. Aufgrund der höchsten Messwerte bei modifizierten LIWC-Attribute wurde mit ihnen begonnen und in absteigender Reihenfolge in Bezug auf den F-Messwert ganze Kategorien in das Baseline-Set inkludiert. Für die Feature-Kombinationen in Tabelle 4.31 werden als Baseline Character N-Grams verwendet. Welche N-Grams von Message- und Character - Features im Detail eingesetzt wurden, ist abhängig vom jeweiligen Classifier und den Erkenntnissen aus den Tabellen 4.19 und 4.24 im Abschnitt 4.4.4, der die Evaluierung der möglichen Baselines beschrieb.

Feature-Kombinationen Message N-Gram Baseline	**SMO**			**NB**			**RDF**		
	P	R	F	P	R	F	P	R	F
Baseline	0,892	0,887	0,887	**0,887**	0,868	0,866	0,899	0,892	0,891
+ LIWC	0,905	0,902	0,901	0,886	**0,88**	**0,88**	0,903	0,901	0,9
+ LIWC + LEX	0,903	0,899	0,899	0,881	0,879	0,879	0,894	0,893	0,893
+ LIWC + LEX + TW	0,915	0,912	0,912	0,802	0,762	0,755	0,903	0,902	0,902
+ LIWC + LEX + TW + LING	0,917	0,914	0,914	0,803	0,77	0,765	0,898	0,897	0,897
+ LIWC + LEX + TW + LING + FO	**0,92**	**0,918**	**0,918**	0,821	0,791	0,787	0,904	0,903	0,903
+ LIWC + LEX + TW + LING + FO + MIS	0,919	0,917	0,916	0,824	0,795	0,791	**0,909**	**0,908**	**0,908**
+ LIWC + LEX + TW + LING + FO + MIS + COMM	0,916	0,915	0,914	0,854	0,851	0,851	0,89	0,889	0,889
+ LIWC + LEX + TW + LING + FO + MIS + TD	0,918	**0,918**	**0,918**	0,827	0,818	0,818	0,905	0,903	0,903

Tabelle 4.29: Klassifikationsresultate durch Feature-Kombinationen bei einer Message N-Gram Baseline

Feature Komb. Message N-Gram	**Baseline**			**Vorgänger Feature-Set**			**Beste Feature-Set** (5)	(2)	(6)
	SMO	NB	RDF	SMO	NB	RDF	SMO	NB	RDF
Baseline (B)							*	*	-
(B) + LIWC (1)	-	-	-	-	-	-	*	-	-
(1) + LEX (2)	-	v	-	-	-	-	*	—	-
(2) + TW (3)	v	*	-	v	*	-	-	*	-
(3) + LING (4)	v	*	-	-	-	-	-	*	-
(4) + FO (5)	v	*	-	-	v	-	—	*	-
(5) + MIS (6)	v	*	-	-	-	-	-	*	—
(6) + COMM (7)	v	-	-	-	v	*	-	*	*
(6) + TD (8)	v	*	-	-	-	-	-	*	-

Tabelle 4.30: Feature-Kombinationen mit Message N-Gram Baseline Signifikanztabelle

Für den SMO reichen bei einer Message N-Gram Baseline die zusätzlichen Feature-Kategorien LIWC, lexikalisch, Twitter, linguistisch und Follower, um sein bestes Klassifikationsresultat mit einer Precision von 0,911 und einem Recall beziehungsweise F-Messwert von 0,908 zu erzielen. Naive Bayes benötigt dafür sogar nur die LIWC-Features als Baseline-Erweiterung, liefert aber mit einer Precision von 0,886 und einem Recall beziehungsweise F-Maß von 0,88 eine schlechtere Performance ab. Unter Einbezug fast aller Feature-Kategorien mit Ausnahme der Comment Embeddings und Typed Dependency Attribute klassifiziert der Random Decision Forest mit einer Precision von 0,92, einem

Feature-Kombinationen Character N-Gram Baseline	**SMO**			**NB**			**RDF**		
	P	R	F	P	R	F	P	R	F
Baseline	0,912	0,912	0,912	0,839	0,839	0,839	0,872	0,86	0,859
+ LIWC	0,928	0,928	0,928	0,84	0,84	0,84	0,892	0,884	0,883
+ LIWC + LEX	0,928	0,927	0,927	0,842	0,842	0,842	0,899	0,893	0,893
+ LIWC + LEX + TW	0,932	0,932	0,932	0,843	**0,843**	**0,843**	0,897	0,891	0,89
+ LIWC + LEX + TW + LING	0,934	0,933	0,933	0,842	0,842	0,842	0,901	0,893	0,893
+ LIWC + LEX + TW + LING + FO	0,934	0,934	0,934	0,841	0,841	0,841	**0,905**	**0,901**	**0,9**
+ LIWC + LEX + TW + LING + FO + MIS	0,935	0,934	0,934	0,841	0,841	0,841	0,899	0,893	0,892
+ LIWC + LEX + TW + LING + FO + MIS + COMM	0,935	0,935	0,935	0,843	**0,843**	**0,843**	0,884	0,88	0,879
+ LIWC + LEX + TW + LING + FO + MIS + TD	**0,938**	**0,937**	**0,937**	**0,844**	**0,843**	**0,843**	0,903	0,896	0,895

Tabelle 4.31: Klassifikationsresultate durch Feature-Kombinationen bei einer Character N-Gram Baseline

Feature Komb. Character N-Gram	**Baseline**			**Vorgänger Feature-Set**			**Beste Feature-Set** (8)	(3)	(5)
	SMO	NB	RDF	SMO	NB	RDF	SMO	NB	RDF
Baseline (B)							*	*	*
(B) + LIWC (1)	v	-	v	v	-	v	*	-	*
(1) + LEX (2)	v	-	v	-	-	-	*	-	-
(2) + TW (3)	v	v	v	-	-	-	-	—	-
(3) + LING (4)	v	-	v	-	-	-	-	-	-
(4) + FO (5)	v	-	v	-	-	-	-	-	—
(5) + MIS (6)	v	-	v	-	-	-	-	-	-
(6) + COMM (7)	v	v	-	-	-	-	-	-	-
(6) + TD (8)	v	-	v	-	-	-	—	-	-

Tabelle 4.32: Feature-Kombinationen mit Character N-Gram Baseline Signifikanztabelle

Recall von 0,918 und einem F-Messwert von 0,918 am effektivsten.

Die Signifikanz der Messwert-Änderungen durch die Feature-Sets gegenüber der Baseline, seinem jeweiligen Vorgänger und jener Feature-Kombination, die aufgrund der Messwerte und Signifikanztests im aktuellen Kontext als “Bestes Feature-Set“ angesehen wird, wurden getestet und die Resultate in Tabelle 4.30 festgehalten. Erwähnenswert dabei ist, dass für den Naive Bayes Klassifikator die Kombination aus Message Baseline und LIWC-Attributen zwar im Hinblick auf die Messwerte der Kennzahlen die besten Ergebnisse

liefert, aber erst das Feature-Set inklusive der lexikalischen Attribute aufgrund der geringeren Varianz der Werte signifikant besser ist als die Message-Baseline. In weiterer Folge wurde eben diese Kombination aus Features als bestes Feature-Set für den Naive Bayes bei einer Baseline aus Message N-Grams definiert. Ab Feature-Set (3) sind die korrespondierenden Resultate durch den SMO signifikant besser als bei der Baseline. Im Bezug auf den Random Decision Forest kann trotz einer Steigerung des F-Scores von 0,891 auf 0,908 durch die Feature-Kombination (6) keine Signifikanz für diese nachgewiesen werden, lediglich das Feature-Set (7) ist signifikant schlechter.

Im Kontext der Character N-Gram Baseline performen SMO und Naive Bayes mit nahezu allen Features bei einem F-Messwert von 0,937 beziehungsweise 0,843 am besten. Die höchsten Werte der dargestellten Kennzahlen werden im Fall einer Klassifizierung durch den Random Decision Forest aufgrund zusätzlicher LIWC, lexikalischer, Twitter, linguistischer und Follower Attribute ermöglicht und betragen für Precision, Recall und F-Maß jeweils 0,905, 0,901 und 0,9.

Vergleicht man die höchsten Resultate je Classifier und Baseline, so schneidet der SMO mit einer Character N-Gram Baseline besser ab, Naive Bayes und Random Decision Forest liefern mit Hilfe von Message N-Grams als Baseline bessere Ergebnisse.

Tabelle 4.32 enthält die Signifikanz-Testergebnisse für die Feature-Kombinationen mit Character N-Gram Baseline. Interessanterweise sind auch hier die höchsten Messwerte des Naive Bayes-Classifiers, die durch das Feature-Set (8) hervorgerufen werden, nicht signifikant besser als die der Baseline. Für die Feature-Kombinationen (3) und (7), die bei Recall und F-Score dieselben Werte wie (8) erreichen, ist die Performance-Steigerung jedoch signifikant. Aufgrund der geringeren Anzahl an Features wurde das Set (3) für die weiteren Signifikanztests als “Bestes Feature-Set“ für den Naive Bayes herangezogen. SMO und Random Decision Forest klassifizieren beinahe mit jeder analysierten Kombination signifikant besser als mit der Character N-Gram Baseline.

Die Diagramme 4.26 und 4.27 bilden die F-Messwerte für die in den Tabellen 4.29 beziehungsweise 4.31 angeführten Feature-Sets und den sich dadurch ergebenden Verlauf je Klassifikator ab, wodurch die zuvor getroffenen Schlussfolgerungen noch einmal unterstrichen werden. Während sich die F-Score Verläufe von SMO und Random Decision Forest überwiegend ähneln, so fällt beim Naive Bayes und einer Message Baseline ein großer Einbruch des Verlaufs durch das Hinzufügen der Twitter Features auf. Der F-Score erlebt einen Absturz von 0,879 auf 0,755 (-0,124). Nach genauerer Untersuchung kann gesagt werden, dass aufgrund der Verwendung von ausschließlich Hateful Message Unigrams (118), Twitter-Attribute einen verhältnismäßig großen negativen Einfluss haben, der durch die geringe Menge an Message-Features nicht ausgeglichen werden kann. Beim Einsatz von hasserfüllten Uni- und Bigrams ändert sich der F-Messwert zwischen den besagten Feature-Sets lediglich von 0,888 auf 0,872 (-0,016). Das entspricht im Vergleich zur vorherigen Verschlechterung nur etwas mehr als einem Achtel. Auf Basis der Tatsache, dass mithilfe von Typed Dependency oder Comment Embedding Features, die den Tweet-Inhalt abbilden, die Klassifikation ebenfalls stark verbessert wird, kann diese Annahme bestätigt werden.

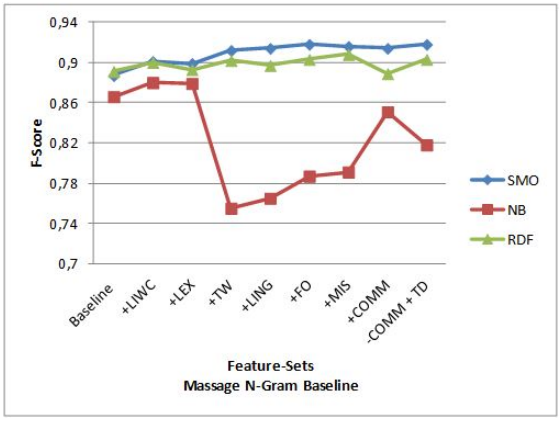


Abbildung 4.26: Verlauf der F-Messwerte der Feature-Sets bei einer Message N-Gram Baseline.

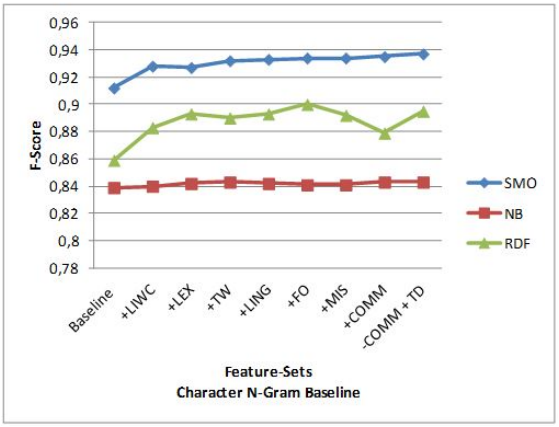


Abbildung 4.27: Verlauf der F-Messwerte der Feature-Sets bei einer Character N-Gram Baseline.

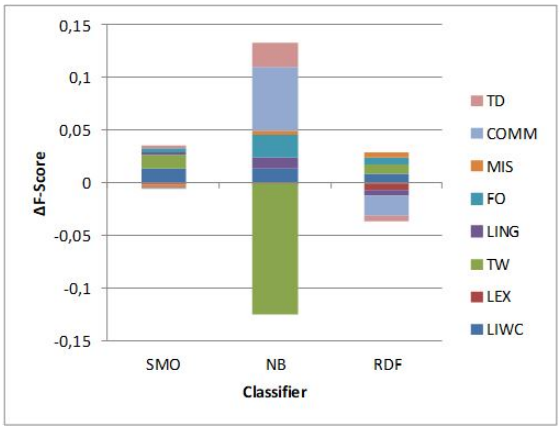


Abbildung 4.28: F-Score Veränderungen durch die Feature-Kategorien bei einer Message N-Gram Baseline.

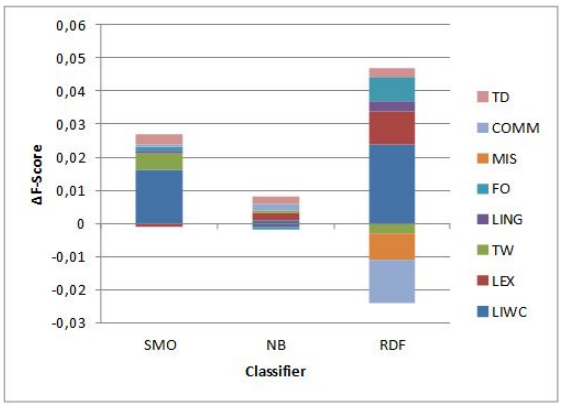


Abbildung 4.29: F-Score Veränderungen durch die Feature-Kategorien bei einer Character N-Gram Baseline.

Die Balkendiagramme 4.28 und 4.29 visualisieren die absolute Veränderung der F-Messwerte im Bezug auf das Vorgänger-Feature-Set, die durch die jeweilige zusätzliche Feature-Kategorie hervorgerufen wird. Somit lassen sich die Auswirkungen je Kategorie noch besser darstellen. Im Fall der Message Baseline wirken sich LIWC und Follower Features immer positiv aus, lexikalische Merkmale bringen für keinen Classifier Vorteile, sondern verschlechtern die Messwerte. Comment Embeddings wirken sich außer für den Naive Bayes aufgrund zuvor genannter Gründe ebenfalls negativ aus. Gut zu erkennen ist auch der bereits erwähnte schlechte Einfluss des Twitter Features auf die Klassifikation durch den Naive Bayes Algorithmus, wodurch dieser Classifier auch den gravierendsten F-Score Schwankungen unterliegt. Das genaue Gegenteil tritt bei einer

Character Baseline ein. Hier ändern sich die F-Messwerte des Naive Bayes kaum, die der zwei anderen sich jedoch verhältnismäßig bedeutend. Besonders auffällig sind hier die positiven Auswirkungen der LIWC Attribute oder im Fall von Random Decision Forest auch die der lexikalischen Merkmale. Comment Embeddings und das Mistake-Attribut verschlechtern das Resultat des Random Decision Forests wiederum merklich.

