Hochschule für Technik und Wirtschaft Berlin
Fachbereich 4

Themenspezifische Gruppierung deutscher Online-Zeitungen mit Natural Language Processing

Bachelorarbeit zur Erlangung des akademischen Grades
Bachelor of Science (B.Sc.), Internationaler Studiengang Medieninformatik

Georg Donner
Matrikel-Nummer 553821

Erstprüfer Prof. Dr. Gefei Zhang
Zweitprüfer Prof. Dr. Barne Kleinen

Abgabetermin: 04.02.2019

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Natural Language Processing ist ein großes Feld, welches besonders in der letzten Zeit im Zuge der Digitalisierung viel an Aufmerksamkeit und Wichtigkeit gewonnen hat. Es ermöglicht uns, Informationen schneller zu finden, Systeme durch gesprochene Sprache zu steuern oder ganze Texte zu generieren. Eine weitere häufige Aufgabe ist es, eine große Menge an Texten in Kategorien einzuteilen, um die Daten auf eine gewünschte Teilmenge für eine spezifischere Suche oder Analyse zu reduzieren.

1.1 Motivation

Diese Arbeit wird am Beispiel deutscher Online-Zeitungen untersuchen, welche Möglichkeiten es gibt, Texte unabhängig von ihrem Inhalt zu vergleichen. Dabei werden sowohl lexikalische, als auch syntaktische Merkmale extrahiert, um den Schreibstil der Texte zu repräsentieren. Das Ziel ist es, die Zeitungen nur unter Berücksichtigung dieser Merkmale zu gruppieren. Dabei wird mithilfe von Klassifizierung überprüft, wie gut sich diese Merkmale als Features für die Unterscheidung eignen. Die Auswahl von Features unabhängig des Inhaltes ermöglicht es, allgemeingültigere Aussagen über deren Effektivität zu treffen, die schwächer von den analysierten Texten abhängig sind. Hierfür werden nur Texte des gleichen groben Themas, wie z.B. Politik oder Sport, untereinander vergleichen, um den Einfluss des Inhaltes zu minimieren. Es besteht zudem die Vermutung, dass sich der Schreibstil einer Zeitung je nach Thema stark unterscheiden kann.

1.2 Related Work

Texte ihren Autoren zuzuordnen ist bereits Ziel unzähliger Forschungsarbeiten. Es werden verschiedene Methoden verwendet, um eine Erkennung der Autorschaft (engl. authorship attribution) zu erreichen und die Extraktion inhaltsunabhängiger Merkmale der Texte hat sich hier als sehr hilfreich erwiesen [Sta09]. Auch im Bereich der Erkennung des Genres, spielen diese Merkmale eine Schlüsselrolle zur korrekten Klassifizierung der Texte [CWD+17]. Die Erkenntnisse dieser beiden Bereiche werden in dieser Arbeit insofern erweitert, dass es sich bei der Zuordnung von Artikeln zu ihren Zeitungen um Texte verschiedener Autoren handelt, anstatt wie bei der authorshipp attribution um die eines einzelnen Autors. Zudem sind sich die Texte wesentlich ähnlicher als bei der Identifizierung des Genres. Es wird also überprüft, inwiefern sich diese Ansätze auch für heterogenere Gruppen von Texten wie Online-Zeitungen eignen.

2 Grundlagen

Für die Verarbeitung und Analyse von Zeitungsartikeln sind zwei Teilgebiete der Informatik besonders wichtig: Natural Language Processing und Machine Learning. Im Folgenden werden die für die Arbeit relevantesten Konzepte der beiden Bereiche genauer erklärt.

2.1 Natural Language Processing

Natural Language Processing ist ein Teilgebiet der Informatik, das Konzepte und Techniken der künstlichen Intelligenz und des Machine Learning verwendet, um natürliche Sprache zu verarbeiten. Der Begriff natürliche Sprache wird verwendet, um menschliche Sprachen zu beschreiben, die im Gegensatz zu künstlich entwickelten Plansprachen eine historische Entwicklung durchlebt haben. Diese Sprachen befinden sich in einem dauerhaften Entwicklungsprozess und sind häufig sehr variabel in ihrer Verwendung durch den Menschen. Die Analyse der Semantik eines Wortes oder Satzes ist für Computer besonders schwierig, da sich die Bedeutung häufig erst durch den Kontext ergibt.

Mit den schnellen Fortschritten im Bereich des Machine Learning in den letzten Jahrzehnten, eröffneten sich für die Verarbeitung natürlicher Sprache jedoch völlig neue Möglichkeiten. Die Erkennung von Syntax und Semantik wurde damit immer präziser und das Teilgebiet immer relevanter. Gegenwärtig basiert die Klassifizierung hauptsächlich auf Algorithmen des Supervised Learning, für die die Texte vor dem Trainingsprozess manuell mit relevanten Markierungen versehen werden müssen. Ein bekanntes Beispiel für einen Korpus deutscher Sprache mit solchen Annotationen ist der TIGER Corpus [1].

1 http://www.ims.uni-stuttgart.de/forschung/ressourcen/korpora/tiger.html

2.1.1 Pipeline

Bei der Analyse eines Textes werden in der Regel verschiedene Schritte abgearbeitet, die jeweils eigene Merkmale der Sprache untersuchen. Es entsteht eine sogenannte Pipeline, die je nach Anwendungsfall unterschiedlich aussieht. Das sequenzielle Ausführen dieser einzelnen Vorgänge ist notwendig, da beispielsweise die Analyse der Syntax voraussetzt, dass das Dokument bereits in Token zerlegt wurde. Im Folgenden werden die für diese Arbeit relevanten Schritte beschrieben.

Tokenisierung

Tokenisierung beschreibt den Prozess, einen gegebenen Text in gleiche Einheiten, sogenannte Token, zu zerlegen. Meistens handelt es sich dabei um Wörter und Satzzeichen, je nach Anwendungsfall können es aber auch Wortgruppen oder Sätze sein. Die Tokenisierung ist häufig eine Voraussetzung einer tiefgehenderen Analyse des Textes, daher ist eine hohe Genauigkeit hier besonders wichtig. Eine Teilung an jedem Satzzeichen um eine Liste von Sätzen zu erhalten, ist zum Beispiel eine triviale Lösung, die bereits gute Ergebnisse erzielt. Es müssen jedoch viele Sonderregeln wie Abkürzungen und Zahlen beachtet werden, sodass das Problem wesentlich komplexer ist, als es zunächst scheint. Ein Ansatz um höhere Genauigkeit zu erreicheren, ist ein Modell auf Basis bereits mit Annotationen versehener Korpora zu trainieren, welches die Regeln automatisch erlernt.

Part-of-speech-Tagging

Das Part-of-speech-Tagging, auch POS-Tagging, ist ein Verfahren, bei dem jedem Wort oder Satzzeichen die jeweilige Wortart zugeordnet wird. Bei der Analyse ist hierbei vor allem der Kontext in dem das Wort erscheint wichtig, da sich daraus häufig erst die Bedeutung ergibt. Die Informationen über die Wortart und oft auch weitere Details, geben sogenannte Tags, die meist aus einem festen Tagset stammen. In dieser Arbeit werden die Tags aus dem Stuttgart-Tübingen-Tagset (STTS) [2] verwendet. Der Satz „Martin findet eine grüne Blechdose.“ sieht nach dem POS-Tagging beispielsweise so aus:

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

Martin/NE findet/VVFIN eine/ART grüne/ADJA Blechdose/NN ./$.

Die Tags geben Auskunft über mehr als nur die Wortart. Zum Beispiel sind die beiden Wörter *Martin* und *Blechdose* jeweils Nomen. Da jedoch beim POS-Tagging auch die Definition des Wortes überprüft wird, kann *Martin* korrekt als Eigenname mit dem Tag NE identifiziert werden. Weiterhin wurde in diesem Satz die Verbform und der Adjektiv-Typ korrekt erkannt.

Dependency Parsing

Die Analyse der syntaktischen Struktur eines Satzes ist ein weiterer wichtiger Schritt in der Verarbeitung natürlicher Sprache. Beim Dependency Parsing wird zunächst jeder Satz nur auf seine Wörter reduziert, um dann die Beziehungen, sogenannte Dependency Relations, der Wörter innerhalb dieses Satzes zu bestimmen.

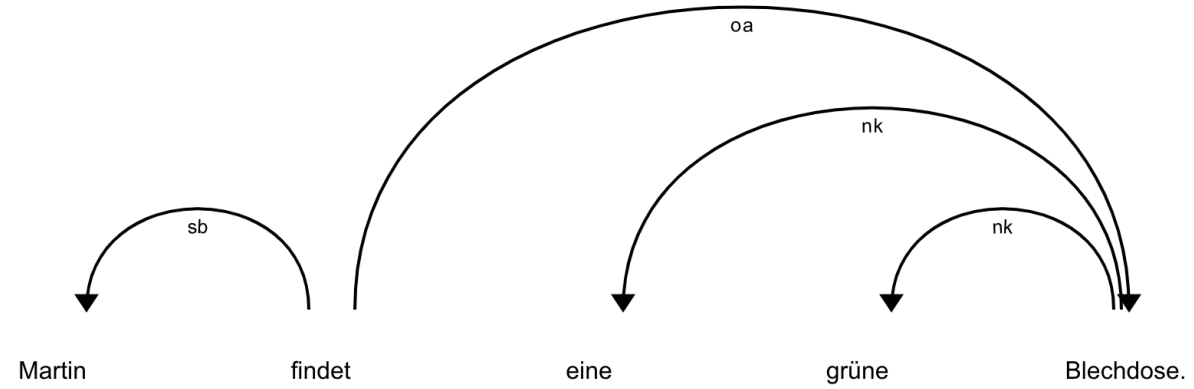


Abbildung 2.1: Visualisierung der Beziehungen mit Pfeilen

Diese Beziehungen ergeben letztendlich einen Baum, der navigiert werden kann und auch Aufschlüsse über die Komplexität eines Satzes zulässt. Die Ergebnisse des Dependency Parsing finden neben der Erkennung semantischer Beziehungen zwischen Wörtern auch noch weitere Anwendungen. So werden sie z.B. von dem Natural Language Processing Tool spaCy für die Erkennung der Satzenden verwendet [Exp19b].

Lemmatisierung

Für viele Analysen eines Textes ist es von Vorteil oder sogar notwendig, dass nicht jedes Wort, welches unterschiedlich geschrieben wird, als ein anderes Wort behandelt wird. Um dies zu erreichen, werden alle Wörter auf ihre Grundform reduziert. Dieser Prozess heißt *Lemmatisierung*. Dies ist besonders hilfreich für die Feststellung der Häufigkeit eines Wortes in einem Dokument oder Korpus. So wäre die Frequenz der Wörter *findet*, *fand* und *finden* zunächst jeweils eins. Nach der Lemmatisierung gibt es nur noch das Lemma *finden* mit einer Frequenz von drei. Dies hilft dabei, das Rauschen innerhalb eines Textes zu reduzieren und das tatsächliche Vokabular genauer zu beurteilen.

2.2 Machine Learning

Die meisten der zuvor in Kapitel 2.1.1 erläuterten Schritte basieren auf Machine Learning und auch in dieser Arbeit werden einige Verfahren aus diesem Bereich angewandt, um den Schreibstil der Texte zu ermitteln und zu analysieren. Das Ziel des Machine Learning ist es, aus bestehenden Beobachtungen ein Modell zu trainieren, welches in diesen ein Muster oder Zusammenhänge erkennen soll und anschließend Aussagen über eine bisher unbekannte Beobachtung treffen kann. In diesem Kapitel werden einige für diese Arbeit relevante Themenbereiche aus dem Machine Learning genauer beschrieben.

2.2.1 Feature Engineering

Alle Beobachtungen aus welchen ein zu trainierendes Modell lernt, haben immer eine bestimmte Anzahl an Features, die jede Beobachtung so gut wie möglich repräsentieren sollen. Die Auswahl dieser Features bildet die Grundlage des Trainings und ist ausschlaggebend für die Performance des Modells. Die Verwendung voneinander unabhängiger Features, die mit der vorherzusagenden Klasse in Beziehung stehen, ist anzustreben, um gute Resultaten zu erhalten [Dom12, Kap. 8]. Die Auswahl der richtigen Features ist also der Schlüssel für das erfolgreiche Lernen, aber es ist auch der schwierigste Teil. „Coming up with features is difficult, time-consuming, requires expert knowledge.“ [Ng13] Häufig werden viele Features generiert bzw. bestimmt, die möglicherweise nicht alle zielführend

oder unabhängig sind. Hier gibt es verschiedene Prozesse, um die Anzahl an Features zu reduzieren und gleichzeitig kaum an Informationsgehalt zu verlieren. Die Auswahl von Features aus der großen Menge heißt *Feature Selection*. Die Kombination von Features, um daraus neue zu generieren, wird *Feature Extraction* genannt. Diese Reduktion ist oftmals erforderlich um das statistische Rauschen in den Daten zu minimieren und beim Training Zeit zu sparen. Besonders bei sehr rechenintensiven Algorithmen ist dies absolut notwendig.

