Universität zu Köln
Institut für Digital Humanities
Sommersemester 2019
Verarbeitung textueller Daten
Prof. Dr. Andreas Witt

Sentimentanalyse von Bundestagsreden mit Python

3. April 2020

Simon Lukas Henne

shenne@smail.uni-koeln.de
Matrikelnr. 7352690
M.A. Linguistik

Inhaltsverzeichnis

1 Einleitung

Diese Hausarbeit begleitet ein Programmierprojekt, das sich mit der Frage beschäftigt, wie sich etablierte Methoden der Sentimentanalyse für komplexe Texte aus dem politischem Bereich eignen und entsprechend für digitalisierte Versionen der Plenarprotokolle des Deutschen Bundestags nutzen lassen. Ziel des Projekts ist die Entwicklung eines für die Sentimentanalyse vorbereiten Datensatzes und eines interaktiven Tools zur Sentimentanalyse von Redetexten aus dem Deutschen Bundestag.

Der Umfang der Daten beläuft sich auf die ersten 150 Sitzungen der 19. Wahlperiode des Bundestags. Der Datensatz kann voraussichtlich in der Zukunft ohne größeren Aufwand erweitert werden, da kurz nach einer Bundestagssitzung die entsprechenden Protokolle verfügbar gemacht werden. Daten zum Projekt lassen sich über ein GitHub-Repositorium unter `https://github.com/lhenne/sentimentanalyse` beziehen.

Die Ursprungsidee widerspricht den üblichen Ansätzen der Sentimentanalyse, die auf die aggregierte Analyse sehr kurzer Texte abzielen. Auch die sprachlichen Besonderheiten politischer Texte können eine Herausforderung darstellen, da sie tendenziell komplexere Ausdrucksweisen verwenden als die üblichen Materialien der Sentimentanalyse.

1.1 Endprodukt des Projektes

Zum Schluss der Projektarbeit soll eine Art annotiertes Korpus vorliegen, das zur Sentimentanalyse genutzt werden kann. Zusätzlich soll ein interaktives Tool einige der Möglichkeiten aufzeigen, die durch die Entwicklung des Korpus entstanden sind. Auch in der vorliegenden Arbeit wird von einigen beispielhaften Ergebnissen dieser Analysemöglichkeiten berichtet.

1.2 Gliederung der Arbeit

In Abschnitt 2 wird zunächst der theoretische Hintergrund des Projekts erläutert. Dazu wird das Feld der Sentimentanalyse ausführlicher beleuchtet, sowohl aus historischer Perspektive (2.1) als auch hinsichtlich der verbreiteter Methodologien (2.2) und domänenspezifischer Anforderungen (2.3). In 2.4 wird außerdem das Feld der (automatischen) politischen Textanalyse kurz porträtiert.

Abschnitt 3 dokumentiert die Projektentwicklung. 3.1 betrifft die methodologischen Entscheidungen im Vorfeld, die die Entwicklung beeinflussten. Dazu werden in diesem

Abschnitt außerdem alle zur Entwicklung genutzten Ressourcen erläutert, wie z.B. Quelldaten und Programmbibliotheken. 3.2 protokolliert die Entwicklung bzw. den Aufbau der verschiedenen Bestandteile des Korpus, und in 3.3 werden anschließend die Möglichkeiten zur Analyse beschrieben.

Mit Abschnitt 4 folgt eine kurze Erläuterung einiger beispielhafter Ergebnisse, die mithilfe des Korpus erzielt werden konnten, und in Abschnitt 5 wird ein Fazit zum Projekt gezogen und zukünftige Zielsetzungen vorgeschlagen.

2 Hintergrund und aktueller Forschungsstand

Dieser Abschnitt nähert sich der spezifischen Problematik des Projekts Schritt für Schritt an. Dazu wird zunächst das Feld der Sentimentanalyse eingeführt und anschließend besonders relevante Aspekte näher beleuchtet. Am Schluss des Abschnitts soll ein hoffentlich klares Bild zu der genauen Herausforderung entstanden sein, die dieses Projekt in Angriff nimmt, sowie Informationen zu den Problemen und Chancen genannt worden sein, die durch die in Abschnitt 3 weiter erläuterte gewählte Herangehensweise entstehen.

2.1 Sentimentanalyse und *Opinion Mining*

Die begriffliche Trennung zwischen *Sentimentanalyse* und *Opinion Mining* ist in der Literatur nicht immer eindeutig, zum Teil werden sie sogar weitgehend synonym verwendet (Pang und Lee 2008). In dieser Arbeit soll jedoch eine klare Trennung erfolgen, da das hier vorgestellte Projekt sich ausschließlich auf die Sentimentanalyse befasst. Sentimentanalyse bezeichnet hier die computergestützte Analyse von Texten mit Sicht auf die darin ausgedrückten Gefühle und emotionalen Zustände, die eine zentrale Thematik oder ein Objekt betreffen. Mithilfe der Ergebnisse dieser Analyse soll dann auf die subjektive Gesamthaltung der Verfasser in Bezug auf ein Thema oder Objekt geschlossen werden können. Werden die Ergebnisse aggregiert, was in den meisten Anwendungen der Fall ist, lassen sich anschließend allgemeine Aussagen über Haltungstendenzen innerhalb einer Gruppe von Nutzern oder Käufern, oder innerhalb der Bevölkerung machen.

Bei der Sentimentanalyse handelt es sich, trotz fortschreitender Spezialisierung und immer aufwändigerer Systeme, oft um eine recht grobe Analyse, die z.B. das mit einem Abschnitt, Satz oder Wort verknüpfte Sentiment in die Kategorien *negativ, positiv* und

neutral einteilt. Eine andere Möglichkeit ist die Zuordnung von numerischen Werten auf einer Skala, oft zwischen -1 und 1 oder zwischen 0 und 5. Die allgemeine Ausrichtung z.B. eines Wortes ist seine *Polarität*, Zahlenwerte zu dieser Polarität werden entsprechend *Intensitäten* oder *Polaritätswerte* genannt (Liu 2015). Während der Prozess viele andere Bereiche der Sprachverarbeitung berührt, bleibt die Analyse an sich damit theoretisch simpel. Eine wirklich korrekte Einschätzung des Sentiments, die in den meisten Definitionen möglichst genau der Einschätzung eines menschlichen Betrachters entsprechen soll, ist und bleibt jedoch trotz stetiger Fortschritte ein ungelöstes Problem (Cambria u. a. 2017).

Liu (2015: 14) charakterisiert Sentimentanalyse als 'mini NLP', das Kontakt zu allen anderen Bereichen des *Natural Language Processing* herstellt und dessen Lösung auch konstruktiv für diese anderen Bereiche wäre:

> [S]entiment analysis touches every core area of NLP, such as lexical semantics, coreference resolution, word sense disambiguation, discourse analysis, information extraction, and semantic analysis.

Das eng mit der Sentimentanalyse verbundene und ähnlich verbreitete *Opinion Mining* stellt dagegen deutlich ambitioniertere Ziele auf. In dieser Art der Analyse sollen durch Textanalyse möglichst genau die Meinungen der Verfasser nachgezeichnet werden. Ziel der Analyse ist oft, ein zusammenfassendes Bild über die Eindrücke einer Gruppe z.B. zu einem Produkt oder einer politischen Entscheidung oder Maßnahme zu gewinnen (Breck und Cardie 2017).

2.1.1 Ursprünge der Sentimentanalyse

Auch wenn die Begriffe Sentimentanalyse und Opinion Mining erst seit der Jahrtausendwende verbreitet sind, gab es für diese oder ähnliche Konzepte schon vorher ein beträchtliches Forschungsinteresse. Als eine der ersten Arbeiten, die sich mit der Fragestellung der maschinengestützen inhaltlichen Textanalyse auseinandersetzt, ist der *General Inquirer* (Stone 1968) zu sehen. Mithilfe dieses Computersystems können Texte im Hinblick auf ihre statistische Nähe zu 182 semantischen Kategorien annotiert werden. Die einzelnen Kategorien bestehen aus Listen von Wörtern, die zu einem semantischen Komplex wie z.B. *negative* oder *dirty* gehören, die Kategorien setzen sich also insgesamt zu einer Art Themenlexikon zusammen. Zum eigentlichen System gehören ansonsten heute übliche Textverarbeitungsverfahren wie Stemming und Retrieval. Insgesamt kann

so eine semantische Analyse und ein Vergleich von Texten durchgeführt werden. Der General Inquirer wurde ursprünglich mit Lochkarten entworfen, in späteren Versionen wurde dann das grundlegende Lexikon verbessert und erweitert, und noch heute ist eine digitalisierte Version des Systems verfügbar[1].

Der General Inquirer weist zwar eine ähnliche Zielsetzung auf, wie sie später z.B. für das Opinion Mining deklariert wurde, er ist aber ausdrücklich nur für die inhaltliche Analyse von Texten in den Sozialwissenschaften gedacht (Hurwitz 2020) und funktioniert zudem nur halbautomatisch. Erst deutlich später entwickelte sich das Forschungsfeld bestimmter in Richtung der heutigen Definition von Sentimentanalyse, zunächst jedoch ohne Benennung dieser Disziplin.

In den 1990er-Jahren beschäftigen sich einige Arbeiten wie z.B. Wiebe (1994) und Hatzvassiloglou und McKeown (1997) mit verwandten Problemen wie der Messung von Subjektivität in Narrativtexten bzw. Einschätzungen zur Polarität von Adjektiven. Damit thematisieren sie einige der zentralen Analysedimensionen, die dem Feld auch heute noch zuzuordnen sind(Breck und Cardie 2017).

Bei Turney (2002) ist das Ziel zum ersten Mal die automatische Klassifikation von Nutzertexten einer Online-Rezensionsplattform, ebenso wie bei Pang, Lee und Vaithyanathan (2002), wo wohl erstmals maschinelles Lernen zum Einsatz kommt, um Dokumente als positiv oder negativ zu klassifizieren. Nasukawa und Yi (2003) tauften diese Art der Analyse dann *sentiment analysis* in einer rein methodologischen Arbeit, die Möglichkeiten zur Extraktion von Polaritätswerten aus Texten erkundet.