Feature-Sets auf Basis des Information-Gains

Dieser Abschnitt untersucht Kombinationen von Features die einen gewissen Information Gain Wert übersteigen. Ausgenommen von der Analyse sind in diesem Kontext Comment Embeddings, die teuer zu berechnen sind und die Ergebnisse in den meisten Fällen verschlechtern. Im Detail wurden Feature-Sets mit einem IG-Wert ihrer Merkmale ab 0,005, 0,015 und 0,030 untersucht, wobei einerseits jeweils alle möglichen Feature-Kategorien dieser Einschränkung unterliegen und andererseits nur die Features aus den Bereichen Linguistik, Lexikon, Twitter, Mistake und Follower für jedes dieser Limits berücksichtigt werden, Message-, Character- und Typed Dependency- N-Grams also im vollen Umfang im Set enthalten sind. Zum Vergleich wurden die Messwerte der Kennzahlen bei einer Kombination aller verfügbaren Features evaluiert. Tabelle 4.33 zeigt die Resultate der sich daraus ergebenden Experimente. Die Classifier SMO und Naive Bayes klassifizieren unabhängig des Information Gain Schwellenwerts für die Fälle signifikant besser, bei denen nur die ausgewählten Feature-Kategorien nach IG beschränkt werden und aus den restlichen Attribut-Kategorien alle Features erhalten bleiben. Die höchsten Werte werden dabei jeweils mithilfe von Features mit einem IG größer gleich 0,015 erzielt. Der Random Decision Forest Algorithmus verhält sich unterschiedlich und erreicht sein klar bestes Ergebnis mit einem F-Maß von 0,921 bei einem IG >=0,005 aller Features.

Feature-Kombinationen Information Gain (IG)		**SMO**			**NB**			**RDF**		
		P	R	F	P	R	F	P	R	F
Alle[14] Featurekategorien		0,936	0,936	0,936	0,849	0,848	0,848	0,905	0,897	0,896
IG >=0,005	Alle[14]	0,922	0,918	0,918	**0,871**	0,841	0,837	**0,925**	**0,921**	**0,921**
	ausgw. FK	0,937	0,936	0,936	0,85	0,85	0,85	0,908	0,902	0,901
IG >=0,015	Alle[14]	0,872	0,872	0,872	0,811	0,727	0,704	0,892	0,89	0,89
	ausgw. FK	**0,938**	**0,937**	**0,937**	0,852	**0,852**	**0,852**	0,909	0,902	0,901
IG >=0,030	Alle[14]	0,865	0,864	0,864	0,812	0,724	0,699	0,871	0,87	0,87
	ausgw. FK	0,936	0,935	0,935	0,849	0,848	0,848	0,907	0,901	0,9

Tabelle 4.33: Resultate von Feature-Sets basierend auf dem Information Gain.

Bei Naive Bayes und Random Decision Forest sind jene Feature-Sets, die jeweils zu den höchsten Messwerten beitragen, signifikant besser als jenes, das sich durch die Verwendung aller möglicher Features ergibt. Beim Einsatz von SMO unterscheidet sich das

[14] Ausgenommen Comment Embedding - Features

Ergebnis durch Features mit einem IG ausschließlich größer gleich 0,015 aus ausgewählten Kategorien von dem durch alle einsetzbaren Attribute nicht signifikant.

Feature-Sets auf Basis der CFS Subset Evaluierung

Weitere Feature-Kombinationen sollen aufgrund der CFS-Subset Evaluierung aus Abschnitt 4.3.8 gebildet werden. Dabei wird so vorgegangen, dass die Attribute aus den subjektiv wichtigsten Kategorien Message, Character, Typed Dependency und LIWC als Grundlage dienen und mit jenen Features kombiniert werden, die ins Subset während der Evaluierung aufgenommen wurden. Die Feature-Kombination entstehen genauer gesagt dadurch, indem die Häufigkeit der Auswahl der einzelnen Attribute in das CFS Subset während einer 10-fachen Kreuz-Validierung begutachtet wird. Angefangen wird mit allen Features, die zumindest einmal in der Subset-Evaluierung vorkommen, daraufhin werden nur solche für die Klassifikation herangezogen, die mindestens dreimal berücksichtigt werden, dann mindestens sechsmal und zuletzt nur jene Attribute die in allen zehn Durchläufen der Kreuz-Validierung im Subset enthalten sind. Tabelle 4.34 zeigt die Resultate der beschriebenen Experimente. Zur Gegenüberstellung der Messwerte wurde auch hier das Ergebnis aus der Klassifikation mit allen zur Verfügung stehenden Features in der Tabelle erfasst. Während die aufgrund dieser Methode resultierenden Feature-Sets die Messwerte für den Naive Bayes kaum verbessern und sich die des Random Decision Forest Classifiers sogar verschlechtern, steigert SMO seine Performance, bei Berücksichtigung jener Features aus zuvor genannten Kategorien, die mindestens einmal für das Subset ausgewählt wurden, von 0,936 auf 0,939. Diese Steigerung ist jedoch wie die bei der Verwendung des Naive Bayes Algorithmus nicht signifikant.

Feature-Kombinationen CFS Subset Evaluierung		**SMO**			**NB**			**RDF**		
		P	R	F	P	R	F	P	R	F
Alle[15] Featurekategorien		0,936	0,936	0,936	**0,849**	0,848	0,848	**0,905**	**0,897**	**0,896**
>=1	ausgw. FK	**0,94**	**0,939**	**0,939**	**0,849**	0,848	0,848	0,9	0,891	0,89
>=3	ausgw. FK	0,935	0,934	0,934	**0,849**	**0,849**	**0,849**	0,893	0,882	0,881
>=6	ausgw. FK	0,938	0,937	0,937	**0,849**	**0,849**	**0,849**	0,901	0,891	0,89
=10	ausgw. FK	0,937	0,936	0,936	**0,849**	**0,849**	**0,849**	0,899	0,89	0,889

Tabelle 4.34: Resultate von Feature-Sets basierend auf der CFS-Subset Evaluierung

4.4.6 Zusammenfassung der Feature-Set Evaluierung

Aufgrund der verschiedenen verfolgten Ansätze für den Erhalt des besten Feature-Sets je Classifier sollen die effektivsten Resultate, beziehungsweise jene Features die dafür nötig waren, zur besseren Übersicht noch einmal gesondert zusammengefasst werden, um Classifier-übergreifend verglichen werden zu können. Dabei sollen ferner die Messwerte je Klasse und die von weiteren Kennzahlen wie der True-Positiv- und False Positiv-Rate

[15] Ausgenommen Comment Embedding - Features

präsentiert werden. Die Zeiten für die Extraktion der jeweiligen Features und die Zeiten für die Evaluierung (10-fache Kreuz-Validierung) ist ebenfalls Inhalt der Tabelle 4.35.

	Resümee Feature-Set Evaluierung							
	Feature-Set	#Feat.	Klasse	TP	FP	P	R	F
SMO	CFS Subset-Evaluierung >=1 bei ausgw. Features	50816	Neutral	0,962	0,086	0,923	0,962	0,942
			Hass	0,914	0,038	0,958	0,914	0,935
			Avg.	0,939	0,063	0,94	0,939	0,939
NB	Message HW Unigram Baseline + LIWC + LEX	202	Neutral	0,918	0,162	0,858	0,918	0,887
			Hass	0,838	0,082	0,905	0,838	0,87
			Avg.	0,879	0,124	0,881	0,879	0,879
RDF	IG>=0,005 bei allen Features	399	Neutral	0,97	0,131	0,888	0,97	0,927
			Hass	0,869	0,03	0,965	0,869	0,915
			Avg.	0,921	0,082	0,925	0,921	0,921

Tabelle 4.35: Besten Feature-Sets je Klassifikator - Zusammenfassung

Der Klassifikator mit der höchsten Performance nach F-Score ist die Support Vector Machine SMO (0,939), gefolgt vom Random Decision Forest (0,921). Mit etwas Abstand verweilt der Naive Bayes mit dem F-Maß von 0,879 auf letzter Stelle. Die wenigsten Features für ihr bestes Resultat benötigt der Naive Bayes mit 202 Attributen, Random Decision Forest braucht dafür etwas weniger als doppelt so viele (399). SMO bezieht fast alle möglichen Features (50816) für die Klassifikation ein. Die große Anzahl erklärt sich dadurch, dass Message, Character und Typed Dependency Features uneingeschränkt enthalten sind.

Zeitliche Performance(s)	Extraktion	Evaluation	Modellbildung	Klassifikation/Posting
SMO	484,367s ∼8min	330,92s ∼5min 30s	47,357s	∅ 0,001335s
NB	168,697s ∼2min 50s	1,485s	0,234s	∅ 0,000172s
RDF	806,365s ∼13min 30s	84926s ∼1min 15s	7,785s	∅ 0,000423s

Tabelle 4.36: Zeitliche Performance je Classifier für das beste Feature-Set

Für die Extraktion dieser großen Menge an Features wird rund acht Minuten in Anspruch genommen, die 10-fache Kreuz-Validierung erledigt der SMO in rund fünf Minuten und 30 Sekunden. Am längsten wird das Feature-Set des Random Decision Forests berechnet

(∼13min 30s), da sich ein Mehraufwand durch die Information Gain Kalkulation je Attribut ergibt. Ebenfalls lange dauert die Modellbildung und Evaluierung durch den Random Decision Forest (∼1min 15s), bedenkt man, dass die Anzahl an Features im Set verhältnismäßig gering ist. Im Bezug auf die zeitliche Performance schneidet Naive Bayes am besten ab und erledigt Extraktion (∼2min 50s), Modellbildung und Klassifikation (∼1,485s) für seine beste Kombination an Attributen in rund drei Minuten. Betrachtet man ausschließlich die Zeit für die Modellbildung, ergeben sich verhältnismäßig ähnliche Resultate. Der SMO benötigt, aufgrund der zuvor bereits erwähnten hohen Anzahl an Features, mit 47,357s mit Abstand am längsten, gefolgt vom Random Decision Forest mit rund einem sechstel der Zeit (7,785s). Die aufgebrachte Zeit für die Modellbildung des Naive Bayes ist mit 0,234s zu vernachlässigen und ergibt sich aufgrund der Einfachheit des dahinterliegenden Modells. Diese Rangordnung der Klassifikatoren wird nach der Messung der Klassifikationszeit pro Posting ebenfalls festgestellt. Der SMO stellt die Klasse in rund 1,335 Millisekunden(ms) fest, der Random Decision Forest in 0,423ms und der Naive Bayes in 0,172ms. Um Schwankungen auszugleichen wurden von den Classifiern die selben 215 Tweets klassifiziert und der Durchschnitt der Klassifikationszeit je Posting berechnet.

Parametertuning

Durch das Anpassen der Parameter der Klassifikatoren sollen die in Tabelle 4.35 enthaltenen Messwerte noch weiter verbessert und so ein endgültiges Setting für die Klassifikation von Hasspostings im Kontext des jeweils optimalen Feature-Set gefunden werden. Im nächsten Abschnitt werden die getesteten Parameter je Classifier beziehungsweise im Fall von Verbesserungen die angepassten Parameterwerte und dazugehörigen Ergebnisse beschrieben.

Im Fall von SMO wurden verschiedene Konfigurationen für den *Komplexiätsparamter c*, den *Epsilon-Wert* und den zu verwendeten Kernel getestet. Als Kernel wurde der *Polynomial-Kernel* beibehalten. Für den Naive Bayes Classifier wurde, wie zuvor erwähnt, ein Kerndichteschätzer für nummerische Attribute ausgewählt anstelle von einer Normalverteilung auszugehen und die Auswirkungen bei Aktivierung der Option "Supervised Discretization", die nummerische in nominale Attribute konvertiert, untersucht. Für den Random Forest Algorithmus wurde die maximale Baumtiefe, die Anzahl an zu erzeugenden Bäumen für die Entscheidungsfindung und die Menge an Features für die Erstellung des Baumes variiert.
Damit die besten Parameterkonfigurationen nun evaluiert werden konnten, wurden die Werte der Parameter nach einer bestimmten Vorgehensweise verändert, wobei zwischen der Art der Parameter unterschieden werden muss. Bei Parametern mit kontinuierlichen Werten werden diese schrittweise innerhalb des möglichen Wertintervalls bis zu den jeweiligen Extremen verändert. Ergab sich bei einer Anpassung eine Verbesserung wurde in diesem Wertebereich feiner justiert, damit möglicherweise noch höhere Performance-Steigerungen erzielt und mögliche Tendenzen erkannt werden. Der Komplexitätsparameter c wurde so jeweils vom Standardwert 1,0 um 0,1 in beide Richtungen verändert, der

Epsilon-Wert ausgehend vom Standardwert 1.0E-12 um Zehnerpotenzen ebenfalls in die positive und negative Richtung angepasst.
Im Kontext des Random Decision Forest wurden die Werte für die maximale Baumtiefe, die Feature-Anzahl händisch festgelegt und sogar immer nur um 1er-Schritte modifiziert, die Menge an Bäumen ausgehend von 100 in 50er-Schritten.
Bei Classifier-Parametern mit diskreten, nominalen Werten wurde jede der Auswahlmöglichkeiten getestet. So zum Beispiel wurden alle verfügbaren Kernel des SMO-Klassifikators in ihrer Standardeinstellung einzeln eingesetzt und ermittelt, ob sich für das gegebene Feature-Set ein besser geeigneter Kernel anbietet. Generell wurde immer nur ein Parameter gegenüber der Standardeinstellung verändert und getestet. Verbesserten aber mehrere Parameter das Ergebnis wurden diese ebenfalls in Kombination justiert.
Für die Klassifikatoren SMO und Naive Bayes konnten keinerlei Optimierungen erreicht werden und auch die Verbesserungen bei der Klassifikation durch den Random Decision Forest waren so minimal, dass diese nicht signifikant ausfielen. So zum Beispiel stieg der F-Score bei Festlegung der Feature-Anzahl auf 16 von 0,921 auf 0,925 an. Wurde die Menge an Bäumen von 100 auf 400 erhöht, konnte ebenfalls ein Performance-Anstieg auf 0,924 verzeichnet werden, jedoch nahm die Evaluierung auch 10-mal so viel Zeit in Anspruch. Durch das Setzen des Limits für die maximale Baumtiefe auf 44 wurde sogar ein F-Messwert von 0,926 erzielt, doch auch dies stellte keine signifikante Verbesserung dar. Die Kombinationen aus diesen Einstellungen verschlechterte das Resultat.

Vergleich mit Related Work-Resultaten

In diesem Kapitel soll ein Vergleich der erzielten Ergebnisse mit jenen aus ähnlichen Studien für den englischen Sprachraum angestellt werden. Im Detail wurden dafür die Arbeiten von Warner et al. [40], Nobata et al. [39] und Zia et al. [44] herangezogen. Grund für genau diese Auswahl ist, dass Warner's et al. Studie den Grundstein, speziell für die Hassrede-Erkennung legten, Nobata et al. mit ihrer Arbeit den Stand der Technik in ihrem Algorithmus umsetzten und Zia et al. die neueste Forschung auf dem Gebiet der Hassrede Identifikation gegenüber anderen Religionen und ethnischen Gruppen, eine Art von Hass, die in dieser Arbeit einen Schwerpunkt einnimmt, veröffentlichten. Tabelle 4.37 stellt die einzelnen besten Resultate der Kennzahlen aus den Studien mit dem besten Klassifikations-Ergebnis aus dieser Arbeit gegenüber. Dabei ist ersichtlich, dass sich die Klassifikitaionsresultate im Laufe der Zeit stetig verbessert haben, wobei der Vergleich etwas hinkt, da sich die Identifikationsziele der Machine Learning Algorithmen auf Hass gegenüber spezifischen Gruppen beschränken, die Datenquelle eine andere ist und allen voran kein gemeinsamer Referenzkorpus für die Evaluierung herangezogen wurde. Stellt man die Studien trotzdem gegenüber wäre der im Zuge dieser Arbeit erarbeitete Algorithmus mit einem F-Score von 0,939 an der Spitze anzusiedeln. Außerdem kann festgestellt werden, dass bei Studien, die mehrere Classifier testeten, die Support Vector Machine als leistungsfähigster Classifier hervorging. [41] [44]

	Vergleich mit ähnlichen Studien					
	Classifier	Identifikationsziel	Datenquelle	P	R	F
Warner et al., Juni 2012	SVM	Antisemitismus	Yahoo, offensive Webseiten	0,68	0,6	0,63
Nobata et al., April 2016	keine Angabe	Alle Arten von Hassrede	Yahoo	0.794	0.773	0.783
Zia et al., November 2016	SVM	Anti-Religion (Christentum, Islam, Judentum)	Twitter	∅ 0,944	∅ 0,944	∅ 0,944
Referenzwerte	SVM(SMO)	Hassposting 3.5.3	Twitter	0,94	0,939	0,939

Tabelle 4.37: Vergleich mit Related Work-Resultaten

4.5 Qualitative Evaluierung

Neben der bisherigen quantitativen Analyse soll der Algorithmus auch qualitativ untersucht werden. Dafür werden die False Positive beziehungsweise False Negative Klassifikationen, die im Rahmen des besten Ansatzes[1] für die Hassposting-Identifikation gemacht werden, genauer analysiert. Gleichzeitig wird versucht die Gründe aufzudecken, die dafür verantwortlich sind, dass Tweets in die falsche Klasse eingeteilt wurden. Diese Erkenntnisse können in weiterführenden Arbeiten zu einem möglicherweise noch besseren Algorithmus führen. Der SMO-Classifier mit dem besten Feature-Set, bestehend aus Message-, Character-, Typed Dependency- und LIWC-Features beziehungsweise Merkmalen die nach der CFS-Subset Evaluierung aufgenommen wurden, klassifiziert insgesamt 170 Instanzen falsch, davon sind 116 False Negative und 54 False Postive Klassifikationen. Nachfolgend werden die interessantesten False Positive Fehlklassifikationen, die sich negativ auf die Precision auswirkt, aufgelistet:

@KidsPaper: "Wie heißt denn ihre Maus?" "Ratten sie mal!" (1)

Dieser Tweet beinhaltet einen Witz, der auch das Schimpfwort "Ratte" enthält, wodurch es dem Algorithmus schwer fiel diesen als neutral einzuordnen, obwohl sonst keinerlei Wörter aus dem typischen Hassredekontext vorkommen.