2.2.2 Dimensionsionalitätsreduktion

Bei der Generierung von d verschiedenen Features bzw. Variablen für n Beobachtungen, entsteht ein Raum der Dimension d. Um so viele Informationen wie möglich zu jeder Beobachtung zu speichern und für z.B. eine Klassifizierung zu verwenden, werden sehr viele Features generiert, welche wiederum einen hochdimensionalen Raum ergeben. Dabei entsteht das Problem, dass die Datenverarbeitung immer mehr Zeit beansprucht, da auch einerseits die Analyse einer einzigen Beobachtung länger dauert und andererseits mehr Beobachtungen notwendig sind um die steigende Dimensionalität zu kompensieren. Weiterhin führt es zu Schwierigkeiten bei der Bestimmung von Distanzen zwischen einzelnen Datenpunkten. In solch einem Raum ist es unvermeidbar, dass die Verteilung der Punkte sehr dünn ist. Darauf basierend wurde festgestellt, dass in einem hochdimensionalen Raum alle Punkte annähernd die gleiche Entfernung haben und die Suche nach dem nächsten Nachbarn nur sehr ungenau ist [HAK00, Kap. 1]. Eine häufige Annahme bei der Betrachtung dieser Räume ist, dass die Variablen nicht alle voneinander unabhängig sind und die Anzahl an Dimensionen reduziert werden kann, ohne dabei einen signifikanten Anteil an Informationen zu verlieren.

Ein dafür häufig verwendetes Verfahren ist die *Hauptkomponentenanalyse* oder englisch *Principal Component Analysis (PCA)*. Hierbei werden iterativ, je nach gewünschter Anzahl an Dimensionen auf die reduziert werden soll, in jedem Durchlauf orthogonale Achsen so ausgewählt, dass die Varianz auf der jeweiligen Achse am höchsten ist. Die Einheitsvektoren, welche die Achsen beschreiben werden auch *Hauptkomponenten* genannt [Gér17, S.213-214]. Bei einer Reduktion auf zwei oder drei Dimensionen können die Daten dann gut visualisiert werden, dabei gehen aber meistens zu viele Informationen verloren.

Für das Finden von Clustern eignet sich zudem das *t-Distributed Stochastic Neighbor Embedding* (t-SNE) Verfahren [MH08]. Die Dimensionalität der Daten werden hier reduziert, indem versucht wird, Beobachtungen, die sich im hochdimensionalen Raum ähneln, auch im niedrigdimensionalen Raum benachbart anzuordnen. Sich unterscheidende Beobachtungen werden hingegen weit voneinander entfernt dargestellt. T-SNE eignet sich damit gut dazu, Cluster innerhalb eines Datensatzes zu visualisieren [Gér17, S. 226].

2.2.3 Klassifizierung

Um die Performance der Features messen zu können, wird in dieser Arbeit Klassifizierung verwendet. Dafür werden alle Beobachtungen zunächst in ein Trainings- und ein Testset unterteilt, um das trainierte Modell nach dem Lernen mit den Testdaten beurteilen zu können. Nach dem Training ist die Aufgabe des Modells, jeder neuen Beobachtung aus dem Testset eine Klasse aus einer Liste bereits bekannter Klassen zuzuordnen. Die Klassen der jeweiligen Beobachtungen aus dem Trainingsset sind bekannt.

Falls es nur zwei mögliche Klassen gibt, handelt es sich um *Binäre Klassifikation*. Eine häufige Herangehensweise bei mehr als zwei Klassen ist es, die Aufgabe in mehrere binäre Probleme zu zerlegen, bei der nach der *One-vs.-rest* Strategie entschieden wird, welche Klasse zugewiesen wird. Dabei wird zur Klassifizierung einer Beobachtung zunächst für jede mögliche Klasse eine Wahrscheinlichkeit bzw. Sicherheit berechnet und anschließend die mit der höchsten Wahrscheinlichkeit ausgewählt.

Klassifizierung ist ein häufiges Problem, für das es aufgrund der diversen Anwendungsbereiche und damit völlig unterschiedlichen Anforderungen, keine Pauschallösung gibt. Die meisten vorhandenen Ansätze basieren auf der Erstellung einer linearen Funktion, in die der Featurevektor anschließend eingesetzt wird. Jedes Feature bekommt hierbei eine eigene Gewichtung, die sich aus dem vorherigen Training ergibt. Dieses Training unterscheidet sich je nach Algorithmus und auch der zurückgegebene Score wird je nach Algorithmus unterschiedlich interpretiert.

2.2.4 Logistische Regression

Einer dieser Trainingsalgorithmen ist die logistische Regression, welche in ihrer klassischen Form eine binäre Entscheidung darüber trifft, welche Klasse einer neuen Beobachtung zugeordnet wird. Dies wird erreicht, indem aus einem Trainingsset ein Vektor mit Gewichten für jedes Feature und ein Bias gelernt werden. Jedes Gewicht w_i steht dafür, wie wichtig das Feature x_i des Inputs für die Klassifizierung ist. Je höher der Betrag des Gewichtes ist, umso entscheidender ist es für die Klassifizierung. Die Wahrscheinlichkeit $\hat{y}$ für eine Klasse wird berechnet, indem das Skalarprodukt des Featurevektors x und des Vektors der Gewichte w zum Bias b addiert und anschließend in die Sigmoidfunktion eingesetzt wird, welche die Werte in den Wertebereich 0 - 1 skaliert:

$$\hat{y} = \frac{1}{1 + e^{-w \cdot x + b}}$$

Die Gewichte und das Bias werden optimiert, indem iterativ für jede Beobachtung im Trainingsset die vorhergesagte Wahrscheinlichkeit $\hat{y}$ mit der wirklichen Klasse y verglichen wird. Bei der logistischen Regression soll hierfür die Wahrscheinlichkeit der korrekten Klasse maximiert werden. Da es nur zwei mögliche Ergebnisse gibt, ergibt sich daraus folgende Berechnung der Wahrscheinlichkeit, welche maximiert wird:

$$p(y|x) = \hat{y}^y (1 - \hat{y})^{1-y}$$

Dies wird anschließend logarithmiert und das Vorzeichen umgekehrt, um eine Kostenfunktion zu bekommen, die minimiert werden soll. Das Ergebnis ist der Kreuzentropie-Verlust L_{CE}:

$$L_{CE}(\hat{y}, y) = -\log p(y|x)$$

Dieses Optimierungsproblem wird dann mit dem Gradientenverfahren gelöst. Dabei wird die Richtung des steilsten Anstiegs der aktuellen Position entlang der N Dimensionen annähernd bestimmt und anschließend die Gewichte und das Bias in die entgegengesetzte Richtung verändert. Da die Kostenfunktion konvex ist, ist das Finden des Minimums garantiert.
Mit der multinomialen logistischen Regression können auch Wahrscheinlichkeiten für mehr als nur zwei Klassen berechnet werden. Hierbei gibt es für jede Klasse je einen Vektor mit Gewichten und die Werte werden hier mit der Softmax-Funktion in den Wertebereich 0 - 1 skaliert. Daher muss für die Optimierung auch eine etwas modifizierte Kostenfunktion

verwendet werden. Die Gewichte zeigen anschließend pro Klasse, welche Features für die Entscheidung am ausschlaggebendsten waren, bei einem positiven Wert dafür und bei einem negativen Wert dagegen [JM, K. 5].

3 Datenverarbeitung

Die Grundlage dieser Arbeit bildet ein großer Datensatz mit Artikeln verschiedener Online-Zeitungen. Damit diese Artikel verglichen werden können oder eine Klassifizierung durchgeführt werden kann, müssen diese zunächst jeweils in einen Featurevektor umgewandelt werden. Dieses Kapitel beschreibt die verwendeten Tools und nötigen Schritte um dieses Ziel zu erreichen.

3.1 Verwendete Tools

3.1.1 Python

Python ist eine sehr universelle Programmiersprache und findet in den unterschiedlichsten Bereichen Anwendung. Die Syntax ist einfach zu erlernen und legt besonderen Wert auf gute Lesbarkeit, was wiederum die Produktivität der Programmierer erhöht [Fou19]. Des Weiteren gibt es eine große Auswahl an Bibliotheken für viele verschiedene Anwendungsfälle. Besonders in der wissenschaftlichen Arbeit hat sich die Verwendung von Python zusammen mit den Packages SciPy, NumPy, Matplotlib und Pandas bewährt. Auf Basis dieser Kombination entwickelte sich die Anaconda Distribution [1], welche die wichtigsten Bibliotheken für Data Science und Machine Learning beinhaltet und die Verwaltung dieser pro Projekt erleichtert. Die Verwendung einer solchen Distribution ermöglicht es, out-of-the-box Optimierungen mathematischer Berechnungen durchzuführen, unter anderem auf Basis der Intel™ Math Kernel Library (MKL). Dadurch konnte in dieser Arbeit zum Beispiel die Analyse der Artikel mit spaCy um etwa 27% beschleunigt werden.

1 https://www.anaconda.com/what-is-anaconda/

3.1.2 SpaCy

Auch für Natural Language Processing gibt es eine Vielzahl an Python Bibliotheken, die jedoch meist unterschiedliche Aufgaben erfüllen. *SpaCy* ist ein exzellentes Open-Source-Tool, welches die in Kapitel 2.1.1 beschriebenen Schritte der Pipeline vollständig abdeckt. Es legt besonderen Wert auf die Genauigkeit der Resultate und die Geschwindigkeit der Berechnungen. Dies wird unter anderem durch die Auswahl und Optimierung des passenden Algorithmus durch die Bibliothek erreicht, anstatt dieses Aufgaben dem Nutzer zu überlassen und verschiedene Algorithmen zur Verfügung zu stellen. SpaCy stellt trainierte Modelle für derzeit sieben verschiedene Sprachen zur Verfügung, weitere befinden sich bereits in der Entwicklung. Deutsch wurde im Jahr 2016 als erste Fremdsprache hinzugefügt und liefert trotz der reichhaltigeren Morphologie der Sprache sehr gute Ergebnisse [See16].

Eine häufig verwendete Alternative zu spaCy ist NLTK, kurz für Natural Language Toolkit. Es wurde in dieser Arbeit an einigen Stellen eingesetzt und für die Berechnung der linguistischen Merkmale in Betracht gezogen. Für die deutsche Sprache existiert hier jedoch kein fertiges Modell für das POS-Tagging, dieses müsste auf Basis eines annotierten Korpus selbst trainiert werden. Des Weiteren unterstützt NLTK kein Dependency Parsing, ein wichtiger Schritt der in dieser Arbeit verwendeten Pipeline. Die Verwendung und Ergebnisse dieser beiden Bibliotheken werden in Kapitel 3.3 genauer untersucht.

3.1.3 Scikit-learn

Scikit-learn ist eine Python Bibliothek, welche eine Vielzahl an effizienten Algorithmen aus dem Bereich des Machine Learning zur Verfügung stellt [PVG+11]. Sie baut auf den bereits in Kapitel 3.1.1 genannten Bibliotheken SciPy, NumPy und Matplotlib auf und bietet eine nahtlose Integration mit diesen. Zudem gibt es eine sehr ausführliche und einsteigerfreundliche Dokumentation, die über die Beschreibung der in der Bibliothek enthaltenen Funktionen hinaus auch ausführliche Tutorials beinhaltet.

3.2 Datenselektion

Bevor die gegebenen Daten für das Extrahieren von Features verwendet werden, ist es ratsam, zunächst einen Überblick über die Struktur zu bekommen. Häufig befinden sich fehlerhafte Daten im Datensatz und auch das Format entspricht nicht immer den Erwartungen oder ist unregelmäßig. Die in dieser Arbeit getroffenen Maßnahmen, um den Datensatz bereit für die Textverarbeitung zu machen, werden in diesem Kapitel genauer erläutert.

3.2.1 Datensatz

Der in dieser Arbeit verwendete Datensatz wurde von Ole Wendt erstellt und enthält etwas mehr als zehn Millionen Artikel in einem Zeitraum bis Juni 2018 für neun verschiedene Zeitungen. Tabelle 3.1 lässt jedoch erkennen, dass die Anzahl der Artikel je nach Zeitung sehr unterschiedlich ist. Für die Speicherung der Daten wurde das relationale Datenbanksystem *SQLite* gewählt.

Zeitung	Anzahl der Artikel
Ärzte-Zeitung	132753
Handelsblatt	942362
RP Online	1549537
SHZ	347335
Spiegel Online	620213
Der Tagesspiegel	975483
Westfälischer Anzeiger	318910
Welt	2980800
Zeit Online	2242198

Tabelle 3.1: Größe des Datensatzes

Die für diese Arbeit relevanten Datenfelder eines Artikels sind `content`, `url` und `date`, welche jeweils in Textform gespeichert sind. Es existieren weitere interessante Felder für z.B. den Autor oder die Zusammenfassung eines Artikels. Ein genauerer Blick in den

Datensatz ergibt jedoch, dass diese bei dem Großteil der Einträge fehlen und somit nicht für die Analyse genutzt werden können.

3.2.2 Normalisierung

Für die Normalisierung der Daten werden zunächst alle Artikel entfernt, die weniger als 100 Zeichen haben. Dies ist notwendig, da ein kleiner Teil der Artikel entweder gar keinen Inhalt besitzt oder nur eine kurze Eilmeldung ist.

Eine weitere Auffälligkeit bei der Betrachtung des Datensatzes ist, dass das Datum je nach Zeitung in einem unterschiedlichen Format gespeichert ist. Zudem müssen Artikel ohne Datum herausgefiltert werden. Die Sortierung der Artikel nach Datum funktioniert trotzdem sehr zuverlässig, sodass die Artikel erst sortiert und anschließend das Datum formatiert werden kann. Obwohl das Datum später nicht direkt weiterverarbeitet wird, ist ein einheitliches Format für die Visualisierung der Daten notwendig. In dieser Arbeit wird der ISO 8601 Standard verwendet. Für diesen Schritt ist die Python Funktion `datetime.strptime` sehr hilfreich, da damit auch Daten wie „1. März 2018, 9:11 Uhr“ einfach geparst werden können.

