

LEHRBUCH

Chris Biemann  
Gerhard Heyer  
Uwe Quasthoff

# Wissensrohstoff Text

Eine Einführung in das Text Mining

2. Auflage



Springer Vieweg

---

## Wissensrohstoff Text

---

Chris Biemann · Gerhard Heyer · Uwe Quasthoff

# Wissensrohstoff Text

Eine Einführung in das Text Mining

2., wesentlich überarbeitete Auflage



Springer Vieweg

Chris Biemann  
House of Computing and Data Science  
Universität Hamburg  
Hamburg, Deutschland

Uwe Quasthoff  
Institut für Informatik, Universität Leipzig  
Leipzig, Deutschland

Gerhard Heyer  
Institut für Informatik, Universität Leipzig  
Leipzig, Deutschland

ISBN 978-3-658-35968-3      ISBN 978-3-658-35969-0 (eBook)  
<https://doi.org/10.1007/978-3-658-35969-0>

Die Deutsche Nationalbibliothek verzeichnetet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

1. Aufl.: © Springer Nature Campus GmbH | w31. 2006  
2. Aufl.: © Springer Fachmedien Wiesbaden GmbH, ein Teil von Springer Nature 2022  
Das Werk einschließlich aller seiner Teile ist urheberrechtlich geschützt. Jede Verwertung, die nicht ausdrücklich vom Urheberrechtsgesetz zugelassen ist, bedarf der vorherigen Zustimmung des Verlags. Das gilt insbesondere für Vervielfältigungen, Bearbeitungen, Übersetzungen, Mikroverfilmungen und die Einspeicherung und Verarbeitung in elektronischen Systemen.  
Die Wiedergabe von allgemein beschreibenden Bezeichnungen, Marken, Unternehmensnamen etc. in diesem Werk bedeutet nicht, dass diese frei durch jedermann benutzt werden dürfen. Die Berechtigung zur Benutzung unterliegt, auch ohne gesonderten Hinweis hierzu, den Regeln des Markenrechts. Die Rechte des jeweiligen Zeicheninhabers sind zu beachten.  
Der Verlag, die Autoren und die Herausgeber gehen davon aus, dass die Angaben und Informationen in diesem Werk zum Zeitpunkt der Veröffentlichung vollständig und korrekt sind. Weder der Verlag noch die Autoren oder die Herausgeber übernehmen, ausdrücklich oder implizit, Gewähr für den Inhalt des Werkes, etwaige Fehler oder Äußerungen. Der Verlag bleibt im Hinblick auf geografische Zuordnungen und Gebietsbezeichnungen in veröffentlichten Karten und Institutionsadressen neutral.

Planung: David Imgrund  
Springer Vieweg ist ein Imprint der eingetragenen Gesellschaft Springer Fachmedien Wiesbaden GmbH und ist ein Teil von Springer Nature.  
Die Anschrift der Gesellschaft ist: Abraham-Lincoln-Str. 46, 65189 Wiesbaden, Germany

---

## Vorwort

Die Flut von Text, die dank des Internets heute in den großen natürlichen Sprachen der Welt vorhanden ist, ist ein bedeutsamer Wissensrohstoff von zunehmender Relevanz. Wie dieser Wissensrohstoff Text aufbereitet, verarbeitet und genutzt werden kann, ist der Gegenstand dieses Buches.

Die Verfahren, welche für das Text Mining verwendet werden, haben sich in den Jahren seit der Erstauflage unseres Buches rasant entwickelt. Standen in den Nullerjahren vor allem noch musterbasierte und statistische Ansätze im Vordergrund, so bildet heute maschinelles Lernen – mit traditionellen Methoden, jedoch insbesondere mit neuronalen Netzen – die Grundlage für die automatische Verarbeitung von Text. Wer diese neuesten Entwicklungen des Text Minings nachvollziehen und für sich nutzen möchte, braucht eine Orientierung über die Grundannahmen und zu methodischen Ansätzen, ohne die man sich leicht in diesem hochdynamischen Gebiet verliert. Mit der vorliegenden, sehr umfangreichen Überarbeitung wollen wir dieser Entwicklung Rechnung tragen und eine aktuelle Einführung in die Grundlagen, Verfahren und beispielhaften Anwendungen des Text Minings bieten.

Dabei haben wir versucht, durch die Verbindung von Grundlagen und Anwendungen einen guten Mittelweg zu finden zwischen den sich rasant entwickelnden algorithmischen Details auf der einen und kaum anwendbaren Überblicksinformationen auf der anderen Seite. Aus unserer Sicht wichtige Schwerpunktsetzungen für das Text Mining, wie die Verfügbarkeit großer Textkorpora, die Nutzung linguistischen Wissens und statistische Ansätze, haben wir beibehalten und durch einen Einblick in aktuelle Methoden zum Aufsetzen einer Verarbeitungspipeline und zur Erstellung von sprachverarbeitenden Komponenten mit maschinellem Lernen ergänzt.

Dieses Buch richtet sich gleichermaßen an Studierende und Fachleute mit einem fachlichen Schwerpunkt in der Informatik, Wirtschaftsinformatik und/oder Linguistik und Digital Humanities, die sich über die Grundlagen, Verfahren und Anwendungen des Text Minings informieren möchten und Anregungen für die Implementierung eigener Anwendungen suchen. Bewusst sprechen wir Anwendende von Text-Mining-Verfahren an, welche sich ein allgemeines Verständnis von Text Mining aneignen wollen. Diese Zielgruppe verweisen wir für weiterführende Themen auf die entsprechende Literatur.

Die dargestellten Inhalte decken alle Bereiche von Text Mining ab und basieren teilweise auf Arbeiten, die während der letzten Jahre an der Abteilung Automatische Sprachverarbeitung am Institut für Informatik der Universität Leipzig sowie der Language Technology Group an der Universität Hamburg entstanden sind.

Das erste Kapitel führt an die Materie heran, motiviert die Anwendung von Text Mining und legt mit der Einführung von Grundbegriffen rund um das Thema Text die Grundlagen für die folgenden Kapitel. Linguistisches Wissen, welches für die Automatisierung der Verarbeitung von Text erforderlich ist, wird in Kap. 2 behandelt. Hier und auch in späteren Kapiteln unterfüttern wir die Erklärungen und Darstellungen mit vielen Sprachbeispielen, welche die Leser und Leserinnen auch mit dem Rohstoff Text in all seiner ausnahmebehafteten Regelhaftigkeit vertraut machen. Die Umsetzung der linguistischen Ebenen für die automatische Vorverarbeitung wird in Kap. 3 diskutiert; hier werden auch Grundkonzepte des maschinellen Lernens eingeführt und die Skalierung der Verarbeitung auf große Textmengen behandelt. Nach der Diskussion der Verfahren behandelt Kap. 4 die Sprach-Daten und beschreibt die Schritte bei der Erstellung von Korpora und bei der Erstellung oder Erweiterung von lexikalischen Ressourcen wie Wörterbüchern oder semantischen Wortnetzen. Sind Korpora in ausreichender Größe verfügbar, können auf diesen die im Kap. 5 beschriebenen Verfahren der Sprachstatistik ausgeführt werden – von der deskriptiven Statistik und sprachlichen Gesetzmäßigkeiten über semantische Modelle wie Topic Modelle, Sprachmodelle und Word Embeddings bis hin zu statistischen Methoden des Korpusvergleichs. Kap. 6 stellt für das Text Mining relevante maschinelle Lernverfahren dar. Dies umfasst die Repräsentation der Daten, Ansätze zum Clustering und zur Klassifikation und die Erstellung von Trainingsdaten mit Annotation. Schließlich beschreiben wir ausgewählte Anwendungsszenarien von Text Mining in Kap. 7, wo die vorher beschriebenen Konzepte zusammengeführt und produktiv umgesetzt werden.

Das Buch wird ergänzt von einem Glossar, welches die Begrifflichkeiten im Text Mining übersichtlich und kompakt zusammenstellt. Bei der Wahl der Terminologie haben wir uns bewusst dafür entschieden, nur vergleichsweise übliche deutsche Begriffe als solche zu verwenden und ansonsten die aus dem englischen Sprachgebrauch stammenden Begriffe beizubehalten, welche immer auch als Übersetzung im Glossar angegeben sind.

Umfangreiche Textressourcen – nicht nur fürs Deutsche – finden sich unter der URL <http://wortschatz.uni-leipzig.de>, weiterführende Beschreibungen von Verfahren und Anwendungen unter den URLs <https://toolbox.wortschatz.uni-leipzig.de> und <https://www.inf.uni-hamburg.de/en/inst/ab/lit/resources/software/asvtoolbox.html>. Die Webseiten bieten zahlreiche Web Services an, die über die genannten URLs kontaktiert werden können. Auf der dieses Buch ergänzenden SpringerLink-Seite sind die Anhänge und auch das Glossar öffentlich verfügbar.

**Tab. 1** Übersicht Autor-Abschnitt

Autor	Abschnitt/Wissensbaustein
Chris Biemann	3.1, 3.2, 3.3.1, 5.4–5.7, 6.1–6.10, 7.2, 7.3, A7
Gerhard Heyer	1.1–1.3, 1.5, 2.1–2.5, 3.3.1, 5.2, 5.6, 5.9, 6.4, 7.1, 7.3, 7.4, 7.6, A1–A4, A6–A8
Uwe Quasthoff	1.4, 2.6, 3.3.2, 4.1–4.6, 5.1, 5.3, 5.8, 7.5, A5–A7
Christian Berger	1.3 (Zuarbeit rechtliche Aspekte)
Christian Kahmann	7.6, A8
Roman Schneider	1.2, 3.3.1
Christopher Schröder	5.6, 7.4

Um den vielfältigen Aspekten des Text Minings und der Dynamik seiner Entwicklung gerecht zu werden, haben die drei Autoren untereinander fachliche Schwerpunkte abgesprochen. An der Erstellung einzelner Wissensbausteine haben sich darüber hinaus Christian Kahmann, Roman Schneider und Christopher Schröder beteiligt. Die Tab. 1 gibt eine Übersicht darüber, wer an der Erstellung welcher Abschnitte mitgewirkt hat.

Besonders danken möchten wir Marie Annisius, Cathleen Kantner und Lydia Müller sowie Thomas Eckhard, Hans Ole Hatzel und Fynn Schröder für Anmerkungen an früheren Versionen des Manuskriptes und dem Verlag und seinen Mitarbeitern und Mitarbeiterinnen für die Geduld und für die Hilfestellungen bei der technischen Umsetzung.

Leipzig/Hamburg  
im September 2021

Chris Biemann  
Gerhard Heyer  
Uwe Quasthoff

---

# Inhaltsverzeichnis

<b>1</b>	<b>Text und Text Mining .....</b>	<b>1</b>
1.1	Text Mining .....	1
1.1.1	Text ist Wissensrohstoff .....	1
1.1.2	Text Mining und Text-Mining-Werkzeuge .....	3
1.1.3	Text-Mining-Umgebungen .....	4
1.1.4	Was leistet Text Mining? .....	6
1.2	Text und Big Data .....	8
1.3	Aufbau und Struktur von Text .....	10
1.3.1	Arten von Text und ihre Merkmale .....	10
1.3.2	Zeichen, Types und Wörter .....	11
1.3.3	Nachricht, Information .....	12
1.3.4	Information und Wissen .....	14
1.3.5	Beispieltexte und Textressourcen .....	15
1.4	Redundanz in Texten .....	18
1.4.1	Arten von Redundanz .....	18
1.4.1.1	Nahe Wiederholung von Wörtern im Text .....	18
1.4.1.2	Wiederholung von Wortteilen .....	19
1.4.1.3	Wiederholung in der Struktur: Eigennamen .....	19
1.4.1.4	Kongruenz .....	20
1.4.1.5	Feste Wendungen .....	20
1.4.1.6	Explizite Wiederholung typischer Zusammenhänge in verschiedenen Sätzen oder Texten .....	21
1.4.2	Wirkung von Redundanz .....	21
1.4.2.1	Wiederholte Wörter: Wichtige Namen oder Schlagwörter .....	22
1.4.2.2	Wiederholte Substrings im Text: Klassifikation von Texten und Wörtern .....	22
1.5	Linguistische Strukturen .....	23
1.5.1	Texte und linguistische Ebenen .....	23