Jene,die nun in Überflutungsgebieten plündern und rauben sind der Abschaum der Gesellschaft. Menschen in Not berauben ist das Allerletzte! (2)

Dieser Tweet ist sehr schwer für den Classifier einzustufen, da sowohl ein Hasswort wie "Abschaum" vorkommt, als auch negativ behaftete Wörter, wie "plündern" oder "berauben". Dennoch handelt es sich hier um kein Hassposting, welcher Umstand jedoch nicht richtig erkannt wird.

[1] SMO mit Features aus CFS-Subset (ausgew. Features)

@kuehne_f Ok, in den 80ern haben wir noch "Neger" gesagt, wussten aber von Astrid Lindgren, Mark Twain und co., wie absurd Vorurteile sind! (3)

Die Bezeichnung "Dreckspack" ist doch zumindest denkenswert justitiabel, oder? Frage für einen Freund. https://t.co/DDjGdOyLyz (4)

Diese Tweets sind typische Beispiele für die Verwendung von Schimpfwörtern, bei denen nicht die Absicht besteht jemanden zu verunglimpfen, sondern um einen Umstand aufzuzeigen. Dies wird oft durch das Setzen des Schimpfwortes in Hochkommata zum Ausdruck gebracht. Trotz des Features bei der die Wörter in Hochkomma je Posting gezählt werden, wurden diese beiden Postings aufgrund dieser extrem einschlägigen Terme fälschlicherweise als Hasspostings identifiziert.

@saraxliont: @diese_ramona Find das halt echt nicht gut, sowas ist einfach nur asozial https://t.co/pk4GlFQDc0 (5)

Auch dieser Tweet bekundet nur eine Meinung, wird jedoch aufgrund des Terms "asozial", welcher oft im Zusammenhang mit Flüchtlingen in Hasspostings verwendet wird, falsch klassifiziert.

Heute ist wieder so ein Tag, an dem ich gerne meinen Berufswunsch als Kameltreiber in der Sahara verwirklichen würde... #ichwillmeineRuhe (6)

In diesem Video zeigt euch Musel wie man eine Gartenteich Brücke selbst baut. #Dzango #Zuschauervideo https://t.co/2eHuyd9aZI (7)

Der erste Tweet beinhaltet zwar das einschlägige Wort "Kameltreiber", das sehr oft im Zusammenhang mit syrischen Flüchtlingen verwendet wird, um diese zu verunglimpfen beziehungsweise zu verspotten, wird hier aber in einem völlig anderem Kontext eingesetzt. Speziell dieser Term wird jedoch eher selten in einem völlig neutralem Posting vorkommen. Im zweiten Tweet stellt der Term "Musel" einen Spitznamen dar. Üblicherweise werden damit jedoch Moslems herabgewürdigt. Dieser Spezialfall kommt mit Sicherheit auch äußerst selten vor und wird für den Classifier immer sehr schwer zu klassifizieren sein.

@ZDF Grenzen dicht, Kontrollen. Kontrollen! (8)

#Merkelmussweg Verbrechen: Wie Extremisten und Kriminelle um Flüchtlinge werben https://t.co/M1bWyal6u5 via @SPIEGELONLINE (9)

Diese Postings behandeln Themen, die oft Hass beinhalten und sind zusätzlich eher kontrovers zu betrachten, können jedoch aufgrund der Kriterien nicht als Hassposting

angesehen werden. Für den Algorithmus sind diese Art von Beiträge deshalb schwer zu differenzieren.

> *Würde am liebsten jeden, der die Flüchtlinge abschieben will, für paar Monate nach Syrien schicken. Ist ja anscheinend nicht so schlimm da?* (10)

Der Tweet enthält negativ behaftete Phrasen wie "abschieben" oder "nach Syrien schicken", die jedoch nicht im sonst üblichen Kontext eingesetzt sind, da sie sich nicht auf die Migranten beziehen. Trotzdem ist dieses Posting ebenfalls eher grenzwertig anzusehen, wurde jedoch dennoch als neutral kategorisiert, da weder Personengruppen verunglimpft oder beleidigt wurden, noch zu Gewalt gegenüber jemanden aufgerufen wird. Der Text wurde eher so bewertet, dass der Verfasser mithilfe von Sarkasmus versucht, dass sich in ihrer Meinung voreingenommene Personen in die Lage der Flüchtlinge versetzen sollen.

> *Ich nehme das Wort "Schwuchtel" mittlerweile als Kompliment. Ich bin gerne eine Schwuchtel, besser als ein Arschloch zu sein?* (11)

Dieser Beitrag gilt als typisches Beispiel, für jene Art von Tweets, die dem Klassifikations-Algorithmus die Grenzen aufzeigen. Trotz der Verwendung von starken Hasswörtern handelt es sich hierbei um kein Hassposting, eher versucht der Verfasser mit scheinbarer gespielter Gleichgültigkeit und Sarkasmus seinen Unmut über seine häufig vorkommende Verunglimpfung kundzutun. Die korrekte Klasse eines solchen Tweets ist äußerst schwer zu erkennen.

Die erkenntnisreichsten False Negative Fehlklassifikationen, die sich negativ auf den Recall auswirken, werden im nachfolgenden Abschnitt behandelt:

> *@Ralf_57 - Wer boarisch net versteht, dem konn i a net helfa. Mia brauch ma de Zuagroasten net. Vom Gsindl hamma täglich mehr.* (12)

> *@HCStracheFP wohl eher zutreffend, allerdings nicht besonders diplomatisch, aber orban ist nich immer besser als putin, russisches gsindl* (13)

Der zuerst angeführte Tweet wurde gänzlich in Mundart verfasst, wodurch Wörter enthalten sind, die einzigartig sind und deshalb zum Beispiel im Rahmen von Bag-of-Word-Features irrelevant werden. Der Term "Gsindl" als Abwandlung von "Gesindel" ist zwar im Hasswortlexikon enthalten, jedoch scheint dieser Umstand trotzdem zu wenig Information zu bieten, um die beiden Postings in die Hassposting-Klasse einzustufen.

> *@WillRobar Man darf doch noch auf mangelhafte Ortographie hinweisen, oder? Sind alle #AFD'ler so asozial wie dieser? https://t.co/KyzI9gjsZ3* (14)

@ProSachsen Menschen wie Sie sind der Abschaum, der zu nichts nutze ist. Sieht #AFD-Sympathisanten aber ähnlich. https://t.co/DFqdfNuF1g (15)

Der erste Tweet enthält das Hasswort "asozial" welches sich auch auf eine Personengruppe mit bestimmter politischer Gesinnung bezieht und die Aussage damit zur Hassrede wird. Dieses Wort kommt in dem annotierten Korpus jedoch nicht zwangsweise nur in negativen Postings vor, welcher Umstand, zusammen mit der Tatsache, dass der erste Satz keinerlei sonstigen Hass oder Beleidigungen enthält, den Algorithmus möglicherweise zu dieser Fehlklassifikation veranlasst. Ähnlich verhält es sich mit dem zweiten Tweet. Auffällig ist hier, dass in beiden Fällen Beleidigungen, aufgrund einer möglichen linksextremen Einstellung der Verfasser, vorliegen, dieser Bereich der Hasserkennung also noch verbesserungswürdig ist.

Kastration bei Pädophilie : Ja das ist Richtig, soll man in Deutschland auch machen, dann würden wir die Grünen, SPD & Beamten los (16)

Dieser Tweet besteht aus zwei Teilen, wobei die Beleidigung im ersten Teil passiert und sich spitzfindig auf politische Parteien beziehungsweise Beamte bezieht. Für den Algorithmus ist so ein Sachverhalt trotz Typed Dependency Features, die diesen Zusammenhang jedoch auch nicht erkennen, schwierig.

haha was für genozid es war nur ein prank (17)

Der armenische Genozid ist eine Lüge! Hitler und Himmler real! #Soyk?r?mYalan?naDURde (18)

Die Verleugnung von anerkannten Genozid ist schwer zu erkennen, da meistens keinerlei Hasswörter enthalten sind, die betroffenen ethnischen Gruppen aber dadurch massiv angegriffen werden. Durch die Aufnahme des Wortes "Genozid" wurde zwar versucht den Algorithmus dahingehend zu sensibilisieren, durch die überwiegende Verwendung von Genozid in Nachrichten- und neutralen Tweets reicht diese Maßnahme jedoch oft nicht aus.

Danke Fr.#Merkel für die #Terroristen die sie nach #Deutschland eingeladen haben. Wollen sie das #Deutsche sterben? Legen sie ihr Amt nieder (19)

Diese Aussagen verallgemeinern und stellen Flüchtlinge als Terroristen dar. Durch diese etwas versteckte Verunglimpfung und dem zusätzlich angewandten Sarkasmus ist dieser Text schwer zu klassifizieren.

RT @schlvrws: @B3RKE_S Ich bereite dann mal die Gaskammer vor.. (20)

Obwohl hier der Zusammenhang etwas fehlt, kann hier aufgrund des einschlägigen Wortes "Gaskammer" von einem Hassposting ausgegangen werden. Für den Klassifikations-Algorithmus ist dies jedoch nicht so einfach zu erkennen.

#Edathy lebt ja immer noch. Dachte er hat sich endlich umgebracht o wurde von guten Leuten wenigstens Kastriert. #PädoSchwein #Spon (21)

Erdowahn will Merkelekel absetzen ! Ist er etwa doch ein Freund Europa's ? (22)

Das erste Hassposting, obwohl eindeutig, wurde nicht erkannt. Grund dafür könnte sein, dass sich nur "umbringen" im Hasswortlexikon befindet. Ein Lemmatizer der für Typed Dependencies eingesetzt wird und das Wort auf seine ursprüngliche Grundform zurückführt, könnte hier Abhilfe liefern. Auch die Bezeichnung "Pädoschwein" wird nicht im Rahmen des LIWC-Wörtebuchs identifiziert. Jedoch wird in diesem Fall in diesem Term ein Hasswort erkannt, da für Features, bei denen das Hasswortlexikon zum Einsatz kommt, eine inexakte Übereinstimmung ausreicht und "Schwein" im Lexikon beinhaltet ist. Auch der zweite Tweet enthält eine eigenwillige Schimpfwort-Kreation, nämlich "Merkelekel", welche ebenfalls vom Algorithmus nicht als Hasswort eingestuft wird und zusätzlich ungebräuchlich ist.

@timsabri87 @FLER dan lern halt erstmal wie mann ein twet macht du duma huso (23)

Dieser Tweet enthält ausgesprochen viele Rechtschreibfehler, wodurch auch einzigartige Wörter entstehen, die daraufhin nicht berücksichtigt werden. Das Mistake-Feature, welches die Anzahl der Rechtschreibfehler in den Klassifikationsprozess einbringt, wurde genau für solche Tweets konzipiert. Da im Laufe der Evaluierung festgestellt wurde, dass die Verteilung der Fehler im Bezug auf die Klassen gleichmäßig ist, konnte dieses Feature trotzdem nicht die nötige zusätzliche Information für den Classifier liefern.

RT @BECKENBOERG: @flaneur_entmt und älter als das internet, sag ich ja, kreativ wie ein Ziegel. Und jetzt verpiss dich irrelevanter Hurenso... (24)

Aufgepasst liebe Surensöhne und Gutmenschen ??? https://t.co/VIOmIepsiv (25)

Werden Hasswörter wie "Hurenso" abgekürzt oder wie "Surensöhne" absichtlich falsch geschrieben, lässt sich das richtige Wort von Menschen leicht erahnen. Vom Machine Learning Algorithmus werden solche Fälle im Normalfall durch das Character N-Gram Feature abgedeckt.

DIE BLÖDHEIT DER DEUTSCHEN SPRENGT JEDEN RAHMEN Mein Rücken macht mir zu schaffen. Sehr. Aber ich hatte 40 qm... https://t.co/STDc6bZioM (26)

Dieser Tweet liefert einen äußert ungewöhnlichen Ausnahmefall. Neben der Hassrede im ersten Satz, setzt sich der Tweet aus einem völlig unabhängigen neutralen zweiten Satz zusammen, wodurch des Posting als neutral eingestuft wurde.

Erdowahn ist gegen Verhütung. Ihn hätte man besser mal verhüten sollen. (27)

Schwimm,#Flüchtling,schwimm! :-) lol https://t.co/9FKKFdlsYZ (28)

Diese Beiträge enthalten bis auf die Eigenkreation "Erdowahn" keinerlei hassassozierende Wörter, sind jedoch aufgrund ihrer Aussage auf jeden Fall als Hasspostings zu klassifizieren. Dieser Umstand macht es für den Classifier jedoch ungleich schwerer, Hass oder Beleidigungen zu identifizieren.

Wir sollten auf allen Ebenen versuchen, die Islamisierung zu stoppen! Jeder mit seinen Möglichkeiten! (29)

Allein schon dafür gehört sie eingesperrt! #MerkelMussWeg #DemDeutschenVolke https://t.co/W8j1jN6nkN via @HuffPostDE (30)

Wenn das in Deutschland passiert, ist die deutsche Bevölkerung Freiwild.Da kann ich nur noch zur Selbstjustiz raten. https://t.co/xKiXn9Zcdu (31)

Auch in diesen Tweets werden keine Hasswörter verwendet, wodurch der Aufruf zu Gewalt subtiler verpackt ist.

#Linke ich will eine arische Religion ohne Islamisierung (32)

Dieser Tweet enthält ebenfalls Hasswörter aus dem Lexikon, die Phrase "ohne Islamisierung", welche die Ausgrenzung zum Ausdruck bringt, sollte jedoch durch Typed Dependency- und BoW N-gram Features in den Klassifikationsprozess einfließen und die Entscheidung erleichtern. Hier konnte die Einteilung nicht korrekt getroffen werden, eine zusätzliche Aufnahme des Wortes "arisch", wäre zu überlegen.

Pseudo-Wahrheiten über den IS https://t.co/bYpV2OJUZ0 via @zeitonline- Dumm bis tödlich wäre es, in den sog. Flüchtlingen die guten zu sehen (33)

In diesem Tweet findet keine direkte Hassrede statt sondern wird darin gut versteckt formuliert. In diesem, wie in vielen anderen Hasspostings, fällt auf, dass hasserfüllte Botschaften emotionslos formuliert werden und Argumente regelrecht selbstverständlich, rational und logisch vorgebracht werden. Dieser Umstand erschwert zusätzlich die Identifikation von Hasspostings.