Die Klassifizierung der Zeitungen soll *themenspezifisch* erfolgen, das heißt jedem Artikel soll idealerweise ein Thema bzw. eine Kategorie zugeordnet werden. Dafür werden im Rahmen dieser Arbeit sehr grobe Themen wie Politik, Wirtschaft oder Sport angestrebt. Für die Bestimmung dieser Themen bietet sich die URL des Artikels an, da die Webseiten der Zeitungen, wie Abbildung 3.1 verdeutlicht, meist nach solchen Themen strukturiert sind. Dies spiegelt sich dann im Format der URL wider.

Je nach Zeitung ergeben sich hierbei unterschiedliche Themen und nicht jedem Artikel kann auf diese Weise zuverlässig ein Thema zugeordnet werden. Zudem unterscheidet sich die Anzahl der Artikel je nach Thema stark, sodass einige aufgrund zu kleiner Repräsentation aussortiert werden.

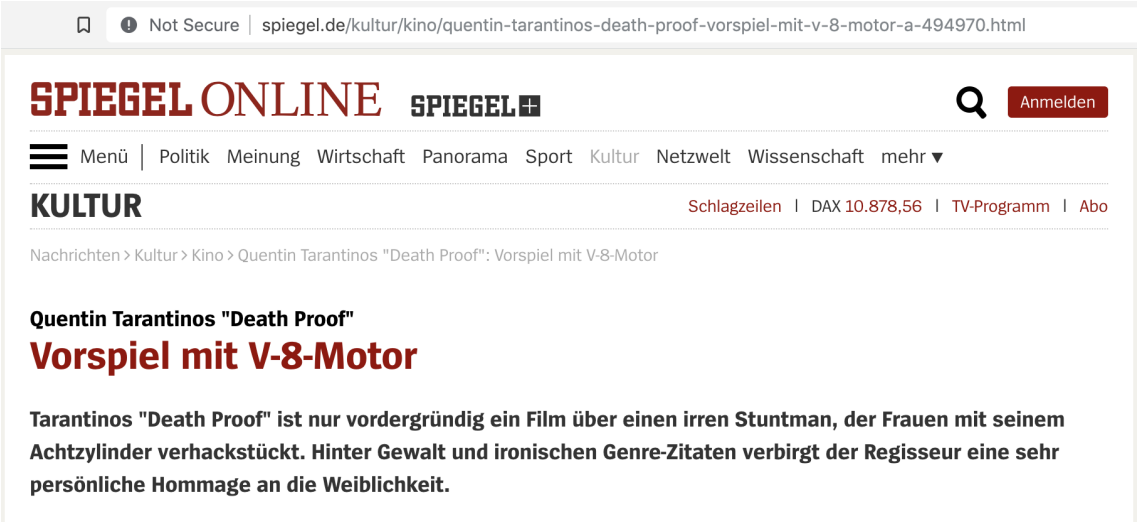


Abbildung 3.1: Die Struktur der Spiegel Online Webseite [Bor07]

3.3 Textverarbeitung

Bei der Textverarbeitung durchläuft eine Auswahl an Artikeln die in Kapitel 2.1.1 beschriebene Pipeline und die Ergebnisse werden anschließend für die weitere Verwendung zwischengespeichert.

Der erste Ansatz für die Verarbeitung der Texte war es, für jeden Schritt in der Pipeline ein explizit für dieses Verfahren entwickeltes Tool zu verwenden. Für die Tokenisierung wurde hier zunächst die populäre Bibliothek NLTK verwendet, welche bereits sehr gute Resultate bei annehmbarer Performance lieferte. Für das POS-Tagging wurde der von Helmut Schmid entwickelte TreeTagger [2] verwendet, welcher zusätzlich das Lemma jedes Wortes bestimmt. Dieser unterstützt eine Vielzahl an Sprachen, da pro Sprache lediglich eine Parameterdatei notwendig ist. Die Verwendung des TreeTaggers bringt jedoch auch einige Nachteile mit sich. Er wird nur als Perl Skript zur Verfügung gestellt und kann so nur über Umwege in die Python Pipeline integriert werden. Zudem ist die Startzeit des Skriptes pro Artikel sehr hoch, sodass die Berechnung zu zeitaufwendig wird. Dies kann umgangen werden, indem die Artikel zuvor zu einem großen Text zusammengefügt und anschließend wieder getrennt werden. Diese Methode führt zwar zu einer sehr hohen Performance, erhöht aber auch das Risiko für Ungenauigkeiten in den Ergebnissen. Ein weiteres Problem dieses Ansatzes ist, dass durch die Ausführung der Pipeline in einzelnen

2 http://www.cis.uni-muenchen.de/ schmid/tools/TreeTagger/

Schritten die Ergebnisse jedes Schrittes zwischengespeichert werden müssen und eventuell später zu groß für den Arbeitsspeicher werden.

Im zweiten Ansatz wurde mehr Wert auf die Zuverlässigkeit der Ergebnisse statt der Optimierung der Geschwindigkeit des Prozesses gelegt. Hierfür wurden zunächst 1000 Artikel pro Zeitung pro Thema verwendet, insgesamt 43000 Artikel, und mit der Bibliothek spaCy analysiert. Die Erstellung und Ausführung einer Pipeline mit spaCy ist sehr intuitiv und der initiale Aufwand ist sehr gering. Die Bibliothek bietet dennoch genügend Flexibilität, um beispielsweise die Performance zu optimieren.

TODO: Überblick spaCy Geschwindigkeit und Optimierungen

Die Anzahl der analysierten Artikel wurde zunächst beschränkt, um die Größe der Dateien, welche die Ergebnisse der Textverarbeitung beinhalten, gering zu halten. Für 43000 Artikel werden hier etwa 3,4 GB benötigt, indem nur die für diese Arbeit relevanten linguistischen Merkmale der Token gespeichert werden. Dies ermöglicht einen schnellen, iterativen Prozess für die Generierung, Auswahl und Bewertung der in Kapitel 3.4 beschriebenen Features. Nachdem das Featureset feststand, wurde die Anzahl der Artikel pro Zeitung pro Thema auf 25000 erhöht, falls so viele zur Verfügung standen, da vermutet wurde, dass eine Erhöhung der Anzahl an analysierten Artikel sich positiv auf die Genauigkeit der Klassifizierung und des Clustering auswirkt. Hierfür mussten die Ergebnisse der spaCy Pipeline nicht mehr zwischengespeichert werden, sondern konnten direkt für die Berechnung der Features weiterverwendet werden.

Nach der Analyse der Texte wurde eine weitere Filterung der Artikel vorgenommen, um den Anteil fehlerhafter Daten zu reduzieren. Es wurden hierfür jegliche Sätze mit weniger als vier Wörtern entfernt. Diese rigorose Filterung war notwendig da viele Artikel Wörter beinhalten, die nicht zum eigentlichen Inhalt gehören. Beispiele dafür sind Kürzel der Nachrichtenagentur, der Ort, der Autor, Quellen oder Verweise am Anfang oder Ende des Artikels.

Zudem wurden Artikel der Nachrichtenagentur dpa herausgefiltert, um nur die Artikel von eigenen Autoren der Zeitungen zu vergleichen. Wie in Abbildung 3.2 zu sehen, unterscheidet sich der Anteil je nach Zeitung sehr stark, ist allerdings auch abhängig von der Kategorie. So ist der Gesamtanteil an dpa Artikeln der *Zeit* etwa 25%, betrachtet man nur die Politik Artikel sind es nicht einmal 0,1%.

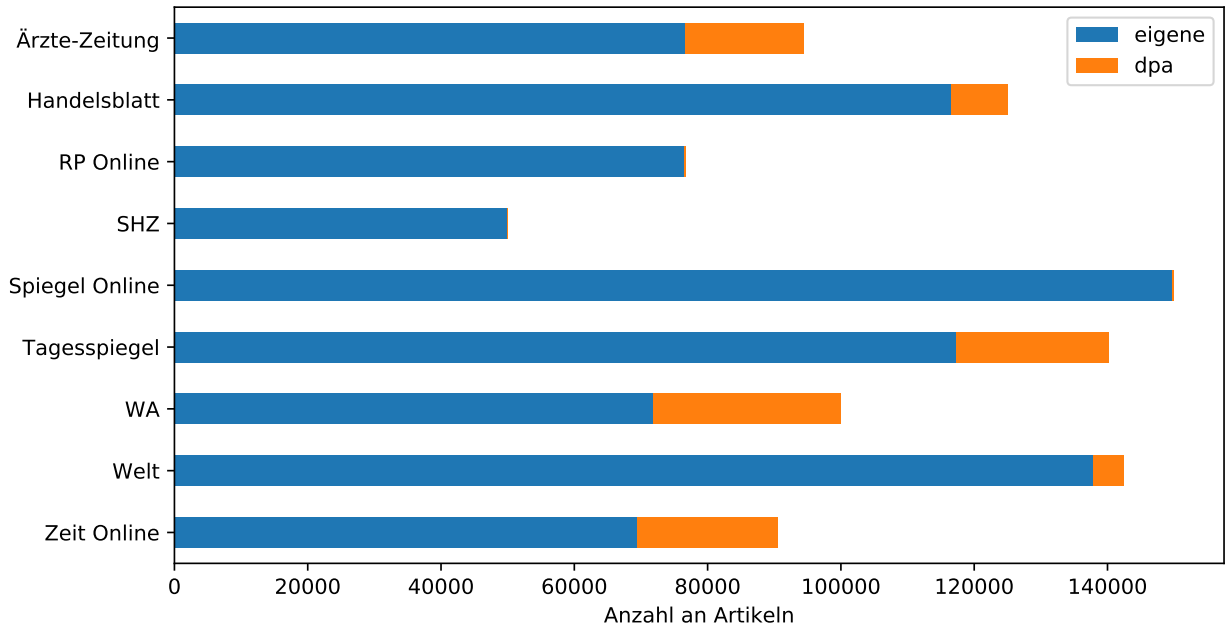


Abbildung 3.2: Anteil der dpa Artikel pro Zeitung

3.4 Featuregenerierung

Es gibt eine Vielzahl an Features, die für einen Text generiert bzw. ausgewählt werden können. In dieser Arbeit werden jedoch nur Features verwendet, welche unabhängig vom Inhalt der Texte sind. Auf diese Weise wird nur der *Schreibstil* einer Zeitung analysiert und die Auswahl der Features wird demnach für jeden weiteren Korpus einen sehr ähnlichen Einfluss besitzen. Besonders im Bereich der Erkennung der Autorschaft ist es ein häufiger Ansatz, die Texte anhand ihrer linguistischen Merkmale zu vergleichen. E. Stamatatos unterteilt diese weiter in lexikalische, syntaktische und semantische Features [Sta09]. Auch in der automatischen Erkennung von Textgenres, wie z.B. Literatur und Wissenschaft, wird geforscht, wie gut sich linguistische Merkmale für die Klassifizierung von Texten eignen [CWD+17]. Im weiteren Verlauf dieses Kapitels werden die in dieser Arbeit verwendeten Features genauer beschrieben und deren Verwendung begründet.

3.4.1 Lexikalisch

Für die Bestimmung lexikalischer Features ist lediglich die Tokenisierung eines Textes notwendig, sie erfordert also einen vergleichsweise geringen Aufwand. Die einfachsten Merkmale sind hierbei die durchschnittliche Satz- oder Wortlänge, welche schon seit den Anfängen der Forschung zur Erkennung der Autorschaft verwendet werden [SFK00, S. 473]. Obwohl sie nicht besonders allgemeingültig sind, spielen diese Merkmale doch immer noch eine Schlüsselrolle in der Unterscheidung von Textgenres [CWD+17] und sind daher auch in dieser Arbeit Teil des Featuresets.

Weiterhin existieren diverse Indizes, welche versuchen, die Lesbarkeit auf einer Skala von 0 (schwer) bis 100 (einfach) zu ermitteln oder dem Text eine Schulstufe zuzuordnen, die abgeschlossen sein muss, um den Text verstehen zu können. Diese berechnen sich meist aus einer Kombination der Anzahl an Silben, Wörtern und Sätzen im Text. Einer der ältesten und populärsten Indizes ist der 1948 veröffentlichte Flesch-Grad [MP82]. Dieser wurde zwar ursprünglich für die englische Sprache entwickelt und wäre damit nicht besonders gut geeignet für den in dieser Arbeit verwendeten Korpus, konnte aber inzwischen von Toni Amstad auf die deutsche Sprache übertragen werden [Rot10, K. 9]. Dieser berechnet sich wie folgt:

$$FG = 180 - \frac{W}{T} - (58,5 \cdot \frac{S}{W})$$

S ist die Gesamtzahl der Silben
W ist die Gesamtzahl der Wörter
T ist die Gesamtzahl der Sätze

Tabelle 3.2 zeigt, wie die Ergebnisse der Berechnung interpretiert werden sollen.

In dieser Arbeit soll überprüft werden, ob die Lesbarkeit eines Artikels zur Erkennung der Zeitung beiträgt und wird daher zunächst in das verwendete Featureset aufgenommen und später genauer analysiert.