1.5.2	Warum erfordert die Verarbeitung natürlicher Sprache linguistisches Wissen? . . . . .	24
1.5.3	Zwei Ansätze für die Repräsentation und Verarbeitung linguistischen Wissens . . . . .	27
	Literatur . . . . .	31
<b>2</b>	<b>Linguistische Repräsentationen</b> . . . . .	35
2.1	Theoretische Grundlage: Strukturalismus . . . . .	36
2.1.1	Was ist die Grundidee des Strukturalismus? . . . . .	36
2.1.2	Kontexte . . . . .	37
2.2	Morphologie . . . . .	40
2.2.1	Grundbegriffe . . . . .	40
2.2.2	Verarbeitungsparadigmen für die Morphologie . . . . .	43
2.3	Syntaktische Repräsentationen . . . . .	46
2.3.1	Begriffsbestimmung: Was sind syntaktische Strukturen? . . . . .	46
2.3.2	Konstituenten-Syntax . . . . .	47
2.3.3	Dependenzen . . . . .	50
2.3.4	Probabilistisches Parsen . . . . .	51
2.4	Semantische Repräsentationen . . . . .	53
2.4.1	Grundbegriffe und Definition . . . . .	53
2.4.2	Semantische Relationen . . . . .	57
2.5	Fachtexte und Terminologie . . . . .	62
2.5.1	Fachtexte . . . . .	62
2.5.2	Terminologie . . . . .	63
2.6	Die Rolle von Ausnahmen in der Sprache . . . . .	65
2.6.1	Falsche Schreibweisen . . . . .	65
2.6.2	Seltene Ausnahmen . . . . .	66
2.6.3	Auswirkungen dieser Sonderfälle . . . . .	67
	Literatur . . . . .	68
<b>3</b>	<b>Maschinelle Verarbeitung von Text</b> . . . . .	73
3.1	Verarbeitungsparadigmen für Text . . . . .	73
3.1.1	Regelbasierte Verarbeitung . . . . .	75
3.1.2	Überwachte Statistische Verarbeitung . . . . .	79
3.1.3	Neuronale Verarbeitung . . . . .	81
3.2	Die Linguistische Pipeline . . . . .	85
3.2.1	Pipeline-Modell . . . . .	85
3.2.2	Einlesen und Vorverarbeitung . . . . .	89
3.2.3	Segmentierung . . . . .	93
3.2.4	Morphologische und Syntaktische Verarbeitung . . . . .	99
3.2.4.1	Grundformreduktion und Stammformreduktion . . . . .	99
3.2.4.2	Tagging mit Wortarten . . . . .	101

3.2.4.3	Chunking . . . . .	102
3.2.4.4	Syntaxparsing . . . . .	103
3.2.5	Semantische Verarbeitung . . . . .	106
3.2.5.1	Eigennameerkennung . . . . .	106
3.2.5.2	Entity Linking . . . . .	107
3.2.5.3	Koreferenzauflösung . . . . .	108
3.2.5.4	Wortbedeutungsdisambiguierung . . . . .	109
3.2.6	Anwendungsorientierte Verarbeitung . . . . .	110
3.2.6.1	Pipeline für Terminologieextraktion . . . . .	111
3.2.6.2	Pipeline für Entitätenzentriertes Retrieval und Facettierte Suche . . . . .	111
3.2.6.3	Pipeline für Sentimentanalyse . . . . .	112
3.2.6.4	Pipeline für Open Information Extraction . . . . .	113
3.3	Skalierung auf große Datenmengen . . . . .	114
3.3.1	Datenparallelität . . . . .	114
3.3.2	Zusammenhang von Korpusgröße und Qualität . . . . .	117
3.3.2.1	Der More-Data-Effect . . . . .	118
3.3.2.2	Quantitatives Wachstum von Resultatmengen . . . . .	119
3.3.2.3	Qualitative Verbesserung von Resultatmengen . . . . .	122
Literatur . . . . .		123
<b>4</b>	<b>Sprachdaten: Lexika und Korpora . . . . .</b>	<b>131</b>
4.1	Korpusauswahl . . . . .	131
4.1.1	Generische Korpora . . . . .	132
4.1.2	Selbst erstelltes Korpus . . . . .	133
4.1.3	Dokumente oder Sätze? Nur wohlgeformte Sätze? . . . . .	134
4.1.4	Repräsentativität und Ausgewogenheit der Zusammensetzung oder zufällige Auswahl? . . . . .	135
4.1.5	Parallele Korpora . . . . .	136
4.2	Satzkorpuserstellung auf Webdaten . . . . .	136
4.2.1	Crawling . . . . .	137
4.2.2	Beschränkung auf HTML-Dokumente . . . . .	137
4.2.3	Text aus HTML-Dokumenten extrahieren . . . . .	137
4.2.4	Qualitätssicherung . . . . .	138
4.2.4.1	Sprachseparierung auf Dokumentenebene . . . . .	139
4.2.4.2	Sprachüberprüfung auf Satzebene . . . . .	140
4.2.4.3	Umgang mit Dubletten und Quasidubletten . . . . .	140
4.2.4.4	Musterbasierte Entfernung nicht-wohlgeformter Sätze . . . . .	146
4.2.5	Evaluierung und Ranking von Sätzen . . . . .	149
4.2.5.1	Ranking mittels GDEX . . . . .	149
4.2.5.2	Typische Sätze . . . . .	152

4.3	Speicherformate . . . . .	153
4.4	Indexierung . . . . .	156
4.4.1	Indexstrukturen . . . . .	157
4.4.1.1	Klassisch: Einzelwort-Index, evtl. mit zusätzlichen Wortgruppen . . . . .	157
4.4.1.2	Klassisch: Einzelwort-Index mit genauer Position . . . . .	157
4.4.1.3	Spezielle Datenstrukturen für Textsuche: PAT Trees und die Anwendung in der NoSketch-Engine . . . . .	157
4.4.1.4	Ranking der Suchergebnisse . . . . .	158
4.4.2	Verschiedene Typen von Suchanfragen . . . . .	158
4.4.2.1	Beispielsätze für Wörter, Wortgruppen oder Kookkurrenzen . . . . .	158
4.4.2.2	Beispielsätze für Grundformen . . . . .	159
4.4.2.3	Kombination mehrerer Suchkriterien . . . . .	159
4.4.2.4	Extraktion von Wortlisten mit bestimmten Eigenschaften . . . . .	159
4.4.2.5	Extraktion von Relationen . . . . .	160
4.4.2.6	Suche unter Verwendung von Satzsignaturen . . . . .	160
4.5	Manuell erstellte lexikalische Ressourcen . . . . .	162
4.5.1	Klassische Wörterbücher . . . . .	163
4.5.2	Verzeichnisse . . . . .	164
4.5.2.1	Verzeichnisse von Personen . . . . .	165
4.5.2.2	Verzeichnisse von Vornamen und Nachnamen . . . . .	165
4.5.2.3	Geographische Eigennamen . . . . .	165
4.5.3	Relationen zwischen Mengen von Wörtern . . . . .	166
4.5.3.1	Synsets . . . . .	166
4.5.3.2	Erweiterung von Wortnetzen: Semantische Ähnlichkeit mit Word Embeddings . . . . .	167
4.6	Automatische Erweiterung lexikalischer Ressourcen . . . . .	167
4.6.1	Klassische Wörterbuchangaben . . . . .	168
4.6.1.1	Lemmatisierung: Vollform zu Grundform . . . . .	168
4.6.1.2	Flexionsklasse zu Grundform . . . . .	168
4.6.1.3	Wortart zu Grundform: POS-Tagging und NER . . . . .	169
4.6.1.4	Grammatisches Geschlecht zu Nomen . . . . .	170
4.6.1.5	Kompositzerlegung . . . . .	170
4.6.2	Statistische Angaben . . . . .	171
4.6.2.1	Worthäufigkeiten . . . . .	171
4.6.2.2	Wortpaare: Kookkurrenzen . . . . .	172
4.6.2.3	Sentiment . . . . .	173
4.6.2.4	Sachgebietangaben . . . . .	173
	Literatur . . . . .	174

<b>5 Sprachstatistik</b>	177
5.1 Statistische Messungen und ihre Zuverlässigkeit	178
5.1.1 Aufgaben aus der Sprachstatistik	178
5.1.2 Messgrößen der Sprachstatistik	179
5.1.3 Präsentation der Ergebnisse	179
5.1.4 Messungen und Messwerte	180
5.1.4.1 Beschreibung und Nachvollziehbarkeit der Messung	180
5.1.4.2 Abhängigkeit von der Wortdefinition am Beispiel der Type-Token-Ratio	181
5.1.4.3 Abhängigkeit von der Korpusgröße	182
5.1.4.4 Zählen mit oder ohne Wiederholungen: Mittlere Wortlänge	183
5.1.5 Abhängigkeit von der Zusammensetzung des Korpus	185
5.1.5.1 Abhängigkeit vom Text-Genre	185
5.1.5.2 Fehlen gesprochener Sprache	186
5.1.6 Abhängigkeit von Optionen bei der der Korpuserstellung	186
5.1.6.1 Verzerrte Ergebnisse durch mangelnde Qualität der Rohdaten	186
5.1.6.2 Fragwürdige Optionen	188
5.1.6.3 Umgang mit seltenen Ereignissen	188
5.2 Zipfsches Gesetz	189
5.2.1 Zusammenhang zwischen Rang, Häufigkeit und Wortlänge	189
5.2.2 Vorhersagen und Anwendungen	192
5.3 Kookkurrenzen	195
5.3.1 Struktur von signifikanten Kookkurrenzen	195
5.3.2 Maße für die statistische Signifikanz einer Kookkurrenz	197
5.3.3 Plausibilität der Ergebnisse	200
5.3.4 Andere Signifikanzmaße	200
5.3.5 Signifikante Kookkurrenzen – Beispiele und Anwendungen	201
5.3.6 Erste Anwendungen für signifikante Kookkurrenzen	203
5.3.6.1 Fremdsprachliche Daten im Text	203
5.3.6.2 Mundart	203
5.3.7 Signifikante Kookkurrenzen und Polysemie	204
5.3.8 Semantische Relationen zwischen signifikanten Kookkurrenzen	205
5.3.9 Visualisierung von signifikanten Kookkurrenzen	206
5.4 Distributionelle Semantik	208
5.5 Sprachmodelle	212
5.6 Dense Vector Embeddings	218
5.6.1 Statische Word Embeddings	219
5.6.2 Statische Word Embeddings: Wortähnlichkeit und Analogie	222

5.6.3	Kontextualisierte Word Embeddings . . . . .	224
5.6.4	Embeddings für Sätze und Texte . . . . .	226
5.6.5	Evaluation von Embeddings . . . . .	227
5.7	Wortbedeutungsinduktion . . . . .	228
5.8	Qualitätsmaße für Korpora . . . . .	231
5.8.1	Statistische Abweichungen von der erwarteten Verteilung . . . . .	232
5.8.2	Verwendung von Vergleichskorpora . . . . .	236
5.8.3	Musterbasierte Abweichungen . . . . .	240
5.9	Statistischer Korpusvergleich . . . . .	242
5.9.1	Voraussetzungen und Ziele . . . . .	242
5.9.2	Wortvergleiche . . . . .	243
5.9.2.1	Log-likelihood-Ratio . . . . .	244
5.9.2.2	tf·idf (term frequency/inverse document frequency) . . . . .	246
5.9.2.3	Statistische Hypothesentests und Burrows Zeta . . . . .	246
5.9.3	Kontextvergleiche . . . . .	247
5.9.3.1	Embeddingbasierte Verfahren . . . . .	247
5.9.3.2	Kookkurrenzstatistik . . . . .	248
Literatur . . . . .		251
<b>6</b>	<b>Maschinelles Lernen für Sprachverarbeitung . . . . .</b>	<b>257</b>
6.1	Merkmalsextraktion . . . . .	258
6.2	Clustering . . . . .	261
6.3	Beispiele für Clustering . . . . .	264
6.3.1	Hierarchisches Clustern von Wortarten . . . . .	265
6.3.2	Wortbedeutungsinduktion mit Graph Clustering . . . . .	267
6.3.3	Eventerkennung mit inkrementellem Clustering . . . . .	268
6.3.4	Dokumentclustering mit k-Means . . . . .	269
6.4	Topic-Modelle . . . . .	270
6.4.1	Dokumente enthalten Themenstränge ( <i>Topics</i> ) . . . . .	270
6.4.2	Modellierung von Topics . . . . .	272
6.4.3	Evaluation und Best Practices . . . . .	275
6.5	Evaluation von Clustering . . . . .	277
6.6	Klassifikation . . . . .	279
6.6.1	Definition und Arten von Klassifikation . . . . .	280
6.6.2	Aufteilung der Beispieldaten . . . . .	280
6.6.3	Beispiele für Klassifikationsalgorithmen . . . . .	283
6.6.3.1	Naïve Bayes . . . . .	283
6.6.3.2	Entscheidungsbäume . . . . .	284
6.6.3.3	Neuronale Netzwerke für die Klassifikation . . . . .	285
6.7	Annotation: Erstellung von Trainingsdaten . . . . .	286
6.7.1	Durchführen von Annotationsprojekten . . . . .	286