4.6 Kosten-Nutzen-Analyse

In diesem Kapitel wird, wie bereits im Abschnitt "Asymmetrische Kostenanalyse"3.3 angekündigt, mit den ermittelten Kosten für FP und FN der optimale Threshold je Klassifikator kalkuliert. Basis dafür sind jene in Abschnitt 4.4.6 evaluierten Feature-Sets, welche im Bezug auf den jeweiligen Classifier das beste Klassifikationsresultat geliefert haben.

Support Vektor Machine - SMO
Für den SMO-Algorithmus muss eine logistische Funktion, in Weka "Logistic Model" genannt, herangezogen werden, um den einzelnen Entscheidungen Wahrscheinlichkeitswerte zuzuordnen. Dies ist notwendig, da der Standard SVM-Algorithmus nicht probabilistisch ist und bekanntermaßen die optimale Grenze, welche der Hyperplane entspricht, selbst festlegt. Ein Anpassen des Thresholds wäre hier also andernfalls nicht möglich.

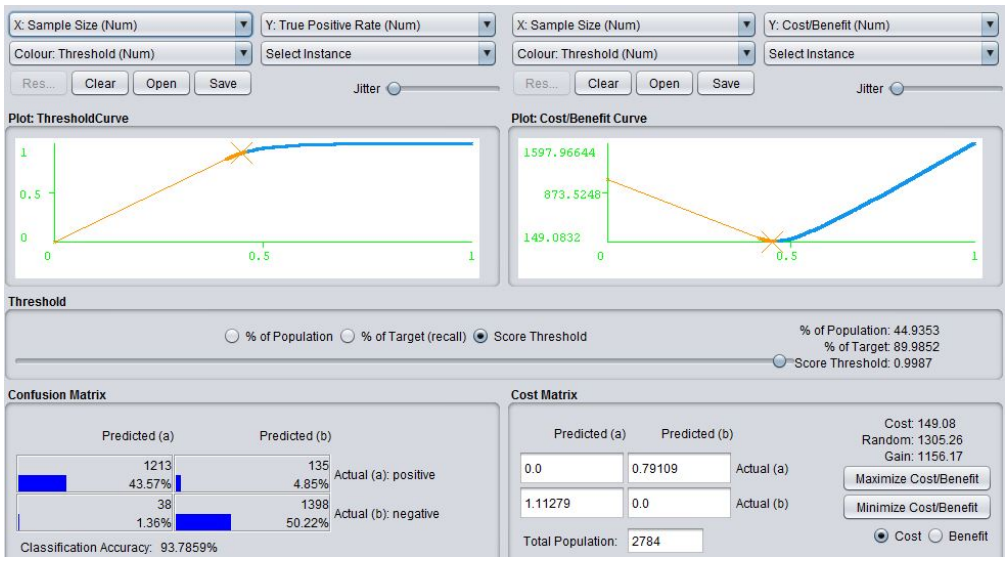


Abbildung 4.30: Weka Kosten/Nutzen Analyse für eine automatische Filterung und einem SMO Classifier.

Abbildung 4.30 stellt die Ergebnisse der Kosten-Nutzen Analyse in Weka für einen automatischen Ansatz dar. Der optimale Threshold bei asymmetrischen Kosten liegt bei rund 0,9987, das heißt ab einer Wahrscheinlichkeit größer als 99,87% wird ein Beitrag als Hassposting eingestuft. Bei diesem Grenzwert beträgt die Anzahl an Instanzen, die als

Hassbeitrag klassifiziert wurden, 44,9353% und der Recall für Hasspostings 89,9852%. Damit diese Werte berechnet werden konnten, wurden die in Abschnitt 3.3 ermittelten Kosten für FN(0,79109¢) und FP(1,11279¢) in die Kostenmatrix, die sich in der Abbildung rechts unten befindet, eingetragen. Links unten in der Abbildung ist die Confusion-Matrix platziert, aus der die genauen Zahlen für TP, TN, FP und FN entnommen werden können. Zusätzlich ist dort auch die bei diesem Threshold erzielte Klassifikations-Accuracy angeführt, die hier bei 93,7859% liegt und nur minimal schlechter als die evaluierte optimale Accuracy von 93.8937% ist.

Das Diagramm links oben gibt weiters das Verhältnis zwischen TP-Rate und Anzahl der Instanzen, die als Hassbeitrag deklariert wurden, bei einem bestimmten Threshold an. Der mit "X" markierte Koordinatenpunkt stellt den berechneten Grenzwerts mit minimalen Kosten dar. Die Farbe dieser sogenannten Threshold-Kurve ist dabei abhängig, wie viele Instanzen der jeweiligen Klasse zugeordnet werden. Gelb gibt an, dass die Mehrheit in diesem Wahrscheinlichkeitsbereich als Hassposting klassifiziert wurde. Desto "blauer" die Farbe der Kurve wird, desto mehr Instanzen wurden als neutral eingestuft und umso niedriger wird der Wahrscheinlichkeitswert, dass es sich um ein Hassposting handelt.

Das rechte Diagramm der Abbildung stellt die angefallenen Kosten und den prozentualen Anteil an klassifizierten Hasspostings, bei einem bestimmten Grenzwert, gegenüber, wobei die einzelnen Werte des Thresholds die im Diagramm visualisierte Kurve ergeben. Daraus ist abzulesen, dass die minimalen Kosten bei 2784 klassifizierten Instanzen bei *149,0832*¢ oder rund *1,5* US-Dollar liegen. Würde man nun diese Menge an FN- und FP-Raten für eine Hochrechnung beibehalten, würden sich bei einer Anzahl von rund 9 Millionen Hasspostings bei insgesamt 500 Millionen täglich abgesetzten Tweets die Kosten auf 151.715$ belaufen. Zieht man die Werte für False Positives(1,94% bei 54 FP) und False Negatives(4,17% bei 116 FN) aus Abschnitt 4.4.6 heran, die bei ungewichtetem Standard Grenzwert auftreten, summieren sich diese täglichen Gesamtkosten auf 211.590$. Die Kosten könnten durch die Veränderung des Grenzwerts also um rund 60.000$ oder 28% verringert werden.

Bei einem semi-automatischen Ansatz würde der Grenzwert von 0.99999998464 bereits beinahe bei 1 liegen, die Wahrscheinlichkeit, dass es sich um ein Hassposting handelt, um es als solches einzustufen, müsste aufgrund der hohen Kosten von 3,95544¢ für FP-Fehlklassifikationen schon fast 100% betragen. Es werden deshalb auch nur 43,4267% der Instanzen im Korpus als Hassbeiträge bewertet, wodurch auch der Recall auf 87,6113% im Vergleich zu einem automatischen Ansatz sinkt. Die anfallenden minimalen Kosten für alle Instanzen aus dem Korpus, die sich aus 167 FN und 28 FP ergeben, sind bei dieser Methode zusätzlich um mehr als einen US-Dollar höher und betragen 284,22¢ oder rund 2,8 US-Dollar. Ein Umstand der bei dieser kleinen Menge an Daten zwar gering aussieht, rechnet man dies jedoch auf die Anzahl an täglichen Postings(500 Mio.) hoch, fallen bereits Summen von rund 700.000$ an. Abbildung 4.31 zeigt die Weka Analyse und beschriebenen Ergebnisse zusammengefasst.

Naive Bayes

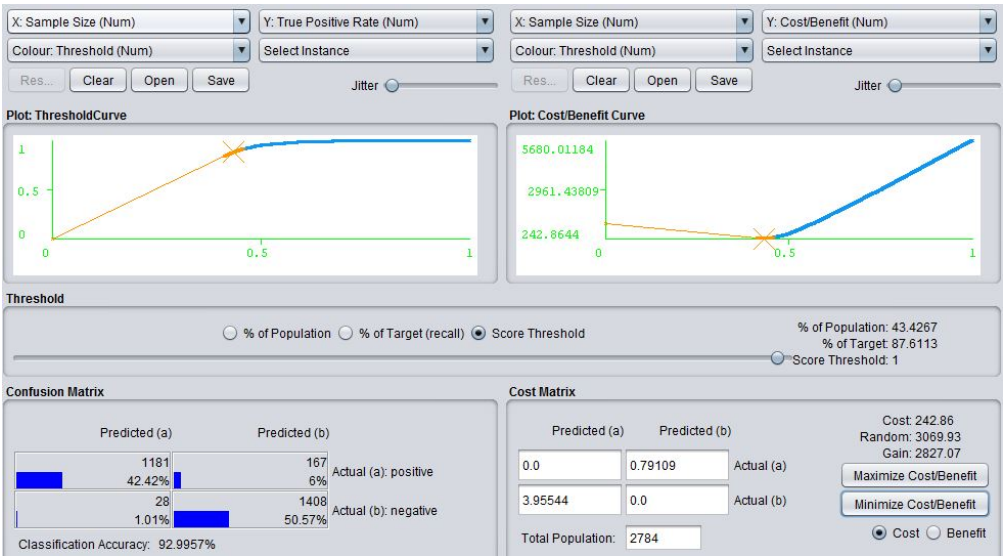


Abbildung 4.31: Weka Kosten/Nutzen Analyse für eine semi-automatische Filterung und einem SMO Classifier.

Bei der Kalkulation des besten Thresholds für den Naive Bayes Klassifikator und dessen optimalen Feature-Sets mussten keine Parameterwerte des Classifiers verändert werden, da es der Natur des Naive Bayes entspricht Wahrscheinlichkeitswerte je Instanz zu berechnen. Die evaluierten Werte sind gesammelt den Tabellen 4.38, 4.39 und 4.40 zu entnehmen.

Random Decision Forest
Auch der Random Decision Forest benötigt keinerlei Anpassungen, da dieser durch verschiedene Decision Trees aufgebaut wird. Diese wiederum weisen jedem ihrer Blätter eine gewisse Klasse und eine korrespondierende Wahrscheinlichkeit zu, beziehungsweise basieren generell auf einer Wahrscheinlichkeitsverteilung. Die Resultate der Kosten-Nutzen Analyse sind ebenfalls in den nachfolgenden Tabellen enthalten.

Zusammenfassung
Analysiert man die Werte fällt auf, dass sich die teils massiven Änderungen des Thresholds nur gering verschlechternd auf die Accuracy auswirken, die Kosten aber trotzdem gesenkt werden können. Der Recall der Hassposting-Klasse verändert sich jedoch schon stärker ins Negative. Das Verhältnis der Anzahl von FN zu FP verändert sich natürlich auch in Richtung FN bei höheren Kosten von FP-Klassifikationen. Für die Kalkulation der täglichen Kosten wurde wie in Abschnitt 3.3 angenommen, dass 9 Millionen der täglichen Tweets einem Hassposting(HP) entsprechen und folgende Gleichungen dafür aufgestellt, wobei NP für neutrale Postings steht:

$$Kosten_{ungewichtet} = (FN\_Rate \cdot Anz.\ HP + FP\_Rate \cdot Anz.\ NP) \cdot 1¢ \qquad (4.9)$$

$$Kosten_{auto} = FN\_Rate \cdot Anz.\ HP \cdot Kosten\ FN + \\ + FP\_Rate \cdot Anz.\ NP \cdot Kosten\ FP \tag{4.10}$$

$$Kosten_{semi} = FN\_Rate \cdot Anz.\ HP \cdot Kosten\ FN + \\ + FP\_Rate \cdot Anz.\ NP \cdot Kosten\ FP + \\ + TP\_Rate \cdot Anz.\ HP \cdot Kosten\ TP \tag{4.11}$$

Vergleicht man nun die täglichen Kosten je Classifier und Methode fällt auf, dass Kosten für einen semi-automatisierten Ansatz aufgrund der Personalkosten doch sehr hoch sind und jedes gefundene Hassposting nochmals evaluiert werden muss. So zum Beispiel würden für den SMO-Classifier rund 700.000$ anfallen. Der RDF-Classifier verursacht zwar "nur" 560.000$, der Grund dafür ist jedoch, dass nicht so viele Hasspostings erkannt werden. Der automatische Ansatz mit rund 150.000$ Kosten täglich und einem SMO-Classifier arbeitet hier schon viel kosteneffizienter. Folgekosten aufgrund unrechtmäßiger Filterung können aber ebenfalls hoch sein. Abhilfe könnte hier ein Hybrid-Modell mittels logistischer Regression bieten, bei dem Beiträge, welche mit nahezu 100%er Sicherheit Hass beinhalten, automatisch gefiltert werden und Tweets, die in einem gewissen Graubereich liegen, in dem es unsicher ist, ob es sich tatsächlich um Hasspostings handelt, zusätzlich von Personen manuell untersucht werden. Weiters ist anzumerken, dass der Grenzwert hier für den annotierten Korpus, bei dem Hass- und neutrale Postings zu ungefähr gleichen Teilen vorhanden sind, optimiert worden ist, dieser Grenzwert also im Praxiseinsatz an das wahre Verhältnis angepasst werden kann, um die Kosten zu minimieren. Die hochgerechneten täglichen Kosten in diesen Tabellen können also auch nur mit Vorbehalt betrachtet werden und sollen einen gewissen Richtwert liefern.

Kosten-Nutzen Analyse Standard-Gewichtung		**SMO**	**NB**	**RDF**
Accuracy(%)		93,93	88,00	92,64
Threshold		0,32164	0,54158	0,44
Racall HP(%)		91,47	83,61	90,36
HP-Klassifikationen(%)		46,23	44,54	46,44
FN		115	221	130
FP		54	113	75
FN-Rate		0,08531	0,16395	0,09644
FP-Rate		0,03760	0,07869	0,05223
minimale Kosten (FN & FP je 1¢)	Korpusinstanzen($)	1,69	3,34	2,05
	täglich($)	192.316	401.127	265.121

Tabelle 4.38: Kennzahlen der durchgeführten Kosten-Nutzen Analyse bei Kostengleichheit von FN und FP und einem Einheitswert (1¢) für Fehlklassifikationen

Kosten-Nutzen Analyse automatische Filterung		**SMO**	**NB**	**RDF**
Accuracy(%)		93,79	87,2847	92,57
Threshold		0,99874	0,92163	0,48
Racall HP(%)		89,98	77,89	88,72
HP-Klassifikationen(%)		44,94	39,73	44,94
FN (0.79109¢)		135	298	152
FP (1.11279¢)		38	56	55
FN-Rate		0,10015	0,22107	0,11276
FP-Rate		0,02646	0,039	0,0383
minimale Kosten	Korpusinstanzen($)	1,4908	2,9806	1,8145
	täglich($)	151.715	228.813	209.268

Tabelle 4.39: Kennzahlen der durchgeführten Kosten-Nutzen Analyse bei automatischem Filterungsansatz

4.7 Bias- und Performance-Analyse

In diesem Kapitel wird der sogennante Bias und die Performance des Trainingskorpus analysiert. Unter dem Bias versteht man jenen systematischen Fehler der Klassifikationsfunktion, der durch falsche Annahmen im Lernalgorithmus entsteht und dadurch nicht

Kosten-Nutzen Analyse halb-automatische Filterung		**SMO**	**NB**	**RDF**
Accuracy(%)		93,64	86,42	90,45
Threshold		0.99999998	0.98794	0.61
Racall HP(%)		87,61	73,89	81,68
HP-Klassifikationen(%)		43,43	37,00	40,23
FN (0.79109¢)		167	352	247
FP (3.95544¢)		28	34	19
FN-Rate		0,12389	0,27596	0,18323
FP-Rate		0,01950	0,02089	0,01323
TP-Rate		0,87611	0,73887	0,81677
minimale Kosten	Korpusinstanzen($)	2,8422	4,1415	2,7131
	täglich($)	699.394	688.415	560.772

Tabelle 4.40: Kennzahlen der durchgeführten Kosten-Nutzen Analyse bei halb-automatischem Filterungsansatz

die richtigen Zusammenhänge zwischen Trainingsdaten und den annotierten Klassen modelliert werden. Im Gegensatz zur Varianz handelt es sich dabei um eine Unteranpassung. Generell stellt es das Ausmaß der Abweichung zwischen den Erwartungswerten und den tatsächlichen Werten der Trainingsdaten dar und wird auch Verzerrung genannt.
Über einen langen Zeitraum gesehen steigt der Bias natürlich, da sich die Themen beziehungsweise gesellschaftspolitische Situationen, auf die sich Hasspostings beziehen, ändern und sich mit dieser ständigen Änderung auch der Sprachgebrauch kontinuierlich angepasst wird. Als Beispiel kann angeführt werden, dass vor ein bis zwei Jahren noch der Flüchtlingsstrom und die von Hasspostern oft geforderte Grenzschließung im Vordergrund stand. Ein Jahr später nach dem die Massenbewegungen einigermaßen unter Kontrolle gebracht wurden, sprechen Beiträge, durch die Anschläge in einigen Städten Europas, im Zuge von Flüchtlingen oft von potentiellen "Gefährdern" und stellen die Ausweisung von Asylanten in den Vordergrund. Ebenso ist das Repertoire an Schimpf- und Hasswörten dynamisch und unerschöpflich, wodurch nie alle solche Wörter durch das Lexikon abgebildet werden können. Um nun das Bias und generell die Performance des Trainingskorpus zu analysieren, wurden Beiträge annotiert, die ein halbes Jahr nach dem Erstellungszeitpunkt des Korpus gepostet wurden. Im Detail stammen diese neu annotierten Tweets vom 12.01.2017 bis zum 08.02.2017, wobei der resultierende "Bias-Korpus" 2691 manuell klassifizierte Tweets enthält. Pro Woche wurden zufällig 250 neutrale und 250 hasserfüllte Postings ausgewählt, um die Kennzahlen aus den Tabellen 4.41 und 4.42 zu berechnen.