Ein weiteres, sehr häufig gemessenes Merkmal eines Textes ist die Reichhaltigkeit des Vokabulars. Ein typisches Beispiel hierfür ist die Type-Token-Relation V/N, wobei V für die Gesamtzahl der einzigarten Wörter und N für die Gesamtzahl aller Wörter steht [Sta09, S. 540]. Für ein genaueres Resultat, vor allem bei kurzen Texten, ist es empfehlenswert, die Lemmata der Wörter zu verwenden, was zu einer Erhöhung der Frequenzen der

Flesch-Grad	Lesbarkeit	Bildungsstand
0-30	Sehr schwer	Studium abgeschlossen
30-50	Schwer	Studierend
50-60	Mittelschwer	10. - 12. Klasse
60-70	Standard	8. - 9. Klasse
70-80	Mitteleinfach	7. Klasse
80-90	Einfach	6. Klasse
90-100	Sehr einfach	5. Klasse

Tabelle 3.2: Flesch-Grad Bewertungen und äquivalenter Bildungsstand [CH02, S. 406]

einzelnen Wörter und damit zu einer Verminderung des Rauschens des Textes führt. Es zeigte sich leider, dass dieses und weitere Maße für die Reichhaltigkeit des Vokabulars abhängig von der Länge des Textes sind und besonders für kürzere Texte an Aussagekraft verlieren [TB98]. A. Cimoni et al. zeigten allerdings, dass die Type-Token-Relation für die Erkennung bestimmter Genres eine wichtige Rolle spielt [CWD+17, S. 5] und auch für die Erkennung der Autorschaft wird dieses Maß immer noch verwendet [Sta09, S. 540]. Deshalb wird die Type-Token-Relation in dieser Arbeit dennoch in das Featureset aufgenommen.

Die Frequenzen der Wörter oder Lemmata eines Textes zu berechnen, ist ein sehr einfacher, aber doch sehr typischer Ansatz, um den gegebenen Text in einen Featurevektor umzuwandeln. Um diese miteinander vergleichen zu können, muss vorher bekannt sein, welche Wörter dafür insgesamt berücksichtigt werden. Eine sehr simple Methode hierfür ist das Bag-of-words Model, bei dem für jeden Text die Frequenz jedes im gesamten Korpus vorkommenden Wortes oder Lemmas berechnet wird. Kombiniert mit einem Index wie dem Tf-idf-Maß, liefert diese Darstellung eines Textes auch für die Erkennung der Autorschaft gute Ergebnisse [Seb02, S. 12]. Mit dieser Herangehensweise wird neben dem Schreibstil aber auch maßgeblich der Inhalt des Textes repräsentiert, weswegen sie in dieser Arbeit nicht in Betracht gezogen wird. Hier ergibt es Sinn, nur für Synsemantika, also Wörter die keine inhaltliche Relevanz besitzen, die Frequenzen zu berechnen. Auf diese Weise ist die Analyse des Textes weniger vom Inhalt abhängig und die verwendeten Wörter sind repräsentativer, da sie von jedem Autor genutzt werden. Zudem wird argumentiert, dass die Verwendung von Synsemantika durch den Autor weniger kontrolliert ist und somit den Schreibstil stärker prägen [Kes14, S. 60-61]. Die Verwendung dieser Frequenzen als

Features hat sich sowohl im Bereich der Erkennung der Autorschaft, als auch des Genres als sehr nützlich erwiesen [CWD+17, Sta09, ZLCH06, AL05]. Dafür werden in Anlehnung an [CWD+17] in dieser Arbeit die 100 häufigsten Lemmata des Korpus bestimmt und anschließend für jeden Artikel die Frequenzen errechnet.

Es gibt viele weitere Ansätze, Texte nur mithilfe von lexikalischen Merkmalen in einen Featurevektor umzuwandeln. Eine sehr bewährte Methode ist es, statt ganzen Wörtern oder Lemmata, N-Gramme der Buchstaben als Features zu verwenden [Sta09, S. 542]. So wird bei der Berechnung von Trigrammen zum Beispiel das Wort *Rausch* in *Rau*, *aus*, *usc* und *sch* zerlegt. Untersuchungen von E. Stamatatos ergaben, dass diese Methode selbst bei der Verwendung von Trainings- und Testkorpora zu einem jeweils unterschiedlichen Thema, noch gute Ergebnisse erzielt und N-Gramm-Frequenzen der Buchstaben als Features wesentlich robuster als Wortfrequenzen sind [Sta13]. Ein hiermit einhergehendes Problem ist jedoch die unglaublich hohe Dimensionalität der daraus resultierenden Featurevektoren. Für das Clustering oder die Gruppierung von Artikeln sind diese Ansätze damit weitaus schlechter geeignet als für die Klassifizierung.

3.4.2 Syntaktisch

Ein wesentlich aufwendigeres Verfahren um weitere Features zu generieren, ist die Analyse der Syntax. Es wird argumentiert, dass sich Autoren unbewusst syntaktische Muster beim Schreiben angewöhnen und die Syntax damit den Schreibstil zuverlässiger repräsentiert als lexikalische Merkmale [Sta09, S. 542]. Auch für die Erkennung von Genres wurde festgestellt, dass syntaktische Merkmale eine Schlüsselrolle spielen und eine Verwendung dieser, die Klassifizierungsgenauigkeit signifikant erhöht [CWD+17].

Für die Bestimmung syntaktischer Merkmale eines Textes ist allerdings nicht nur die Tokenisierung notwendig, sondern auch das POS-Tagging. Da dieses Verfahren nicht komplett akkurat ist, wird durch die Verwendung darauf aufbauender Features möglicherweise weiteres Rauschen hinzugefügt, welches die Ergebnisse beeinträchtigen kann. Da in dieser Arbeit nur Zeitungsartikel analysiert werden, welche kaum syntaktische Fehler enthalten und die Genauigkeit für das POS-Tagging von spaCy derzeit bei 97.15% liegt [Exp19a], wird davon ausgegangen, dass die Verwendung syntaktischer Features keinen ausschlaggebenden negativen Einfluss hat. Es wurden außerdem die Ergebnisse des Dependency Parsing mit in Betracht gezogen, da diese einen noch tieferen Einblick in die Struktur der

Sätze erlauben. Das Risiko des Rauschens ist aufgrund der geringeren Genauigkeit von 89.75% von spaCy [Exp19a] jedoch hier wesentlich höher.

Ähnlich zu den Wortfrequenzen als lexikalisches Merkmal, werden in dieser Arbeit auch für die POS-Tags die Frequenzen für jeden Artikel berechnet, hierbei spricht man auch von POS-Monogrammen. Analysen von A. Cimoni ergaben, dass sich die Verwendung von Bigrammen der POS-Tags im Vergleich zu Monogrammen nicht signifikant auf die Genauigkeit der Klassifizierung auswirkt [CWD+17], daher wurde diese Möglichkeit in dieser Arbeit ebenfalls nicht weiter erforscht. Es ist wichtig zu erwähnen, dass die Frequenzen der einzelnen Wortarten für jeden Artikel teilweise nur sehr gering sind. Hier erhöht sich die Aussagekraft der Frequenzen als Features ebenfalls mit der Länge des Artikels.

Die lexikalische Dichte eines Textes ist sein Verhältnis von Autosemantika bzw. Inhaltswörtern zu der Gesamtzahl an Wörtern. Eine höhere lexikalische Dichte bedeutet, dass der Text die enthaltenen Informationen präziser und verständlicher wiedergibt. Für die Berechnung des Anteils der Autosemantika ist das POS-Tagging die Grundlage, da in der deutschen Sprache die Wortart bestimmt, ob ein Wort eine eigene lexikalische Bedeutung besitzt. Diese Wortarten sind Substantive, Verben, Adjektive und Adverbien.

Weiterhin werden die Monogramme der beim Dependency Parsing berechneten Dependency Relations als Features verwendet, indem die jeweiligen Frequenzen pro Artikel bestimmt werden. Auch hier ist es möglich, Bigramme zu berechnen, allerdings würde das Einbeziehen dieser in die Featurevektoren deren Dimensionalität beträchtlich erhöhen. Zudem würde es die Frequenzen dieser Features verringern, was auch in diesem Fall dazu führen würde, dass kürzere Texte schlechter klassifiziert werden können.

3.4.3 Featureset

In dieser Arbeit werden insgesamt 201 Features verwendet und untersucht. Tabelle 3.3 gibt einen Überblick über alle in dieser Arbeit verwendeten Features.

1.	Durchschnittliche Satzlänge
2.	Flesch-Grad
3.	Type-Token-Relation
4.	Lexikalische Dichte
5-59.	Frequenzen der Monogramme aller Wortarten (POS-Tags)
60-101.	Frequenzen der Monogramme aller Dependency Relations
102-201.	Frequenzen der 100 häufigsten Lemmata des Korpus

Tabelle 3.3: Überblick über das gesamte Featureset

4 Datenauswertung

Das Ziel dieser Arbeit ist es, zu untersuchen, inwiefern die Zeitungen gruppiert werden können. Der Vergleich der Zeitungen erfolgt dabei anhand des ausgewählten Featuresets. Bevor jedoch ein Clustering bzw. eine Gruppierung der Artikel oder Zeitungen durchgeführt werden kann, ist es notwendig, zunächst die Performance der Features zu überprüfen. Hierbei wird untersucht, inwiefern die Features ausreichend sind, um Zeitungsartikel ihrer zugehörigen Zeitung nach zu unterscheiden. Um dies auswerten zu können, wird eine Klassifizierung der Artikel durchgeführt und die Ergebnisse mit verschiedenen Methoden ausgewertet. Für das Clustering wird zunächst überprüft, ob Cluster entstehen, wenn jeder Artikel aller Zeitungen als eigener Wert verwendet wird. Weiterhin werden die Durchschnitte der Artikel einer Zeitung berechnet und anschließend für das Clustering verwendet. Die Klassifizierung wird *themenspezifisch* durchgeführt, folglich werden nur Artikel eines bestimmten Themas zum Training eines dazugehörigen Models verwendet. Dadurch wird sichergestellt, dass der Inhalt der Texte eine nur sehr untergeordnete Rolle spielt. Zudem wird überprüft, ob es je nach Thema Unterschiede in der Performance der Features gibt.

4.1 Überblick

Um einen Überblick über die Verteilung der Artikel zu bekommen, werden zunächst einige ausgewählte Features auf ihre Verteilung und mögliche Korrelationen überprüft. Dafür bieten sich die Features 1. bis 4. an, da sie unabhängig von anderen Features bereits einige Aussagekraft besitzen und ihre Werte am besten vom Menschen interpretiert werden können.

Die Tabelle 4.1 zeigt die Korrelationen der Features, indem der Korrelationskoeffizient berechnet wird. Je höher der Betrag des Koeffizienten, umso stärker ist die Korrelation. Ein

	Satzlänge	Flesch-Grad	Type-Token-Relation	Lexikalische Dichte
Satzlänge	1,00	-0,39	-0,01	-0,03
Flesch-Grad	-0,39	1,00	-0,25	-0,30
Type-Token-Relation	-0,01	-0,25	1,00	0,29
Lexikalische Dichte	-0,03	-0,30	0,29	1,00

Tabelle 4.1: Pearson-Korrelation der Features 1. bis 4.

Wert von unter 0,5 signalisiert eine schwache Korrelation, ab 0,8 wird von einer starken Korrelation der Variablen gesprochen [BCK12, S. 268-273].

Die Tabelle 4.1 zeigt bereits, dass die Variablen untereinander nicht alle unabhängig sind. Dennoch sind alle Korrelationskoeffizienten nicht besonders hoch und signalisieren damit nur einen schwachen Zusammenhang zwischen den Variablen. Da die durchschnittliche Satzlänge Teil der Formel für den Flesch-Grad ist, ist die Abhängigkeit hier besonders offensichtlich. Diese Abhängigkeit hat den höchsten Koeffizienten mit -0,4, die Artikel werden hier jedoch trotzdem weit verteilt liegen. Dies wird durch Abbildung 4.1 sehr gut verdeutlicht. Hier ist jeder der ingesamt 865797 analysierten Artikel in das Diagramm eingezeichnet. Dabei hat jeder Datenpunkt einen Alphawert von 0,005, ist also sehr transparent. Da bei dieser Darstellung der Features keine Linie oder Ähnliches entsteht, ist von ausreichender Unabhängigkeit auszugehen.

In Abbildung 4.2 wurden nur die Artikel mit dem Thema Politik berücksichtigt und erneut jeder Artikel mit dem gleichen Alphawert wie in Abbildung 4.1 eingezeichnet. Die schwarzen Kreuze im Diagramm symbolisieren jeweils den Durchschnitt der Features aller Artikel einer Zeitung. Es ist bereits deutlich erkennbar, dass sich die Durchschnitte der Variablen pro Artikel einer Zeitung nicht sehr voneinander unterscheiden. Lediglich die Artikelmenge der *Ärzte-Zeitung* einen sichtbar anderen Durchschnittswert für den Flesch-Grad auf, der jedoch in Anbetracht der weiten Verteilung der Daten auch für diese Zeitung bei Weitem kein zuverlässiges Unterscheidungsmerkmal ist.