6.7.2	Annotationsebenen und Tools zur manuellen Annotation . . . . .	288
6.7.3	Datensatzerstellung mit Crowdsourcing . . . . .	290
6.8	Sequenzklassifikation . . . . .	291
6.9	Evaluation von Klassifikation . . . . .	295
6.9.1	Mengenorientierte Evaluationsmaße . . . . .	296
6.9.2	Andere Evaluationsmaße im Text Mining . . . . .	298
6.10	Neuronale Methoden: End-to-End, Transfer Learning . . . . .	299
6.10.1	End-to-End-Lernen . . . . .	299
6.10.2	Transferlernen . . . . .	301
Literatur . . . . .		302
<b>7</b>	<b>Beispielanwendungen . . . . .</b>	<b>311</b>
7.1	Terminologieextraktion . . . . .	312
7.1.1	Arbeitsablauf der Terminologieextraktion . . . . .	312
7.1.2	Verfahren der Terminologieextraktion . . . . .	313
7.2	Facettierte Suche mit Eigennamen . . . . .	317
7.2.1	Tagesnetzwerk: Visuelle Nachrichtenzusammenfassung . . . . .	318
7.2.2	Storyfinder: Eigene Lesehistorie . . . . .	319
7.2.3	Investigativtool New/s/leak . . . . .	319
7.2.4	Zusammenfassung . . . . .	321
7.3	Sentimentanalyse . . . . .	322
7.3.1	Begriffsbestimmung, Einsatzbereiche, Herausforderungen . . . . .	322
7.3.2	Lösungsansätze und Aufgaben . . . . .	323
7.4	Trendanalysen und News Monitoring . . . . .	327
7.4.1	Trends und schwache Signale . . . . .	327
7.4.2	News Monitoring . . . . .	327
7.4.3	Wörter des Tages . . . . .	328
7.5	Neologismen . . . . .	332
7.5.1	Neologismenwörterbücher . . . . .	333
7.5.2	Hochfrequente Neologismen 2010–2020 . . . . .	337
7.6	Kontextvolatilität . . . . .	338
7.6.1	Kurze Zusammenfassung des Verfahrens . . . . .	338
7.6.2	Beispielanwendung . . . . .	340
Literatur . . . . .		342
<b>Glossar . . . . .</b>		<b>347</b>
<b>Stichwortverzeichnis . . . . .</b>		<b>381</b>



## Zusammenfassung

Text repräsentiert Wissen. Im Unterschied zu den strukturierten Daten in einer Datenbank stellen Texte unstrukturierte Daten dar. Eine wichtige Eigenschaft natürlicher Sprachen ist Redundanz. Im Text Mining werden verschiedene Arten von Redundanz genutzt, um Wörter oder ganze Texte zu klassifizieren sowie strukturelle oder inhaltliche Zusammenhänge zwischen Wörtern zu ermitteln. Dabei ist zu unterscheiden zwischen der Makro-Sicht auf ein ganzes Textkorpus und der Mikro-Sicht auf einzelne Wörter. Text Mining verwendet regelbasierte, statistische sowie neuronale Verfahren und erlaubt somit innovative Anwendungen, bei denen sehr große Mengen an Text sehr schnell und sehr umfangreich ausgewertet werden. Für das Text Mining ist es meist erforderlich, die linguistische Struktur von Texten zu berücksichtigen. Die Beschreibung linguistischer Strukturen setzt voraus, dass sich Texte in verschiedene linguistische Ebenen untergliedern lassen. Wichtige Ebenen für das Text Mining sind die Ebenen der Morpheme, der Wörter, der Phrasen und der Sätze. Auf allen Ebenen finden sich Mehrdeutigkeiten, sprachstatistische Gesetzmäßigkeiten und sprachliche Dynamiken.

---

## 1.1 Text Mining

### 1.1.1 Text ist Wissensrohstoff

Für die Zwecke des Text Mining soll im Folgenden unter **Text** eine Abfolge von Wörtern verstanden werden. Üblicherweise wird Text von Menschen für Menschen verfasst, genauer: von Autorinnen und Autoren für Leserinnen und Leser. Die Anliegen der Autorinnen und Autoren sowie die Inhalte der Texte umfassen ein weites Spektrum.

Sie können **Information** über Gegenstände und Sachverhalte enthalten oder sich als Appell oder als Ausdruck von Meinungen und Befindlichkeiten an die Lesenden wenden (vgl. Bühler 1934, vgl. hierzu Box im Anhang A2). Damit ein Text verstanden werden kann, muss er entsprechend der allgemein üblichen Regeln der verwendeten Sprache verfasst sein. Dies betrifft sowohl die grammatischen Regeln der Satzbildung wie auch die übliche Verwendung von Wörtern. Mit Hilfe des Text Mining können für einen Text auffällige Wörter, ihre Verteilung und ihr gemeinsames Auftreten mit anderen, thematisch verwandten, Wörtern automatisch erkannt und extrahiert werden und für eine nachfolgende inhaltliche Analyse verfügbar gemacht werden.

Der englische Begriff *Mining* bedeutet Bergbau. Wie beim Bergbau der Boden erkundet wird und Rohstoffvorkommen identifiziert und abgebaut werden, so werden im Text Mining Texte exploriert und relevante inhaltliche Zusammenhänge extrahiert. Dabei ist es möglich, dass neue und überraschende Zusammenhänge entdeckt werden, die für die Nutzerinnen und Nutzer von hohem Wert sind. Bei der interaktiven Anwendung von Text-Mining-Werkzeugen werden aus einer sehr großen Menge von Textdaten zunächst Hypothesen generiert, die schrittweise überprüft, modifiziert und verfeinert werden können.

Je nachdem, welche Verfahren im Vordergrund stehen und welche Ergebnisse von Interesse sind, finden sich in der Literatur unterschiedliche Definitionen von Text Mining, z. B. aus Sicht des Data Mining (vgl. Witten et al. 2017; Berry 2010; Aggarwal und Zhai 2012), des Information Retrieval (vgl. Feldman und Sanger 2006), des Wissensmanagements (vgl. Bodendorf 2006; Meier und Beckh 2000, S. 165–167) oder der Computerlinguistik (vgl. Dörre et al. 2001, S. 425–441).

Unter dem Begriff des **Data Mining** werden Verfahren aus der Statistik und künstlichen Intelligenz zusammengefasst, mit denen sich in strukturierten Datenbeständen Muster und statistische Zusammenhänge berechnen lassen (vgl. Witten et al. 2017). Unter strukturierten Datenbeständen werden dabei Daten verstanden, die ein vorgegebenes Format erfüllen, z. B. nach einem vorgegebenen Schema aufbereitete Daten in einer Datenbank. Das Retrieval von Daten aus einer Datenbank wird allgemein als Fakten-Retrieval bezeichnet, während das Retrieval von Informationen im Sinne einer Inhaltsabfrage auf großen Dokumentenkollektionen als **Information Retrieval** bezeichnet wird (vgl. Henrich 2008). Im Unterschied zu den Daten in einer Datenbank sind Texte unstrukturierte Daten. Sie folgen keinem vorgegebenen Schema, sondern die explizite Strukturierung von unstrukturierten Daten erfolgt erst nachträglich, etwa durch manuelle Annotation oder das Text Mining. **Text Mining** ähnelt dem Information Retrieval, insofern sein Gegenstand ebenfalls unstrukturierte Daten sind. Es ähnelt aber auch dem Data Mining, insofern es zum Ziel hat, durch die Berechnung von Mustern und statistischen Zusammenhängen einen Datenbestand inhaltlich zu analysieren. Der Unterschied zwischen Text Mining und Data Mining besteht darin, dass für das Text Mining meist die linguistische Struktur von Text berücksichtigt werden muss, beispielsweise die Wortart von Wörtern oder die syntaktische Struktur von Sätzen (vgl. dazu Kap. 2, Linguistische Repräsentationen). Hierfür steht aus dem Bereich des **Natural**

**Tab. 1.1** Retrieval und Mining von strukturierten und unstrukturierten Daten

	Retrieval	Analyse
Strukturierte Daten	Datenabfrage <i>Datenbank-Systeme</i>	Datenanalyse <i>Data-Mining-Werkzeuge</i>
Unstrukturierte Daten	Inhaltsabfrage <i>Suchmaschinen, Dokumentenmanagement-Systeme</i>	Inhaltsanalyse <i>Text-Mining-Werkzeuge</i>

**Language Processing (NLP)** eine Vielzahl von Verfahren bereit, die beim Text Mining verwendet werden können (vgl. Kao und Poteet 2007). Beiden gemeinsam ist jedoch, dass das Ergebnis des Mining im Unterschied zum Retrieval von Daten oder Text eine inhaltliche Analyse darstellt.

Den Zusammenhang zwischen strukturierten und unstrukturierten Daten einerseits sowie zwischen Retrieval und Analyse andererseits verdeutlicht Tab. 1.1.

### 1.1.2 Text Mining und Text-Mining-Werkzeuge

Im Rahmen dieses Buches verstehen wir unter **Text Mining** computergestützte Verfahren für die semantische Analyse von Texten, welche die automatische bzw. semi-automatische **Annotation** und **Klassifikation** von Texten, insbesondere von sehr großen Mengen von Texten, unterstützen. Diese können verwendet werden, um aus Texten neue und relevante Informationen zu extrahieren. Als Grundlage dienen regelbasierte, statistische und neuronale Verfahren. Diese Verfahren werden in Kap. 3 und in Kap. 6 ausführlich beschrieben und sollen an dieser Stelle deshalb nur kurz erläutert werden.

**Regelbasierte Verfahren** zielen darauf ab, Muster von Wörtern in Texten zu erkennen. Die Muster werden in Form von Regeln definiert, welche es ermöglichen, Wörter und Wortfolgen in Texten zu finden. Hierfür werden eigene, vollständig definierte Kunstsprachen wie z. B. **reguläre Ausdrücke** (vgl. Kleene 1956, S. 3–42) oder die Sprache DIAL (Dedicated Information Extraction Language for Text Mining, vgl. Feldman und Sanger 2006) verwendet.

Bei den **statistischen Verfahren** werden Texte unter Ausnutzung sprachstatistischer Gesetzmäßigkeiten nach verschiedenen Kriterien analysiert. Abhängig davon, wie der Begriff der Wahrscheinlichkeit gefasst wird, können wir zwischen frequentistischen und evidenzbasierten Ansätzen unterscheiden (vgl. Howie 2002). Bei den **frequentistischen Ansätzen** wird Wahrscheinlichkeit als Approximation der **relativen Häufigkeit** verstanden. Text-Mining-Verfahren, die auf diesem Ansatz aufbauen, untersuchen die Häufigkeit und statistische Verteilung von Sprachdaten. Hierzu gehören bspw. **Korpusvergleiche** zur Ermittlung von statistisch signifikanten Unterschieden in der Verwendung von **Vokabularen**, sowie **Kookkurrenzanalysen** und **Clusteranalysen** (vgl. Kap. 6, insbesondere Abschn. 6.2 und 6.5). **Evidenzbasierte Ansätze** bauen auf der **Bayesschen**

**Statistik** auf. Anstelle von Häufigkeiten werden Wahrscheinlichkeitsverteilungen verwendet, in denen Vorwissen und A-Priori-Annahmen explizit im Modell ausgedrückt werden. Hierzu gehören bspw. **Topic-Modelle** (vgl. Abschn. 6.4).

### Beispiel Häufigkeitsanalyse

Nehmen wir an, es sollen auffällige Wörter in einem Fachtext aus der Finanzbuchhaltung mit einem frequentistischen Verfahren identifiziert werden. Hierzu vergleichen wir (in einer noch festzulegenden Weise) die relative Häufigkeit von Wörtern aus diesem Fachtext (**Analysekörper**) mit der relativen Häufigkeit von Wörtern in einer großen, allgemeinsprachlichen Sammlung von Texten, beispielsweise einer Sammlung von Nachrichtentexten (**Referenzkorpus**). Schon dieser einfache Vergleich zeigt, dass z. B. die Wörter *Auftrag*, *Lieferung*, *Ware* oder *Kosten* für die Finanzbuchhaltung charakteristisch sind (mehr dazu im Abschn. 5.9). ◀

Bei der Anwendung von statistischen Verfahren, beispielsweise für die Textklassifikation, werden nur solche Wörter oder Wortfolgen betrachtet, die bestimmte, vorher festgelegte **Merkmale (Features)** erfüllen, beispielsweise Endungen wie *-itis* oder *-roid* als statistisch auffällige Endungen für Wörter in einem medizinischen Kontext. Im Unterschied dazu sind **neuronale Verfahren** in der Lage, für die Aufgabenstellung geeignete Feature-Repräsentationen selbst zu lernen, um eine Lösung für eine Aufgabe zu finden. Dabei werden Instanzen in neuronalen Architekturen immer durch reellwertige Vektoren repräsentiert, welche bezüglich einer Zielfunktion optimiert werden (mehr dazu in Kap. 3).

### 1.1.3 Text-Mining-Umgebungen

Mit dem rasanten Wachstum des Internets, der sozialen Medien und der zunehmenden Digitalisierung gedruckter Texte in maschinenlesbaren Text mithilfe der **Optical Character Recognition (OCR)** hat das Text Mining in den letzten Jahren stark an Bedeutung gewonnen. Erfolgreiche Anwendungen erfordern allerdings nicht nur inhaltsanalytische Verfahren, sondern auch den Aufbau einer vollständigen Prozesskette, welche insbesondere die folgenden Arbeitsschritte umfasst:

- Auswahl und Bereitstellung von Text (z. B. durch Sammeln von Texten im Internet oder OCR),
- Vorverarbeitung von Text (z. B. Entfernen von HTML-Tags und anderen Steuerzeichen, Tokenisierung, Dublettenbeseitigung),
- Einbindung von Wissensquellen für die Inhaltsanalyse (z. B. Lexika oder Enzyklopädien wie Wikipedia),

- Nutzung oder Entwicklung aufgabenangemessener Algorithmen für die Inhaltsanalyse,
- Einbindung der Inhaltsanalyse in einen aufgabenspezifischen Workflow,
- interaktive Visualisierung der Ergebnisse,
- Speicherung der Ergebnisse.