Um das Ergebnis vergleichbar zu machen, wurde ein zusätzliches Feature-Set für die

Bias-Analyse herangezogen. Bei der Frage welche Eigenschaften dieses Set aufweisen soll, wurden mehrere Überlegungen miteinbezogen. Grundsätzlich war eine Reduktion der Feature-Anzahl angedacht, wobei speziell Feature-Kategorien gestrichen wurden, welche die einzelnen verwendeten Wörter eines Tweets abdecken, wie beispielsweise BoW-Attribute. Damit soll eine veränderte Sprachwahl nicht zu viel Einfluss in die Klassifikation nehmen und die wirklich relevanten Begriffe(Hasswörter) im Rahmen von Hasspostings in den Vordergrund rücken. Die Wahl fiel schließlich auf ein Feature-Set, das die modifizierten LIWC-Features und das Netzwerkfeature, welches die bereits geposteten Hassbeiträge des Users abbilden, beinhaltet. Bag-of-Word, Character N-Gram und Typed Dependency- Features wurden bei dieser radikalen Feature Reduktion auf insgesamt 68 Merkmale komplett außen vor gelassen.

Bias-Varianz Zerlegung
Ergebnis der Bias-Varianz Zerlegung, welche mittels Weka für den am besten performenden Trainingskorpus und einem SMO-Classifier durchgeführt wurde, war folgendes(9/10 als Traininsset 1/10 als Testset für die Biasanalyse):

SMO BEST

- $GesamtError$: 0.0694
- $Sigma^2$: 0
- $Bias^2$: 0.0417
- $Varianz$: 0.0249

Zum Vergleich wurde das Feature-Set, bestehend aus LIWC- und einem Netzwerkfeature, nämlich Anzahl der bereits verfassten Hasspostings eines Users, der Bias-Varianz-Zerlegung unterzogen:

SMO LIWC+#HP

- $GesamtError$: 0.1119
- $Sigma^2$: 0
- $Bias^2$: 0.1032
- $Varianz$: 0.0078

Dabei ist die Unteranpassung durch den vergleichsweise hohen Bias-Wert gut ersichtlich. Die Varianz ist jedoch relativ gering, da aufgrund der generischen Ausprägung des Modells, dieses nicht so sehr auf größere Schwankungen beziehungsweise Ausreißer in den Feature-Werten reagiert. Führt man diese Bias-Varianz Zerlegung nun erneut durch,

verwendet jedoch als Testset die Instanzen aus dem Bias-Korpus verändern sich die Werte wie folgt:

SMO BEST

- $GesamtError$: 0.1236
- $Sigma^2$: 0
- $Bias^2$: 0.0845
- $Varianz$: 0.0352

Dabei ist ersichtlich, dass sich der Wert des Bias verdoppelt, die Varianz sich aber nur leicht nach oben bewegt und der Fehler insgesamt fast doppelt so hoch ist. Das Modell ist also unterangepasst und müsste mit aktuelleren Daten neu trainiert werden. Als Referenz wiederum die Ergebnisse, durch das zuvor beschriebene kleinere Feature-Set, wobei hier der gesamte Fehler durch die sehr geringe Varianz sogar kleiner ist:

SMO LIWC+#HP

- $GesamtError$: 0.1139
- $Sigma^2$: 0
- $Bias^2$: 0.1084
- $Varianz$: 0.005

Als Vergleich würde man Trainings- und Testset nur aus dem Bias-Korpus bilden, ergibt die Bias-Varianz Zerlegung wieder klar bessere Ergebnisse (9/10 als Traininsset 1/10 als Testset für die Biasanalyse): *SMO BEST*

- $GesamtError$: 0.092
- $Sigma^2$: 0
- $Bias^2$: 0.0649
- $Varianz$: 0.0244

SMO LIWC+#HP

- $GesamtError$: 0.0695
- $Sigma^2$: 0

- $Bias^2$: 0.069
- $Varianz$: 0.0004

Performanceanalyse
Nachfolgend wird gezeigt wie sich das erhöhte Bias auf die Performance Kennzahlen auswirkt. Tabelle 4.41 beinhaltet dabei die berechneten Kennzahlen für den SMO-Classifier mit jenem Feature-Set das durch die Evaluierung als bestes befunden wurde. Dabei fällt auf, dass die Performance bereits um einiges schlechter wurde. Betrug das F-Maß im Juni noch rund 0,939 so machte dies in der Woche 2 der Bias Analyse nur mehr rund 0,754 aus, welches einer absoluten Einbuße von 0,185 entspricht. Generell schwankte das F-Maß zwischen 0,754 und 0,817 und entsprach über die vier Wochen hinweggesehen einem durchschnittlichen F-Maß von 0,782. Es ist jedoch zu erwähnen, dass die Performanceeinbuße durch die große Anzahl an False-Negative Klassifikationen verursacht wird ($\tilde{4}00$) und es kaum zu False-Positive Klassifikationen ($\tilde{1}8$) gekommen ist.

Für den realen Einsatz in Twitter wäre dieser Umstand sogar rentabler, wobei die täglichen Kosten bei den im Abschnitt 4.6 getroffenen Annahmen und einem ungewichteten automatischen Ansatz im Detail bei 126899$ (FP 1,11279¢ FN 0,79109¢) beziehungsweise bei einem ungewichteten halb-automatischen Ansatz bei 378132$ (FP 3,95544¢ FN 0,79109¢) liegen würden. Im Vergleich dazu machten die täglichen Kosten im Juni der ungewichteten Ansätze, bei denen also der Schwellenwert für die Klassifikation nicht nach den Kosten angepasst wird, des besten evaluierten SMO-Classifiers (FP-Rate 0,0376/FN-Rate 0,08531) bei gleichen Kostenraten für Falschklassifikationen wie zuvor 211513$ (automatisch) beziehungsweise 736311$ (halb-automatisch) aus. Grund dafür sind die geringeren Kosten für FN. Andererseits ist das Ziel so viele Hasspostings wie möglich zu identifizieren, weshalb zwar ökonomische Kriterien erfüllt wären, jedoch nicht die ideellen Ziele erreicht werden. Der Korpus müsste deshalb zumindest vierteljährlich angepasst werden oder als bessere Alternative die Live-klassifizierten Beiträge immer wieder in den Korpus aufgenommen werden, um die entstandene Unteranpassung zu verhindern. Die Veränderung der Sprache beziehungsweise Gegebenheiten würden so wieder einbezogen werden.

Die Ergebnisse der Bias/Performance Analyse mit dem Feature-Set, bestehend aus LIWC-Attributen und der Anzahl bereits verfasster Hasspostings des Users, finden sich in Tabelle 4.42 wieder und fallen tatsächlich im Schnitt 10% besser aus, als jene der vorherigen Analyse. Bei einem Vergleich des Resultats der Feature-Kategorien Evaluierung bei der das erweiterte LIWC-Featureset einen F-Score von 0,859 erzielte, kann man feststellen, dass dieser Wert in jeder Woche des Analysezeitraums sogar übertroffen wird.

Die Kosten für Falschklassifikationen würden sich für diesen Zeitraum und dieses Feature-Set auf rund 360861$ (automatisch) beziehungsweise 1254136$ (halb-automatisch) belaufen. Das sind um einiges mehr an Kosten als jene die durch das zuvor betrachtete Feature-Set anfallen würden. Zusätzlich teurer als mit dem nach der Evaluierung als besten für den SMO Classifier befundenen Feature-Set, welches aus Message-, Character-,

Bias Analyse 12.01.2017 - 08.02.2017								
	Feature-Set	#Features	Klasse	TP	FP	P	R	F
SMO	CFS Subset-Evaluierung >=1 bei ausgew. Features	50817	**Woche 1**					
			Neutral	0,968	0,42	0,697	0,968	0,811
			Hass	0,58	0,032	0,948	0,58	0,72
			Avg.	0,774	0,226	0,823	0,774	0,765
			Woche 2					
			Neutral	0,984	0,452	0,685	0,984	0,808
			Hass	0,548	0,016	0,972	0,548	0,701
			Avg.	0,766	0,234	0,828	0,766	0,754
			Woche 3					
			Neutral	0,988	0,344	0,742	0,988	0,847
			Hass	0,656	0,012	0,982	0,656	0,787
			Avg.	0,822	0,178	0,862	0,822	0,817
			Woche 4					
			Neutral	0,988	0,388	0,718	0,988	0,832
			Hass	0,612	0,012	0,981	0,612	0,754
			Avg.	0,8	0,2	0,849	0,8	0,793

Tabelle 4.41: Resultate der Performance-Analyse des SMO mit der evaluierten besten Konfiguration für Daten vom 12.01.2017-08.02.2017

Typed Dependency- und LIWC-Features beziehungsweise Merkmalen die nach der CFS-Subset Evaluierung aufgenommen wurden, zusammengesetzt ist. Trotz der insgesamt besseren Klassifikation sind die Kosten hier um einiges höher, da die FP-Klassifikationen ansteigen. Das lässt darauf schließen, dass nicht nur das F-Maß eine gewisse Grenze erreichen soll, um ausreichend Hasspostings zu identifizieren, sondern gleichzeitig auch die FP-Rate einen gewissen Wert nicht übersteigen darf, damit die Kosten nicht exorbitant ansteigen. Grundsätzlichen ist schließlich ein F-Maß von über 90% die Zielsetzung und eine FP-Rate für die Hass-Klasse von unter 0,05 erforderlich, damit der Algorithmus effizient und effektiv eingesetzt werden kann. Tabelle 4.43 verschafft noch mal einen Überblick der Kostenschätzungen und wie diese berechnet wurden.

In den nachfolgenden Diagrammen 4.32 und 4.33 werden die Verläufe der Messwerte der Bias-Analysen abgebildet. Auffällig ist der Ausreißer nach oben in Woche 3. Grund für die bessere Performance waren vermehrte Hassbeiträge und Beleidigungen betreffend der Ablehnung gegenüber Flüchtlingen und der damit verbundenen Flüchtlingspolitik der deutschen Bundeskanzlerin Merkel, welche in ähnlicher Form auch zum Zeitpunkt der Korpuserstellung aktuell war. Als Folge dessen konnten die Beiträge darüber besser eingeordnet werden. Generell ist auffällig, dass die Sprache in Postings sehr abhängig von der Sprachwahl in aktuellen politischen Diskussionen ist. So zum Beispiel wurde in einer parlamentarischen Anfrage in Deutschland das Wort “Gefährder“ für Islamisten, Rechts- und Linksextreme gewählt, woraufhin viele Hassbeiträge in Anspielung darauf

Bias Analyse 12.01.2017 - 08.02.2017								
	Feature-Set	#Features	Klasse	TP	FP	P	R	F
SMO	LIWC + #HatePosts	68	**Woche 1**					
			Neutral	0,916	0,160	0,851	0,916	0,882
			Hass	0,84	0,084	0,909	0,84	0,873
			Avg.	0,878	0,122	0,880	0,878	0,878
			Woche 2					
			Neutral	0,972	0,196	0,832	0,972	0,897
			Hass	0,804	0,028	0,966	0,804	0,878
			Avg.	0,888	0,112	0,899	0,888	0,887
			Woche 3					
			Neutral	0,928	0,084	0,917	0,928	0,922
			Hass	0,916	0,072	0,927	0,916	0,922
			Avg.	0,922	0,078	0,922	0,922	0,922
			Woche 4					
			Neutral	0,928	0,188	0,832	0,928	0,877
			Hass	0,812	0,072	0,919	0,812	0,862
			Avg.	0,87	0,13	0,875	0,87	0,870

Tabelle 4.42: Resultate der Bias-Analyse des SMO mit LIWC und #Hasspostings der User für Daten vom 12.01.2017-08.02.2017

Gegenüberstellung der Kosten		SMO BEST CFS Subset Juni 2016	SMO BEST CFS Subset Jan./Feb. 2017	SMO LIWC + #HP Jan./Feb. 2017
FN-Rate (Avg.)		0,08531	0,401	0,157
FP-Rate (Avg.)		0,0376	0,018	0,064
# täglicher neutraler Tweets		491000000		
# täglicher Hass-Tweets		9000000		
Kosten / FP	automatisch	1,11279¢		
	halb-automatisch	3,95544¢		
Kosten / FN		0,79109¢		
Tägliche Kosten	automatisch	211.513$	126.899$	360.861$
	halb-automatisch	736.311$	378.132$	1.254.136$

Tabelle 4.43: Gegenüberstellung der Kosten - Basiskorpus Evaluierung vs. Biasanalyse-korpus Evaluierung

Flüchtlinge als Gefährder bezeichneten. Ähnlich verhielt es sich im Frühjahr 2016 als Siegmar Gabriel Rechtsextreme und Fremdenhasser als “Pack“ bezeichnete und dieses Wort in Hasspostings ebenfalls auf Politiker und Flüchtige umgelegt wurde. Zusätzlich

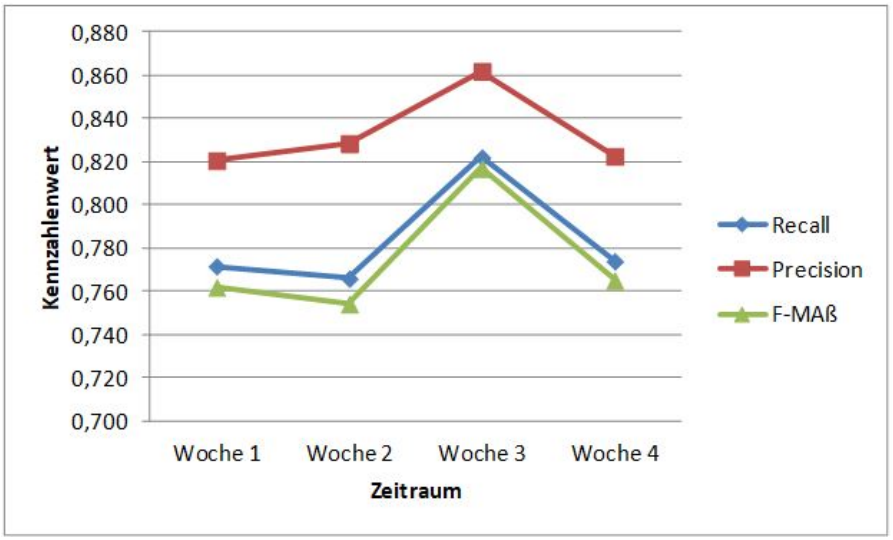


Abbildung 4.32: F-Score Verlauf des SMO Classifiers mit dem besten evaluierten Feature-Set.