Aus diesen Erkenntnissen lässt sich bereits vermuten, dass die Klassifizierung und das Clustering der Artikel nur auf Basis dieser Features nicht erfolgreich sein wird. Daher ist es unbedingt notwendig, weitere Features zu verwenden, um die Artikel gruppieren zu können. Mit steigender Anzahl der Features lässt sich deren Unabhängigkeit jedoch nicht mehr

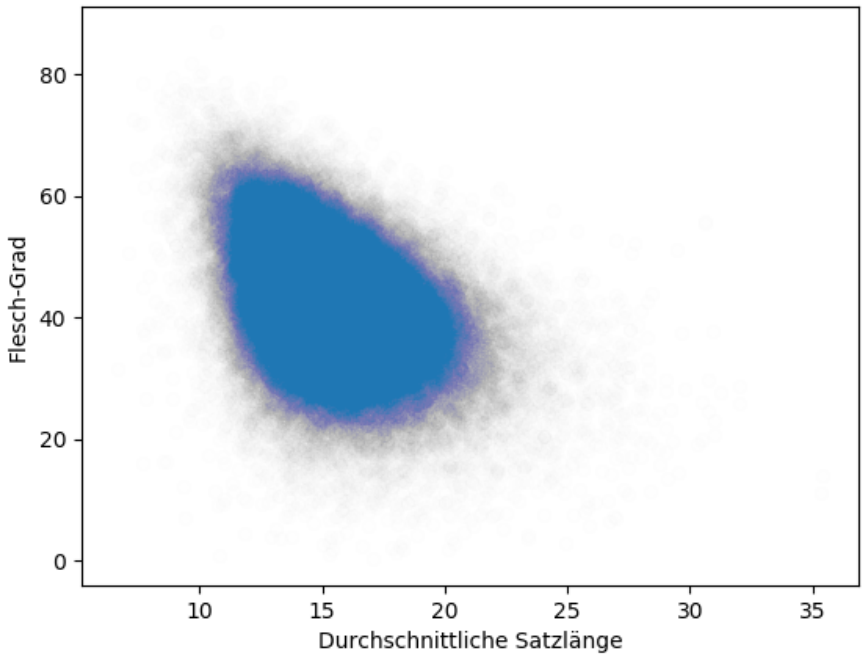


Abbildung 4.1: Korrelation der durchschnittlichen Satzlänge und dem Flesch-Grad

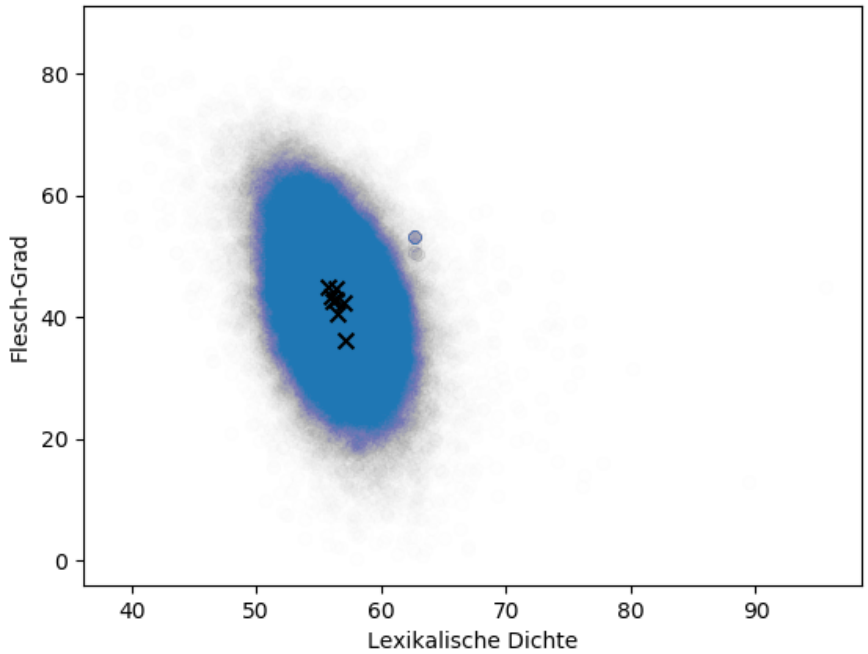


Abbildung 4.2: Korrelation der lexikalischen Dichte und dem Flesch-Grad und Durchschnitte der Zeitungen für alle Politik-Artikel

mit bloßem Auge erkennen, weshalb im weiteren Verlauf die Klassifizierungsgenauigkeit als Performancemaß verwendet wird.

4.2 Klassifizierung

Die Klassifizierung wird auf einem Datensatz mit maximal 25000 Artikeln pro Zeitung pro Thema durchgeführt. Es wird vermutet, dass sich mit einer größeren Menge an Daten auch die Genauigkeit der Klassifizierung erhöht. Die Ergebnisse werden anschließend anhand verschiedener Performanceindikatoren bewertet. Zudem wird überprüft, welche Features besonders geeignet sind und inwiefern eine Reduktion der Featureanzahl das Ergebnis beeinflussen.

4.2.1 Vorbereitung des Datensatzes

Die Skalen der Features 1. bis 4. unterscheiden sich sowohl voneinander als auch von den restlichen auf Frequenzen beruhenden Features. Daher ist es notwendig, dass vor deren Verwendung eine Standardisierung aller Features durchgeführt wird. Hierbei werden diese so transformiert, dass für jedes Feature die Varianz 1 und der Erwartungswert 0 beträgt. Die Standardisierung der Beobachtungswerte $x_1, \ldots, x_n$ mit einer positiven empirischen Standardabweichung s_x und dem arithmetischen Mittel $\bar{x}$ wird wie folgt berechnet [BCK12, S. 104]:

$$z_i = \frac{x_i - \bar{x}}{s_x}, i \in \{1, \ldots, n\}$$

Für das Training werden 70% der Daten verwendet. Die restlichen 30% ergeben das Testset, welches für die Evaluierung des Klassifikators verwendet wird. Da nicht für jede Zeitung bei jedem Thema die gleiche Anzahl an Artikeln für die Klassifizierung zur Verfügung steht, wird zudem untersucht, ob sich Undersampling hier positiv auf das Ergebnis auswirkt. Undersampling ist der Prozess, die Anzahl der Daten jeder Klasse auf die Anzahl der am geringsten repräsentierten Klasse zu reduzieren. Dies resultiert in einem ausgeglichenen Datenset, welches je nach verwendetem Algorithmus zu zuverlässigeren Ergebnissen führen kann.

Weiterhin werden vor der Klassifizierung Artikel herausgefiltert, welche fehlerhafte Daten besitzen, wie z.B. einen negativen Flesch-Grad. Aufgrund der Abhängigkeit vieler Features

von der Länge des Textes, wurden zudem nur Artikel mit mehr als 100 Wörtern für die Analyse berücksichtigt. Dies dient erneut dem Herausfiltern nicht repräsentativer Artikel wie Eilmeldungen.

4.2.2 Feature Selection

Zunächst werden alle Features herausgefiltert, welche nach der Standardisierung keine Varianz von 1 besitzen. Ein abweichender Wert für die Varianz kann nach der Standardisierung nur vorkommen, wenn alle Artikel für das Feature den gleichen Wert haben.

Um den Einfluss der verschiedenen Gruppen an Features auf die Genauigkeit der Klassifizierung zu untersuchen, werden sieben Featuresets getestet. Diese werden in Tabelle 4.2 vorgestellt und sind für die bessere Referenzierung mit einem Label versehen.

Label	Features	Anzahl an Features[1]
pos	Features 4.-59.	55
dep	Features 60.-101.	42
lemma	Features 102.-201.	97
-pos	Alle Features außer *pos*	144
-dep	Alle Features außer *dep*	156
-lemma	Alle Features außer *lemma*	100
alle	Alle Features	198

Tabelle 4.2: In dieser Arbeit verwendete Featuresets

Für die Analyse einzelner Features wird *Recursive Feature Elimination* (RFE) verwendet. Dies funktioniert mit einem linearen Klassifikator, wie der logistischen Regression, welcher den Features Gewichte zuordnet. Dabei werden zunächst alle Features verwendet und anschließend die mit dem geringsten Einfluss auf die Klassifizierung eliminiert. Dies wird rekursiv wiederholt, bis die gewünschte Anzahl an Features erreicht ist [Sl18, Seite: sklearn.feature_selection.RFE]. In den Untersuchungen dieser Arbeit wird dabei pro

1 Die Anzahl der Features wird *nach* der Aussortierung von Features mit einer Varianz von weniger als 1 gemessen

Durchlauf genau ein Feature eliminiert, da dabei keine Kombinationen von Features übersprungen werden.

4.2.3 Verfahren

Bevor eine Klassifizierung durchgeführt werden kann, muss zunächst erarbeitet werden, welche Verfahren dafür verwendet werden sollen. Da Featuresets für die Erkennung der Autorschaft üblicherweise aus tausenden Features bestehen, werden hierfür Algorithmen eingesetzt, welche mit diesen hochdimensionalen Daten in einer annehmbaren Laufzeit gute Ergebnisse erzielen. Hierbei gilt es Overfitting zu vermeiden. Overfitting liegt dann vor, wenn das Model zur Klassifizierung zu sehr auf die Trainingsdaten spezialisiert ist. Neue Daten werden dann falsch klassifiziert, da keine Generalisierung des Problems vorliegt und das System nur die Ergebnisse zu den Trainingsdaten erlernt hat. Es zeigte sich, dass die Support Vector Machine (SVM) hierfür eine der besten Methoden ist [LZC06, Sta09].

In dieser Arbeit soll die Klassifizierung jedoch auch für die Beurteilung der Features verwendet werden. Voraussetzung dafür ist, dass die Gewichtungen der Features gut interpretiert werden können. Das ist besonders bei linearen Klassifikatoren der Fall, da sie den Score einer Klasse berechnen, indem das Skalarprodukt aus einem Vektor mit Gewichtungen und dem Featurevektor gebildet wird. Je nach Verfahren gibt der Vektor der Gewichtungen anschließend Aufschluss darüber, welches Feature wie stark zum Ergebnis der Klassifizierung beiträgt. Für die Beurteilung der Gewichtungen in einem Multiklassenproblem eignet sich eine Maximum-Entropie-Methode [CWD$^+$17, K. 2], wie die multinomiale logistische Regression. Ein weiterer Vorteil linearer Klassifikatoren ist, dass diese weniger rechenintensiv als eine SVM sind und somit auch für zehntausende Stichproben eine vergleichsweise kurze Laufzeit haben. Die Laufzeit der nicht-linearen SVM skaliert mehr als quadratisch mit dem Stichprobenumfang, sodass solch eine Datenmenge selbst für nur einen Durchlauf schwer zu verarbeiten ist [Sl18, Seite: sklearn.svm.SVC].

Die Genauigkeit der Klassifizierung wird außer als Maß der Performance der Features noch dazu verwendet werden, zu überprüfen, wie gut sich die Artikel nach zugehöriger Zeitung unterscheiden lassen. Daraus kann bereits abgeleitet werden, inwiefern eine anschließende sinnvolle Gruppierung der Artikel überhaupt möglich ist.

In dieser Arbeit wird multinomiale logistische Regression, auch Maximum-Entropie-Klassifikator genannt, als Verfahren für die Klassifizierung verwendet. Der größte Vorteil hierbei ist die gute Interpretationsmöglichkeit der Gewichtungen. Je höher der Betrag einer Gewichtung ist, umso entscheidender ist die Rolle, die sie bei der Berechnung des Scores einer Klasse spielt. Als linearer Klassifikator ist er zudem auch für den hohen Stichprobenumfang von etwa 100000 Artikeln immer noch gut geeignet.

4.2.4 Messung der Performance

Für die Evaluierung der Ergebnisse einer Klassifizierung gibt es diverse Metriken. Besonders bei einer Klassifizierung mit mehr als zwei Klassen ist es wichtig, die Performance pro Klasse beurteilen zu können. Dafür ist es notwendig, dass mehr als nur die Anzahl an korrekt klassifizierten Beobachtungen, sondern auch auch die *false positives* mit in die Metrik einfließen. Einen guten Überblick gibt hier die Wahrheitsmatrix, welche, wie in Tabelle 4.3 zu sehen, zeilenweise die vorhergesagten Klassen und spaltenweise die wirkliche Klasse zeigt. Aus dieser Matrix können zwei wichtige Maße ermittelt werden. Die Trefferquote (recall) gibt an, wie viele der Beobachtungen einer Klasse korrekt klassifiziert wurden. Die Genauigkeit (precision) gibt an, wie viele der Vorhersagen für eine Klasse tatsächlich korrekt sind. Für den Spiegel ergibt sich hier eine gute Trefferquote von 0,9. Die Genauigkeit ist mit rund 0,64 jedoch vergleichsweise schlecht. Das F_1-Maß, auch F-Maß genannt, kombiniert diese beiden Metriken, um dieses Verhältnis mit nur einem Maß aussagekräftig beurteilen und damit zwei Klassen oder Klassifikatoren gut vergleichen zu können [Gér17, S. 88-89]:

$$F_1 = \frac{2 \cdot (precision \cdot recall)}{precision + recall}$$

In diesem Beispiel hat der Spiegel folglich ein F-Maß von rund 0,75. Um das F-Maß zur Evaluierung von Klassifizierungsmodellen mit mehr als zwei Ergebnissen anzuwenden, gibt es zwei Ansätze. Es können einerseits alle einzelnen Klassifizierungen für die Berechnung des F-Maßes verwendet werden (*micro*), sodass eine ungleiche Verteilung der Klassen das Ergebnis nicht verzerrt. Es ist jedoch auch möglich, den Durchschnitt der F-Maße der einzelnen Klassen zu berechnen (*macro*).

		wirkliche Klasse		
		Spiegel	Welt	Zeit
	Spiegel	9	5	2
vorhergesagte Klasse	Welt	1	12	0
	Zeit	0	1	5

Tabelle 4.3: Beispiel einer Wahrheitsmatrix

4.3 Ergebnisse

4.3.1 Stichprobenumfang

Zunächst wird die Auswirkung des Stichprobenumfangs auf die Zuverlässigkeit der Klassifizierung untersucht. Hierfür wird das Thema *Politik* verwendet, da 8 der 9 Zeitungen Artikel zu diesem Thema verfasst haben und die Menge der zur Verfügung stehenden Artikel hier am größten ist. Für jeden ausgewählten Stichprobenumfang, werden 10 Durchläufe ausgeführt und anschließend sowohl das durchschnittliche als auch das beste *micro* F-Maß als Metrik für die Zuverlässigkeit verwendet. Für die Klassifizierung wird ausschließlich logistische Regression verwendet, da die im Rahmen dieser Arbeit zur Verfügung stehenden Rechenkapazitäten nicht ausreichen, um mit einer SVM in angemessener zeit Ergebnisse zu erzielen.