Weltweit finden sich zahlreiche Anbieter von Produkten und Services im Text Mining. Zusammenfassende Übersichten, welche die Produkte und Services nach verschiedenen Kriterien bewerten, werden regelmäßig von großen Beratungsunternehmen erstellt. Je nachdem, ob die gesamte Prozesskette im Vordergrund steht oder nur spezielle inhaltsanalytische Verfahren, lässt sich der Markt für Text Mining in drei große Gruppen teilen:

- Software für generische Text-Mining-Anwendungen,
- Software für anwendungsspezifische Text-Mining-Anwendungen,
- Algorithmen und Programmierumgebungen für die Inhaltsanalyse.

Eine Auswahl von Open-Source-Produkten im Text Mining, die für Lehrzwecke oder Machbarkeitsstudien eingesetzt werden können, findet sich im Anhang A1.

Generische Statistikprogramme und Software für das Data Mining, wie z. B. R (<https://www.r-project.org/>) und Weka (<https://sourceforge.net/projects/weka/>), können zwar auch für Text-Mining-Anwendungen eingesetzt werden. Im Unterschied zum Data Mining ist beim Text Mining jedoch zu beachten, dass Sprachdaten meist eine linguistische Verfahren einbeziehende Vorverarbeitung erfordern und eigenen sprachstatistischen Gesetzmäßigkeiten unterliegen.

---

#### Beispiel Rechtschreibungsnormen und Sprachstatistik

Für die Verarbeitung eines Textes ist zunächst festzulegen, welche Wörter als die kleinsten Einheiten auf Wortebene betrachtet werden sollen, also z. B. ob ein Mehrwortbegriff wie *Bundesrepublik Deutschland* oder *H-Milch* bei der Textanalyse als ein oder zwei Wörter verarbeitet werden soll. Solche Festlegungen sind sehr sprachspezifisch und haben einen unmittelbaren Einfluss auf die Qualität der Text-Mining-Ergebnisse (vgl. Kap. 4).

Die Verteilung von Wörtern in einem Text ist statistisch nicht gleich- oder normalverteilt, da wenige Wörter sehr häufig, aber viele Wörter selten vorkommen (mehr dazu im Abschn. 5.2). Diese Gesetzmäßigkeit ist bei der Auswahl des Mining-Algorithmus und der Interpretation der Ergebnisse zu berücksichtigen. ◀

### 1.1.4 Was leistet Text Mining?

Grundlage von Text-Mining-Anwendungen bilden natürlichsprachliche Texte wie Presseartikel, technische Dokumente, digitalisierte Bücher und Korrespondenzen, Internetforen oder Meldungen in sozialen Medien. Welche Texte für eine Anwendung verwendet werden, hängt davon ab, welche Fragestellung beantwortet werden soll. Dabei ist zu beachten, dass die inhaltliche Auswahl der Textquellen einen entscheidenden Einfluss auf die Qualität der Text-Mining-Ergebnisse hat. Die für eine Anwendung ausgewählten Texte bezeichnen wir als **Textkorpus**. Technische Aspekte der Textkorpuserstellung behandeln wir in Kap. 4.

In der Praxis existiert in bestimmten Bereichen von Organisationen erfolgskritisches Wissen in Form von Texten, welches ständig im Blick behalten werden muss, und für das Text Mining sinnvoll eingesetzt werden kann. Einige dieser Bereiche sind:

- Herstellung und Entwicklung von Produkten (Patente, Produktfeatures),
- frühzeitige Erfassung von Technologietrends,
- Dienstleistungen,
- Qualitätskontrolle,
- Marktanalysen,
- Optimierung von Betriebsabläufen.

Auch die Erschließung von neuen Wissensgebieten und Zusammenhängen kann durch die Analyse von Texten unterstützt werden. So können beispielsweise durch Patentanalysen aufkommende Technologietrends oder Abhängigkeiten und Nutzungsmöglichkeiten verschiedener Technologien schneller erkannt und deren relevante Komponenten effektiver identifiziert werden.

Für die automatische inhaltsbasierte Recherche in großen Textkorpora ist in Anlehnung an Morettis Konzept eines Distant Reading (Moretti 2013) zwischen der **Makro-Sicht** und der **Mikro-Sicht** zu unterscheiden (vgl. Jockers 2013). Die Makro-Sicht ermöglicht es, sehr umfangreiche Textkorpora zu strukturieren, z. B. durch das **Clustering** ähnlicher Texte oder der Identifizierung von Themen und deren Verteilung im Textkorpus. Die Makro-Sicht unterstützt damit die Suche nach relevanten Informationen und Zusammenhängen in einem sehr großen Suchraum. Die Mikro-Sicht ermöglicht es, aus einem Text auffällige Informationen zu extrahieren und zusammen mit den Informationen aus anderen Texten zu verbinden. In der Praxis fallen dabei typischerweise folgende Text-Mining-Aufgaben an:

- Identifizierung relevanter und anwendungsspezifischer **Fachausdrücke**,
- Identifizierung relevanter **Eigennamen (Named Entities)** in Bezug auf Personen, Organisationen, Produkte, Stoffe oder Orte,
- Berechnung **semantischer Relationen** zwischen Fachausdrücken oder Eigenamen,

- Berechnung semantisch ähnlicher Fachausdrücke, Eigennamen oder Relationen,
- Berechnung semantisch ähnlicher Texte,
- Auffinden von Definitionen, Erläuterungen und Referenzen in Texten.

Durch die Einbindung in einen Workflow ergeben sich daraus beispielsweise folgende Anwendungsfelder:

- effiziente und hochselektive Recherche in großen Textkorpora,
- automatische Filterung und Verteilung von Nachrichten (z. B. Verfolgen von Pressemitteilungen, E-Mails oder Tweets, klassifiziert nach Themengruppen oder Schlüsselwörtern),
- semi-automatischer Aufbau von fachspezifischen Glossaren, Thesauren und Synonymwörterbüchern,
- semi-automatische Erstellung von semantischen Netzen für das Wissensmanagement.

Zu diesen Anwendungen kommt hinzu, dass durch die Erkennung von Personennamen auch die Recherche nach handelnden Personen in bestimmten Domänen ermöglicht wird. Weiterhin können durch Text-Mining-Verfahren Stimmungen (Bewertungen für positiv/negativ/neutral) in Texten identifiziert werden. Durch die Anwendung von **Sentimentanalysen** wird deutlich, welche Wörter oder Wortfolgen negativ oder positiv besetzt sind.

Erweitert man den automatischen Charakter dieser Verfahren und Anwendungen mit dem professionellen Wissen von Domänenspezialisten und -spezialistinnen, so können weitere Vorteile aus Textanalysen gezogen werden, beispielsweise:

- detaillierte Analyse von Geschäftsberichten bzw. Reports,
- Analysen von Nachrichtenströmen,
- Beurteilung von Forschungssituationen,
- Identifikationen von Gefahren,
- Reputationsüberwachung,
- Identifikation von Trends über die Zeit oder verschiedene Orte,
- Beobachtung öffentlicher Reaktionen auf Aktivitäten wie Produktlaunches, Werbe- und PR-Kampagnen.

In den Sozialwissenschaften kann Text Mining die Inhaltsanalyse im Sinne einer regelgeleiteten und interpretativen Auswertungsmethode von Texten ergänzen (vgl. Kuckartz 2018). Ausdrücklich sei jedoch darauf hingewiesen, dass Text Mining die inhaltliche Analyse von Texten unter Anwendung hermeneutischer Methoden nicht ersetzen sondern lediglich unterstützen kann, weil die Operationalisierung komplexer geistes-, sozial- und kulturwissenschaftlicher Begriffe in hohem Maße von der zu bearbeitenden Forschungsfrage abhängt und geleistet werden muss, bevor die Text-Mining-Verfahren zum Einsatz kommen (vgl. Kantner und Overbeck 2020).

## 1.2 Text und Big Data

Mit dem Begriff *Text Mining*, also dem Abbau und der Aufbereitung des Wissensrohstoffs Text, wird bereits auf den Umstand hingewiesen, dass wir es mit der Erschließung und Auswertung sehr großer Mengen von Texten zu tun haben. Für die Verarbeitung sehr großer Datenmengen, insbesondere auch in Echtzeit, hat sich der Begriff **Big Data** durchgesetzt. Damit wird eine breite Palette an Technologien und Algorithmen bezeichnet, die für die Speicherung und Auswertung digitaler Massendaten zum Einsatz kommen (vgl. Meier 2018; Sakr und Zomaya 2019). Charakteristische Eigenschaften beschreibt das sogenannte **V-Modell** (siehe auch Zikopoulos et al. 2012):

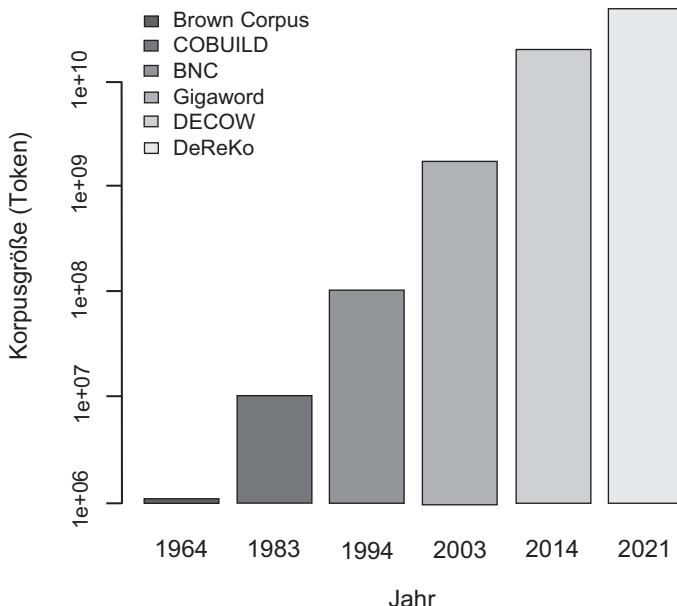
*Volume* (Volumen): Die zu handhabende enorme Datenmenge erfordert spezielle informatische Strategien, wobei sich die Grenze, ab wann Daten als *big* einzustufen sind, nicht konkret in Giga- oder Terabyte angeben lässt.

*Variety* (Vielzahl): Die abzubildenden Inhalte und deren wechselseitigen Beziehungen sind hochkomplex und umfassen gleichermaßen strukturierte, semistrukturierte und unstrukturierte Daten.

*Velocity* (Verarbeitungsgeschwindigkeit): Zeitoptimierte Recherche- und Analyse schritte tragen zur Nutzungsakzeptanz bei, wobei grundsätzlich alle Datenoperationen betroffen sind, also das Anlegen, Lesen, Verändern und Löschen.

Je nach Schwerpunktsetzung haben sich neben diesen drei initialen Kriterien eine ganze Reihe weiterer *Vs* etabliert, etwa *Veracity* (Verlässlichkeit), *Value* (Nutzen), *Variability* (Varianz) oder *Validity* (empirische Validität).

Den damit befassten Unternehmen oder Forschungseinrichtungen kommt die umfassende Digitalität moderner Produktions- und Kommunikationskanäle zugute. Ob Zeitschriftenartikel, Kriminalroman, Parlamentsrede oder Online-Diskussions forum: Computerbasiert erstelltes Sprachmaterial ist mittlerweile flächendeckend für viele moderne Textsorten und Kommunikationssituationen verfügbar. Unter Beachtung rechtlicher, insbesondere datenschutz- und urheberrechtlicher Rahmenbedingungen lässt es sich vergleichsweise einfach in eine linguistische Verarbeitungspipeline (vgl. Abschn. 3.2) einspeisen und muss nicht, wie beispielsweise bei historischen Quellen, vermittels aufwendiger Erkennungsverfahren nachdigitalisiert werden. Die rapide technologische Weiterentwicklung von Speichermedien hat im Übrigen dafür gesorgt, dass bei der für reproduzierbare Analysen unabdingbaren langfristigen („persistenten“) Archivierung solcher Massendaten kaum noch physische Kapazitätsprobleme anfallen. Entsprechend entwickeln sich die Volumina von Textressourcen; Abb. 1.1 illustriert das Wachstum am Beispiel prominenter Textkorpora. Es wird eine nahezu exponentielle Zunahme der Datenmenge sichtbar: Während das erste digitale Korpus („Brown Corpus“, vgl. Francis und Kučera 1967) Mitte der 1960er Jahre gerade einmal etwas mehr als eine Million gespeicherter Wörter aufwies, enthielt die zwanzig Jahre später als „Bank of England“ bekannt gewordene COBUILD-Sammlung (vgl. Moon 2009) 10 Mio. und das in den 1990er Jahren kompilierte British National Corpus (BNC; vgl.