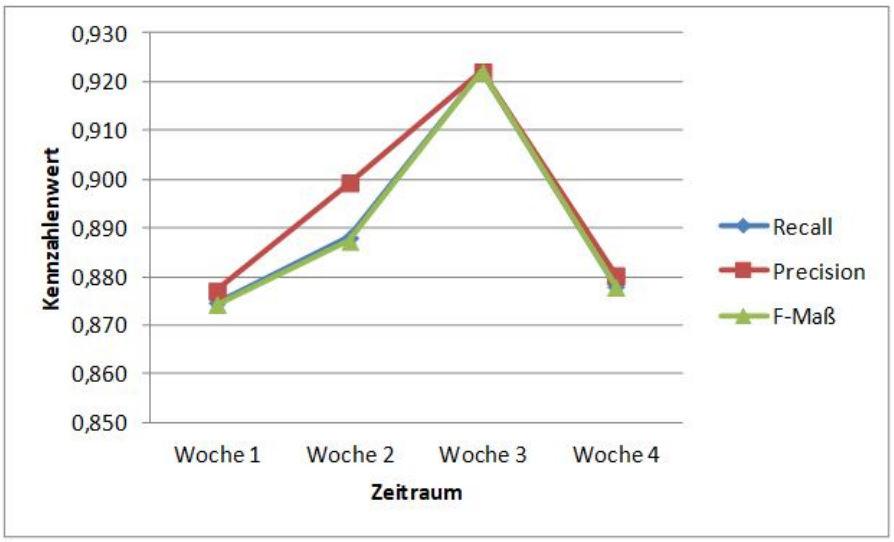


Abbildung 4.33: F-Score Verlauf des SMO Classifiers mit einem Feature-Set bestehend aus LIWC-Features und dem Feature über die Anzahl bereits verfasster Hasspostings.

der Verlauf des F-Scores in Diagramm 4.34 der beide Analysen gegenübergestellt, um den Unterschied noch einmal grafisch zu visualisieren.

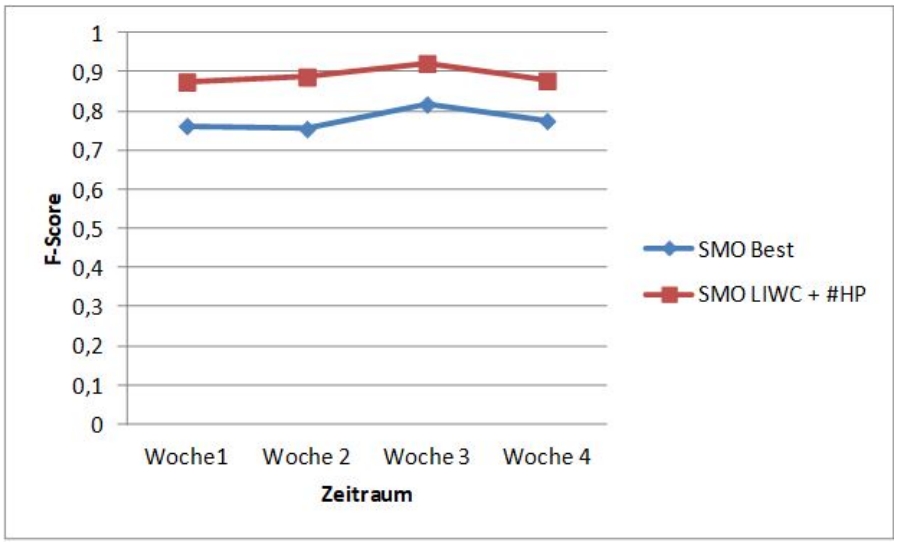


Abbildung 4.34: Gegenüberstellung der F-Scores der Klassifikationen des SMO-Classifiers mit den beiden evaluierten Feature-Sets.

4.8 Zusammenfassung

In dem vorangegangen Kapitel wurden die verschiedenen Evaluierungsmethoden für einzelne Features und Feature-Sets vorgestellt. Für einzelne Features wurde der Information Gain und χ^2-Wert berechnet, nach diesen geordnet und analysiert. Zusätzlich wurde eine CFS Subset Evaluierung durchgeführt, welche informative, nicht korrelierende Attribute kombiniert. Für die Feature-Kategorien wurden die Messwerte der Kennzahlen Precision, Recall und F-Measure für die drei Klassifikatoren ausgewertet. Die Features wurden auf verschiedene Arten kombiniert, zum Beispiel auf Basis der Kategorien, des Information Gains oder der CFS Subset Evaluierung. Bei der Kombination ganzer Attribut-Kategorien wurde zuvor eine Baseline, die im Endeffekt entweder aus Message- oder Character N-Grams bestehen konnte, aufgrund zuvor kalkulierter Messwerte je Classifier festgelegt. Als weiterer Indikator wurden die Zeiten für die Extraktion und Durchführung der 10-fachen Kreuzvalidierung für Feature-Kategorien und -Kombinationen gemessen und in Diagrammen aufbereitet. Als Abschluss wurden die besten Feature-Sets je Klassifikator und deren Parameter Einstellungen zusammengefasst. Während des Prozesses zur Findung der optimalen Feature-Kombination je Klassifikator wurden auch Signifikanztests durchgeführt, um vorhandene Verbesserungen auch auf Signifikanz zu untersuchten. Zusätzlich zur quantitativen wurde eine qualitative Evaluierung des Algorithmus durchgeführt, indem die False Positve und False Negative Klassifikationen untersucht wurden. Damit sollen auch die Gründe für deren Auftreten festgestellt werden. Die ankündigte Kosten-Nutzen Analyse wurde ebenfalls im Kapitel Methodik durchgeführt, um den optimalen Grenzwert für die Hassposting-Entscheidung zu ermitteln. Ein weiterer wichtiger Punkt des Kapitels war die Bias-Analyse des Trainings-Korpus, die Aufschluss über dessen Qualität gibt und

wie sich das Klassifikations-Ergebnis durch unterschiedlichste Einflüsse verändert hat.

KAPITEL 5

Deployment

In Übereinstimmung mit dem CRISP-DM Modell folgt auch in dieser Arbeit als letztes Kapitel das Deployment, welches die evaluierten Ergebnisse und Schritte die notwendig wären, um dieses Modell produktiv einzusetzen, noch einmal kurz zusammenfasst und vergegenwärtigt.[74]

Halb-Automatisch oder automatische Filterung
Damit die Notwendigkeiten für das Modell im Realeinsatz überhaupt diskutiert werden kann, muss zuvor festgelegt werden, wie bei der Filterung von Hasspostings grundsätzlich vorgegangen wird. Eine voll automatische Filterung wäre natürlich wünschenswert, da effizienter, schneller und bei entsprechender Genauigkeit kostengünstiger. Liegt die Accuracy wie im Zuge dieser Diplomarbeit jedoch bei maximal rund 94% würden bei 500 Mio. Tweets pro Tag falsch klassifiziert werden. Aufgrund der dazugehörigen FP-Rate von 0,038 und der FN-Rate von 0,086 wären das ausgehend von 9 Mio. Hasspostings pro Tag fast 19 Mio. (18658000) Beiträge die zu Unrecht gefiltert werden respektive rund 800000 (774000) Hasspostings, welche veröffentlicht werden und bei welchen auf eine Meldung durch den User gehofft werden muss. Diese Zahlen zeigen, dass dieses Modell also eher kritisch agiert und eine manuelle Begutachtung notwendig wird. Eine tägliche Filterung von 19 Mio. neutralen Postings, die zu enormen Reputationsverlusten, durch drohende Zensurvorwürfe und mangelnde Usability, führen, ist keinesfalls wirtschaftlich. Der Einsatz eines halb-automatischen Modells, welches Tweets nur vorfiltert, ist also trotz erhöhten Personalaufwands zu empfehlen.

Ethische Aspekte
Der Grad zwischen der Befreiung von diskriminierenden, rechtswidrigen oder beleidigenden Beiträgen auf sozialen Plattformen und der massiven Einschränkung der Meinungsfreiheit beziehungsweise Zensur ist ein schmaler, weshalb auch auf ethische Punkte eingegangen werden muss. Im Speziellen Beiträge, die sich im Graubereich zwischen der eigenen Meinung und strafbaren Aussagen bewegen, sind für Menschen und vor allem für Algorithmen schwer zuzuordnen.

Als Referenz, welche ethischen Fragen für einen halb- beziehungsweise automatischen Algorithmus zur Entscheidungsfindung bei Hasspostings beachtet werden müssen, wurde das Paper SStatement on Algorithmic Transparency and Accountability"des ACM US Public Councils herangezogen und auf die sieben darin enthaltenen Punkte bezugnehmend auf diese Arbeit eingegangen: [75]

1. **Bewusstsein:** Beteiligte Personen eines analytischen Systems müssen sich über möglichen Bias bewusst sein. Auch das in dieser Arbeit vorgestellte System unterliegt einem solchen. Einerseits wurde festgestellt, dass die Zeitdauer, über welche die Trainingsdaten gesammelt wurde, zu kurz war und anderseits, dass aufgrund der verschiedenen Methoden, wie neutrale und hasserfüllte Tweets gesammelt wurden, es zu einem Bias kommen konnte, da der Umfang an neutralen Postings für den Trainingskorpus im Kontext von spannungsgeladenen, oftmals hasserregenden Themen zu gering ausfiel.

2. **Zugriff und Entschädigung:** Nutzern muss eine Funktion geboten werden, Entscheidungen des Algorithmus zu hinterfragen und zu Unrecht gelöschte Beiträge wiederherstellen zu lassen. Im Falle von Twitter wäre das beispielsweise ein einfacher Button, ähnlich dem "Melden-Button", damit der gelöschte Tweet noch einmal eingehend von einem Mitarbeiter untersucht wird. Erklärungen, warum der Tweet gelöscht wurde, sollten jedoch nicht zu detailreich sein, da es für User sonst leichter wäre mithilfe dieser Informationen die Löschung durch spitzfindige Umformulierungen oder dergleichen zu umgehen.

3. **Rechenschaftspflicht:** Institutionen sollen für die Entscheidungen, die durch die Algorithmen getroffen werden, verantwortlich sein, gleichwohl nicht festzustellen ist, wie die Resultate im Detail produziert werden. Die Entscheidung, einen Algorithmus wie diesen einzusetzen, muss also bei dem jeweiligen Kunden liegen. Die in dieser Arbeit evaluierten Modelle und Herangehensweisen können maximal als Empfehlung angesehen werden, wobei die jeweiligen Stärken und Schwächen aufgezeigt werden, den Entscheidungsträgern also bewusst gemacht werden und in weiterer Folge als Entscheidungsgrundlage dienen.

4. **Erläuterung:** Systeme und Institutionen sind dazu angehalten Erläuterungen zur Funktionsweise des Algorithmus und seinen Entscheidungen abzugeben. Im Falle von Twitter könnte dies zusätzlich in den Unternehmensrichtlinien, innerhalb jenes Punktes, welcher den Inhalt von Tweets regelt, verankert beziehungsweise darauf hingewiesen werden, wobei wiederum darauf hingewiesen werden muss, dass der Detailgrad der Erklärung bezüglich der Entscheidungsfindung über Hass- oder neutrales Posting nicht zu hoch sein darf, da das System ansonsten korrumpiert werden könnte.

5. **Datenherkunft:** Eine Beschreibung des Vorgangs, auf welche Art und Weise die Trainingsdaten gesammelt wurden, sollte durch die Ersteller der Algorithmen gewartet werden, inklusive Untersuchungen bezüglich eines möglichen Bias, der

infolge des menschlichen oder algorithmischen Datenerfassungsprozesses induziert wurde. Zusätzlich würde eine Veröffentlichung der Trainingsdaten die Möglichkeit bieten Verzerrungen zu korrigieren, wobei im Fall von Twitter eine Beschränkung dieser Offenlegung aufgrund der zu schützenden Privatsphäre und der sich dadurch ergebenden Erleichterung zur Umgehung des Systems gerechtfertigt wäre. In der vorliegenden Arbeit wurde sowohl der Prozess der Trainingsdatenerfassung 3.5.3 als auch umfassende Bias-Analysen durchgeführt 4.7.

6. **Prüfbarkeit:** Modelle, Algorithmen, Daten und Entscheidungen sollten aufgezeichnet werden, damit sie in jenen Fällen auditiert werden können, in denen ein Fehler vermutet wird. Für ein Unternehmen wie Twitter bedeutetet das also zusätzlich große Mengen an zu haltenden Daten, da für jeden Tweet die Entscheidung des Algorithmus, welches Modell und Algorithmus eingesetzt wurde und wie sich die Trainingsdaten zu diesem Zeitpunkt zusammengesetzt haben, verfügbar sein muss. Speziell in einem System, bei dem vom Algorithmus getroffene Entscheidungen ständig in den Trainingskorpus aufgenommen werden, ist eine exakte Nachvollziehbarkeit nicht trivial.

7. **Validierung und Testen:** Schließlich werden Institutionen dazu aufgefordert gründliche Methoden zur regelmäßigen Validierung ihrer Modelle anzuwenden und diese Methoden und Ergebnisse zu dokumentieren. Insbesondere sind auch routinemäßige Tests notwendig, um beurteilen beziehungsweise feststellen zu können, ob das Modell einen diskriminierenden Schaden für Teile der User verursacht. Die Ergebnisse von solchen Tests sollten veröffentlicht werden. Wie schon in Kapitel 4.7 Performanceanalyse hingewiesen, wurde auch in der vorliegenden Arbeit eine Validierung des Modells in regelmäßigen Abständen empfohlen, da sich Sprache und Gesprächsthemen in sozialen Netzwerken rasant ändern. Routinemäßig durchgeführte stichprobenartige Tests können dabei helfen, eine negative Veränderung des Modells festzustellen und Hinweise auf eine Neuausrichtung des Modelles und der Trainingsdaten geben.

KAPITEL 6

Zusammenfassung und Ausblick

6.1 Zusammenfassung

Ziel dieser Arbeit war es, einen Supervised Machine Learning Algorithmus umzusetzen, der Hassbeiträge in deutscher Sprache aus Twitter automatisch als solche klassifiziert. Die Klassifikation beruhte auf State-of-the Art Features für die Hasserkennung und wurde um Features die mit Hilfe eines LIWC-Wörterbuchs kalkuliert oder aus der Netzwerkstruktur gewonnen werden, erweitert. Die Besonderheit lag darin, dass gerade in der Identifikation von deutschen Hasspostings keinerlei vorangegangene Forschung angestellt wurde. LIWC-Attribute und diese Art an Netzwerk-Features, bei denen beispielsweise die Anzahl an Hassposter, denen ein User folgt, herangezogen wird, wurden in diesem Kontext ebenfalls noch nicht eingesetzt.