Von diesen Jahren an beschleunigte sich das Wachstum des nun konkreteren Forschungsfeldes der Sentimentanalyse zunehmend. Pang und Lee (2008: 7) erklären diese Entwicklung vor allem damit, dass erstens maschinelles Lernen eine immer mehr verbreitete Methode wurde, zweitens entsprechende Datensätze zum Training solcher Systeme wegen des Wachstums des *World Wide Web* im Allgemeinen und Online-Meinungsportalen im Besonderen verfügbar wurden, und sich drittens Forscher der Möglichkeiten und Herausforderungen des Feldes zunehmend bewusst und dadurch motiviert wurden. Mit diesem Wachstum kam um das Jahr 2006 auch die fortschreitende Ausweitung der Anwendungsgebiete, indem mit unterschiedlichem Erfolg die verschiedensten Fragestellungen angegriffen wurden (Liu 2015).

[1] `http://www.wjh.harvard.edu/~inquirer/`, zuletzt abgerufen am 27.03.2020.

2.1.2 Anwendungsgebiete

Heutzutage finden Sentimentanalyse und Opinion Mining in vielen verschiedenen Gebieten Anwendung. Wie bereits festgestellt, erlauben die Ansätze Rückschlüsse auf implizite Haltungen und Gefühlszustände, was unter Umständen ein zentraler Faktor für Entscheidungen sein kann und damit dieser Art von Informationen einen hohen Wert zukommen lässt. Unterschiede in der Anwendung lassen sich hauptsächlich anhand von zwei Aspekten beschreiben.

Erstens kann der Zweck der Anwendungen unterschieden werden, also das Endziel oder Endprodukt, das ein System zur Sentimentanalyse hervorbringen soll. Forschungsarbeiten zur Sentimentanalyse sind entweder primär methodologisch oder primär inhaltlich orientiert. Arbeiten, die ersteres Ziel verfolgen, haben das Ziel, den methodologischen Status Quo zu verbessern und neue Ansätze für die Konstruktion von Systemen zur Sentimentanalyse auszuprobieren. Diese Arbeiten sind vor allem für die Zusammenfassung in 2.2 relevant.

Für inhaltlich orientierte Arbeiten sollen aus den Ergebnissen der Systeme generalisierbare Aussagen gewonnen werden, z.B. zu Gefühlszuständen von Verfassern in einer Textsammlung.

Jedoch gibt es abseits des wissenschaftlichen Interesses noch weitere Zwecke. So sind die häufigsten Hintergründe der Entwicklung von Analysesystemen Markt- und Meinungsforschung (Pang und Lee 2008). In der Marktforschung sollen diese vor allem die Meinungen von (potenziellen) Kunden zu Produkten abbilden oder zusammenfassen, in der Meinungsforschung sollen sie z.B. die von Bevölkerungsgruppen oder Wählern repräsentieren. Letzteres Potenzial wurde in den vergangenen Jahren auch für wissenschaftliche Untersuchungen entdeckt, vor allem also in Politologie und Soziologie (Busch und Kaupert 2018; Mullen und Malouf 2006; Rauh 2018; Young und Soroka 2012).

Zweitens kann sich das Textmaterial, welches analysiert wird, deutlich unterscheiden. Schon in frühen Arbeiten waren aufgrund ihrer massenhaften Verfügbarkeit, der klaren Verbindung zu Nutzerhaltungen und des ursprünglich digitalen Formats vor allem Texte wie Rezensionen aus Online-Meinungsportalen (Pang, Lee und Vaithyanathan 2002; Turney 2002) und Einträge aus Internet-Diskussionsforen (Abbasi und Chen 2007; Mullen und Malouf 2006) oft Untersuchungsobjekte. Liu (2015: xiii) bezeichnet diese Art von Text wegen ihrer starken Prägung durch persönliche Meinungen auch als „opinion document“.

Im vergangenen Jahrzehnt hat sich insbesondere die Analyse von kurzen Texten in sozialen Netzwerken wie Facebook und Twitter als Hauptfeld der Sentimentanalyse gezeigt (Giachanou und Crestani 2016). Die Beschränkung von *Tweets* auf 140 bzw. 280 Zeichen sowie ihre massenhafte Verfügbarkeit zu den verschiedensten Themen qualifizieren individuell zugeschnittene Twitter-Korpora als ideale Ressource, um etwa in breiten Gruppen existierende Meinungstendenzen synchronisch oder diachronisch verfolgen zu können. Schon frühe Arbeiten zeigen, dass Autoren sich dieser Voraussetzungen und ihrer spezifischen Bedingungen durchaus bewusst sind, und auch um den Nutzen der verfügbaren Metadaten wissen. Thelwall u. a. (2011: 411) nutzen ein für kurze Texte angepasstes, ursprünglich für das soziale Netzwerk *MySpace* entwickeltes Sentimentlexikon, das für ihren Algorithmus mit zusätzlichen linguistischen Regeln kombiniert wird. Bollen u. a. (2011) machen von genauen Informationen zur Publikationszeit von Tweets Gebrauch, um die Entwicklung von Haltungen der Nutzer auf einem Zeitstrahl nachzuzeichnen. Auch maschinelles Lernen kam in Verbindung mit Twitter-Daten schon früh zum Einsatz (Kouloumpis u. a. 2011).

Wie in 2.4 erläutert wird, spielt die Sentimentanalyse allerdings auch in Disziplinen, in denen mit langen komplexen Texten zu rechnen ist, eine nennenswerte Rolle.

2.2 Methodologien der Sentimentanalyse

Auch bei Methodologien der Sentimentanalyse müssen mehrere Dimensionen und zentrale systemische Designfragen unterschieden werden. Zwei der wichtigsten werden im Folgenden dargestellt.

2.2.1 Analyseebenen

Eine der grundlegendsten und zugleich entscheidenden Fragen, die sich oft gleich zu Anfang eines Projektes stellt, ist die nach der Analyseebene des Systems. Auf welche Informationseinheit soll die Untersuchung ausgerichtet sein? Repräsentiert die zusammengesetzte Polarität eines gesamtes Dokument am besten die dadurch ausgedrückte emotionale Haltung des Verfassers?

Abwägungen dieser Art sollten zielgerichtet unternommen werden. Um z.B. einer Rezension eine grundsätzlich positive oder grundsätzlich negative Gesamthaltung zuordnen zu können, ist eine Analyse auf Dokumentebene wahrscheinlich die beste Herangehensweise, da ein entsprechender Gesamtwert aussagekräftig genug für eine solche

Klassifikation ist (Liu 2015). Ähnliches muss für sehr kurze Texte wie Tweets vermutet werden, die höchstwahrscheinlich nur eine zusammenhängende Haltung ausdrücken. Kouloumpis u. a. (2011) erwähnen, dass aufgrund dieser Kürze die Analyse von Tweets eher einer Analyse auf der Satzebene gleicht.

Als weitere, differenzierendere Granularitäten als die Dokumentebene, werden auch Abschnitte, Sätze, Phrasen und Wörter genutzt (Breck und Cardie 2017). Für längere Texte verschiedener Arten eignet sich oft eine dieser Ebenen am besten, da sie Veränderungen innerhalb des Textes einfangen können und dazu beitragen, zwischen verschiedenen Referenten bzw. Evaluationsobjekten zu unterscheiden (Liu 2015). Thematisiert z.B. eine politische Rede mehrere Maßnahmen, von denen der Sprecher einige ablehnt und anderen wiederum zustimmt, so könnte diese Unterscheidung wiedergegeben werden und, anders betrachtet, auch bei der Erkennung von semantischen Zusammenhängen hilfreich sein.

Eine ambitionierte Idee aus der jüngeren Vergangenheit ist *concept-level sentiment analysis*, die auf der Ebene von semantischen Konzepten arbeitet (Bisio u. a.; in Cambria u. a. 2017). Diese müssen jedoch vorher erst zuverlässig ermittelt werden, was eine separate, ungemeisterte Herausforderung ist.

Implizit erfolgt in der Literatur oft eine Evaluation auf Wortebene, auch wenn deren Ergebnisse oft nicht an die Oberfläche gelangen, sondern mit ihrer Hilfe Gesamtwerte errechnet werden. Komplexere Typen wie N-Gramme wurden durchaus eingesetzt, allerdings werden nur solche auf der Subwortebene als hilfreich eingeschätzt (Kouloumpis u. a. 2011; Wiebe u. a. 2004). Wortartklassifikation (POS-Tagging) hat für manche zweifelhaften Nutzen (Kouloumpis u. a. 2011), wird aber oft zumindest als eine Art von *word sense disambiguation* genutzt und kann zusätzlich Tendenzen dazu offenbaren, welche Wortarten besonders ausdrucksstark in der Wiedergabe des Sentiments sind (Pang und Lee 2008).

2.2.2 Klassifikationsmethoden

Eine weitere zentrale Entscheidung ist, auf welche Art die Klassifikation von Wörttypen erfolgen soll, wie also das Basismodell der Analyse erstellt und aufgebaut wird. Die aktuell meistverbreiteten Verfahren stehen hier größtenteils in klarer Opposition zu den in 2.1.1 beschriebenen Ursprüngen des Feldes.

Einführungen in das Thema – ob mit wissenschaftlichem Anspruch (z.B. Breck und

Cardie 2017; Liu 2015; Pang und Lee 2008) oder pragmatischer wie in zahlreichen Marketingblogs – legen seit dem in 2.1.1 erwähnten Umschwung in den frühen 2000er-Jahren den größten Fokus auf Methoden des maschinellen Lernens, mit denen Sentimentklassifikatoren auf Basis von Trainingsdaten gelernt werden. Auf diese Weise kann ein Sentimentlexikon generiert werden, das nicht nur allgemeine Werte für die Polarität von Wörtern enthalten kann, sondern auch die in den Trainingsdaten vorhandenen Kontexte mit einbezieht. Der Trainingsprozess kann dafür z.B. so entworfen werden, dass das System *multi word expressions* erfasst, um für „lange Batterielaufzeit“ und „langer Startvorgang“ in Rezensionen zu einem Laptop nicht die gleiche Polarität eintragen zu müssen (Wang und Zhai; in Cambria u. a. 2017: 124).

Lexikonbasierte Ansätze, die in der Frühphase des Forschungsfelds am weitesten verbreitet waren, spielen eine untergeordnete Rolle besonders bei der Analyse von Texten aus sozialen Netzwerken, wo Trainingsdaten oft in praktisch unbegrenztem Umfang verfügbar sind. In Bereichen mit nach Erwartung spezielleren sprachlichen Eigenschaften (siehe 2.3) kommen diese jedoch auch heute noch zum Einsatz, weswegen auch zur Erstellung von domänenspezifischen Lexika geforscht wird (Bross und Ehrig 2013; Haselmayer und Jenny 2017).