**Abb. 1.1** Größenentwicklung ausgewählter Textkorpora (logarithmisierte Darstellung)

Aston und Burnard (1998) bereits 100 Mio. Wörter. Im Folgejahrzehnt überschritt das „English Gigaword Corpus“ (vgl. Graff und Cieri 2003) die Milliardengrenze. Auch deutschsprachige Kollektionen verschieben die Grenzen kontinuierlich nach oben: Sowohl das auf Internet-Crawling basierende DECOLW (deutschsprachiges Archiv der „Corpora from the Web“; vgl. Schäfer und Bildhauer 2012) mit über 20 Mrd. als auch das Deutsche Referenzkorpus DeReKo des Leibniz-Instituts für Deutsche Sprache (IDS) (vgl. Kupietz et al. 2018) mit über 50 Mrd. Wörtern haben den Umfang nochmals erhöht; die Leipzig Corpora Collection (LCC; vgl. Goldhahn et al. 2012) hält neben dem Deutschen Korpora für gar mehr als 250 weitere Sprachen vor.

An diesem Punkt verlangt die Analogie von Text Mining zum Erz- oder Kohlebergbau jedoch nach einer Präzisierung. Während bei Bodenschätzten neben der absoluten Fördermenge stets eine möglichst einheitliche Rohstoffkonsistenz erwünscht ist, liegt die Attraktivität des Wissensrohstoffs Text maßgeblich in seiner Heterogenität. Als Beispiel sei hier die Erkennung initialer Sprachwandelprozesse genannt. Selbst vordergründige „Fehler“ im Datenmaterial, also ungewöhnliche Schreibungen, unbekannte Wörter oder syntaktische Auffälligkeiten, beinhalten nicht nur aus deskriptiver linguistischer Sicht wertvolle Informationen. In einem dynamischen System wie der menschlichen Sprache können die Abweichungen von heute Hinweise auf den Standard von morgen liefern und sollten deshalb keinesfalls aussortiert werden.

## 1.3 Aufbau und Struktur von Text

### 1.3.1 Arten von Text und ihre Merkmale

Ein Text besteht aus Wörtern, die nach Art und Zweck des Textes zu größeren Einheiten zusammengefasst werden (Sätze, Absätze, Abschnitte, Kapitel), vgl. auch Lissmann (2008).

Von besonderem Interesse für das Text Mining sind Texte, die in der Absicht geschrieben worden sind, Wissen zu vermitteln. Hierzu zählen u. a.:

- wissenschaftliche Aufsätze,
- Fachbücher,
- enzyklopädische Nachschlagewerke und Lexika,
- technische Dokumentationen, Benutzerhandbücher und Produktbeschreibungen,
- Normen, Gesetze, Kommentare und Verträge,
- Organisationsanweisungen,
- Korrespondenzen.

Wissen kann aber auch aus vielen anderen Arten von Texten extrahiert werden, deren Zweck zwar nicht primär die Formulierung von Wissen ist, die aber implizit wissensrelevante Wörter wie z. B. Personen oder Ortsnamen enthalten und inhaltliche Zusammenhänge herstellen oder abzuleiten ermöglichen, beispielsweise welche Personen mit welchen Orten oder welchen Funktionen in Verbindung gebracht werden. Beispiele für diese Arten von Texten sind:

- digitalisierte historische Texte wie Romane, Journale, Reisebeschreibungen,
- Zeitungen,
- E-Mails,
- Kommunikation in sozialen Medien,
- Kommunikation mit Chatbots und Frage-Antwort-Systemen.

Unabhängig von der inneren Struktur eines Textes und dem in ihm explizit oder implizit repräsentierten Wissen, sind für die weitere Verarbeitung von Texten auch die folgenden externen Merkmale relevant, da sie für die Einordnung und Nutzung eines Textes oft wichtige Hinweise geben:

- In welcher Organisation ist der Text entstanden?  
(z. B. Wirtschaft, Behörden, Journalismus, Wissenschaft, usw.)
- In welcher Sprache ist der Text verfasst?  
(z. B. Deutsch (BRD), Deutsch (AT), Deutsch (CH), usw.)
- Um welche Form der textuellen Mitteilung handelt es sich?  
(z. B. Buch, Zeitung, Zeitschrift, Bulletin, E-Mail, usw.)

- Um welchen Typ von Text handelt es sich?  
(z. B. Aufsatz, Dokumentation, FAQ, Katalog, Protokoll, Werbung, Webseiten, usw.)
- Von welchem Sachgebiet handelt der Text?  
(z. B. Wirtschaft, Politik, Naturwissenschaft, Kunst, usw.)

Für die weitere Verarbeitung von Texten werden die genannten Merkmale in Form von Metadaten notiert. **Metadaten** für Texte sind Attribut-Wert-Paare, welche Texte in technischer, organisatorischer und inhaltlicher Hinsicht beschreiben. Mit dem Standard **Dublin Core** hat die Dublin Core Metadata Initiative (DCMI) einen de facto Standard für Metadaten geschaffen (vgl. DCMI 2020).

Im Folgenden wird Text als Wissensrohstoff betrachtet, aus dem mit Hilfe von Text-Mining-Verfahren das im Text enthaltene Wissen extrahiert und aufbereitet werden kann. Als Grundlage hierfür wird zunächst das Verhältnis von Text und Wissen im Allgemeinen beschrieben.

### 1.3.2 Zeichen, Types und Wörter

Ein Text besteht aus *Wörtern* und Wörter aus *Buchstaben* (einschließlich Leer- und Sonderzeichen) wie sie beispielsweise im Codierungsstandard UNICODE (vgl. unicode.org 2021) festgelegt sind. Damit ist ein Text letztlich eine Aneinanderreihung von Zeichen (vgl. Anhang A4, Formale Rekonstruktion linguistischer Ebenen).

Ein Text kann aus zwei Perspektiven betrachtet werden: Aus Sicht des Text Mining steht dabei die Praxis der Erstellung, Speicherung und Verarbeitung von Textkorpora im Mittelpunkt, aus Sicht der Linguistik aber die linguistischen Aspekte der linguistischen Ebenen, speziell Morphologie, Lexikon, Syntax, Semantik und Pragmatik (vgl. Anhang A3, Linguistische Ebenen).

Für das Text Mining ist die Unterscheidung zwischen sog. **Types** und **Tokens** von zentraler Bedeutung. Sie geht auf den englischen Logiker C.S. Peirce (2009) zurück und dient der Unterscheidung zwischen einem allgemeinen Muster (Type) und den Vorkommen oder Instanziierungen dieses Musters (Token) beim Zählen von Wörtern in einem Text. Das **Type-Token-Verhältnis** (engl. Type-token-ratio) gibt Hinweise auf die lexikalische Vielfalt in einem Korpus (Abschn. 5.1).

Den hier verwendeten Begriff des *Musters* (engl. *pattern*) wollen wir wie folgt präzisieren: Für die Zerlegung eines Textes in Tokens gibt es meist verschiedene Möglichkeiten, für die eindeutige Regeln festgelegt werden müssen. Im Folgenden bezeichnen wir diese Regeln als **Tokenregeln** (vgl. Kap. 3 und 4). Indem eine Tokenregel festlegt, welche Zeichenketten als Token betrachtet werden, legt sie aber auch fest, welche Zeichenketten als gleich betrachtet werden. Die Anwendung einer Tokenregel auf einen Text, nachfolgend als **Tokenisierung** bezeichnet, liefert daher als Ergebnis eine Menge von Zeichenketten, die – entsprechend der Tokenregel – als gleich betrachtet werden. Eine Menge von Elementen, die eine gemeinsame Eigenschaft haben, wird als **Äquivalenz-**

**klasse** bezeichnet. Während also „Token“ das tatsächliche Vorkommen einer Zeichenkette in einem Text bezeichnet, verstehen wir unter dem Type einer Zeichenkette die Äquivalenzklasse derjenigen Zeichenketten, die als gleich betrachtet werden. Die Anzahl von Types und Tokens in einem Text hängt direkt ab von den angewendeten Tokenregeln. Für ein und denselben Text kann es dabei deutliche Unterschiede geben (vgl. Abschn. 5.1).

Bei der linguistischen Betrachtung eines Textes sind auch die linguistischen Aspekte der Morphologie und Syntax zu berücksichtigen. Weil einzelne Token syntaktisch mehrdeutig sein können, beispielsweise kann das Token *einen* einerseits die Funktion des unbestimmten Artikels haben, andererseits aber auch die Funktion eines Verbs, wollen wir aus Perspektive der Linguistik auch zwischen den Wortformen und den Wörtern in einem Text unterscheiden, wobei wir bewusst einen datengetriebenen Ansatz wählen, der von dem sonst in der Linguistik üblichen abweicht (vgl. Anhang A3, Linguistische Ebenen):

Eine **Wortform** ist die flektierte Erscheinungsform eines Wortes, wie sie in einem syntaktischen Kontext vorkommt. Als Bezeichner für ein Wort, beispielsweise in einem Lexikon, wird in der Regel eine ausgezeichnete Wortform verwendet, die als **Grundform** oder auch **Lemma** bezeichnet wird. Wortformen, welche dieselbe **Grundform** haben, können wir zu einer Äquivalenzklasse zusammenfassen, die wir als **Wort** bezeichnen (z. B. alle flektierten Formen von *schreiben*).

### Wortform und Wort

*einen* als Artikel hat als Grundform *ein*, weitere Wortformen von *ein* als Artikel sind: *eine, einem, eines, einer* usw.

*einen* als Verb hat als Grundform *einen*, weitere Wortformen von *einen* als Verb sind: *eine, einst, eint, einte, geeint* usw. ◀

Zur Vereinfachung werden wir in diesem Buch immer von Wörtern sprechen und nur nach Bedarf zwischen Types, Tokens, Wortformen, Grundformen und Wörtern (im engeren Sinne) unterscheiden.

### 1.3.3 Nachricht, Information

Ein Text als eine (geordnete) Folge von Wörtern repräsentiert eine Menge von Aussagen, die wir intuitiv als Wissen bezeichnen. Allerdings erschließt sich uns dieses Wissen erst, wenn wir den Text (in einem bestimmten Sinne) verstanden haben. Das aber setzt voraus, dass die Folgen von Wörtern semantisch richtig verstanden worden sind.

### Beispiel Text in einer fremden Sprache

Betrachten wir zur Verdeutlichung den folgenden Text, einen Auszug aus der Firmengeschichte des finnischen Mobilfunkherstellers Nokia (2020):

Vuonna 1982 Nokia toimitti ensimmäisen täysin digitaalisen puhelinkeskuksen Euroopassa ja samana vuonna esittelimme maailman ensimmäisen autopuhelimen analogiselle NMT-standardille. 1980-luvulla kehitetty GSM-standardi mahdollisti korkealaatuiset äänipuhelut, hyödynsi entistä tehokkaammin radiotaajuuksia ja tarjosi korkealaatuiseen äänentoiston. Ensimmäinen GSM-puhelu soitiin Nokian matkapuhelimella yhtiön Radiolinja-operaattorille rakentamassa verkossa vuonna 1991.

Samoihin aikoihin Nokia teki strategisen päätöksen keskittyä televiestintälaitteisiin. Nokian muut liiketoiminnot, mukaan lukien alumiini-, kaapeli-, kemikaali-, paperi-, kumi-, sähkötuotanto- ja televisiotoiminnot myytiin.

Vuoteen 1998 mennessä Nokia oli maailman johtava matkapuhelinvalmistaja, ja yhtiö piti kiinni tästä johtoasemastaan yli 10 vuoden ajan.

Liiketoimintaympäristö ja teknologia jatkoivat kuitenkin kehittymistään, ja näin teki myös Nokia.

Wer als Leser oder Leserin nicht des Finnischen mächtig ist, wird sich schwertun, den Text zu verstehen. Intuitiv sind einzelne Wörter zu erkennen, von denen wir annehmen können, dass sie im Deutschen eine ähnliche Bedeutung haben, wie im Finnischen, z. B. *Nokia, 1982, GSM* usw. Insgesamt aber ist nicht erkennbar, welche Aussagen der Text enthält, sofern wir den Text nicht im engeren Sinne verstehen. ◀

Die theoretische Grundlage für die automatische Verarbeitung von Zeichen und Informationen ist die Informationstheorie (vgl. Bauer et al. 1991; Bauer und Goos 1992; Shannon und Weaver 1949). **Zeichen** sind Elemente eines endlichen geordneten Zeichenvorrats, **Alphabet** genannt. Durch Ordnungsregeln (insbesondere eine Grammatik) lassen sich Zeichen zu Zeichenketten kombinieren. Eine nach vorher festgelegten Regeln zusammengestellte, endliche Folge von Zeichen bezeichnen wir als **Nachricht**. Eine Nachricht zusammen mit ihrer Bedeutung für einen Empfänger<sup>1</sup> ist eine **Information**. Physikalisch werden Zeichen als **Signale** zwischen einem Sender und Empfänger ausgetauscht. Charakteristisch für die Informatik ist dabei die Verwendung *digitaler Signale*, d. h. Signale, deren Signalhöhe nur eine begrenzte Anzahl von Werten, meist nur die zwei Werte „an“ oder „aus“, annehmen kann. Die so ausgetauschten Nachrichten werden als **Daten** bezeichnet. Daten sind also noch nicht interpretierte Zeichen bzw. Zeichenfolgen, die erst durch die Herstellung eines Interpretationsbezugs zu Informationen werden.