Wie in Abbildung 4.3 zu sehen, hat der Stichprobenumfang nur bei einer sehr niedrigen Anzahl an Artikeln einen Einfluss auf die Zuverlässigkeit der Klassifizierung. Ab 2000 Artikeln pro Zeitung, also insgesamt 16000 Artikeln, können keine relevanten Unterschiede in der Genauigkeit mehr festgestellt werden. Auffällig ist zudem, dass das F-Maß des Besten der 10 trainierten Klassifikatoren für 500 Artikel pro Zeitung höher als bei über 7000 Artikeln pro Zeitung ist. Da die Artikel jedes Mal zufällig ausgewählt werden, ist es möglich, dass bei diesem Durchlauf ein besonders gut zu klassifizierendes Set von Artikeln ausgewählt wurde. Dafür spricht auch der Fakt, dass das durchschnittliche F-Maß für diesen Stichprobenumfang deutlich schlechter ist als in den Durchläufen mit 2000 oder mehr Artikeln. Insgesamt wird also deutlich, dass sich die Robustheit des Klassifikators mit einem größeren Stichprobenumfang erhöht und die Ergebnisse damit zuverlässiger sind. Die Erwartung, dass sich mit der Erhöhung des Stichprobenumfangs

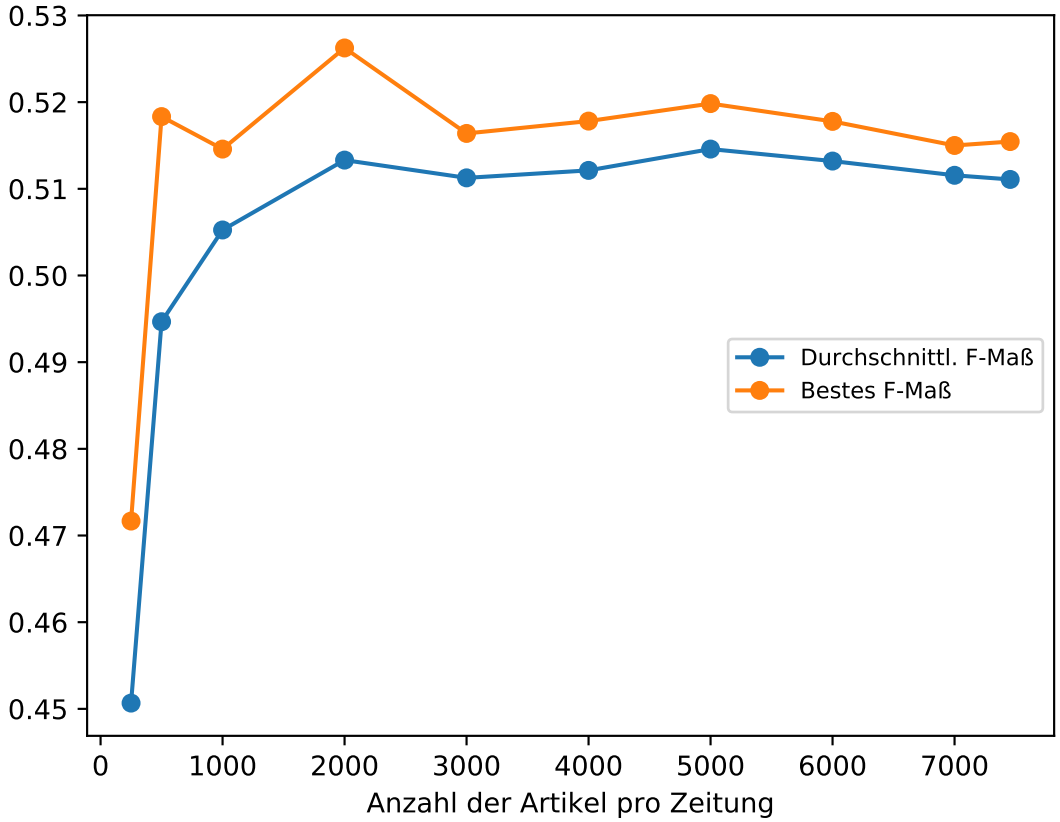


Abbildung 4.3: Zusammenhang zwischen Stichprobenumfang und Zuverlässigkeit der Klassifizierung

auch die Performance des Klassifikators verbessert, konnte allerdings nur bis zu einem unerwartet geringen Umfang bestätigt werden. Ein durchschnittliches F-Maß von 0,51 bedeutet zudem, dass die Artikel mit diesem Featureset nicht besonders gut voneinander unterschieden werden können und eine zuverlässige Klassifizierung aller Artikel nicht möglich ist.

4.3.2 Undersampling

Aus der Artikelanzahl der Zeitung *RP Online* von etwa 7500 zum Thema Politik, ergibt sich die maximal verwendbare Anzahl an Artikeln für das Trainings des Modells. Diese Zeitung hat von allen Zeitungen die wenigsten Politikartikel. Um ein Bias gegen *RP Online*

auszuschließen, wird die Artikelanzahl der anderen Zeitungen deshalb auf 7500 reduziert. Für die meisten Zeitungen sind allerdings mehr als doppelt so viele Politik-Artikel im Datensatz enthalten. Daher wird im Folgenden überprüft, ob die Verwendung aller Politik-Artikel trotz ungleicher Verteilung die Zuverlässigkeit der Klassifizierung erhöhen kann. Dafür wird sowohl das *micro* als auch das *macro* F-Maß mithilfe der multinomialen logistischen Regression bestimmt und evaluiert.

	F-Maß	
	micro	macro
Mit Undersampling	0,5125	0,5078
Ohne Undersampling	0,5245	0,4766

Tabelle 4.4: Vergleich der Performance mit und ohne Undersampling der Daten

Die Tabelle 4.4 zeigt, dass sich die Performance des Klassifikators leicht verbessert, wenn alle Daten verwendet und kein Undersampling mehr angewandt wird. Das deutlich schlechtere *macro* F-Maß ist jedoch ein Indikator dafür, dass nicht jede Klasse gleich gut klassifiziert werden kann. Das liegt meist daran, dass Beobachtungen von Klassen mit geringem Support wesentlich schlechter und seltener erkannt werden und damit vor allem die Trefferquote für Klassen mit geringem Support sehr schlecht ausfällt. Support steht hierbei für die Anzahl der Beobachtungen einer Klasse in den Testdaten. Er gibt zudem eine Annäherung für die Performance des Klassifikators, in dem Fall, dass in den Testdaten jede Klasse in etwa gleicher Zahl vertreten ist. Bei Betrachtung der individuellen F-Maße der Zeitungen eines Durchlaufes in Tabelle 4.5 kann dies bestätigt werden. Folglich hat Undersampling einen positiven Einfluss auf die Zuverlässigkeit der Klassifizierung und ermöglicht eine unverzerrte Evaluierung der Performance.

4.3.3 Unterschiede pro Zeitung

Weiterhin stellt sich heraus, dass die Genauigkeit und Trefferquote während der Klassifizierung je nach Zeitung sehr unterschiedlich ausfallen kann. Tabelle 4.6 zeigt für einen Durchlauf, dass die *Ärzte-Zeitung* weit besser von den restlichen Zeitungen unterschieden werden kann. Dies kann dadurch begründet werden, dass diese auch in ihren Politik-Artikeln überwiegend medizinische Themen behandelt und daher einen komplexeren Schreibstil verwendet. Innerhalb der restlichen Zeitungen gibt es ebenfalls Unterschiede in

	Genauigkeit	Trefferquote	F-Maß	Support
Ärzte-Zeitung	0.76	0.82	0.78	5542
Handelsblatt	0.52	0.58	0.55	6519
RP Online	0.32	0.11	0.16	2221
Spiegel Online	0.50	0.56	0.53	7384
Tagesspiegel	0.47	0.37	0.42	6122
WA	0.37	0.24	0.29	4304
Welt	0.47	0.50	0.48	7400
Zeit Online	0.53	0.65	0.58	7404

Tabelle 4.5: Performance der Klassifizierung der Zeitungen ohne Undersampling

	Genauigkeit	Trefferquote	F-Maß
Ärzte-Zeitung	0.80	0.84	0.82
Handelsblatt	0.52	0.60	0.56
RP Online	0.47	0.42	0.44
Spiegel Online	0.45	0.47	0.46
Tagesspiegel	0.47	0.40	0.43
WA	0.40	0.33	0.36
Welt	0.45	0.47	0.46
Zeit Online	0.50	0.59	0.54

Tabelle 4.6: Performance der Klassifizierung der Zeitungen mit Undersampling

der Zuverlässigkeit der Klassifizierung. Das *Handelsblatt* und die *Zeit Online* werden ein wenig besser und der *Westfälische Anzeiger* etwas schlechter als der Durchschnitt klassifiziert. Insgesamt ist die Klassifizierung der Zeitungen mit dem in dieser Arbeit verwendeten Featureset jedoch sehr unzuverlässig. Dies kann einerseits dadurch begründet werden, dass die Artikel der Zeitungen von vielen verschiedenen Autoren geschrieben werden und sich daraus kein kohärenter Schreibstil identifizieren lässt. Andererseits ist es möglich, dass der Schreibstil einer Zeitung durch die verwendeten Features nicht akkurat genug repräsentiert wird. Aus diesem Grund wird der Einfluss der einzelnen Features in Kapitel 4.3.5 genauer untersucht.

4.3.4 Unterschiede pro Thema

Die Einteilung der Artikel in vordefinierte, grobe Themenbereiche erlaubt es, die Zuverlässigkeit der Klassifizierung je nach Thema zu beurteilen. Hierbei ist es jedoch wichtig zu erwähnen, dass aufgrund sehr unterschiedlicher Verteilungen der Artikel, nicht für jedes Thema die gleiche Anzahl an Artikeln pro Zeitung für die Klassifizierung verwendet werden können. Zudem variiert die Anzahl der Zeitungen je nach Thema. In Abbildung 4.4 wird deutlich, dass die Zuverlässigkeit der Klassifizierung stark davon abhängig ist, wie viele möglichen Klassen es gibt. Des Weiteren sind die Ergebnisse davon abhängig, welche Themen bei den jeweiligen Zeitungen vertreten sind. Es zeigte sich, dass die *Ärzte-Zeitung* insgesamt wesentlich besser von anderen Zeitungen unterschieden werden kann, unabhängig vom Thema. Trotzdem ist es auffällig, dass die Klassifizierung in der Kategorie *Lokales* verhältnismäßig gut funktioniert. Dies kann jedoch auch durch die größeren inhaltlichen Differenzen der Artikel über lokale Ereignisse begründet werden, die so groß sind, dass dabei auch der Schreibstil maßgeblich beeinflusst werden könnte. Insgesamt kann selbst unter Verwendung von nur vier verschiedenen Klassen kein sehr zuverlässiges Modell trainiert werden – das maximale F-Maß liegt hier bei nur reichlich 0,7. Im Vergleich erreichte die Erkennung von 8 verschiedenen Autoren, nur auf Basis von 200 Synsemantika als Features, bereits eine Genauigkeit von 99% [AL05, S. 2].

4.3.5 Features

Für die Auswertung der Features werden zunächst die in Tabelle 4.2 beschriebenen Featuresets ausgewertet. Hierfür werden erneut 10 Klassifikatoren mit multinomialer logistischer Regression trainiert und anschließend der Durchschnitt der jeweiligen *micro* F-Maße gebildet.