Zur Sentimentanalyse englischsprachiger Texte existieren mehrere Ressourcen, unter anderem *SentiWordNet* (Baccianella u. a. 2010), welches zusätzliche semantische Informationen einbettet. Noch vor einigen Monaten wurde eine neue Version des populärsten deutschen Sentimentlexikons *SentiWS* (Remus u. a. 2010) veröffentlicht[2]. Daneben ist auch *GermanPolarityClues*[3] (*GPC*) eine vergleichbare Ressource.

Cambria u. a. (2017: 5f.) teilen das Feld der Ansätze zur Sentimentanalyse in drei verschiedene Kategorien ein. *Knowledge-based techniques* basieren hauptsächlich auf vordefinierten Lexika, deren Inhalte zur Bewertung des Sentiments von Texten genutzt werden. Diese Methoden haben klare Nachteile, denn eine naive Anwendung dieser Art liefert schon dann eine falsche Bewertung, wenn ein Wort wie *traurig*, das in5 SentiWS erwartungsgemäß eine negative Polarität von -0.1266 aufweist, negiert wird. So würde dem Satz „Seit wir uns getroffen haben, bin ich nie mehr traurig“ wohl ein negativer Gesamtwert zugewiesen, obwohl die damit kommunizierte Haltung zweifellos sehr po-

[2] *SentiWS 2.0*, zu finden unter `https://wortschatz.uni-leipzig.de/de/download`. Zuletzt abgerufen am 30.03.2020.

[3] *GermanPolarityClues* Version 0.2, zu finden unter `http://www.ulliwaltinger.de/sentiment/`. Zuletzt abgerufen am 30.03.2020.

sitiv ist. Deswegen nutzen fortschrittlichere Systeme mit diesem allgemeinen Ansatz zusätzlich linguistische Regeln, um z.B. bei Negationen den Polaritätswert umzukehren, z.B. zu 0.1266 für „nicht traurig“ (Rauh 2018). Dem wissensbasierten Ansatz folgt auch die hier vorliegende Arbeit.

Die zweite Kategorie ist die der statistischen Methoden, die schon seit Entstehung des Feldes meistens mit Methoden des maschinellen Lernens verbunden ist (z.B. Pang und Lee 2004; Pang, Lee und Vaithyanathan 2002; Turney 2002), und sich in den letzten Jahren auch zunehmend die neuen Technologien des *Deep Learning* (Goodfellow u. a. 2016) zunutze machen (z.B. Jianqiang u. a. 2018; Severyn und Moschitti 2015). Diese Kategorie ist wie oben erwähnt aktuell aus mehreren Gründen die am weitesten verbreitete. Erstens, da im Laufe der Zeit oft gezeigt wurde, dass Ansätze des maschinellen Lernens meist eine bessere Systemleistung garantieren können. Zweitens, da generell Verfahren des maschinellen Lernens und in letzter Zeit auch die des Deep Learning in vielen Feldern schnell an Popularität gewonnen haben. Und drittens, weil mit dieser Methodik bei ausreichender Datenbasis mitunter sehr wenig menschliche Annotationsarbeit o.ä. involviert ist, also nur die Eingabedaten entsprechend vorbereitet werden müssen und das System anschließend selbständig angelernt wird (Liu 2015).

Systeme der dritten Kategorie sind Hybrid-Systeme, in denen Strukturen aus den beiden anderen Kategorien implementiert sind und Informationen liefern. Oft wird dementsprechend nach dem Prinzip des *supervised learning* verfahren, indem zunächst die Trainingsdaten mit Wissensressourcen wie Lexika und linguistischen Strukturen annotiert werden, um dann als Eingabe für ein System des maschinellen Lernens zu dienen. Die Herangehensweise ist die jüngste und wohl vielversprechendste, allerdings sind entsprechende Systeme auch deutlich aufwendiger zu konstruieren, da der Arbeitsaufwand eines wissensbasierten Systems anfällt und anschließend ein ML-System trainiert werden muss, welches die Annotationen auch nutzen kann.

2.3 Domänenspezifische Anforderungen

Wie bereits angedeutet, ist damit zu rechnen, dass verschiedene Textsorten, inhaltliche Kontexte und zahlreiche weitere Aspekte verschiedene Anforderungen mit sich ziehen, um eine verlässliche Sentimentanalyse zu gewährleisten. Liu (ebd.: 63) warnt vor der Anfälligkeit der Klassifizierung gegenüber Anwendung in anderen Domänen als der, aus der z.B. die Trainingsdaten eines ML-Modells stammen:

> It has been shown that sentiment classification is highly sensitive to the domain from which the training data are extracted. A classifier trained using opinion documents from one domain often performs poorly on test data from another domain because words and even language constructs used for expressing opinions in different domains can be quite different.

Neben der Gefahr der Fehldeutung von Wörtern deckt ein Sentimentlexikon oft das domänenspezifische, oft werttragende Vokabular nur unzureichend ab – in Kontexten, in denen eine große gelabelte Datenbasis verfügbar ist, sind daher generell *supervised learning*-Ansätze zu bevorzugen (Liu 2015).

Allgemeine Richtlinien und Empfehlungen für bestimmte Domänen (wie z.B. mündliche politische Texte, die hier analysiert werden sollen) sind meines Wissens nicht verfügbar.

2.4 Politische Textanalyse

Wie bereits in 2.1.1 erwähnt, ist der *General Inquirer* als Analysesystem für Texte aus den Sozialwissenschaften entwickelt worden und wurde so gesehen in anderen Studien nur zweckentfremdet. Es ist nicht das einzige solche System, da die politische Textanalyse bereits seit längerer Zeit ein wichtiger Teil der entsprechenden Wissenschaften ist. Diese Tradition beruht auf der Wichtigkeit von Sprache als Ausdrucksmittel und als Verhandlungsmedium für Konflikte, wie Grimmer und Stewart (2013: 267) konstatieren: „Language is the medium for politics and political conflict." Die qualitative Herangehensweise an Text- bzw. Diskursanalyse, also ihre manuelle linguistische Untersuchung, war und ist ein großes Forschungsfeld an der Schnittstelle zwischen Sozialwissenschaften und Linguistik (van Dijk 1997).

Die wachsenden Möglichkeiten zur automatischen Textanalyse kam dieser Disziplin besonders zugute, da die Datenbestände, die für die umfassende Betrachtung eines Problems anfallen, oft schlicht zu groß für eine manuelle Untersuchung sind (Grimmer und Stewart 2013: 267). Heutzutage scheint der Ansatz zudem dadurch begünstigt zu sein, dass Texte aus offiziellen Quellen immer öfter in maschinenlesbarer oder zumindest generell in digitalisierter Form vorliegen. Oft sind diese außerdem, die Pflicht zur Information der Bevölkerung berücksichtigend, frei verfügbar. Dies gilt auch für die Plenarprotokolle, die Gegenstand dieses Projekts sind. Diese Voraussetzungen können Forscher sich zum Vorteil machen, weswegen in den vergangenen Jahren zahlreiche entsprechende Arbeiten veröffentlicht wurden.

2.4.1 Sprachressourcen

Für Forschungsprojekte in der automatischen Textanalyse werden meist eigens kompilierte oder von Anderen bezogene Textkorpora verwendet. Es existieren mehrere dieser Ressourcen für politische Texte, die hier folgende Aufzählung soll lediglich einige der umfassendsten und populärsten nennen. Besonders zu erwähnen sind auch kleinere, speziell kuratierte Korpora wie das von Truan (2016), welches nur für die Untersuchung von parlamentarischen Debatten zum Thema Europa erstellt wurde.

Ein Beispiel aus dem deutschsprachigen Raum ist das *PolMine*-Projekt[4], welches das Ziel hat, aus frei verfügbaren offiziellen Protokolldaten Sprachressourcen zu machen. Als Teil des Projekts ist das *GermaParl*-Korpus veröffentlicht worden (Blatte und Blessing 2018). Es enthält alle Plenarprotokolle des Deutschen Bundestags zwischen Februar 1996 und Dezember 2016 und kann entweder als XML-Datensatz oder als Bibliothek für die Programmiersprache *R*[5] bezogen werden. GermaParl wurde erstellt, indem automatisch generierte Textdateien und in einigen Fällen PDF-Dateien aus den offiziellen Quellen verarbeitet wurden.

Abercrombie und Batista-Navarro (2018) konstruieren ein Korpus von Plenarprotokollen des englischen Parlaments mithilfe der offiziellen *Hansard*-Transkripte, um auf dessen Basis Sentimentanalysen auf Dokument- bzw. Redeebene durchführen zu können. Die inhaltlichen Schlüsse der Autoren zum Wert einer Sentimentanalyse fallen jedoch bescheiden aus, da sie in ihr z.B. keine klaren Vorteile gegenüber einer einfachen Betrachtung der Abstimmungsverläufe einzelner Abgeordneter sehen, wenn mithilfe des ermittelten Sentiments eine Meinung zu einer Fragestellung gedeutet werden sollte (ebd.: 4180).

Das *ParlSpeech*-Korpus (Rauh und Schwalbach 2020) enthält Texte der Plenarprotokolle aus neun Ländern, jeweils für die zurückliegenden 21 bis 32 Jahre. Insgesamt beläuft sich der Datensatz auf 6,3 Mio. Reden. Neben seines riesigen Umfangs ist das Korpus wohl vor allem deswegen interessant, weil hier erstmals Daten aus mehreren Parlamenten mithilfe eines vereinigten Annotationsschemas aufbereitet und somit gleich verarbeitbar und vergleichbar gemacht wurden. ParlSpeech enthält auch Plenarprotokolle des deutschen Bundestags bis zum Jahr 2019, allerdings wurden diese auch hier nur auf Basis der offiziellen PDF-Dokumente kompiliert. Die Autoren weisen darauf hin,

[4] `https://polmine.github.io/`
[5] `https://www.r-project.org/`

dass für weitere Projekte die offiziellen XML-Dokumente zur Verfügung stehen.

Schon deutlich früher entstanden ist außerdem das *Europarl*-Korpus (Koehn 2005), welches mehrsprachige parallele Texte aus Sitzungen des EU-Parlaments enthält, die vor allem als Trainingsmaterial für maschinelle Übersetzung entwickelt wurden.