<sup>1</sup> Da es sich hier um Bezeichnungen aus dem Kommunikationsmodell nach Shannon und Weaver (1949) handelt, wurden die Begriffe *Empfänger* und *Sender* ohne Angleichung an die Geschlechter übernommen. Sie können sich sowohl auf Personen jeglichen Geschlechts als auch auf Maschinen beziehen.

### Beispiel zum Interpretationsbezug

Welche Information wird durch die binäre Zeichenfolge *01101101* ausgedrückt? Damit diese Zeichenfolge eine Information erhält, muss zunächst festgelegt werden, ob sie beispielsweise als Binärzahl, als **ASCII**-codiertes Zeichen oder als eine Anweisung im Maschinencode (Assembler) interpretiert werden soll.

Die konkrete Bedeutung der Zeichenfolge erschließt sich erst, wenn der Interpretationsbezug festgelegt worden ist. So bedeutet die Zeichenfolge z. B. in der ASCII-Codierung den kleinen Buchstaben *m*, oder mit dem Interpretationsbezug Dualzahl die Dezimalzahl *109*. ◀

Nachrichten enthalten Informationen. Mehrfach vorhandene Informationseinheiten werden in der Nachrichtenübertragung als redundant bezeichnet. Diese können ohne Informationsverlust weggelassen werden, erhöhen aber die Übertragungssicherheit und können zur Fehlerkorrektur benutzt werden. Unter **Redundanz** in Texten versteht man das wiederholte Auftreten der gleichen Information, oft in unmittelbarer Nachbarschaft. Natürliche Sprachen enthalten sehr viel Redundanz, sodass auch z. B. verstümmelte Sätze noch verständlich bleiben, wie der folgende Beispielsatz verdeutlicht, bei dem jedes 6. Zeichen weggelassen worden ist:

„Das Gwandhus zuLeipzg befndet ich a Augutuspltz“

#### 1.3.4 Information und Wissen

Als Nachricht, die für die empfangende Person bzw. Maschine nach einem festgelegten Informationsschlüssel eine Bedeutung hat, besteht eine **Information** aus Daten, die in einem Bedeutungskontext stehen. Damit allerdings diese Information für die empfangende Person bzw. Maschine auch wertvoll ist, muss sie in der Regel mit anderen aktuellen oder in der Vergangenheit gespeicherten Informationen vernetzt werden. Diese Vernetzung wird durch das Wissen einer Person oder Organisation geleistet. Wissen ist ein Prozess der zweckdienlichen Vernetzung und Interpretation von Informationen. In Anlehnung an gängige Definitionen in der Künstlichen-Intelligenz-Forschung kann unter **Wissen** somit die Gesamtheit der Kenntnisse, Fähigkeiten und Fertigkeiten verstanden werden, die Personen zur Lösung von Problemen einsetzen (vgl. Encyclopedia of Artificial Intelligence, Shapiro et al. 1987, S. 291). Dies umfasst sowohl theoretische Erkenntnisse als auch praktische Alltagsregeln und Handlungsanweisungen.

Wird dagegen der Inhalt von Informationen nicht ausgewertet, sondern werden die Informationen nur als sinnhafte Datenobjekte behandelt, sprechen wir von Content. **Content** wird heute meist mit Hilfe von Content-Management-Systemen strukturiert erfasst, verwaltet und medienneutral verarbeitet. Als Wirtschaftsgut wird Content als **Asset** bezeichnet. Die Tab. 1.2 fasst die skizzierten Zusammenhänge zusammen.

**Tab. 1.2** Text und seine semantischen Anreicherungen

Text	Uninterpretierte, strukturierte oder unstrukturierte Daten vom Typ <i>string</i>
Information	Interpretierte Daten
Wissen	Vernetzte Informationen
Content	Information als Objekt und Austauschgegenstand
Asset	Information als Wirtschaftsgut

### 1.3.5 Beispieldateien und Textressourcen

Für die beispielhafte Anwendung von Text-Mining-Verfahren finden sich zahlreiche öffentlich zugängliche Textquellen, beispielsweise die Daten der Deutschen Digitalen Bibliothek (<https://www.deutsche-digitale-bibliothek.de/>), CLARIN-D (<https://www.clarin-d.net/de/>), eine Forschungsinfrastruktur für die Geistes-, Kultur- und Sozialwissenschaften, die Forschende beim Auffinden, Aufbewahren und Auswerten von textbasierten Forschungsdaten unterstützt, oder die umfangreichen Textsammlungen des Internet Archives (<https://archive.org/>).

Für die Evaluation von Text-Mining-Verfahren haben sich in den letzten Jahren spezielle Evaluationsplattformen etabliert. Diese Plattformen enthalten Dokumente, Themenzuordnungen und Relevanzbewertungen, die im Rahmen von jährlich stattfindenden Vergleichswettbewerben für die Evaluation und den Vergleich von Text-Mining-Verfahren genutzt werden. Unter Bereitstellung einer vorgegebenen **Ground Truth** werden Texte aus verschiedenen Wissenschaftsbereichen wie Medizin, Recht, Chemie oder Bioinformatik sowie verschiedenen Medien, beispielsweise Blogs, Nachrichtentexten, Frage-Antwort-Systemen oder natürlichsprachlichen Dialogsystemen, angeboten. Beispielhaft genannt sei an dieser Stelle SemEval (<http://alt.qcri.org/semeval2020/index.php?id=tasks>), eine Plattform der Association for Computational Linguistics (ACL) für die Evaluation von Verfahren der Computational Semantics. Diese Plattform dient der Herstellung der Vergleichbarkeit von Text-Mining-Verfahren durch die Verwendung gleicher Textkollektionen und Evaluierungsverfahren. Neben der Bereitstellung realistisch großer Textkorpora umfasst SemEval auch eine Umgebung für die Testdurchführung und -auswertung, wobei das Wettbewerbsprinzip auch der Weiterentwicklung der Evaluationssystematik selber dient. Zusätzlich können bereitgestellte Texte auch zum Trainieren von Verfahren verwendet werden, die mit maschinellen Lernverfahren arbeiten.

Als Referenztexte, an denen die in diesem Buch beschriebenen Text-Mining-Verfahren erprobt werden können, verwenden wir Zeitungstexte und Wikipedia-Artikel aus dem Projekt „Deutscher Wortschatz“, (vgl. Anhang A6). Diese Daten basieren auf öffentlich verfügbaren Quellen, die in den letzten Jahren täglich durch Crawling gesammelt worden sind (und weiterhin gecrawlt werden) und für das Text Mining rechts sicher aufbereitet worden sind. Die automatisch erzeugten Korpora stehen sortiert nach

Sprache, Umfang und Zeitraum unter Verwendung gleicher Formate und vergleichbarer Quellen für wissenschaftliche Zwecke zur Verfügung.

Grundsätzlich sind Texte als Sprachwerke aller Art durch das **Urheberrecht** in Deutschland geschützt (vgl. § 2 Abs. 1 Nr. 1 UrhG 2021). Eigentümer bzw. Eigentümerin aller Verwertungsrechte eines Textes ist sein Urheber bzw. seine Urheberin. Ohne deren Zustimmung ist insbesondere die Speicherung und Verbreitung von Textpassagen, die mehr als einen Satz enthalten, grundsätzlich verboten. Ausdrücklich erlaubt ist es allerdings, einen Satz unter Angabe der Quelle zu zitieren, ohne dass es einer Zustimmung des Urhebers und sonstiger Rechteinhaber bedarf (§ 51 UrhG 2021). Auch das Text Mining ist erlaubt. Die 2021 neu geschaffene Grundlage dafür ergibt sich aus dem Gesetz zur Anpassung des Urheberrechts an den digitalen Binnenmarkt (neue Fn.: BGBl I, Nr. 27, S. 1204). Grundlage für Text Mining zu wissenschaftlichen Zwecken ist § 60d UrhG 2021. Danach dürfen Texte vervielfältigt und einem begrenzten Kreis der Öffentlichkeit auch zugänglich gemacht werden. Das gilt auch für die aus verschiedenen Werken erstellten Korpora. Jedoch sind bestimmte Vorgaben einzuhalten, insbesondere die Löschung des Ursprungsmaterials nach Abschluss der Forschungsarbeiten (vgl. 60d Abs. 5 UrhG 2021). Bei der rechtlichen Bewertung einer Text-Mining-Anwendung sind im Einzelfall stets die kommerziellen Interessen der Urheber, die wissenschaftlichen und kommerziellen Interessen der Entwickler von Text-Mining-Software und -Anwendungen sowie die Schutzrechte von natürlichen und juristischen Personen, bspw. Persönlichkeitsschutzrechte, miteinander in Einklang zu bringen, wofür in der Regel juristischer Rat unerlässlich ist.<sup>2</sup>

Die Korpora des Projekts „Deutscher Wortschatz“, die wir in diesem Buch als Referenztexte verwenden, enthalten zufällig ausgewählte Sätze aus verschiedenen Korpusprachen und können in Größen von 10.000 bis 1.000.000 Sätzen von der angegebenen Webseite heruntergeladen werden. Als Quelle werden typischerweise entweder Nachrichtentexte oder das Ergebnis des allgemeinen Webcrawls verwendet. Aus urheberrechtlichen Gründen sind die verwendeten Texte immer in einzelne Sätze zerlegt und diese zufällig sortiert, sodass eine Wiederherstellung des Ursprungstextes nicht möglich ist. Ungrammatische Sätze und fremdsprachliches Material wurden bestmöglich entfernt (vgl. Goldhahn et al. 2012).

Das nachfolgende Beispiel (deu\_newscrawl-public\_2019\_10K-sentences.txt) zeigt die ersten 12 Sätze (von insgesamt 10.000) aus dem Normgrößenkorpus „Deutsche Zeitungstexte aus Online-Quellen 2019“ in alphabetischer Sortierung als Ergebnis des automatischen Korpuserstellungsprozesses (siehe Anhang A5 und Anhang A6):

---

<sup>2</sup>Dem Gesetzestext entsprechend, auf den sich dieser Absatz bezieht, wurde hier das generische Maskulinum verwendet.

**Die ersten 12 Sätze (von insgesamt 10.000) aus „Deutsche Zeitungstexte aus Online-Quellen 2019“ in alphabetischer Sortierung**

1. „70 Prozent aller Menschen in Deutschland brauchen irgendwann im Leben Blutkonserven“, so Hackstein. (<https://www.giessener-allgemeine.de/giessen/unspektakulaere-art-menschenleben-retten-12169303.html>, 2019-06-03)
2. Aachen (an-o) – Am Freitag Abend startet wieder die Nacht der offenen Kirchen. (<https://www.aachener-nachrichten.de/lokales/archiv/2002/10/10/>, 2019-06-03)
3. Aachener Oberstufenschüler zeigen Verantwortung für sozial Schwächeren: Am Freitagabend, 6. Juli, ist es so weit. (<https://www.aachener-nachrichten.de/lokales/archiv/2007/07/02/>, 2019-05-30)
4. Ab 1297 hatte Dunwich das Recht, zwei Abgeordnete ins Parlament zu entsenden, das entsprach damals der Größe der Stadt. (<https://www.stern.de/panorama/wissen/mensch/britisches-atlantis-tauchgang-zur-versunkenen-stadt-3220302.html>, 2019-06-03)
5. Ab 16.30 Uhr berichtet Maximilian Semsch im Göttinger GDA-Wohnstift von seiner viermonatigen Reise mit dem E-Bike durchs Heimatland. (<https://www.goettinger-tageblatt.de/Die-Region/Goettingen/Klingebiel-Zelle-Thema-im-Landtag>, 2019-06-03)
6. Ab 2018 können die Bundesländer ihre Schüler früher in die Sommerferien schicken und sich beim Beginn weiter abwechseln. ([https://www.allgaeuer-anzeigeblatt.de/index.shtml?artikelarchiv\\_2014=0000017592](https://www.allgaeuer-anzeigeblatt.de/index.shtml?artikelarchiv_2014=0000017592), 2019-05-29)
7. ABB hat in China einen grösseren Auftrag von SAIC Volkswagen erhalten. (<https://www.cash.ch/aktien/abb-n-1222171/swl/chf>, 2019-05-29)
8. Ab dann werden Lkws einer regionalen Spedition, die mit einem speziellen Stromabnehmer ausgestattet sind, regelmäßig diese Strecke befahren. (<https://www.aktiv-online.de/news/ehighway-auf-der-autobahn-a1-entsteht-teststrecke-mit-oberleitungen-fuer-hybrid-lkws-3213>, 2019-05-29)
9. Ab dem 1. Januar ist der frühere Bürgermeister von Lundtoft ohne politisches Amt – sowohl kommunal als auch regional. (<https://www.nordschleswiger.dk/de/nordschleswig-apenrade/tietje-senior-hat-es-kommen-sehen>, 2019-05-31)
10. Ab diesem Donnerstag treten die Stars um Tiger Woods und Co. beim zweiten Major des Jahres an. (<https://www.muensterschezeitung.de/Sport/Sportarten/Golf>, 2019-05-29)
11. Abensberg (dpa/lby) – Rund ein Jahr vor dem nächsten Weihnachtsfest haben in Niederbayern unbekannte Täter eine Christbaumplantage mit rund 500 Bäumen verwüstet. (<https://www.bild.de/regional/muenchen/unbekannte-verwuesten-christbaumplantage-27919320.bild.html>, 2019-05-31)
12. Aber alles verlief gut“, schildert Michael Fröhlich, „zeigte aber auch, wie wichtig unser Einsatz ist“. (<https://www.vogtland-anzeiger.de/vogtland/theaterwachen-sichern-spielbetrieb-ab-artikel110470224>, 2019-05-30) ◀

## 1.4 Redundanz in Texten

Redundanz kommt in der Sprache häufig vor und hilft beim Verständnis. Redundanz geht weit über die einfache Wiederholung als stilistisches Mittel der Rhetorik hinaus und liegt auch dann vor, wenn Informationen aus einem Textteil mehrfach geschlossen werden können (vgl. Pompino-Marschall 2010, S. 552). So steckt in der Phrase *die Hintertür des Hauses* doppelt die Information, dass es sich um einen Genitiv handelt: sowohl im Artikel *des* wie auch in der flektierten Form *Hauses*. Auch in *Bundeskanzlerin Angela Merkel* ist die Information, dass es sich um eine Frau handelt, doppelt kodiert: in dem weiblichen Titel *Bundeskanzlerin* und in dem weiblichen Vornamen *Angela*.