Abbildung 4.5 zeigt deutlich, dass die POS-Tags für die Klassifizierung sehr entscheidend sind. Während das Weglassen der Features aus *dep* und *lemma* die Performance nur leicht verschlechtert, bricht diese bei *-pos* ein. Sie ist dort sogar noch unter der für das Featureset *pos*. Dies kann dadurch begründet werden, dass die Bestimmung der POS-Tags von spaCy sehr akkurat ist und daher kein zusätzliches Rauschen, sondern zuverlässige Erkenntnisse über die Verteilung der Wortarten bringt. Es ist zudem anzunehmen, dass diese den Schreibstil eines Autors bzw. einer Zeitung besser repräsentieren können als die restlichen

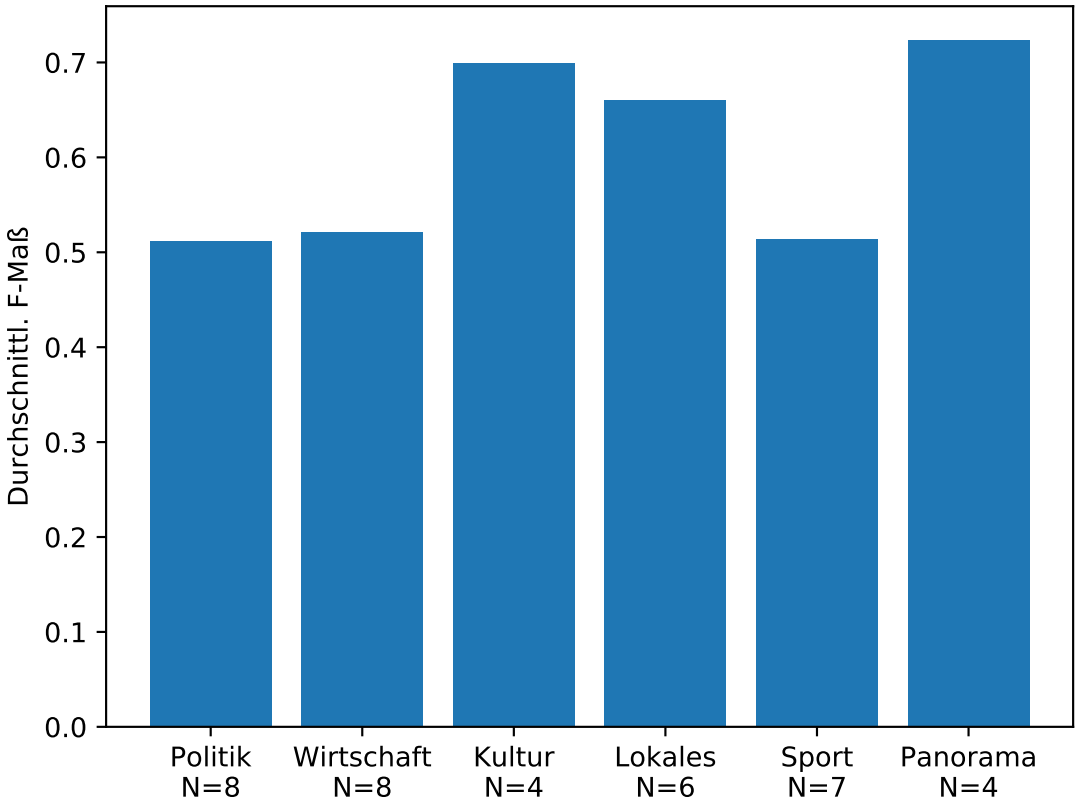


Abbildung 4.4: Performance der Klassifizierung pro Thema - N steht für die Anzahl der Zeitungen

verwendeten Features.
Die Monogramme der Dependency Relations sind für die Klassifizierung nicht als Features geeignet und geben kaum Aufschluss über den Schreibstil einer Zeitung. Ein möglicher Grund ist die bereits angesprochene, geringere Zuverlässigkeit der Bestimmung, wodurch die Features auf Basis der Dependency Relations möglicherweise zu viel Rauschen in den Datensatz bringen.
Das Featureset *lemma* ist auch nicht ausreichend, um eine gute Klassifizierung der Zeitungen zu erreichen, obwohl es aus insgesamt 97 Features besteht. Die leicht bessere Performance im Vergleich zu *dep* ist dadurch bedingt, dass es mehr als doppelt so viele Features sind. Obwohl sich die häufigsten Lemmata eines Korpus für die Erkennung der Autorschaft und des Genres eines Textes in bisherigen Studien als entscheidende Features

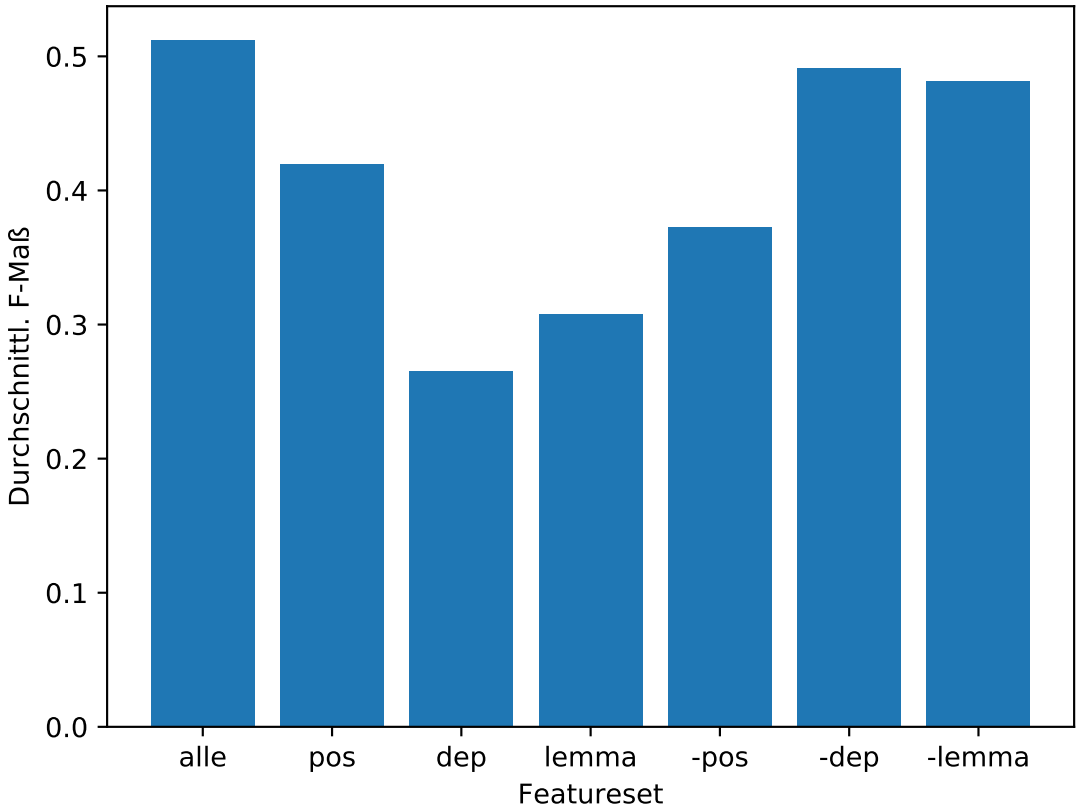


Abbildung 4.5: Performance der Featuresets

bewährt haben, eignen sich diese für die Klassifizierung der in dieser Arbeit verwendeten Texte nicht. Die Effektivität dieser Features beruht jedoch vor allem auf der Annahme, dass Autoren diese Wörter unterbewusst verwenden und daher bei jedem die Verteilung etwas anders ausfällt. In diesem Datensatz sind pro Zeitung jedoch eine Vielzahl an Autoren für die Artikel verantwortlich und so ist es sehr unwahrscheinlich, dass sich über diese hinweg, innerhalb der Zeitung ein kohärenter Schreibstil entwickelt. Dafür müsste sich die unterbewusste Verwendung der Synsemantika innerhalb der Autoren einer Zeitung stark ähneln.

Die Auswertung des Einflusses einzelner Features wird mithilfe von *Recursive Feature Elimination* (RFE) durchgeführt. Dafür werden die für die Klassifizierung mit multinomialer logistischer Regression wichtigsten 10, 50 und 80 Features bestimmt. Abbildung 4.6 zeigt den Anteil der Gruppen *LEX* (Features 1.-3.), *POS* (Features 4.-59.), *DEP* (Features 60.-101.) und

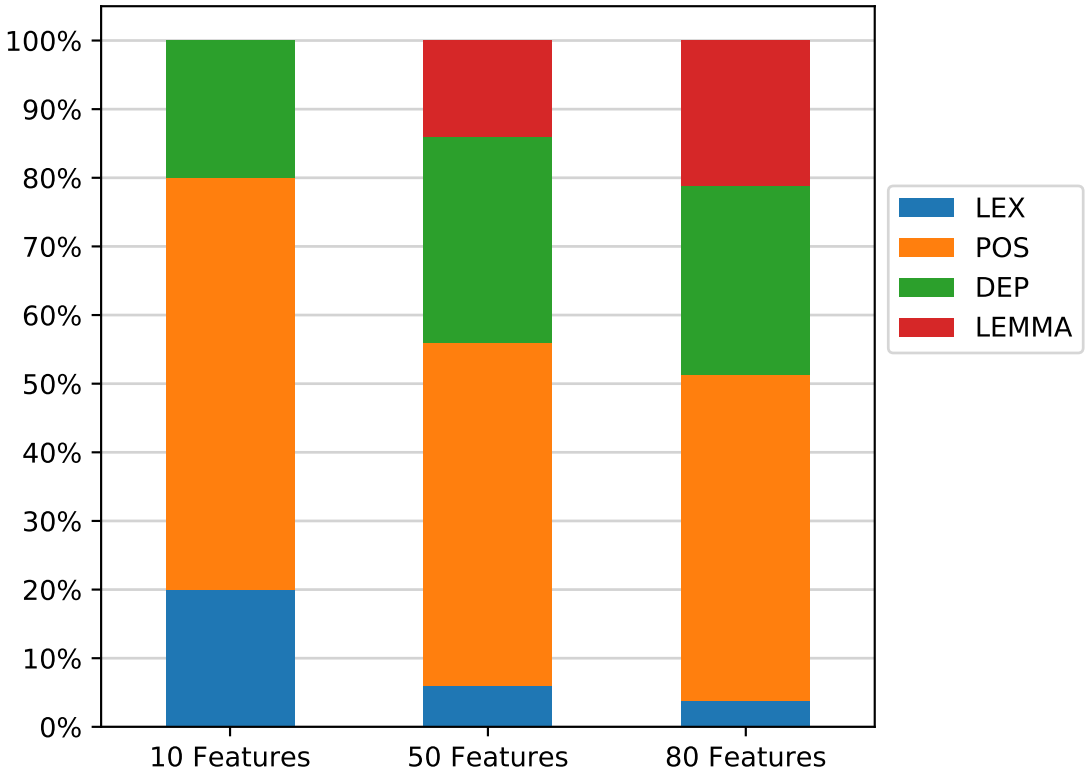
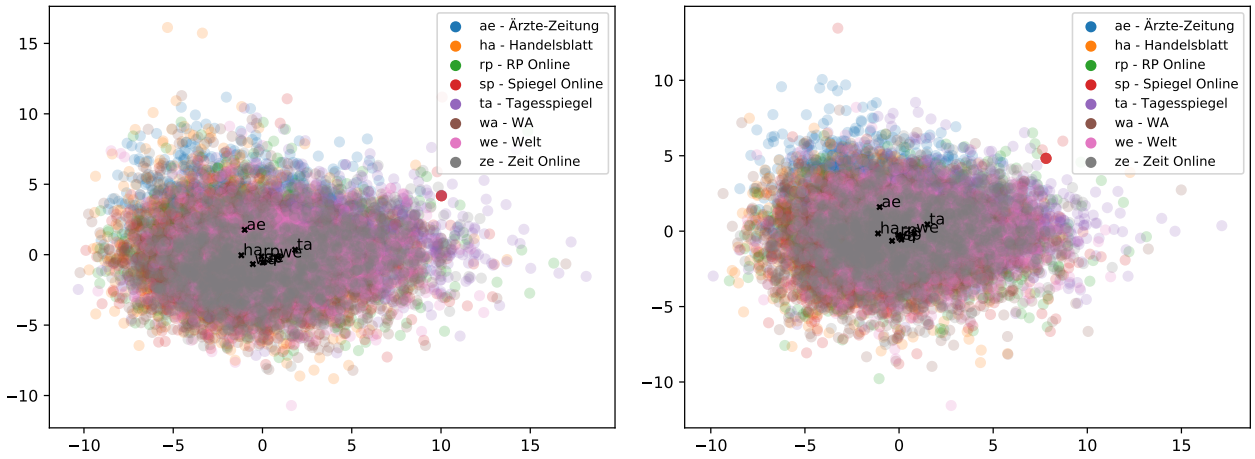


Abbildung 4.6: Die wichtigsten Features nach Recursive Feature Elimination

LEMMA (Features 102.-201.) an der Gesamtheit der nach RFE verbleibenden Features. Es bestätigt sich, dass die POS-Tags einen sehr großen Einfluss auf die Klassifizierung haben und vor allem die häufigsten Lemmata sehr schlecht abschneiden. Bei genauerer Betrachtung der 10 besten Features `['lex_sent_length', 'lex_readability', 'tag_PTKNEG', 'tag_$.', 'tag_$(', 'tag__SP', 'tag_NE', 'tag_$,', 'dep_punct', 'dep_ng']`[2], fällt jedoch auf, dass ein großer Teil der Features die Zeichensetzung des Textes messen und die Satzzeichen damit als ein gutes Unterscheidungsmerkmal der Artikel identifizieren. Dies kann zusätzlich dadurch bedingt sein, dass die Bestimmung dieser POS-Tags besonders einfach und akkurat ist und daher kaum Rauschen hinzufügt.