2.4.2 Sentiment- und Polaritätsanalysen

Die Analyse der Polarität von Texten hat in den Sozialwissenschaften bereits eine lange Tradition (Haselmayer und Jenny 2017: 2625). Gewissermaßen bedeutet die Einkehr der Sentimentanalyse also nur ein Anknüpfen an bestehende Strukturen, vor allem gemäß zweier Aspekte.

Erstens wird wie oben erläutert die Analyse weitgehend automatisiert, was sowohl einen grundlegenden Paradigmenwechsel als auch weitreichende Veränderungen für die analytische Praxis mit sich zieht.

Zweitens weisen die Bedeutungen von Polarität abseits der Sentimentanalyse und innerhalb der Sentimentanalyse durchaus Unterschiede auf. Schließlich ist es Aufgabe der Sentimentanalyse, die Haltung gegenüber einer Entität oder einem Thema aus der Polarität der Aussagen darüber abzuleiten, und zwar mit automatischen Methoden, die solche Zusammenhänge größtenteils rein auf Textbasis erschließen. Wird Polarität z.B. zu einem Thema in den Medien qualitativ untersucht, erschließen sich die Zusammenhänge normalerweise nicht in Form von numerischen Werten, die hauptsächlich durch lexikalische und syntaktische Analyse errechnet werden, sondern sind oft von einem menschlichen Annotator abhängig und intellektuell begründet (ebd.).

An dieser Stelle sollte jedoch auch bedacht werden, dass automatische Sprachanalyse durchaus fehleranfällig ist und politische Texte eher kontextabhängiger und sprachlich wie semantisch komplexer als die anderer Genres sind (Grimmer und Stewart 2013). Dadurch, dass z.B. bei der Arbeit mit Lexika oft Ressourcen benutzt werden, die nicht speziell für diese Domäne adaptiert wurden, kann die Fehleranfälligkeit ungleich höher sein (ebd.: 268).

Die Idee, die im XML-Format angebotenen Plenarprotokolle des Deutschen Bundestags für Datenanalysen zu nutzen, wurde schon an mindestens zwei anderen Stellen aufgegriffen Eine digitale Reportage der *Süddeutschen Zeitung* hat sich mit der AfD-Fraktion im Bundestag beschäftigt (Brunner u. a. 2018), besonders intensiv wurde sich hier mit Zwischenrufen beschäftigt.

Besonders zu erwähnen ist aber auch die Arbeit von Rauh (2018), die zahlreiche Parallelen zu der hier vorliegenden Projektarbeit vorweist. Wie im nächsten Abschnitt erklärt wird, wurde mithilfe der gleichen Sentimentlexika ebenfalls eine Analyse von Bundestagsreden vorgenommen, allerdings mit der Zielsetzung, die Kombination der Sentimentlexika zu validieren. Rauhs Analyse ist deswegen nicht darauf konzentriert, ein annotiertes Korpus zu erstellen, inhaltliche Schlüsse zu ziehen oder die Ergebnisse der Sentimentanalyse analytisch aufzubereiten, was in der vorliegenden Arbeit im Fokus steht.

3 Projektarbeit: Sentimentanalyse von Bundestagsreden

Dieser Abschnitt beschreibt die Vorüberlegungen und Entscheidungen sowie den Entwicklungsverlauf des Projektes zur Sentimentanalyse von Bundestagsreden.

3.1 Methodologische und technische Basis

Für das Baseline-System wurde entschieden, einem wissensbasierten Ansatz zu folgen. Dazu werden die Sentimentlexika *SentiWS* (Remus u. a. 2010) und *GPC* (Waltinger 2010) genutzt. Motivation hierfür waren die erkennbar besseren Ergebnisse für das kombinierte Lexikon, von dem Rauh (2018) berichtet. Jedoch erfolgt der Einsatz der beiden Lexika nicht exakt wie in Rauhs Ansatz. Die Unterschiede werden in 3.3 weiter erläutert.

Generell fiel die Entscheidung auf einen wissensbasierten Ansatz, da der Umfang der Lexika und deren Abdeckung wichtiger Vokabeln aus dem offiziellen politischen Kontext durchaus zufriedenstellend erschienen, und weil Rauhs Arbeit darauf hindeutete, dass sich auf diese Weise gute Ergebnisse erzielen ließen. Die schnelle Konstruierbarkeit eines annotierten Korpus und eines interaktiven Baseline-Systems war ein weiteres Argument. Die Frage nach der Analyseebene muss an dieser Stelle nicht beantwortet werden, da mithilfe des Korpus eine Analyse auf Wort-, Satz-, Absatz- und Dokumentebene möglich ist.

Ein zusätzliches System mit Klassifikation auf ML-Basis wurde ursprünglich als optionale Erweiterung zu einem späteren Zeitpunkt eingeplant, bisher wurde es aber nicht verwirklicht, da es sinnvoller erschien, das Korpus zu finalisieren und anschließend einige Nutzungsmöglichkeiten zu erkunden.

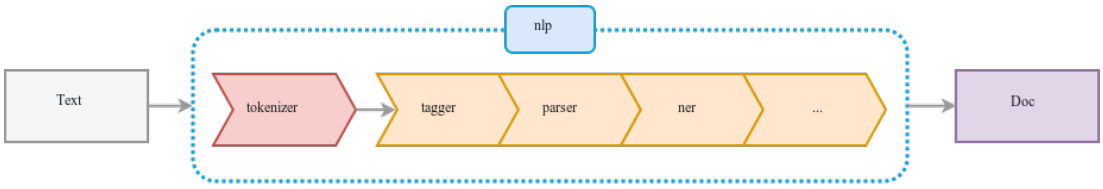

Abbildung 1: Die Textverarbeitungs-Pipeline des *spaCy*-Pakets. Übernommen aus *Language Processing Pipelines. spaCy* Usage Documentation. `https://spacy.io/usage/processing-pipelines`

3.1.1 Frameworks und technische Voraussetzungen

Das System wird in *Python* 3.6[6] entwickelt. Die weite Verbreitung von Python zu Zwecken der maschinellen Sprachverarbeitung und die Verfügbarkeit zahlreicher zusätzlicher Bibliotheken zur Textverarbeitung und Arbeit mit natürlicher Sprache sprechen für dessen Eignung zur Lösung der mit dem Projekt verbundenen Aufgaben.

Die 150 Plenarprotokoll im XML-Format werden mithilfe des Python-Pakets `lxml`[7] eingelesen und geparst. Durch das Modul `lxml.objectify` können die Dokumente dann ähnlich wie native Python-Objekte navigiert und ihre Inhalte entsprechend ausgelesen werden. Hier sollte erwähnt werden, dass auch mit anderen Bibliotheken wie dem in Python enthaltene Modul `xml.etree.ElementTree` und *BeautifulSoup*[8] (im Paket `bs4`) experimentiert wurde. Es zeigte sich allerdings, dass ersteres deutlich langsamer arbeitete und weniger intuitiv zu handhaben war als lxml, und es bei letzterem Probleme mit der Validierung der Dokumente gab, welche die Verarbeitung unnötig erschwert hätten.

Als zentrale Verarbeitungsbibliothek für die Texte der Plenarprotokolle wird *spaCy*[9] eingesetzt. Das Paket `spacy` bietet ein einheitliches System zur Extraktion von Informationen mithilfe eines vorgefertigten Sprachmodells. Dieses Sprachmodell `de_core_news_sm`[10] ist ein optionales, für deutsche Texte optimiertes spaCy-Paket. Texte werden darin entsprechend der Pipeline in Abb. 1 verarbeitet.

Die Ergebnisse aller aufgeführten Module sind potenziell relevant für das Projekt. Anfängliche Tokenisierung ist wichtig, da für die Analyse mit Sentimentlexika einzelne Token als Eingabe nötig sind. Wortartklassifikation mithilfe des Part-of-Speech-Taggers (*tagger*) im zweiten Modul kann bei der Disambiguierung von Homographen helfen, die

[6] `https://www.python.org/`
[7] `https://lxml.de/`
[8] `https://www.crummy.com/software/BeautifulSoup/`
[9] `https://spacy.io/`
[10] verfügbar unter `https://spacy.io/models/de#de_core_news_sm`

semantisch unterschiedlich sind und verschiedenen Wortarten angehören, und verschiedene Polaritätswerte haben können. Die Ergebnisse des *dependency parser* (*parser*), der syntaktische Beziehungen zwischen Token analysiert, könnte die Polaritätsbewertung beeinflussen, indem z.B. der Nukleus eines Ausdrucks höher bewertet wird. *Named entity recognition* (*ner*) lässt sich zur Erkennung des Referenz- oder Evaluationsobjektes eines Satzes, Absatzes oder einer Rede nutzen.

Auch wenn es besonders im Falle der *named entity recognition* oft fehlerhafte Ergebnisse gibt, sind die bei der Initialverarbeitung durch die spaCy-Pipeline automatisch erstellten Informationen von überzeugender Qualität und könnten nicht ohne größeren Aufwand in einem eigens entworfenen System übertroffen werden.

Schlussendlich ist noch `pandas`[11] zu erwähnen, welches mit der Funktion `pandas.read_csv` das Einlesen der Sentimentlexika im TSV-Format und mit der tabellarischen *DataFrame*-Datenstruktur die Handhabung der Analysedaten erleichtert.

3.1.2 Datenbasis

Als Datensätze wurden die Plenarprotokolle der ersten 150 Sitzungen der 19. Wahlperiode des Deutschen Bundestags ausgewählt. Die Sitzungen fanden zwischen dem 24. Oktober 2017 und dem 6. März 2020 statt.

Die Protokolle sind als XML-Dateien als offizielle Dokumente des Deutschen Bundestags über dessen Open Data-Portal verfügbar[12]. Sämtliche Dokumente sind validierbar gegen eine Dokumenttypdefinition (DTD), die von der publizierenden Institution ausführlich dokumentiert wurde (*Bundestags-Plenar-Protokolle im XML-Format: Aufbau der Strukturdefinition – DTD* 2015). Die DTD scheint keinem übergeordneten Annotationsschema zu folgen, sondern isoliert für den gegebenen Zweck entwickelt worden zu sein.

Laut o.g. DTD-Handbuch basieren die XML-Dokumente auf den Plenarprotokollen im PDF-Format. Ob die Überführung hauptsächlich manuell oder automatisch erfolgt ließ sich nicht feststellen.