So betrachtet lässt sich Redundanz beschreiben als das mehrfache Vorliegen derselben Information in expliziter oder impliziter Form. Von besonderem Interesse ist die implizite Form: Hier werden Wörter (meist Wortpaare) durch eine gemeinsame und wiederholte Eigenschaft verbunden und stehen dadurch in einer zunächst nicht näher definierten Relation. Solche Redundanzen lassen sich als Wiederholungen mit statistischen Verfahren extrahieren und sind damit Grundlage für die Erkennung von Zusammenhängen zwischen verschiedenen Wörtern.

Redundanz ist eine wichtige Eigenschaft aller natürlichen Sprachen, allerdings unterscheiden sich die Varianten von Redundanz von Sprache zu Sprache sehr. Im Folgenden sollen verschiedenen Arten von Redundanzen für die deutsche Sprache untersucht werden, um danach deren Auswirkungen für das Text Mining zu betrachten.

### 1.4.1 Arten von Redundanz

Wiederholt werden können vollständige Wörter, Teile von Wörtern oder verschiedene Wörter mit übereinstimmenden Eigenschaften. Auch der Abstand der Wörter kann dabei unterschiedlich sein: Gleiche Wörter werden meist nur mit einem größeren Abstand wiederholt, während zusammengehörige benachbarte Wörter sich häufig durch eine beiden innenwohnende Eigenschaft auszeichnen.

#### 1.4.1.1 Nahe Wiederholung von Wörtern im Text

Aus der Häufigkeit eines Wortes in einem größeren Korpus lässt sich ableiten, wie groß der mittlere Abstand zwischen zwei Vorkommen des Wortes sein sollte. Diese Berechnung können wir später mithilfe des Zipfschen Gesetzes recht genau vornehmen (vgl. Abschn. 5.2). Aber schon jetzt ist klar, dass in einem Text zu einem bestimmten Thema einige Wörter öfter vorkommen als üblich, was mit der Kohärenz von Texten zu tun hat, aufgrund derer dieselben Konzepte mehrfach referiert werden; siehe Church (2000) zum mehrfachen Auftreten von Wörtern innerhalb desselben Textes. Diese im Durchschnitt eher seltenen Wörter, die im betrachteten Text vergleichsweise häufig vorkommen, sind in der Regel wichtig. Dies können Fachwörter aus einem bestimmten Sachgebiet sein oder die Eigennamen der wichtigen Personen oder Orte.

Das häufige Auftreten normalerweise seltener Wörter spricht für deren Wichtigkeit (vgl. Abschn. 7.1).

Statt einzelner Wörter können natürlich auch ganze Wortgruppen wiederholt auftreten. Dies ist der Fall für Fachbegriffe, die aus mehreren Wörtern bestehen oder Personennamen, bestehend aus Titel, Vor- und Nachnamen.

### 1.4.1.2 Wiederholung von Wortteilen

Es können sich natürlich auch kleinere Teile als ganze Wörter wiederholen. Speziell in der deutschen Sprache mit vielen Komposita können sich Teile wiederholen. In einem Text finden wir beispielsweise gleichzeitig die Wörter *Gesetz*, *Gesetzentwurf*, *Gesetzes-*  
*text*, *Gesetzgebungsverfahren* usw. Aus den wiederkehrenden Teilen kann hier auf den Inhalt des Textes geschlossen werden.

#### Wiederholung von Wortendungen

Wiederholen können sich auch nicht bedeutungstragende Wortbestandteile wie Endungen, z. B. die Endung *-ung* im folgenden Beispiel. Hier wird der Sprachstil deutlich. Häufige Substantivierungen lassen den Sprachstil als Verwaltungssprache erscheinen: *Zur Beschleunigung der Planung und in Abstimmung mit der Verwaltung und den Ortschaftsräten tritt die Stadt bei der Planung in Vorleistung.* ◀

Außerdem eignen sich wiederkehrende Wortteile in vielen Fällen auch zur Klassifikation der entsprechenden Wörter. Bei solchen Wortteilen handelt es sich oft um sogenannte Morpheme, siehe Abschn. 1.5. Dabei müssen diese Wortteile sich gar nicht in enger Nachbarschaft wiederholen, sondern nur auffällig oft vorkommen. Die Wörter werden dann aufgrund ihrer Struktur bei jedem einzelnen Vorkommen erkannt. So kann man für ein auf *-ung* endendes Wort schlussfolgern, dass es sich um ein weibliches Nomen handelt. Aber Achtung, für diese Regel gibt es Ausnahmen, siehe Abschn. 2.6. Auch inhaltliche Schlussfolgerungen sind möglich: *Sofja Kowalewskaja* kann man als weiblichen slawischen Namen wegen der Form *-a -aja* erkennen oder *Kobiaschwili* als georgischen Nachnamen wegen der Endung *-wili*.

### 1.4.1.3 Wiederholung in der Struktur: Eigennamen

Woran erkennt man im Text bei einem unbekannten Wort (ohne die oben beschriebenen strukturellen Auffälligkeiten), dass es sich um einen Nachnamen handelt? Vielleicht, weil man an dieser Position im Satz einen Nachnamen erwarten würde. Hier geht es also um Eigenschaften der benachbarten Wörter. Personenbezeichnungen haben häufig die Form *Titel/Beruf Vorname Nachname*. Wenn man beispielsweise liest ... *sagte Tischlermeister Jürgen Arzt* ..., dann schließt man aus der Reihenfolge der Wörter, dass *Arzt* Nachname sein muss, da davor eine Berufsbezeichnung und ein bekannter Vorname stehen. Ähnliche Muster gibt es für Ortsbezeichnungen (... *vor der russischen Doppelinsel Nowaja Semlja* ...) sowie Organisationsnamen (... *die Zeitung El País berichtet*). Hier steht häufig eine nähere Beschreibung vor dem Eigennamen.

Nützlich ist hier das Vorgehen, beim Schreiben die Struktur des Kontextes so zu wählen, dass Redundanz enthalten ist. Sollte einem Teil der Leserschaft der Eigenname allein zur Information nicht ausreichen, so werden zusätzliche, erklärende Angaben in einem wohlbekannten Muster präsentiert. Im Rahmen des Text Mining kann dies genutzt werden, um die zusätzlichen Informationen zu erkennen.

#### 1.4.1.4 Kongruenz

Unter Kongruenz versteht man eine durch die Sprache geforderte regelhafte Übereinstimmung mehrerer Wörter oder Satzglieder im Satz bezüglich solcher Merkmale wie Person, Numerus, Kasus, Genus. Diese Übereinstimmung enthält Redundanz, da übereinstimmende Merkmale mehrfach vorkommen.

Die folgenden Beispiele zeigen solche wiederholten Merkmale:

- *das Auto*: *Auto* trägt das sächliche Geschlecht, dies wird durch den Artikel *das* wiederholt.
- *wir singen*: Sowohl das Personalpronomen *wir* wie auch die Verbform *singen* haben die Eigenschaft „erste Person Mehrzahl“. Allerdings könnte *singen* auch noch andere Eigenschaften besitzen, z. B. Infinitiv im Satz *Ich will nicht singen*.
- *Gestern sagte er ...*: Neben der Kongruenz von *er* und *sagte* gibt es eine weitere wiederholte Eigenschaft: Sowohl *gestern* wie auch die Verbform *sagte* beziehen sich auf die Zeitform Vergangenheit.
- *Erstens ..., zweitens ...*: In einer Aufzählung, die mit *erstens* beginnt, erwarten wir zumindest noch *zweitens*, möglicherweise auch *drittens*. Dies muss nicht mehr im gleichen Satz stehen.

#### 1.4.1.5 Feste Wendungen

Bei festen Wendungen (auch als Redewendungen oder Phraseologismen bezeichnet) handelt es sich um eine feste Verbindung mehrerer Wörter zu einer Einheit (vgl. Burger 2015). Beispiele sind:

##### Feste Wendungen

*die Katze im Sack kaufen,*  
*Licht am Ende des Tunnels sehen,*  
*mit an Sicherheit grenzender Wahrscheinlichkeit,*  
*dumm wie Bohnenstroh.* ◀

Nicht so sehr von Interesse ist hier die Eigenschaft, dass sich die Bedeutung einer festen Wendung oft nicht aus der Bedeutung der Einzelwörter erschließen lässt (wie bei *die Katze im Sack kaufen*: hier wird zwar etwas gekauft, aber keine Katze). Interessant ist die strukturelle Stabilität der festen Wendungen: Sie tauchen oft in exakt derselben Form auf (*dumm wie Bohnenstroh*), manchmal werden einige Teile flektiert (*Da hast du die Katze im Sack gekauft*).

Die Redundanz besteht darin, dass ein Teil einer solchen Wendung ausreicht, um den restlichen Teil zu erraten bzw. vorherzusagen, solange man die entsprechende Wendung kennt. Eine zusätzliche Form der Redundanz findet man in vielen Wendungen der Form (*Adjektiv*) wie (*Artikel*) (*Nomen*), z. B. *dumm wie Bohnenstroh*, *alt wie ein Baum* oder *arm wie eine Kirchenmaus*, bei denen die Nominalphrase nur eine Verstärkungsfunktion für das Adjektiv hat. Analog gibt es Wendungen der Form (*Verb*) wie [*(Artikel)*] (*Nomen*) wie *zittern wie Espenlaub*. Ebenso bei [*(Artikel)*] (*Nomen*) wie [*(Artikel)*] (*Nomen*) wie *ein Kerl wie ein Baum*. In diesen Fällen ist das vollständige Verständnis der Nominalphrase gar nicht nötig, man muss das verstärkende Nomen also nicht einmal kennen. Man versteht, dass jemand, der *Geld wie Heu* hat, vermutlich *viel Geld* hat. Und wenn jemand *schnell wie ein XXX* ist, dann vermutlich sehr schnell. Und dabei ist es egal ob *XXX* für *Wiesel*, *Blitz* oder *Windhund* steht.

#### **1.4.1.6 Explizite Wiederholung typischer Zusammenhänge in verschiedenen Sätzen oder Texten**

Typische Eigenschaften oder typische Zusammenhänge zeichnen sich dadurch aus, dass sie nicht nur einmalig auftreten, sondern wiederholt, und zwar in verschiedenen Situationen und unter verschiedenen Umständen. Damit finden wir diese Wiederholungen typischerweise nicht nahe beieinander, sondern verteilt über mehrere Sätze, Absätze oder verschiedene Dokumente. Wir wollen dabei unterscheiden zwischen Wortpaaren und größeren Wortmengen.

**Wiederholte Wortpaare:** Bei solchen Paaren von Wörtern sprechen wir von Kookkurrenzen, siehe Abschn. 5.3. Dabei handelt es sich oft um Wörter verschiedener Wortarten wie Adjektiv und Nomen (z. B. *schwarz* und *Kaffee*), wobei das Adjektiv eine typische Eigenschaft des Nomens beschreibt, welche sich auch im Text wiederfindet (*Ich trinke meinen Kaffee immer schwarz*). Häufig treten solche Wortpaare auch unmittelbar nebeneinander auf (*schwarzer Kaffee*), in solchen Fällen ist diese Nachbarschaft meist durch die Grammatik der Sprache bestimmt.

**Größere Wortmengen:** Auch Wörter aus größeren Wortmengen tauchen immer wieder zusammen auf. In diesem Falle kommen nicht immer alle Wörter vor, aber oft mehrere zusammen. Die Wörter aus solchen Mengen stammen oft aus einem Fachgebiet (z. B. *Medizin*) oder aus bestimmten Alltagssituationen (z. B. *Schulbesuch*). Das Auftreten einiger Wörter aus einer solchen Menge erhöht die Auftretenswahrscheinlichkeit für andere Wörter derselben Menge.