Da für die deutsche Sprache leider kein Referenzkorpus existiert, musste ein eigener Datensatz für die Bildung des Klassifikations-Modells erstellt werden und Tweets händisch annotiert werden. Insgesamt wurden 2837 Beiträge manuell klassifiziert. Nachteil besteht jedoch darin, dass dadurch kein objektiver Vergleich mit anderen Algorithmen zur Hasserkennung möglich ist.
Besondere Aufmerksamkeit kamen den Features, die durch die Analyse des Twitter-Netzwerks gewonnen werden, zu teil. Dabei stellte sich heraus, dass besonders Informationen über bereits verfasste Hasspostings eines Users für die Klassifikation sehr sinnvoll sind. Auch die Follower-Features, welche Hinweise auf die Einstellung gegenüber einschlägigen Twitter-Seiten oder User liefern, trugen überwiegend positiv zu den Messresultaten bei. Interessante Erkenntnisse im Bezug auf die Qualität bestimmter Eigenheiten von Tweets und User die als Attribute umgesetzt wurden, konnten ebenfalls festgestellt werden. So zum Beispiel geben die Anzahl an verlinkter User in einem Tweet oder die Häufigkeit an Retweets eines Beitrags beziehungsweise die bereits verfassten Tweets eines Nutzers oder die Menge an Gruppen, denen ein User beigetreten ist, Aufschluss über die Klasse eines Beitrags. In direkten Beleidigungen werden beispielsweise andere User erwähnt,

bei Nutzern mit sehr vielen Tweets handelt es sich oft um solche, die lediglich neutrale Kurzberichte über Geschehnisse aus dem untersuchten Kontext verbreiten (Nachrichtendienste). Die Follower Attribute, das Feature über die Anzahl an bereits getwitterten Hasspostings und die zuvor explizit erwähnten Merkmale aus Twitter im Bezug auf Tweets und User weisen zudem einen hohen Information Gain beziehungsweise χ^2-Wert auf und sind in den Ranglisten nach diesen Messwerten weit oben eingestuft.
Im Zuge der Arbeit wurde weiters ein Lexikon aus Hasswörtern entworfen und deren Inhalt in jenes des LIWC-Thesaurus eingegliedert, für die Filterung der Message-N-Grams verwendet und für die Berechnung des lexikalischen Features *numberOfHatefulTerms* benötigt. Zusätzlich wurde die Taxonomie des LIWCs um Hass-spezifische Kategorien erweitert. Dabei stellte sich heraus, dass diese Maßnahmen eine teils beachtliche Performance-Steigerung bewirkten. Beispielsweise stieg der F-Score bei Einsatz des Naive Bayes und Verwendung von Message Unigrams, bei denen es sich ausschließlich um Hasswörter handelte, anstelle aller möglichen Message Unigrams von 0,801 auf 0,866. Durch das Anpassen des LIWC-Wörterbuchs für den Bereich der Hasspostingerkennung stieg das F-Maß im Gegensatz zu dem bei Einsatz der Standard LIWC-Features klassifikatorübergreifend sogar um rund 20%. Im Endeffekt konnte ein Feature-Set aus Message-, Character- , Typed Dependency- und ausgewählten Features auf Basis der CFS Subset Evaluierung gefunden werden bei dem der Support Vector Machine Classifier SMO mit den Standardparametereinstellungen eine Exaktheit der Klassifikation von rund 94% erreicht. Im Detail beträgt die Precision dabei 0,94, der Recall 0,939 und das F-Measure 0,939, bei einer True-Positive-Rate von 0,914 (TP) bei Hasspostings und 0,962(TN) bei neutralen Postings. Die False-Positive-Rate für Hassbeiträge beträgt 0,038(FP), für sachliche Tweets 0,086(FN). Dies stellt die bestmögliche Konfiguration die im Rahmen dieser Arbeit gefunden werden konnte dar, um Hasspostings effektiv automatisch zu identifizieren. Damit wurde auch die ursprüngliche Zielsetzung einen Algorithmus zu entwickeln, der einen F-Score von über 90% aufweist, erreicht.

6.2 Ausblick

Es existieren einige Möglichkeiten um die Identifikation von Hasspostings weiter zu verbessern beziehungsweise anzupassen.
Beispielsweise wird auf bestimmte Sprachstile, wie Sarkasmus oder implizite Sprache kein gesonderter Augenmerk gelegt, weshalb Tweets die solche Elemente enthalten oft falsch klassifiziert werden. Spezielle Features oder mehr Trainingsdaten die auf diese Besonderheiten Bezug nehmen könnten dabei helfen.
Des weiteren wäre ein längerer Zeitraum für die Sammlung der Tweets, die als Grundlage für das Trainingsmodell dienen, oder eine ständige Adaptierung dieser Daten durch Live-klassifizierte Postings sinnvoll. Grund dafür ist, dass der derzeitige Korpus zum Teil einige Tweets enthält, die sich mit polarisierenden Themen befassen, die zum Teil nur zum Zeitpunkt der Erstellung aktuell waren und der annotierte Datensatz deshalb zu spezifisch trainiert ist, wie auch die Bias/Performance Analyse zeigt. Hassbeiträge die auf derzeitige Vorkommnisse Bezug nehmen, können so nicht erkannt werden.

Das System sollte auch in naher Zukunft in der Lage sein die verschiedenen Arten von Hass genauer zu unterteilen und zwischen Hassrede und Beleidigung zu unterscheiden, um detailliertere Aussagen über den Inhalt des Tweets treffen zu können.
Eine weitere Möglichkeit zur Verfeinerung des Prozesses der Hasspost Erkennung würde die Erweiterung des Follower Site Features darstellen. Einerseits könnte man die Anzahl an Seiten, die regelmäßig Tweets mit hasserfüllten Inhalt verbreiten und für die Feature-Kalkulation berücksichtigt werden, erhöhen andererseits diese Seiten noch ausgewogener auswählen. Außerdem könnte die Identifikation von solchen Seiten automatisch funktionieren, indem ähnlich wie in Ting et al. Eigenschaften von Twitter-Seiten extrahiert und auf Basis dieser Features diese automatisch als hassschürend identifiziert werden.[59]
In einem Einsatzgebiet bei dem Tweets live klassifiziert werden, müsste zusätzlich auch das Netzwerk ständig aktualisiert werden, wodurch dieses wachsen und umfangreicher werden würde. Features, die aus der Analyse der Netzwerkstruktur entstammen, würden daher bei umfangreicherem Netzwerk noch mehr an Bedeutung gewinnen, da dadurch ein größerer Beitrag für die Klassifikation geleistet werden kann.
Da sich Sprache und daher Schimpfwörter, Beleidigungen etc. im Laufe der Zeit verändert, müsste das Hasswortlexikon und das LIWC-Wörterbuch ebenfalls immer weiter angepasst werden, damit darauf reagiert werden kann. Weiters könnte das LIWC-Dictionary wie von Mohtasseb et al. zu einer Ontologie erweitert werden, indem Wörter oder Phrasen in bestehenden Ontologien nach LIWC-Standards kategorisiert werden. Aus den Beziehungen zwischen den Wörtern könnten weitere Features für die Identifikation von Hassbeiträgen kalkuliert werden.[47][48]
Auffällig ist zudem, dass in Twitter in manchen Beiträgen zusätzlich Bilder enthalten sind, welche ebenfalls hasserfüllt sein können. Durch Image Klassifikation könnten auch diese erkannt werden und mittels Feature-Fusion ein zusätzliches Merkmal für die Hassposting Identifikation darstellen.
Als Klassifikator mit der besten Performance stellte sich nach Durchführung der Experimente die Support Vector Machine SMO heraus. Noch besser könnten Meta Classifier funktionieren, welche die Klassifikations-Entscheidung aufgrund der Ergebnisse mehrerer unterschiedlicher Classifier treffen. Diese wurden jedoch im Rahmen dieser Arbeit nicht untersucht und könnten daher in zukünftigen Forschungen im Blickpunkt stehen.

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[1] ris.bka.gv.at, "RIS - Strafgesetzbuch § 283 - Bundesrecht konsolidiert, tagesaktuelle Fassung." https://www.ris.bka.gv.at/NormDokument.wxe?Abfrage=Bundesnormen&Gesetzesnummer=10002296&Artikel=&Paragraf=283&Anlage=&Uebergangsrecht=. Zugriffsdatum: 2016-05-30.

[2] Bundesministerium für Inneres, Bundesamt für Verfassungsschutz und Terrorismusbekämpfung (BVT), "Verfassungsschutzbericht für das Jahr 2014." http://www.bmi.gv.at/cms/BMI_Verfassungsschutz/Verfassungsschutzbericht_Jahr_2014.pdf. Zugriffsdatum: 2016-12-20.

[3] Bundesministerium für Inneres, Bundesamt für Verfassungsschutz und Terrorismusbekämpfung (BVT), "Verfassungsschutzbericht 2015." http://www.bmi.gv.at/cms/BMI_Verfassungsschutz/Verfassungsschutzbericht_2015.pdf. Zugriffsdatum: 2016-12-20.

[4] Bundesministerium für Inneres, Bundesamt für Verfassungsschutz und Terrorismusbekämpfung (BVT), "Verfassungsschutzbericht 2016." http://www.bmi.gv.at/cms/BMI_Verfassungsschutz/Verfassungsschutzbericht_Jahr_2016.pdf. Zugriffsdatum: 2017-07-06.

[5] J. Meibauer, "Hassrede – von der Sprache zur Politik," in *Hassrede/Hate Speech - Interdisziplinäre Beiträge zu einer aktuellen Diskussion.* [76], pp. 1 – 16.

[6] ris.bka.gv.at, "RIS - Strafgesetzbuch § 111 - Bundesrecht konsolidiert." https://www.ris.bka.gv.at/Dokument.wxe?Abfrage=Bundesnormen&Dokumentnummer=NOR12029654. Zugriffsdatum: 2017-01-24.

[7] ris.bka.gv.at, "RIS - Strafgesetzbuch § 115 - Bundesrecht konsolidiert, Fassung vom 25.01.2017." https://www.ris.bka.gv.at/NormDokument.wxe?Abfrage=Bundesnormen&Gesetzesnummer=10002296&FassungVom=2017-01-25&Artikel=&Paragraf=115&Anlage=&Uebergangsrecht=. Zugriffsdatum: 2017-01-25.

[8] ris.bka.gv.at, "RIS - Strafgesetzbuch § 297 - Bundesrecht konsolidiert, tagesaktuelle Fassung." https://www.ris.bka.gv.at/NormDokument.wxe?Abfrage=

Bundesnormen&Gesetzesnummer=10002296&Artikel=&Paragraf=297&Anlage=&Uebergangsrecht=. Zugriffsdatum: 2017-01-25.

[9] ris.bka.gv.at, "RIS - Strafgesetzbuch § 152 - Bundesrecht konsolidiert." https://www.ris.bka.gv.at/Dokument.wxe?Abfrage=Bundesnormen&Dokumentnummer=NOR40173657. Zugriffsdatum: 2017-01-25.

[10] ris.bka.gv.at, "RIS - Strafgesetzbuch § 107c - Bundesrecht konsolidiert." https://www.ris.bka.gv.at/Dokument.wxe?Abfrage=Bundesnormen&Dokumentnummer=NOR40177258. Zugriffsdatum: 2017-01-25.

[11] ris.bka.gv.at, "RIS - Strafgesetzbuch § 107 - Bundesrecht konsolidiert, tagesaktuelle Fassung." https://www.ris.bka.gv.at/NormDokument.wxe?Abfrage=Bundesnormen&Gesetzesnummer=10002296&Artikel=&Paragraf=107&Anlage=&Uebergangsrecht=. Zugriffsdatum: 2017-01-25.

[12] amadeu-antonio stiftung.de, "Was ist überhaupt Hate Speech ~ Amadeu Antonio Stiftung." http://www.amadeu-antonio-stiftung.de/hatespeech/was-ist-ueberhaupt-hate-speech/. Zugriffsdatum: 2016-05-30.

[13] amadeu-antonio stiftung.de, "Geh sterben! - Umgang mit Hate Speech und Kommentaren im Internet." https://www.amadeu-antonio-stiftung.de/w/files/pdfs/hatespeech.pdf, 2015. Zugriffsdatum: 2017-1-26.

[14] S. A. Smith, "There's such a thing as free speech. and it's a good thing, too.," in *Hate Speech* (R. K. Whillock and D. Slayden, eds.), pp. 226 – 266, 1995.

[15] L. Hornscheidt, "Der Hate Speech-Diskurs als Hate Speech: Pejorisierung als konstruktivistisches Modell zur Analyse diskriminierender Sprach Handlungen.," in Meibauer [76], pp. 28 – 58.

[16] B. Technau, "Sprachreflexion über politisch inkorrekte Wörter: Eine konversationsanalytische Studie.," in Meibauer [76], pp. 223 – 256.

[17] K. Marker, "Know Your Enemy. Zur Funktionalität der Hassrede für wehrhafte Demokratien.," in Meibauer [76], pp. 59 – 94.

[18] L. C. Bollinger, *The Tolerant So ciety. Freedom of Speech and Extremist Speech in America.* Oxford University Press USA, 1986.

[19] M. Schwarz-Friesel, "„Dies ist kein Hassbrief – sondern meine eigene Meinung über Euch!" – Zur kognitiven und emotionalen Basis der aktuellen antisemitischen Hassrede.," in Meibauer [76], pp. 143 – 164.

[20] M. Dederich, "Behinderung, Körper und die kulturelle Produktion von Wissen – Impulse der amerikanischen Disability Studies für die Soziologie der Behinderten.," in *Soziologie im Kontext von Behinderung. Theoriebildung, Theorieansätze und singuläre Phänomene.* (R. Forster, ed.), pp. 175 – 196, 2004.

[21] D. Unger, "Kriterien zur Einschränkung von hate speech: Inhalt, Kosten oder Wertigkeit von Äußerungen?," in Meibauer [76], pp. 257 – 285.

[22] J. Cohen, "Freedom of Expression.," in *Philosophy & Public Affairs*, vol. 22, pp. 207 – 263, Wiley, 1993.

[23] K. B. Kansara and N. M. Shekokar, "A framework for cyberbullying detection in social network," *2015 International Journal of Current Engineering and Technology (IJCET)*, vol. 5, pp. 494–498, 2015.

[24] K. Reynolds, A. Kontostathis, and L. Edwards, "Using Machine Learning to Detect Cyberbullying," in *Proceedings of the 2011 10th International Conference on Machine Learning and Applications and Workshops (ICMLA)*, vol. 2, pp. 241–244, IEEE, 2011.

[25] A. Kontostathis, K. Reynolds, A. Garron, and L. Edwards, "Detecting cyberbullying: query terms and techniques," in *Proceedings of the 2013 5th Annual ACM Web Science Conference((WebSci)*, pp. 195–204, ACM Press, 2013.

[26] V. Nahar, S. Unankard, X. Li, and C. Pang, "Sentiment Analysis for Effective Detection of Cyber Bullying," in *Proceedings of the 14th Asia-Pacific International Conference on Web Technologies and Applications (APWeb)*, vol. 7235, pp. 767–774, Springer Berlin Heidelberg, 2012.

[27] S. Nadali, M. A. A. Murad, N. M. Sharef, A. Mustapha, and S. Shojaee, "A review of cyberbullying detection: An overview," in *Proceedings of the 2013 13th International Conference on Intellient Systems Design and Applications (ISDA)*, pp. 325–330, IEEE, 2013.

[28] M. Williams and O. Pearson, "Hate Crime and Bullying in the Age of Social Media," in *Conference Report*, p. 32, Welsh Government, 2016.

[29] P. Wadhwa and M. Bhatia, "Tracking on-line radicalization using investigative data mining," in *Proceedings of the 2013 19th National Conference on Communications (NCC)*, pp. 1–5, IEEE, 2013.

[30] M. Yang and H. Chen, "Partially supervised learning for radical opinion identification in hate group web forums," in *Proceedings of the 2012 10th International Conference on Intelligence and Security Informatics (ISI)*, pp. 96–101, IEEE, 2012.

[31] A. Sureka and S. Agarwal, "Learning to Classify Hate and Extremism Promoting Tweets," in *Proceedings of the 2014 1st Joint Intelligence and Security Informatics Conference (JISIC)*, pp. 320–320, IEEE, 2014.

[32] A. H. Razavi, D. Inkpen, S. Uritsky, and S. Matwin, "Offensive Language Detection Using Multi-level Classification," in *Proceedings of the 2010 23rd Canadian Conference on Advances in Artificial Intelligence (AI)*, vol. 6085, pp. 16–27, Springer Berlin Heidelberg, 2010.

[33] B. Vandersmissen, *Automated detection of offensive language behavior on social networking sites*. PhD thesis, Universiteit Gent, 2012.

[34] G. Xiang, B. Fan, L. Wang, J. Hong, and C. Rose, "Detecting offensive tweets via topical feature discovery over a large scale twitter corpus," in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM)*, p. 1980, ACM Press, 2012.

[35] Y. Chen, Y. Zhou, S. Zhu, and H. Xu, "Detecting Offensive Language in Social Media to Protect Adolescent Online Safety," in *Proceedings of the 2012 4th International Conference on Privacy, Security, Risk and Trust (PASSAT) and 2012 5th International Confernece on Social Computing (SocialCom)*, pp. 71–80, IEEE, 2012.

[36] D. Yin, Z. Xue, L. Hong, B. D. Davison, A. Kontostathis, and L. Edwards, "Detection of harassment on web 2.0," *Proceedings of the 2009 Content Analysis in the WEB 2.0 Workshop (CAW 2.0)*, vol. 2, pp. 1–7, 2009.