Das Korpus der verarbeiteten Redetexte beläuft sich auf insgesamt 10.835.000 Token (inkl. Satzzeichen u.ä.). Die Texte der einzelnen Protokolle umfassen zwischen 1.190 (Sitzung 19) und 166.521 Token (Sitzung 107), bei einer durchschnittlichen Länge von 72.232 Token. Einzelne Rede-Elemente sind zwischen 2 (Gerd Müller (CSU) in

[11] `https://pandas.pydata.org/`
[12] zu beziehen unter `https://www.bundestag.de/services/opendata`

Sitzung 103) und 8.225 Token (Angela Merkel (CDU) in Sitzung 22) lang, bei einer durchschnittlichen Länge von 689 Token. Dabei ist zu bedenken, dass bei jedem offiziellen Sprecherwechsel in der XML-Datei das laufende Rede-Element endet und ein neues beginnt. Somit ist Gerd Müllers kurze Wortmeldung „Ja.“ ebenso eine vollständige Rede wie die Ansprache der Bundeskanzlerin.

Die Sentimentlexika enthalten Sentimentwerte zu Lemmata und einige ihrer flektierten Formen. In GPC ist jede Form als ein lexikalischer Eintrag notiert, das Lexikon enthält 19.962 Einträge mit negativer Polarität und 17.627 Einträge mit positiver Polarität, also insgesamt 37.589 Typen. Der GPC-Datensatz mit neutralen Einträgen wurde nicht genutzt, da die Analyse nur zwischen positiv und negativ unterscheiden soll und zudem *SentiWS* nur diese beiden Polaritäten kennt.

SentiWS zählt 1.827 Grundformen bzw. 18.285 flektierte Formen mit negativer Polarität, und 1.644 Grundformen bzw. 16.853 flektierte Formen mit positiver Polarität, also insgesamt 35.138 Typen. Jeder Eintrag hat einen Wert im Bereich [-1 ; 1] , der die Intensität der Polarität wiedergibt. GPC enthält lediglich eine binäre Unterteilung in negative und positive Polarität.

3.2 Entwicklung des Projekts

Das Projekt ist in zwei Teile aufzuteilen, die hier anhand von zwei Schemata dargestellt werden sollen.

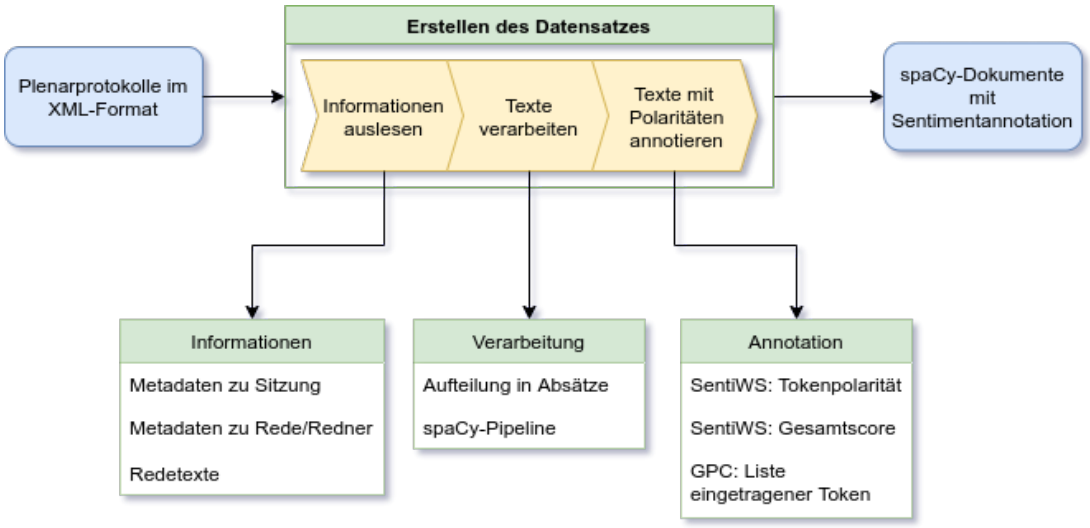

Abbildung 2: Der Arbeitsablauf zur Erstellung des Datensatzes inklusive aller nötigen Annotationen.

Erstens fällt die Erarbeitung des Datensatzes an, der alle zur Sentimentanalyse nötigen Informationen enthält. Für den Analyseprozess bzw. die Nutzung des Jupyter Notebooks als experimentelle Umgebung soll der Datensatz nur noch geladen werden, sodass z.B. die Polaritätswerte der Token einer Rede nicht mehr errechnet bzw. in den Lexika nachgeschlagen werden müssen, sondern einfach abgerufen werden können.

Abb. 2 zeigt den Arbeitsablauf für das Erstellen des finalen Datensatzes. Die ersten beiden Schritte des Erstellungsprozesses werden durch das Python-Skript `process_pp.py` übernommen, der letzte Schritt durch das Skript `dictionary_analysis.py`. Beide sind über das GitHub-Repositorium zum Projekt verfügbar und zusätzlich (unkommentiert) im Anhang zu dieser Arbeit zu finden.

Schlüssel	XML-Pfad	Beschreibung
wahlperiode	`kopfdaten/plenarprotokollnummer/wahlperiode`	Wahlperiode der Sitzung
sitzungsnr	`kopfdaten/sitzungstitel/sitzungsnr`	Nummer der Sitzung
ort	`kopfdaten/veranstaltungsdaten/ort`	Ort der Sitzung
datum	`kopfdaten/veranstaltungsdaten/datum[@date]`	Attribut: Datum der Sitzung

Tabelle 1: Metadaten zu den Protokoll-Elementen, eingetragen in jedem Rede-Element.

Schlüssel	XML-Pfad	Beschreibung
redner_id	`rede/p[@klasse='redner']/redner[id]`	Attribut: ID des Redners gem. Stammdatenverzeichnis
redner_name	`rede/p[@klasse='redner']/vorname + nachname`	Konkatenation aus Vor- und Nachname
redner_info	`rede/p[@klasse='redner']/redner/rolle/rolle_lang`	Besondere Rolle im Bundestag (z.B. Präsident)
redner_partei	`PARTEI_KURZ`	Parteizugehörigkeit (aus dem Stammdatenverzeichnis ausgelesen)
rede_id	`rede[@id]`	Attribut: ID der Rede gem. Sitzungsverzeichnis

Tabelle 2: Metadaten zu den Rede-Elementen.

3.2.1 Informationsextraktion aus XML-Dokumenten

Ausgehend von den XML-Dokumenten werden zuerst mit `lxml` die nötigen Informationen mithilfe von Listen und Wörterbüchern zwischengespeichert. Bei diesen Informationen handelt es sich um die Inhalte der XML-Tags, die in Tabellen 1 und 2 aufgelistet sind, Metadaten zweier verschiedener Kategorien.

Erstens beschreiben Metadaten auf Dokumentebene (Tabelle 1), die also jeweils zwischen einzelnen Plenarprotokollen variieren, einige Informationen, nach denen Reden später gruppiert werden können, z.B. könnte es interessant sein, die durchschnittliche Polarität von Reden chronologisch zu verfolgen.

Zweitens gibt es Metadaten zu jeder Rede (Tabelle 2), die Informationen über den Redner enthalten. Diese sollen dazu verfügbar sein, z.B. die Reden einzelner Mitglieder des Bundestages zu sammeln oder Vergleiche zwischen einzelnen Abgeordneten oder

denen verschiedener Parteien oder Fraktionen zu ziehen. Separat werden noch Zwischenrufe und Kommentare der Stenografen zum Geschehen im Plenarsaal in einer Liste mit dem Schlüssel `kommentare` eingetragen.

Im gleichen Verarbeitungsschritt werden auch die Texte der Reden extrahiert. Die Reden sind in Absätze eingeteilt, die jeweils von <p>-Tags umschlossen werden. Für die XML-Tags dieser Absätze sind im Handbuch der Dokumenttypdefinition (*Bundestags-Plenar-Protokolle im XML-Format: Aufbau der Strukturdefinition – DTD* 2015) drei Werte für das Attribut `klasse` verzeichnet, die normalen Text begleiten: <J>, <J_1> und <O>. <p klasse=J_1> ist das erste Textelement jeder Rede, so kann also automatisch der jeweilige Anfang gefunden werden. Zusätzlich werden auch mehrere Klassen, die Teile der offiziellen Tagesordnung einleiten, mit aufgenommen, da sonst womöglich der Sinn des Textes im Nachhinein nicht mehr zusammengesetzt werden könnte.

3.2.2 Weiterverarbeitung der Redetexte

Die Absätze einer jeweiligen Rede werden anschließend zu einem String zusammengesetzt und Übergänge zwischen ihnen durch Einfügen des Trennzeichens „ || “ signalisiert. Darauf folgt die Weiterverarbeitung dieses Strings mit spaCy. Dazu wird `de_core_news_sm` geladen und in einer Variable `nlp` bereitgestellt, sodass ein Text daraufhin einfach mit dem Befehl `nlp(<eingabetext>)` gemäß der in Abb. 1 notierten Teilschritte verarbeitet werden, resultierend in einem Objekt des Typs `Doc`, welches den (tokenisierten) Text und zahlreiche weitere Daten dazu enthält.

3.2.3 Annotation mit Polaritätswerten

Im dritten und letzten Schritt, mit dem der Datensatz finalisiert wird, werden für jeden Text die Polaritätswerte aus den beiden Sentimentlexika eingetragen. Das dafür geschriebene Skript `dictionary_analysis.py` ist in Anhang 2 zu finden. Dort werden auch noch vor den restlichen Prozessen die durch spaCy erratisch als Entitäten eingetragenen Trennzeichen (s. 3.2.2) entfernt und die bereinigte Liste von Entitäten separat verzeichnet. Diese Liste und die Informationen zu Polaritätswerten werden im dafür vorgesehenen Bereich `user_data` im jeweiligen `Doc`-Objekt gespeichert, wie Tabelle 3 beschreibt.