2 Die Bedeutung der Kürzel für POS-Tags und Dependency Relations sind auf https://spacy.io/api/annotation nachzulesen



(a) PCA mit allen Features **(b)** PCA mit den besten 80 Features

Abbildung 4.7: Hauptkomponentenanalyse (PCA) der Politik-Artikel

4.3.6 Clustering

Für das Clustering wird zunächst eine Hauptkomponentenanalyse (PCA) durchgeführt, welche die Dimensionalität der Daten auf 2 reduziert. Hierfür werden erneut die Politik-Artikel der Zeitungen verwendet. In den zwei so entstandenen Linearkombinationen bleiben unter Verwendung aller Features etwa 10% der Informationen erhalten. Werden nur die besten 80 Features nach Recursive Feature Elimination verwendet, bleiben zwar fast 20% der Informationen erhalten, aber wie in Abbildung 4.7 zu sehen, ergibt sich kaum ein anderes Bild. Dies ist zwar in beiden Grafiken keine genaue Annäherung an den wirklichen hochdimensionalen Raum, den die Featurevektoren bilden, aber oftmals trotzdem nützlich, um einen Überblick über die Verteilung zu bekommen. Die Punkte sind mit einem Alphawert von 0,2 eingezeichnet, um die Konzentration der Punkte visualisieren zu können. In Abbildung 4.7 ist zu sehen, dass die Datenpunkte sehr gemischt verteilt sind und nur in den Durchschnitten aller Artikel einer Zeitung kleine Tendenzen zu erkennen sind. Es ist also festzustellen, dass die Streuung immens hoch ist und demnach aus dieser Grafik keine aussagekräftigen Erkenntnisse über eine mögliche Gruppierung der Zeitungen aus ihren Durchschnitten gewonnen werden können. Den Durchschnitt einer Zeitung für das Clustering zu verwenden ist folglich weder aussagekräftig noch zielführend. Es ist zudem zu erkennen, dass das Clustering der einzelnen Artikel nach der Hauptkomponentenanalyse unmöglich ist.

Da das Clustering nach Hauptkomponentenanalyse keine guten Ergebnisse erzielt hat,

wird zusätzlich t-SNE eingesetzt, um Cluster der Artikel zu finden. Dieses Verfahren wird mit steigender Anzahl an Datenpunkten und vor allem höherer Dimensionalität sehr viel rechenintensiver. Daher wird die Anzahl der Artikel pro Zeitung auf etwa 1000 begrenzt und die Dimensionalität mithilfe von PCA auf 125 reduziert. Hierbei ist es notwendig, verschiedene Kombinationen aus der Anzahl an Iterationen `n_iter`, der Lernrate `learning_rate`, und des Parameters `perplexity`, zu testen, da diese Variablen großen Einfluss auf das Endergebnis besitzen. Für `n_iter` werden 500, 1000 und 2000, für `learning_rate` 10, 20, 40, 100 und 500 und für `perplexity` 10, 25, 50, 60, 75 und 100 als Werte verwendet. Aus allen 90 Kombinationen ergibt sich jedoch kein Diagramm, welches gute Cluster zeigt und Artikel der gleichen Zeitungen sind auch hier über den ganzen Raum verteilt. Abbildung 4.8 zeigt dabei das "beste" Ergebnis mit der Kombination `n_iter = 2000`, `learning_rate = 20` und `perplexity = 50`. Dabei ist zu beachten, dass die Distanzen der Punkte im Diagramm untereinander keine Bedeutung haben und das Verfahren nur Cluster findet und visuell darstellt. Folglich ist es nicht möglich, die Artikel oder Zeitungen anhand der in dieser Arbeit verwendeten Features zu gruppieren.

TODO: T-SNE Bild austauschen

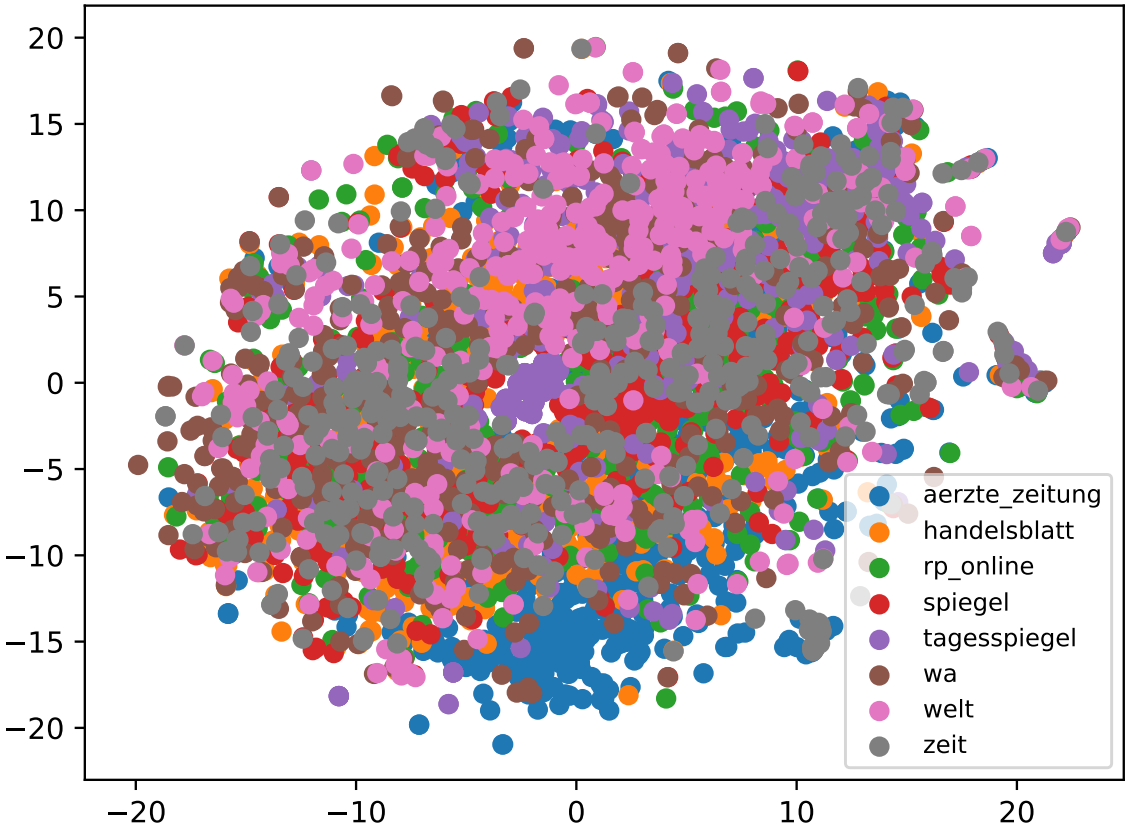


Abbildung 4.8: Clustering der Politik-Artikel mit dem t-SNE Verfahren

5 Fazit und Ausblick

In dieser Arbeit wurde untersucht, inwiefern ausgewählte Zeitungen gruppiert werden können, mit der Restriktion, dass nur Features verwendet werden, welche unabhängig vom Inhalt sind. Die in dieser Arbeit untersuchten Features konnten jedoch den Schreibstil einer Zeitung nicht genügend erfassen und repräsentieren. Für den Bereich der Erkennung der Autorschaft oder des Genres eines Textes haben sich solche Features zwar für die Klassifizierung als sehr geeignet herausgestellt, die Texte weisen dort jedoch größere syntaktische Unterschiede auf. Die Artikel von Zeitungen werden von vielen verschiedenen Autoren verfasst und berichten über sehr ähnliche Themen, sodass die Ergebnisse dieser Arbeit nicht direkt damit verglichen werden können. Die Gruppierung der Zeitungen ist eine noch schwierigere Aufgabe, da dann nicht nur ein Text als Vektor repräsentiert werden muss, sondern ein ganzer Korpus. Der Ansatz, die Artikel zu clustern stellte sich als besser heraus, die Ergebnisse der Klassifizierung zeigen jedoch schon, dass diese untereinander nicht gut voneinander unterschieden werden können und damit ist auch hier eine Gruppierung nicht zielführend.

Es gibt jedoch Potenzial bei der Klassifizierung der Zeitungen unter Verwendung inhaltlicher Features, wie der N-Gramme der Buchstaben. Diese stellten sich auch über Themen hinweg als robuste Merkmale für die Erkennung der Autorschaft eines Textes heraus [Sta13] und könnten auch für die Klassifizierung der Zeitungen erheblich bessere Resultate erzielen. Dabei sind die Ergebnisse dann auch nicht mehr abhängig von den Genauigkeiten der in der Pipeline verwendeten Tools, sondern nur noch von der korrekten Repräsentation der Texte im Datensatz. Es ist zudem möglich, abhand von Topic Models oder ähnlichen Repräsentationen eines Textes, die Zeitungen nur nach ihren Inhalten zu gruppieren. Hierbei spielt jedoch der verwendete Korpus eine maßgebende Rolle. Für einen aussagekräftigen Vergleich, muss der Zeitraum für jede Zeitung gleich und die Sammlung der Artikel annähernd komplett oder zumindest sehr repräsentativ sein.

Literaturverzeichnis

[AL05] Argamon, Shlomo; Levitan, Shlomo: Measuring the usefulness of function words for authorship attribution. In: *Proceedings of the 2005 ACH/ALLC Conference*, 2005

[BCK12] Burkschat, Marco; Cramer, Erhard; Kamps, Udo: *Beschreibende Statistik: Grundlegende Methoden der Datenanalyse*. Springer-Verlag, 2012

[Bor07] Borcholte, Andreas: *Quentin Tarantinos "Death Proof": Vorspiel mit V-8-Motor*. `http://www.spiegel.de/kultur/kino/quentin-tarantinos-death-proof-vorspiel-mit-v-8-motor-a-494970.html`, Juli 2007. – letzter Zugriff: 17.01.2019

[CH02] Courtis, John K.; Hassan, Salleh: Reading ease of bilingual annual reports. In: *The Journal of Business Communication (1973)* 39 (2002), Nr. 4, S. 394–413

[CWD+17] Cimino, Andrea; Wieling, Martijn; Dell'Orletta, Felice; Montemagni, Simonetta; Venturi, Giulia: Identifying Predictive Features for Textual Genre Classification: the Key Role of Syntax. In: *CLiC-it 2017 11-12 December 2017, Rome* (2017), S. 107

[Dom12] Domingos, Pedro: A few useful things to know about machine learning. In: *Communications of the ACM* 55 (2012), Nr. 10, S. 78–87

[Exp19a] ExplosionAI: *German - spaCy Models Documentation*. `https://spacy.io/models/de`, 2019. – letzter Zugriff: 23.01.2019

[Exp19b] ExplosionAI: *Linguistic Features - spaCy Usage Documentation*. `https://spacy.io/usage/linguistic-features#dependency-parse`, 2019. – letzter Zugriff: 12.01.2019

[Fou19] Foundation, Python: *What Is Python? Executive Summary*. `https://www.python.org/doc/essays/blurb`, 2019. – letzter Zugriff: 15.01.2019

[Gér17] Géron, Aurélien: *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems*. Ö'Reilly Media, Inc.", 2017

[HAK00] Hinneburg, Alexander; Aggarwal, Charu C.; Keim, Daniel A.: What is the nearest neighbor in high dimensional spaces? In: *26th Internat. Conference on Very Large Databases*, 2000, S. 506–515

[JM] Jurafsky, Dan; Martin, James H.: *Speech and language processing*. Bd. 3

[Kes14] Kestemont, Mike: Function Words in Authorship Attribution. From Black Magic to Theory? In: *Proceedings of the 3rd Workshop on Computational Linguistics for Literature (CLFL)*, 2014, S. 59–66

[LZC06] Li, Jiexun; Zheng, Rong; Chen, Hsinchun: From fingerprint to writeprint. In: *Communications of the ACM* 49 (2006), Nr. 4, S. 76–82

[MH08] Maaten, Laurens van d.; Hinton, Geoffrey: Visualizing data using t-SNE. In: *Journal of machine learning research* 9 (2008), Nr. Nov, S. 2579–2605

[MP82] McCallum, Douglas R.; Peterson, James L.: Computer-based readability indexes. In: *Proceedings of the ACM'82 Conference* ACM, 1982, S. 44–48

[Ng13] Ng, Andrew: *Machine Learning and AI via Brain simulations*. 2013

[PVG+11] Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, E.: Scikit-learn: Machine Learning in Python. In: *Journal of Machine Learning Research* 12 (2011), S. 2825–2830

[Rot10] Rottensteiner, Sylvia: Structure, function and readability of new textbooks in relation to comprehension. In: *Procedia-Social and Behavioral Sciences* 2 (2010), Nr. 2, S. 3892–3898

[Seb02] Sebastiani, Fabrizio: Machine learning in automated text categorization. In: *ACM computing surveys (CSUR)* 34 (2002), Nr. 1, S. 1–47

[See16] Seeker, Wolfgang: *spaCy now speaks German*. `https://explosion.ai/blog/german-model`, 2016. – letzter Zugriff: 16.01.2019

[SFK00] Stamatatos, Efstathios; Fakotakis, Nikos; Kokkinakis, George: Automatic text categorization in terms of genre and author. In: *Computational linguistics* 26 (2000), Nr. 4, S. 471–495

[Sl18] Scikit-learn: *scikit-learn API Reference*. `https://scikit-learn.org/stable/modules/classes.html`, 2018. – letzter Zugriff: 26.01.2019

[Sta09] Stamatatos, Efstathios: A survey of modern authorship attribution methods. In: *Journal of the American Society for information Science and Technology* 60 (2009), Nr. 3, S. 538–556

[Sta13] Stamatatos, Efstathios: On the robustness of authorship attribution based on character n-gram features. In: *Journal of Law and Policy* 21 (2013), Nr. 2, S. 421–439

[TB98] Tweedie, Fiona J.; Baayen, R H.: How variable may a constant be? Measures of lexical richness in perspective. In: *Computers and the Humanities* 32 (1998), Nr. 5, S. 323–352

[ZLCH06] Zheng, Rong; Li, Jiexun; Chen, Hsinchun; Huang, Zan: A framework for authorship identification of online messages: Writing-style features and classification techniques. In: *Journal of the American society for information science and technology* 57 (2006), Nr. 3, S. 378–393

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, dass alle Stellen der Arbeit, die wörtlich oder sinngemäß aus anderen Quellen übernommen wurden, als solche kenntlich gemacht und dass die Arbeit in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegt wurde.

Ort, Datum Unterschrift