[37] Z. Xu and S. Zhu, "Filtering offensive language in online communities using grammatical relations," in *Proceedings of the 2010 7th Annual Collaboration, Electronic Messaging, Anti-Abuse and Spam Conference (CEAS)*, 2010.

[38] S. H. Yadav and P. M. Manwatkar, "An approach for offensive text detection and prevention in Social Networks," in *Proceedings of the 2015 2nd International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*, pp. 1–4, IEEE, 2015.

[39] C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad, and Y. Chang, "Abusive Language Detection in Online User Content," in *Proceedings of the 2016 25th International Conference on World Wide Web (WWW)*, pp. 145–153, International World Wide Web Conferences Steering Committee, 2016.

[40] W. Warner and J. Hirschberg, "Detecting Hate Speech on the World Wide Web," in *Proceedings of the 2012 2nd Workshop on Language in Social Media (LASM)*, pp. 19–26, Association for Computational Linguistics, 2012.

[41] P. Burnap and M. L. Williams, "Hate speech, machine classification and statistical modelling of information flows on Twitter: Interpretation and communication for policy decision making," in *Proceedings of the 2014 3rd Internet, Policy and Politics Conference (IPP)*, 2014.

[42] P. Burnap and M. L. Williams, "Us and them: identifying cyber hate on Twitter across multiple protected characteristics," *EPJ Data Science*, vol. 5, no. 1, pp. 1–15, 2016.

[43] Z. Waseem and D. Hovy, "Hateful symbols or hateful people? predictive features for hate speech detection on Twitter," in *Proceedings of the 2016 15th Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies (NAACL-HLT)*, pp. 88–93, 2016.

[44] T. Zia, M. Shehbaz Akram, M. Saqib Nawaz, B. Shahzad, M. Abdullatif, M. Raza Ul Mustafa, and M. Ikramullah Lali, "Identification of hatred speeches on Twitter," in *Proceedings of the 2016 52nd Institute of Research Engineers and Scientists (IRES) International Conference*, pp. 27–32, 2016.

[45] F. Del Vigna, A. Cimino, F. Dell'Orletta, M. Petrocchi, and M. Tesconi, "Hate me, hate me not: Hate speech detection on facebook," in *Proceedings of the 2017 1st Italian Conference on Cybersecurity (ITASEC)*, 2017.

[46] I.-H. Ting, S.-L. Wang, H.-M. Chi, and J.-S. Wu, "Content matters: A study of hate groups detection based on social networks analysis and web mining," in *Proceedings of the 2013 5th IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 1196–1201, ACM, 2013.

[47] H. Mohtasseb and A. Ahmed, "Psychonet: a psycholinguistic commonsense ontology.," in *Proceedings of the 2010 2nd International Conference on Knowledge Engineering and Ontology Development (KEOD)*, pp. 159–164, 2010.

[48] H. Mohtasseb Billah, A. Ahmed, A. Altadmri, and D. Cobham, "Psychonet 2: contextualized and enriched psycholinguistic commonsense ontology.," in *Proceedings of the 2011 3rd International Conference on Knowledge Engineering and Ontology Development (KEOD)*, pp. 339–343, 2011.

[49] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*. The MIT Press, 2012.

[50] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer New York, 2006.

[51] S. B. Kotsiantis, "Supervised machine learning: A review of classification techniques," in *Proceedings of the 2007 Conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies*, pp. 3–24, IOS Press, 2007.

[52] I. für Deutsche Sprache (IDS), "ProGr@mm Grammatisches Grundwissen." Zugriffsdatum: 2016-07-18.

[53] J. W. Pennebaker, C. K. Chung, *et al.*, "Computerized text analysis of al-qaeda transcripts," *A content analysis reader*, pp. 453–465, 2008.

[54] Y. R. Tausczik and J. W. Pennebaker, "The Psychological Meaning of Words: LIWC and Computerized Text Analysis Methods," *Journal of Language and Social Psychology*, vol. 29, no. 1, pp. 24–54, 2010.

[55] M. Wolf, A. B. Horn, M. R. Mehl, S. Haug, J. W. Pennebaker, and H. Kordy, "Computergestützte quantitative Textanalyse: Äquivalenz und Robustheit der deutschen Version des Linguistic Inquiry and Word Count," *Diagnostica*, vol. 54, no. 2, pp. 85–98, 2008.

[56] B. Bohnet, "Very high accuracy and fast dependency parsing is not a contradiction," in *Proceedings of the 2010 23rd International Conference on Computational Linguistics (COLING)*, pp. 89–97, Association for Computational Linguistics, 2010.

[57] A. Björkelund, L. Hafdell, and P. Nugues, "Multilingual semantic role labeling," in *Proceedings of the 2009 13th Conference on Computational Natural Language Learning (COLING)*, pp. 43–48, Association for Computational Linguistics, 2009.

[58] A. Björkelund, B. Bohnet, L. Hafdell, and P. Nugues, "A high-performance syntactic and semantic dependency parser," in *Proceedings of the 2010 23rd International Conference on Computational Linguistics (COLING)*, pp. 33–36, Association for Computational Linguistics, 2010.

[59] I.-H. Ting, S.-L. Wang, H.-M. Chi, and J.-S. Wu, "Content matters: A study of hate groups detection based on social networks analysis and web mining," in *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 1196–1201, ACM Press, 2013.

[60] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *Computing Research Repository (CoRR)*, vol. abs/1301.3781, 2013.

[61] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents.," in *Proceedings of the 2014 31st International Conference on Machine Learning (ICML)*, vol. 14, pp. 1188–1196, 2014.

[62] N. Djuric, J. Zhou, R. Morris, M. Grbovic, V. Radosavljevic, and N. Bhamidipati, "Hate speech detection with comment embeddings," in *Proceedings of the 24th International Conference on World Wide Web*, pp. 29–30, ACM Press, 2015.

[63] V. N. Vapnik, *The Nature of Statistical Learning Theory*. Springer New York, 1995.

[64] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the 1992 5th Annual Workshop on Computational Learning Theory (COLT)*, (New York, NY, USA), pp. 144–152, ACM, 1992.

[65] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[66] S.-T. Chen, Y.-H. Hsiao, Y.-L. Huang, S.-J. Kuo, H.-S. Tseng, H.-K. Wu, and D.-R. Chen, "Comparative Analysis of Logistic Regression, Support Vector Machine and Artificial Neural Network for the Differential Diagnosis of Benign and Malignant Solid Breast Tumors by the Use of Three-Dimensional Power Doppler Imaging," *Korean Journal of Radiology*, vol. 10, no. 5, p. 464, 2009.

[67] J. C. Platt, "Advances in kernel methods," ch. Fast Training of Support Vector Machines Using Sequential Minimal Optimization, pp. 185–208, MIT Press, 1999.

[68] G. H. John and P. Langley, "Estimating continuous distributions in bayesian classifiers," in *Proceedings of the 1995 11th Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 338–345, Morgan Kaufmann, 1995.

[69] T. K. Ho, "Random decision forests," in *Proceedings of the 1995 3rd International Conference on Document Analysis and Recognition (ICDAR)*, pp. 278–, IEEE Computer Society, 1995.

[70] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 20, no. 8, pp. 832–844, 1998.

[71] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[72] H. Liu and R. Setiono, "Chi2: Feature selection and discretization of numeric attributes," in *Proceedings of the 1995 7th International Conference on Tools with Artificial Intelligence (TAI)*, pp. 88–, IEEE Computer Society, 1995.

[73] M. A. Hall and L. A. Smith, "Practical feature subset selection for machine learning," in *Proceedings of the 1998 21st Australasian Computer Science Conference (ACSC)*, pp. 181–191, Springer Berlin Heidelberg, 1998.

[74] P. Chapman, *CRISP-DM 1.0: Step-by-step Data Mining Guide*. SPSS, 2000.

[75] ACM U.S. Public Policy Council, "Statement on algorithmic transparency and accountability," no. 1, pp. 1–2, 2017.

[76] J. Meibauer, "Hassrede/hate speech," *Interdisziplinäre Beiträge zu einer aktuellen Diskussion. Gießen: Gießener Elektronische Bibliothek*, 2013.

ANHANG A

A.1 Hasswortlexikon

abbrennen abfackeln abhacken abknallen abschaum abschieben abschlachten abschneiden abtrennen affe analritter anzünden arsch arschficker arschgeige arschkriecher arschlecker arschloch asozial assi asylantenpack aufgehängt aufhängen aushungern ausrotten bastard bescheuert bestie bimbo bimbos bitch bratze brut bumsen degeneriert dreck dreckpack drecksack drecksbestie drecksbrut dreckschwein drecksgesindel dreckshaufen dreckspack drecksschlampe dreckssschmarotzer dreckssschwein drecksviech drecksvolk dreckviech dreschen drescht dumm dummdödel dödel ekelhaft elend erschießen ersäufen ertränken ertränkt eselficker exekutieren exekution fettsack fick ficken fickfehler flittchen foltern foltert folterung fratze fresse gaskammer gedroschen geisteskrank genozid gesindel gesindl gestalt gsindl hackfresse hingerichtet hinrichten hinrichtung honk hundesohn hure hurengesicht hurenkind hurensohn huso hängt höhlenmensch idiot invasoren inzucht inzuchtkind irre judenpack judensau kackbratze kackbraze kackficke kackwurst kaffer kamelficker kamelfickerhackfresse kamelscheiße kameltreiber kampflesbe kanacke kanake kastration kastrieren kastriert kastrierung kinderficker kopfschuss kümmeltürke lümmel milf minderbemittelt minderwertig mischpoke missgeburt mistgeburt mistpack mistsau mongo monster moslempack moslemweiber muffti mufftis muschi musel muselmann muselpack mußel mußelmann mußelpack möpse mörder möse neandertaler neger negger niederbrennen nigger nippel nutte pack packkind parasit penner pimmel pimpern pissbacke pissen pisser poppen primat prügel prügeln rassenschande rassenverunreinigung ratte ratten rattenpack rausschmeissen reudig rosette schabracke schafficker scheiße schlampe schmarotzer schwanzlutscher schwarzafrikaner schwein schweinehund schweinepack schweinevolk schweinspack schwengel schrott schwuchtel sodomie spacken spacko spaken sprengen sprengt sprengung titten treten trullas tschusch töten tötet tötung türkenschlampe umbringen untermensch verbluten verbrecherpack verflucht vergasen verhungern verrecken verreckt versenken versenkt vieh viehzeug vollhonk vollpfosten vögeln weib weiber wichse wichsen wichser widerling wildsau zwangskastration zwangskastrieren

A.2 Liste der Stoppwörter

aber alle allem allen aller alles als also am an ander andere anderem anderen anderer anderes anderm andern anderr anders auch auf aus bei bin bis bist da damit dann der den des dem die das dass derselbe derselben denselben desselben demselben dieselbe dieselben dasselbe dazu dein deine deinem deinen deiner deines denn derer dessen dich dir du dies diese diesem diesen dieser dieses doch dort durch ein eine einem einen einer eines einig einige einigem einigen einiger einiges einmal er ihn ihm es etwas euer eure eurem euren eurer eures für gegen gewesen hab habe haben hat hatte hatten hier hin hinter ich mich mir ihr ihre ihrem ihren ihrer ihres euch im in indem ins ist jede jedem jeden jeder jedes jene jenem jenen jener jenes jetzt kann kein keine keinem keinen keiner keines können könnte machen man manche manchem manchen mancher manches mein meine meinem meinen meiner meines mit muss musste nach nicht nichts noch nun nur ob oder ohne sehr sein seine seinem seinen seiner seines selbst sich sie ihnen sind so solche solchem solchen solcher solches soll sollte sondern sonst über um und uns unse unsem unsen unser unses unter viel vom von vor während war waren warst was weg weil weiter welche welchem welchen welcher welches wenn werde werden wie wieder will wir wird wirst wo wollen wollte würde würden zu zum zur zwar zwischen

A.3 Liste der Konnektivpartikel

aber abermals allein allemal allenfalls allerdings alsbald alsdann also anderenfalls andernfalls andererseits anderseits andernteils anschließend ansonsten anstatt dessen auch aufgrund dessen ausschließlich außerdem bald beispielsweise beziehungsweise bestenfallsbloß da dabei dadurch dafür daher dahingegen damit danach daneben dann darauf daraufhin darüber hinaus darum davor dazu dazwischen dementgegen dementsprechend demgegenüber demgemäß demnach demzufolge denn dennoch derweilen desgleichen deshalb deswegen ferner folglich freilich gegebenenfalls genauso gleichwohl gleichfalls gleichzeitig hernach hierbei hierdurch hiermit hingegen hinterher hinwiederum immerhin im Übrigen indes indessen infolgedessen insbesondere insofern insoweit inzwischen jedoch kaum mal mithin mittlerweile nachher nämlich nebenbei nebenher nichtsdestominder nichtsdestoweniger nichtsdestotrotz noch nun nunmehr nur obendrein ohnedies ohnehin schließlich schon seitdem seither selbst so sodann somit sonst soviel soweit sowieso stattdessen trotzdem überdies überhaupt unterdessen vielmehr vor allem vorher währenddessen weiter weiterhin weiters wieder wiederum wohlgemerkt zum Beispiel z.B. zudem zu guter Letzt zumal zusätzlich zuvor zwar zwischendurch zwischenzeitlich

A.4 Liste der Abtönungspartikel

aber auch bloß denn doch eben eh einfach eigentlich etwa halt ja mal man nicht nur ruhig schon überhaupt vielleicht wohl

A.5 Liste der Modalverben

dürfen darfst durfte gedurft dürfte können kannst konnte gekonnt könnte möchten möchtest mögen magst mochte gemocht möchte müssen musst musste gemusst müsste sollen sollst sollte gesollt sollte wollen willst wollte gewollt wollte

A.6 List der Personalpronomen der ersten Person

ich meiner mir mich wir unser uns

A.7 Liste der Personalpronomen der zweiten Person

du deiner dir dich ihr euer euch

A.8 Liste der Personalpronomen der dritten Person

er seiner ihm ihn sie ihrer ihr es ihnen

A.9 Liste der Demonstrativpronomen

ebendiese ebendies ebendieselbe ebendieselben ebendiesem ebendiesen ebendieser ebendieses ebenjene ebenjenem ebenjenen ebenjener ebenjenes dasjenige dasselbe demjenigen demselbe demselben denjenigen denselben dergleichen derjenige derjenigen derlei derselbe derselben desjenigen desselben diejenige diejenigen diese dies dieselbe dieselben diesem diesen dieser dieses jene jenem jenen jener jenes selber selbst solche solcher solchem solchen solches

A.10 Liste der Indefinitpronomen

all alle allem allen aller alles allesamt andere anderem anderen anderer anderes anderm andern andre andrem andren andrer andres beide beidem beiden beider beides bisschen deinesgleichen eine einem einen einer eines einige einigem einigen einiger einiges eins etliche etlichem etlichen etlicher etliches etwas etwelche etwelchem etwelchen etwelcher etwelches euresgleichen ihresgleichen irgendein irgendeine irgendeinem irgendeinen irgendeiner irgendeines irgendeins irgendetwas irgendjemand irgendjemandem irgendjemanden irgendjemandes irgendjemands irgendwas irgendwelche irgendwelchem irgendwelchen irgendwelcher irgendwelches irgendwem irgendwen irgendwer irgendwessen jede jedem jeden jeder jedermann jedermanns jedes jedwede jedwedem jedweden jedweder jedwedes jegliche jeglichem jeglichen jeglicher jegliches jemand jemandem jemanden jemandes jemands kein keine keinem keinen keiner keines keins man manch manche manchem manchen

mancher manches mehrere mehreren mehrerer meinesgleichen nichts niemand niemandem niemanden niemandes niemands paar seinesgleichen sämtlich sämtliche sämtlichem sämtlichen sämtlicher sämtliches unseresgleichen wenig

A.11 Liste der Interrogativpronomen

wer was wem wen wessen welcher welch was für einer was für ein was für eins welche was für eine was für welche was für eines welches was für eines was für einer was für welcher welchem was für einem welchen was für welchen welchen was für einen was für eine