Zum Sammeln der Werte wird auf Tokenbasis gearbeitet, d.h. jedes `Token`-Objekt in jedem `Doc` wird mit den Lexika abgeglichen, was gleichermaßen rechenintensiv und

Schlüssel	Inhalt
`entitaeten`	Bereinigte Liste der durch spaCy extrahierten Entitäten
`sentiws`	Liste der aus SentiWS gefundenen Token mit Intensitätswerten
`sentiws[,,sentiment_score‘‘]`	Summe aller Intensitätswerte der gefundenen Token
`gpc[,,positiv‘‘]`	Liste der in GPC gefundenen Token mit positiver Polarität
`gpc[,,negativ‘‘]`	Liste der in GPC gefundenen Token mit negativer Polarität

Tabelle 3: Datenstrukturen zu Entitäten und Polaritätswerten, verzeichnet in `Doc.user_data`

unvermeidbar ist. Findet es sich in einem Lexikon, wird der Polaritätswert unter dem entsprechenden Schlüssel einer Liste hinzugefügt.

3.3 Aufbau des Korpus

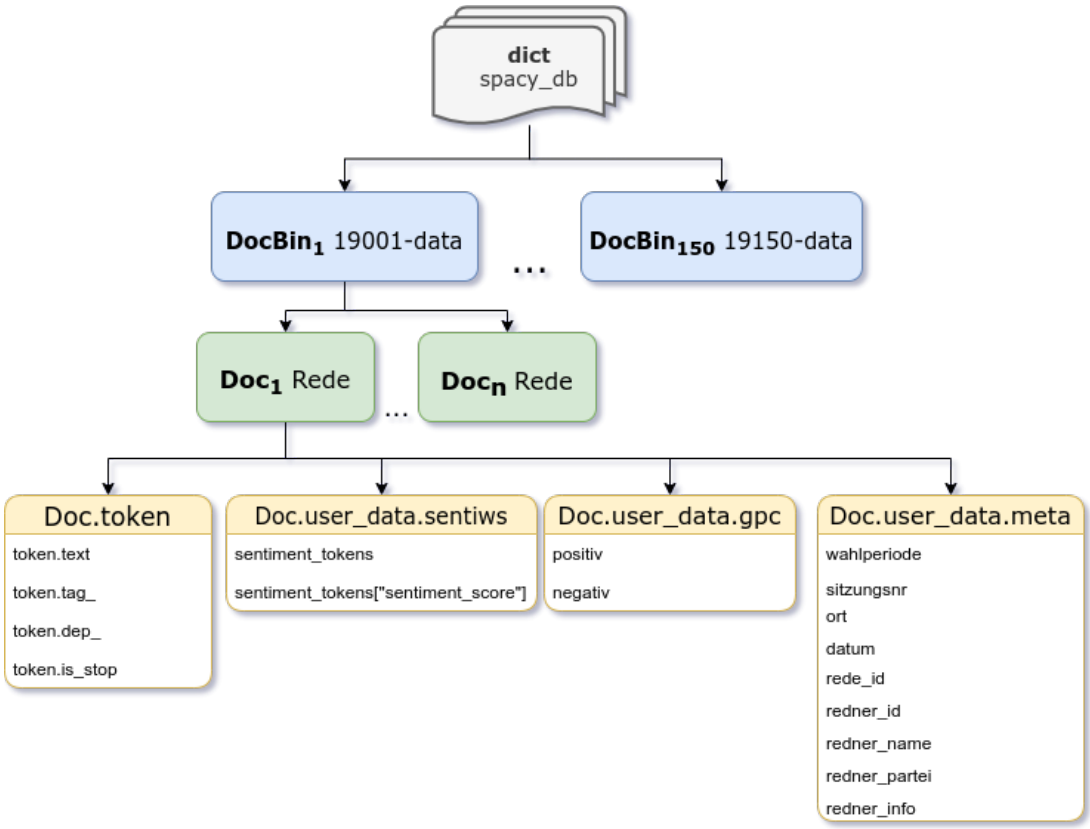

Abbildung 3: Die Organisationsstruktur der für die weitere Analyse relevanten Teile des Datensatzes.

Zum besseren Verständnis der Struktur relevanter Teile des Datensatzes ist dieser in Abb. 3 modelliert. Das oberste Objekt ist ein Wörterbuch mit den Kennzahlen der Plenarprotokolle als Schlüssel und deren Inhalt als Wert. Dieser Wert ist wiederum eine Liste von `Doc`-Objekten, ihre Anzahl entspricht der Anzahl der Reden im jeweiligen Protokoll.

Über die angegebenen Pfade kann innerhalb der `Docs` auf die gespeicherten Annota-

tionen zugegriffen werden, nachdem der Datensatz geladen wurde. Oft ist ein Iterieren über die spaCy-Objekte nötig, um Informationen zu sammeln.

In dem über das GitHub-Repositorium zu dieser Arbeit zu findenden Jupyter Notebook `sentiment_analysis.ipynb` findet sich der nötige Programmcode unter anderem für sämtliche Beispiele, die im nächsten Abschnitt kurz vorgestellt werden. Um diese auch am eigenen Computer ausführen zu können, muss das Repositorium lediglich geklont oder heruntergeladen werden.

4 Analysebeispiele

Im Folgenden sollen als eine Art Anhang einige kurze und keinesfalls wissenschaftliche Analyseansätze illustriert werden, die durch das Korpus möglich werden.

4.1 Beispiel 1: Durchschnittliche Polarität nach Parteizugehörigkeit

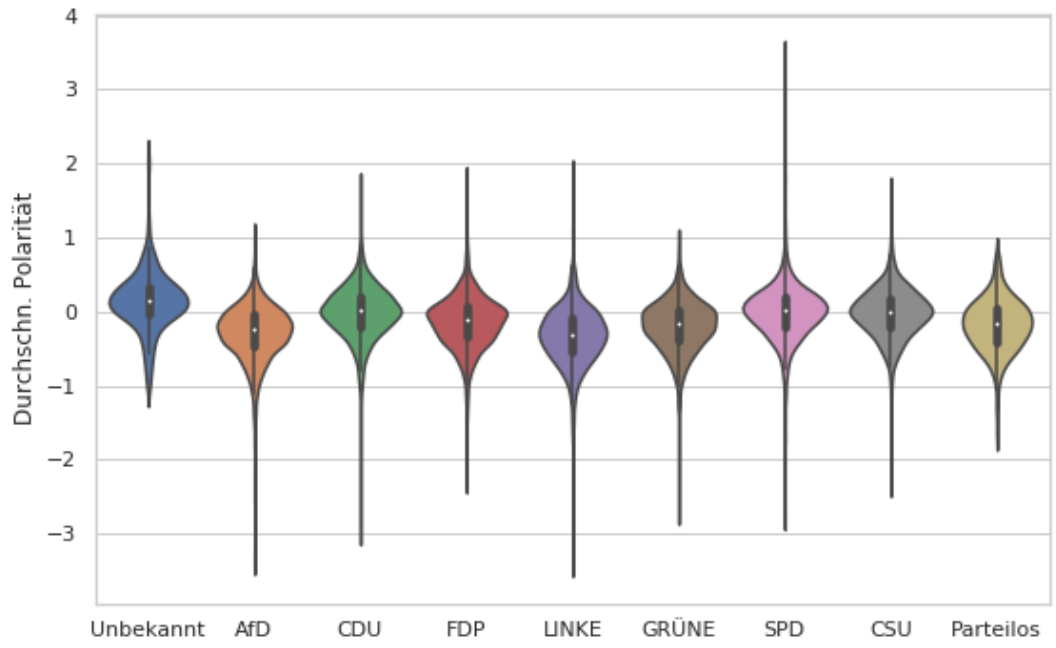

Abbildung 4: Durchschnittliche SentiWS-Intensitätswerte für Reden, nach Parteizugehörigkeit der Redner.

Abb. 4 zeigt die durchschnittlichen SentiWS-Intensitätswerte für Reden, aufgeteilt nach der Parteizugehörigkeit der Redner. Unter „Unbekannt“ fallen im Korpus alle Redner mit ungültiger ID, vor allem Mitglieder des Bundeskabinetts, deren IDs nicht ins Stammdatenverzeichnis eingetragen sind. An den Statistiken zu Mitgliedern anderer

Parteien lässt sich erkennen, dass die Polaritätswerte der Reden stark schwanken, besonders bei AfD und der Linken aber im Median deutlich unter 0 liegen. Aus diesen beiden Parteien kommen auch die insgesamt am negativsten gestimmten Reden. Von Mitgliedern der AfD und den Grünen sowie von parteilosen Rednern gab es außerdem keine Reden, die weit über einem Wert von 1 lagen, also ein besonders positives Sentiment kommunizierten. Reden von Abgeordneten der SPD zeigen die größte Variation.

4.2 Beispiel 2: Polarität von Reden mit Referenz zu Entitäten

Entität	Durchschn. Redepolarität
Aufstiegs-BaföG	2,18
Enquete-Kommission	1,82
Deutschland-Takt	1,71
Hightech-Strategie	1,59
Wolfgang Wiehle	1,57
Nicole Höchst	1,57

Tabelle 4: Die positivsten durchschnittliche Redewerte von Entitäten, die in mindestens 25 Rede-Elementen erwähnt wurden.

Tabelle 4 zeigt die mit den positivsten Reden verknüpften Entitäten im Korpus. Entitäten, die in weniger als 25 Rede-Elementen vorkommen, sind ausgenommen, um zu vermeiden, dass die Beiträge einzelner Redner zu sehr verzerren. Zwischen vielen der Entitäten lassen sich Verbindungen knüpfen, so stehen die ersten vier allesamt in Verbindung zu konstruktiven politischen Programmen z.B. zu Bildungsförderung und Nachhaltigkeit. Darauf folgen zwei Personalien, auf die betreffenden Reden würde sich entsprechend eine weitere Untersuchung lohnen.

Entität	Durchschn. Redepolarität
Walter Lübcke	-5,53
Nazis	-4,60
Bundesbank	-4,17
Katrin Göring-Eckardt	-4,01
Abgrund	-3,98
Herrn Maaßen	-3,90

Tabelle 5: Die negativsten durchschnittliche Redewerte von Entitäten, die in mindestens 25 Rede-Elementen erwähnt wurden.

Tabelle 5 zeigt gegenteilig die am negativsten konnotierten Entitäten. Hierzu zählen erneut mehrere Personalien, von denen zwei im Zusammenhang mit Rechtsterrorismus in der Diskussion standen, und kontextübergreifend negativ konnotierte Wörter wie „Nazis“

und „Abgrund“.

4.3 Beispiel 3: Polarität von Entitäten im Laufe der Zeit

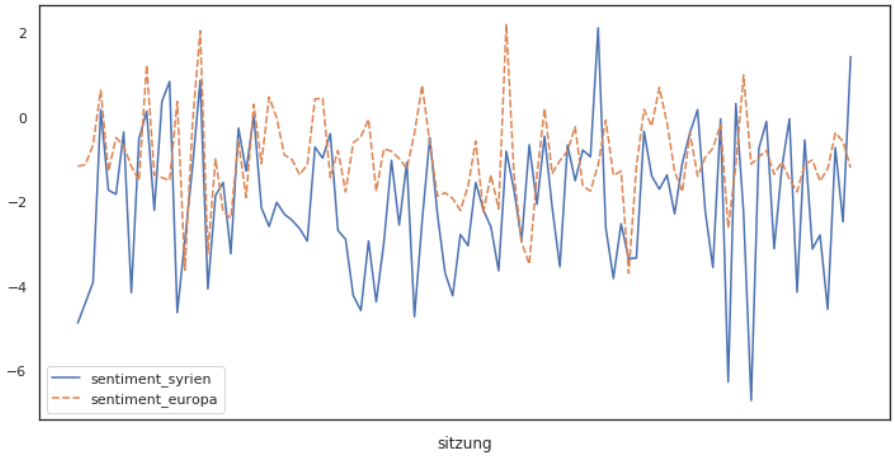

Abbildung 5: Die Polarität von Reden mit Referenz zu den Entitäten „Syrien“ und „Europa“ im Laufe der Sitzungen.

Abb. 5 zeigt die durchschnittliche Polarität von Reden, die die Entitäten *Syrien* bzw. *Europa* erwähnen, im Verlaufe der Sitzungen (und damit auch der Wahlperiode). Trotz des sehr unbeständigen Verlaufs lassen sich deutlich Perioden ausmachen, in denen *Europa* mit deutlich positiveren Reden verknüpft ist als *Syrien*. Zudem variiert die durchschnittliche Polarität von Reden zu Europa deutlich weniger stark als die von Reden, die Syrien erwähnen.

5 Fazit und zukünftige Perspektiven

Die XML-Dokumente des Bundestags haben sich als nützliche Ressource zur unkomplizierten Erstellung eines Korpus zur Textanalyse erwiesen. Mithilfe dieses weitgehend problemlos maschinell verarbeitbaren Formats lassen sich z.B. Probleme beim Parsen vermeiden, die sich in anderen Arbeiten zwangsweise ergaben (Blatte und Blessing 2018; Rauh und Schwalbach 2020).

Das in dieser Arbeit vorgestellte Korpus ist eine umfassende Ressource zur Sentimentanalyse von Bundestagsreden zu aktuellen Problemen der Politik, die viele nötige Annotationen bereits bereitstellt. Das ebenfalls verfügbare Jupyter Notebook ist eine

von vielen Möglichkeiten, mit denen sich das Korpus erkunden lässt. Die auf der spaCy-Bibliothek basierende Datenstruktur bietet zahlreiche Informationen, die sich in eine Analyse einbinden lassen.

Ein fortgeschrittenes Analysesystem konnte im Rahmen des Projekts leider bisher nicht gebaut werden, mithilfe der Daten können jedoch viele Möglichkeiten selbst realisiert werden, wie in 4 bereits angedeutet wurde. Als Ziele für die weitere Entwicklung des Projekts könnte z.B. das Speichern von Analyseergebnissen in Form einer XML-Datenbank in Frage kommen, die dann über eine Website erkundet werden können, ohne Code oder Rechenvorgänge zu benötigen.

Literatur

Abbasi, Ahmed und Hsinchun Chen (2007). „Affect intensity analysis of dark web forums“. In: *2007 IEEE Intelligence and Security Informatics*, S. 282–288.

Abercrombie, Gavin und Riza Theresa Batista-Navarro (2018). „‘Aye’ or ‘No’? Speech-level Sentiment Analysis of Hansard UK Parliamentary Debate Transcripts“. In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC’18)*, S. 4173–4180.

Baccianella, Stefano, Andrea Esuli und Fabrizio Sebastiani (2010). „Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining“. In: *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC’10)*, S. 2200–2204.

Blatte, Andreas und Andre Blessing (2018). „The GermaParl Corpus of Parliamentary Protocols“. In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC’18)*. LREC’18, S. 810–816.

Bollen, Johan, Huina Mao und Alberto Pepe (2011). „Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena“. In: *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*. Association for the Advancement of Artificial Intelligence, S. 450–453.

Breck, Eric und Claire Cardie (2017). „Opinion Mining and Sentiment Analysis“. In: *The Oxford Handbook of Computational Linguistics*. Hrsg. von Ruslan Mitkov. Oxford University Press. DOI: 10.1093/oxfordhb/9780199573691.013.43.

Bross, Juergen und Heiko Ehrig (2013). „Automatic construction of domain and aspect specific sentiment lexicons for customer review mining“. In: *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, S. 1077–1086.

Brunner, Katharina u. a. (2018). „Das gespaltene Parlament“. In: *Süddeutsche Zeitung*. URL: https://projekte.sueddeutsche.de/artikel/politik/die-afd-im-bundestag-e362724/ (besucht am 01. 04. 2020).

Bundestags-Plenar-Protokolle im XML-Format: Aufbau der Strukturdefinition – DTD (2015). Deutscher Bundestag.

Busch, Andreas und Madeline Kaupert (2018). „Die Regierungserklärungen deutscher Bundeskanzler von 1949 bis 2018 im Spiegel automatisierter Textanalyse“. In: *Zeitschrift für Politikwissenschaft* 28, S. 359–370.

Cambria, Erik u. a., Hrsg. (2017). *A Practical Guide to Sentiment Analysis*. Socio-Affective Computing 5. Springer.

Giachanou, Anastasia und Fabio Crestani (2016). „Like It or Not: A Survey of Twitter Sentiment Analysis Methods“. In: *ACM Computing Surveys* 49 (2), S. 1–41. DOI: 10.1145/2938640.

Goodfellow, Ian, Yoshua Bengio und Aaron Courville (2016). *Deep Learning*. MIT Press.

Grimmer, Justin und Brandon M. Stewart (2013). „Text as Data: The Promise and Pitfalls of Automatic Content Analysis Methods for Political Texts“. In: *Political Analysis* 21.3, S. 267–297.

Haselmayer, Martin und Marcelo Jenny (2017). „Sentiment analysis of political communication: combining a dictionary approach with crowdcoding“. In: *Quality and Quantity* 51, S. 2623–2646.

Hatzvassiloglou, Vasileios und Kathleen R. McKeown (1997). „Predicting the semantic orientation of adjectives“. In: *Proceedings of the 35th annual meeting of the Association for Computational Linguistics and eight conference of the European chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, S. 174–181.

Hurwitz, Roger (2020). *How the General Inquirer is used and a comparison of General Inquirer with other text-analysis procedures*. URL: `http://www.wjh.harvard.edu/~inquirer/3JMoreInfo.html` (besucht am 27. 03. 2020).

Jianqiang, Zhao, Gui Xiaolin und Zhang Xuejun (2018). „Deep Convolution Neural Networks for Twitter sentiment analysis“. In: *IEEE Access* 6, S. 23253–23260.

Koehn, Philipp (2005). „Europarl: A parallel corpus for statistical machine translation“. In: *MT Summit 5*, S. 79–86.

Kouloumpis, Efthymios, Theresa Wilson und Johanna Moore (2011). „Twitter sentiment analysis: The good the bad and the OMG!“ In: *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*. Association for the Advancement of Artificial Intelligence, S. 538–541.

Liu, Bing (2015). *Sentiment Analysis. Mining Opinions, Sentiments, and Emotions*. Cambridge University Press.

Mullen, Tony und Robert Malouf (2006). „A Preliminary Investigation into Sentiment Analysis of Informal Political Discourse“. In: *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, S. 159–162.

Nasukawa, Tetsuya und Jeonghee Yi (2003). „Sentiment Analysis: Capturing Faborability Using Natural Language Processing“. In: *Proceedings of the K-CAP-03, 2nd International Conference on Knowledge Capture*.

Pang, Bo und Lillian Lee (2004). „A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts“. In: *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics.

— (2008). „Opinion Mining and Sentiment Analysis“. In: *Foundations and Trends in Information Retrieval* 2.1-2, S. 1–135.

Pang, Bo, Lillian Lee und Shivakumar Vaithyanathan (2002). „Thumbs Up? Sentiment Classification Using Machine Learning Techniques“. In: *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP-2002)*.

Rauh, Christian (2018). „Validating a sentiment dictionary for German political language - a workbench note.“ In: *Journal of Information Technology & Politics* 15.4, S. 319–343.

Rauh, Christian und Jan Schwalbach (2020). *The ParlSpeech V2 data set: Full-text corpora of 6.3 million parliamentary speeches in the key legislative chambers of nine representative democracies*. Version V1. DOI: 10.7910/DVN/L4OAKN. URL: https://doi.org/10.7910/DVN/L4OAKN.

Remus, Robert, Uwe Quasthoff und Gerhard Heyer (2010). „SentiWS - A publicly available German-language resource for sentiment analysis“. In: *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC'10)*, S. 1168–1171.

Severyn, Aliaksei und Alessandro Moschitti (2015). „Twitter Sentiment Analysis with Deep Convolutional Neural Networks“. In: *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '15*. Association for Computing Machinery, S. 959–962.

Stone, Philip (1968). „The General Inquirer: A Computer Approach to Content Analysis.“ In: *Journal of Regional Science* 8.1, S. 113–116.

Thelwall, Mike, Kevan Buckley und Georgios Paltoglou (2011). „Sentiment in Twitter events“. In: *Journal of the American Society for Information Science and Technology* 62 (2), S. 406–418.

Truan, Naomi (2016). *Parliamentary Debates on Europe at the Deutscher Bundestag (1998-2015)*. ORTOLANG (Open Resources und TOols for LANGuage). URL: https://hdl.handle.net/11403/de-parl (besucht am 01. 04. 2020).

Turney, Peter D. (2002). „Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews“. In: *Proceedings of the 40th Annual Meeting of the Association ofr Computational Linguistics (ACL)*. 40th Annual Meeting of the Association for Computational Linguistics (ACL) (Philadelphia), S. 417–424.

van Dijk, Teut A. (1997). „What is political discourse analysis“. In: *Belgian Journal of Linguistics* 11 (1), S. 11–52.

Waltinger, Ulli (2010). „GERMANPOLARITYCLUES: A Lexical Resource for German Sentiment Analysis“. In: *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC)*. Valletta, Malta: electronic proceedings.

Wiebe, Janyce (1994). „Tracking point of view in narrative“. In: *Computational Linguistics* 20 (2), S. 233–287.

Wiebe, Janyce u. a. (2004). „Learning Subjective Language“. In: *Computational Linguistics* 30.3, S. 277–308.

Young, Lori und Stuart Soroka (2012). „Affective news: The automated coding of sentiment in political texts“. In: *Political Communication* 29 (2), S. 205–231. DOI: 10.1080/10584609.2012.671234.

Anhang

Skript 1: process_pp.py

```
1  from lxml import objectify as etree
2  import pickle
3  from glob import glob
4  from pprint import pprint
5  import spacy
6  from spacy.tokens import DocBin
7
8  daten = glob("plenarprotokolle/testing_prep/*.xml")
9  total = len(daten)
10 current = 1
11
12 stammdaten_parse = etree.parse("plenarprotokolle/MDB_STAMMDATEN.XML")
13 mdbs_ohne_daten = {}
14 nlp = spacy.load("de_core_news_sm")
15
16 redetext_klassen = ["J_1", "J", "O", "A_TOP", "T_Beratung", "T_Drs", "T_E_Drs",
       "T_E_E_Drs", "T_E_fett",
17                  "T_NaS", "T_NaS_NaS", "T_ZP_NaS", "T_ZP_NaS_NaS", "
                        T_ZP_NaS_NaS_Strich",
18                  "T_Ueberweisung", "T_fett", "T_ohne_NaS"]
19 redetext_kondition = "./p[@klasse='" + "' or @klasse='".join(redetext_klassen)
       + "']"
20
21
22 def get_partei(redner_id):
23     mdb_id = stammdaten_parse.xpath(".//ID[text()=%s]" % redner_id)[0]
24     mdb_daten = mdb_id.getparent()
25     mdb_partei = mdb_daten.find("./BIOGRAFISCHE_ANGABEN/PARTEI_KURZ")[0].text
26     return mdb_partei
27
28
29 for sitzung in daten:
30     print("[" , current , "/" , total , "]")
31     reden = []
32     spacy_reden = []
33
34     pp_parse = etree.parse(sitzung)
35     metadata = pp_parse.find(".//kopfdaten")
36     xml_reden = pp_parse.findall(".//rede")
37
38     wahlperiode = metadata.find("./plenarprotokoll-nummer/wahlperiode").text
39     sitzungsnr = metadata.find("./sitzungstitel/sitzungsnr").text
40     ort = metadata.find("./veranstaltungsdaten/ort").text
41     datum = metadata.find("./veranstaltungsdaten/datum").get("date")
42
43     for xml_rede in xml_reden:
44         redner_info = xml_rede.find("./p[@klasse='redner']/redner")
45
46         redetext = xml_rede.xpath(redetext_kondition)
47
48         for i in range(len(redetext)):
```

```
49              absatz = redetext[i]
50              if isinstance(absatz, str):
51                  pass
52              else:
53                  absatz = absatz.text
54                  redetext[i] = absatz
55          redetext = " || ".join([x for x in redetext if x])
56          redetext = nlp(redetext)
57
58          kommentare = xml_rede.xpath("./kommentar")
59
60          for i in range(len(kommentare)):
61              kommentar = kommentare[i]
62              if isinstance(kommentar, str):
63                  pass
64              else:
65                  kommentar = kommentar.text
66                  kommentare[i] = kommentar
67
68          redner_id = str(redner_info.xpath("@id")[0])
69
70          try:
71              rednername = redner_info.find("./name/vorname") + " " + redner_info.
                    find("./name/nachname")
72          except:
73              try:
74                  rednername = redner_info.find("./name/nachname")
75              except:
76                  try:
77                      rednername = redner_info.find("./name/vorname")
78                  except:
79                      print("Rednername in Datei", sitzung, "nicht auffindbar.")
80
81          try:
82              redner_partei = get_partei(redner_id)
83          except:
84              mdbs_ohne_daten[rednername] = redner_id
85              redner_partei = 'Unbekannt'
86
87          redetext.user_data["meta"] = {
88                  "wahlperiode": metadata.find("./plenarprotokoll-nummer/
                        wahlperiode").text,
89                  "sitzungsnr": metadata.find("./sitzungstitel/sitzungsnr").text,
90                  "ort": metadata.find("./veranstaltungsdaten/ort").text,
91                  "datum": metadata.find("./veranstaltungsdaten/datum").get("date"
                        ),
92                  "rede_id": str(xml_rede.xpath("@id")[0]),
93                  "redner_id": redner_id,
94                  "redner_name": rednername,
95                  "redner_info": str(redner_info.xpath("./rolle/rolle_lang")),
96                  "redner_partei": redner_partei
97              }
98          redetext.user_data["kommentare"] = kommentare
99
100         spacy_reden.append(redetext)
101
```

```
    pprint(mdbs_ohne_daten, stream=open("mdbs_ohne_daten.txt", "w+"))

    doc_bin = DocBin(attrs=["POS", "TAG", "LEMMA", "IS_STOP", "DEP", "SHAPE", "
        ENT_ID", "ENT_IOB", "ENT_KB_ID", "ENT_TYPE"], store_user_data=True)
    for doc in spacy_reden:
        doc_bin.add(doc)
    spacy_out = doc_bin.to_bytes()
    with open(file=(sitzung + ".xspacy"), mode="wb") as spacy_outfile:
        spacy_outfile.write(spacy_out)

    current += 1
```

Skript 2: dictionary_analysis.py

```
from pandas import read_csv
from glob import glob
import spacy
from spacy.tokens import DocBin

nlp = spacy.load("de_core_news_sm")

daten = glob("plenarprotokolle/pp19/*.xml.spacy")


spacy_db = {}
for datei in daten:
    protokoll = DocBin(store_user_data=True).from_bytes(open(datei, "rb").read
        ())
    protokoll = list(protokoll.get_docs(nlp.vocab))
    datei = datei.split("plenarprotokolle/pp19/")[1]
        spacy_db[datei] = protokoll

for f, protokoll in spacy_db.items():
    for rede in protokoll:
        rede.user_data["entitaeten"] = [x.text for x in rede.ents]
        rede.user_data["entitaeten"] = [x for x in rede.user_data["entitaeten"]
             if not x == "||"]


def collect_classifiers_sentiws(sourcefile):
    with open(sourcefile) as csv_file:
        classifiers = read_csv(csv_file, sep="\t", header=None, names=["lemma",
             "wert", "formen"])
        classifiers["formen"] = classifiers["formen"].astype(str)
        classifiers["formen"] = classifiers["formen"].apply(lambda x: x.split("
            ,"))
        classifiers[["lemma", "pos"]] = classifiers["lemma"].str.split("|",
            expand=True)
        classifiers["lemma"] = classifiers["lemma"].astype(str)
        for formen, lemma in zip(classifiers.formen, classifiers.lemma):
            formen = formen.append(lemma)
        classifiers = classifiers.explode("formen")
    return classifiers


def collect_classifiers_gpc(sourcefile):
```

```
38      with open(sourcefile) as csv_file:
39          classifiers = read_csv(csv_file, sep="\t", header=None,
40                                 names=["form", "lemma", "pos", "polaritaet", "
                                       probabilitaet", "whatever"])
41          classifiers = classifiers.drop(labels=["probabilitaet", "whatever"],
                axis=1)
42      return classifiers
43
44
45  classifiers_neg_sentiws = collect_classifiers_sentiws("SentiWS/SentiWS_v2.0
        _Negative.txt")
46  classifiers_pos_sentiws = collect_classifiers_sentiws("SentiWS/SentiWS_v2.0
        _Positive.txt")
47
48  classifiers_neg_gpc = collect_classifiers_gpc("gpc/GermanPolarityClues-
        Negative-21042012.tsv")
49  classifiers_pos_gpc = collect_classifiers_gpc("gpc/GermanPolarityClues-
        Positive-21042012.tsv")
50
51
52  def sentiws_eval(text):
53      sentiment_tokens = {}
54      sentiment_score = 0
55      for token in text.doc:
56          if token.is_stop is not True:
57              token_row = classifiers_neg_sentiws[classifiers_neg_sentiws.formen
                    == token.text]
58              if not token_row.empty:
59                  sentiment_tokens[token.text] = token_row["wert"].values[0]
60                  sentiment_score += token_row["wert"].values[0]
61              else:
62                  token_row = classifiers_pos_sentiws[classifiers_pos_sentiws.
                        formen == token.text]
63                  if not token_row.empty:
64                      sentiment_tokens[token.text] = token_row["wert"].values[0]
65                      sentiment_score += token_row["wert"].values[0]
66      sentiment_tokens["sentiment_score"] = sentiment_score
67      return sentiment_tokens
68
69
70  def gpc_eval(text):
71      positive_token = []
72      negative_token = []
73      sentiment_tokens = {}
74      for token in text.doc:
75          if token.is_stop is not True:
76              token_row = classifiers_neg_gpc[classifiers_neg_gpc.form == token.
                    text]
77              if not token_row.empty:
78                  negative_token.append(token.text)
79              else:
80                  token_row = classifiers_pos_gpc[classifiers_pos_gpc.form ==
                        token.text]
81                  if not token_row.empty:
82                      positive_token.append(token.text)
83      sentiment_tokens["positiv"] = positive_token
```

```
84      sentiment_tokens["negativ"] = negative_token
85      return sentiment_tokens
86
87  counter = 0
88  for f, protokoll in spacy_db.items():
89      counter += 1
90      print(counter)
91      for rede in protokoll:
92          rede.user_data["sentiws"] = sentiws_eval(rede)
93      for rede in protokoll:
94          rede.user_data["gpc"] = gpc_eval(rede)
95      doc_bin = DocBin(
96          attrs=["POS", "TAG", "LEMMA", "IS_STOP", "DEP", "SHAPE", "ENT_ID", "
                ENT_IOB", "ENT_KB_ID", "ENT_TYPE"],
97          store_user_data=True)
98      for doc in protokoll:
99          doc_bin.add(doc)
100     spacy_out = doc_bin.to_bytes()
101     with open(file=("plenarprotokolle/pp19/" + f + ".sentiment"), mode="wb") as
             spacy_outfile:
102         spacy_outfile.write(spacy_out)
```