#### **1.4.2 Wirkung von Redundanz**

Das Interesse an Redundanz kommt daher, dass die vorliegenden Wiederholungen statistisch nachweisbar sind. Durch die Verfahren wird sichtbar gemacht, dass Wortpaare bzw. größere Gruppen von Wörtern zusammengehören. Dies wiederum liefert uns Informationen sowohl über die Sprache wie auch über die Dinge, die durch die Wörter

beschrieben werden. Deshalb kann Redundanz genutzt werden, um Zusammenhänge zu rekonstruieren, welche Autoren und Autorinnen bewusst oder unbewusst in ihren Texten wiederholt benutzen.

Die folgenden Arten inhaltlicher Zusammenhänge können mit statistischen Mitteln aus Redundanz erschlossen werden und zur Lösung von Problemen im Text Mining beitragen. Auf einzelne solche Anwendungen wird in späteren Kapiteln eingegangen.

#### **1.4.2.1 Wiederholte Wörter: Wichtige Namen oder Schlagwörter**

Treten Wörter, die sonst eigentlich selten sind, plötzlich immer wieder auf, so ist das für die Leserschaft auffällig und solche Wörter werden als wichtig wahrgenommen. Diese Wahrnehmung lässt sich mit automatischen Mitteln nachbilden, um aus Texten die wichtigen Eigennamen oder Schlagwörter zu extrahieren. Beispielsweise lassen sich auf diese Weise die wichtigen Ereignisse eines Tages, verbunden mit den beteiligten Personen und Orten aus den Nachrichten eines Tages extrahieren. Dabei wird die größere relative Häufigkeit der auffälligen Wörter am entsprechenden Tag verglichen mit ihrer relativen Häufigkeit über einen längeren Vergleichszeitraum (z. B. ein Jahr), vgl. Abschn. 5.9 und 7.4.

#### **1.4.2.2 Wiederholte Substrings im Text: Klassifikation von Texten und Wörtern**

Wortteile, insbesondere Teile von Nomen oder Eigennamen, kommen in bestimmten Sachgebieten (z. B. Wissenschaft und Technik) auffällig wiederholt vor. Beispiele sind Teile von Substanznamen in der Chemie (*-thanol*, *-säure*, *-oxid*) oder Teile von Krankheitsnamen aus der Medizin (*-itis*, *-tumor*, *-entzündung*). Diese können zur Klassifikation dieser Texte genutzt werden. Ebenso können diese Teile von Wörtern benutzt werden, um die entsprechenden Wörter zu klassifizieren, siehe auch Abschn. 2.5 und 7.1.

Auch die Textsorte lässt sich häufig durch verwendete Worte, Wortteile oder Abkürzungen erkennen, die im Normalfall nicht oder nur selten verwendet werden. Beispiele dafür sind:

- Stellenanzeigen mit (m/w) oder (m/w/d),
- Annoncen mit typischen Abkürzungen,
- Schreibweise, z. B. in Internet-Foren: alles Kleinschreibung,
- Verwendung von Emojis: Texte aus sozialen Netzen.

Von besonderem Interesse für das Text Mining sind die Wiederholung von Wortpaaren (Kookkurrenzen) und Wortclustern. Wortpaare mit signifikanter Häufigkeit (*schwarzer – Kaffee*, *Männer – Frauen*) werden im Abschn. 5.3 über Kookkurrenzen ausführlich behandelt. Größere Mengen von Wörtern, die typischerweise gemeinsam in Texten zu einem Thema Verwendung finden, werden im Abschn. 6.4 über Topic-Modelle ausführlich behandelt.

## 1.5 Linguistische Strukturen

### 1.5.1 Texte und linguistische Ebenen

Aus Sicht der Informatik ist ein Text zunächst nur eine Menge von Zeichenketten, also eine Datenmenge vom Typ String. Beim Text Mining müssen jedoch auch die linguistischen Strukturen eines Textes mit ausgewertet werden.

Linguistische Strukturen finden sich auf allen Ebenen eines Textes. In Anlehnung an Noam Chomsky wollen wir diese Ebenen als **linguistische Ebenen** bezeichnen. Damit werden (unabhängig von dem verwendeten linguistischen Theorierahmen) die Elemente einer hierarchischen Untergliederung von Texten bezeichnet, die mit den Zeichen eines **Alphabets** (den Buchstaben) beginnt und mit der Ebene des Satzes endet (vgl. Chomsky 1975). Die wesentlichen Ebenen der Schriftsprache sind:

- **Buchstaben** eines Alphabets (einschließlich Ziffern, Leer- und Sonderzeichen),
- **Morpheme** als sinntragende Konkatenation (Aneinanderreihung) von Buchstaben eines Alphabets,
- **Wortformen** als sinntragende und morphologisch zulässige Konkatenation von Morphemen (vgl. Abschn. 1.3.2),
- **Phrasen** als sinntragende und syntaktisch zulässige Konkatenation von Wortformen,
- **Sätze** als sinntragende und syntaktisch zulässige Konkatenation von Phrasen.

In der theoretischen Informatik wird unter der **Konkatenation** von zwei Zeichenketten  $x$  und  $y$  deren Aneinanderreihung zu einer neuen Zeichenkette  $xy$  verstanden, die beiden Zeichenketten werden also ohne Zwischenraum einfach nebeneinander geschrieben.

#### Beispiel – Linguistische Ebenen bei der Zerlegung eines Satzes

Der Satz „*Das Gewandhaus zu Leipzig befindet sich am Augustusplatz.*“ besteht aus:

1. einer Menge von Buchstaben:  
{D,a,s, ,G,e,w,n,d,h,u,z,L,i,p,g,b,f,t,c,m,A,l,. }
2. einer Menge von Morphemen als einer Konkatenation von Buchstaben:  
{Das,Gewand,haus,zu,Leipzig,be,find,et,sich,am,Augustus,platz}
3. einer Menge von Wortformen als einer Konkatenation von Morphemen:  
{Das,Gewandhaus,zu,Leipzig,befindet,sich,am,Augustusplatz}
4. einer Menge von Phrasen als einer Konkatenation von Wortformen:  
{DasGewandhauszuLeipzig,befindetsich,am Augustusplatz} ◀

Abhängig von den zu verarbeitenden Texten führt die Konkatenation von Phrasen nicht immer zur Ebene der Sätze. So finden sich z. B. in sozialen Medien im Unterschied zu Zeitungstexten nicht immer vollständige Sätze, sondern als vergleichbare Einheit sog. Konversationen. Für die Zwecke des Text Mining sollten die zu verarbeitenden

linguistischen Ebenen daher je nach Aufgabe angepasst werden (vgl. Gründer-Fahrer et al. 2018, S. 221–264).

Für die vollständige Beschreibung der linguistischen Ebenen ist neben der Konkatenation ein weiteres Bildungsprinzip erforderlich, die Zusammenfassung einer Menge von Zeichen oder Zeichenketten zu einer **Äquivalenzklasse**, wie wir es bereits bei der Unterscheidung von **Types** und **Tokens** sowie der Zusammenfassung von **Wortformen** zu einem **Wort** kennengelernt haben (Abschn. 1.3, vgl. BOX Formale Rekonstruktion linguistischer Ebenen, Anhang A4).

Einzelne Ebenen sind auch Gegenstand von Teilgebieten der Linguistik: Gegenstand der **Morphologie** sind Morpheme, Gegenstand der **Lexikologie** sind Wörter, Gegenstand der **Syntax** sind Phrasen und deren Kombination zu Sätzen (vgl. Chomsky 1957).

Die Berücksichtigung der linguistischen Ebenen leistet einen wesentlichen Beitrag zur automatischen Ermittlung von inhaltlichen Zusammenhängen aus Texten. Hierbei wird **linguistisches Wissen** verwendet, welches die für eine Sprache typischen Gesetzmäßigkeiten zusammenfasst. Im nachfolgenden Abschnitt stellen wir zunächst einige Besonderheiten natürlicher Sprachen vor, welche für das Text Mining eine besondere Herausforderung darstellen und deren Verarbeitung linguistisches Wissen erfordert.

### 1.5.2 Warum erfordert die Verarbeitung natürlicher Sprache linguistisches Wissen?

Für die automatische Verarbeitung von Daten und Informationen werden in der Informatik künstliche, formale Sprachen, beispielsweise eine Programmiersprache oder eine Auszeichnungssprache wie XML, verwendet. Dabei wird für eine **formale Sprache** durch **Wohlgeformtheitsregeln** festgelegt, welche Zeichenketten in dieser Sprache zulässig sind. Nicht jede wohlgeformte Zeichenkette hat aber auch eine Bedeutung. Deshalb wird für jede formale Sprache mit Hilfe von **Gültigkeitsregeln** zusätzlich festgelegt, welche Zeichenketten gültig sind und welche Bedeutung eine gültige Zeichenkette in der intendierten Anwendung hat (vgl. Kleene 1971). Dadurch wird sichergestellt, dass klar definiert ist, welches die zulässigen Zeichenketten einer formalen Sprache sind, und dass jede gültige Zeichenkette in dieser Sprache genau eine Bedeutung hat.

Natürliche Sprachen sind aber nicht künstlich geschaffen, sondern entwickeln sich nach eigenen Gesetzmäßigkeiten und eigener Dynamik. Insbesondere die für formale Sprachen charakteristische Eindeutigkeit, dass exakt definiert ist, welche Zeichenketten zulässig sind und welche Bedeutung die gültigen Zeichenketten haben, ist für natürliche Sprachen nicht gegeben. Natürlichsprachliche Texte sind deshalb mit Methoden und Verfahren der Informatik aus folgenden Gründen schwierig zu verarbeiten:

- Wörter in natürlichen Sprachen können mehrdeutig sein und erlangen ihre Bedeutung im Kontext,

- Texte unterliegen eigenen sprachstatistischen Gesetzmäßigkeiten,
- Texte spiegeln die Sprachdynamik einer Sprache wider.

Alle drei Aspekte – **Mehrdeutigkeit**, **Sprachstatistik** und **Sprachdynamik** – begegnen uns auf allen linguistischen Ebenen. Für ihre Beschreibung und Verarbeitung ist linguistisches Wissen erforderlich, das für die automatische Verarbeitung natürlicher Sprache in geeigneter Form zur Verfügung stehen muss. Welche Repräsentation linguistischen Wissens dabei verwendet wird, hängt zum einen von der zugrunde gelegten linguistischen Theorie ab, zum anderen von der zu bearbeitenden Aufgabe. Die wesentlichen Ansätze dazu stellen wir in Kap. 2 vor. Zunächst aber sollen die genannten Phänomene genauer beschrieben werden.

Bei der Mehrdeutigkeit ist zwischen einer lexikalischen und strukturellen Mehrdeutigkeit zu unterscheiden. Ein Wort ist lexikalisch mehrdeutig, wenn es mehr als eine Bedeutung oder Funktion hat. Meist werden Bedeutungsmehrdeutigkeiten in Form eines **Lexikons** beschrieben (vgl. Kap. 4).

#### Beispiel – lexikalische Mehrdeutigkeit

Auf Wortebene haben die folgenden Wörter mehr als eine Bedeutung:

*Schloss* (Nomen) – (a) Gebäude, (b) Schließvorrichtung als Teil einer Tür oder eines Tores

*Aufstrich* (Nomen) – (a) Lebensmittel, (b) Strichart bei Streichinstrumenten

*Abstrich* (Nomen) – (a) körpereigenes Material als Teil einer medizinischen Untersuchung, (b) Strichart bei Streichinstrumenten, (c) Einschränkung

*leicht* (Adjektiv) – (a) Gewichtsangabe, (b) Schwierigkeitsangabe

*übersehen* (Verb) – (a) alles im Blick haben, (b) etwas nicht bemerken

*umfahren* (Verb) – (a) um ein Hindernis herumfahren, (b) mit einem Hindernis dieses zerstörend zusammenstoßen ◀

Oft hat ein Wort mehr als eine Bedeutung, weil es verschiedenen Wortarten zuzuordnen ist, z. B. *Rauchen* als Nomen oder Verb in dem Hinweis „*Rauchen gefährdet die Gesundheit*“. Diese Art der lexikalischen Mehrdeutigkeit findet sich vor allem im Englischen wegen des Fehlens der einfachen Unterscheidung von Nomina und Verben bzw. Adjektiven durch Großschreibung, z. B. *light* (Nomen) – Licht, *light* (Verb) – anzünden, *light* (Adjektiv) – leicht.

Auch Morpheme können mehrdeutig sein, z. B. die Endung „-en“ als Indikator des Indikativs eines Verbs („*les-en*“) oder der 1. Person Plural („*wir les-en*“).

Ein Wort ist strukturell mehrdeutig, wenn es mehr als eine sinnvolle Zerlegung zulässt. Meist werden strukturelle Mehrdeutigkeiten in Form eines Strukturaums dargestellt (vgl. Kap. 2).

Die nachfolgenden Beispiele verdeutlichen leicht verallgemeinerbare, strukturelle Mehrdeutigkeiten auf Wort-, Phrasen- und Satzebene:

### Beispiel – strukturelle Mehrdeutigkeit

Wort:

*Wachstube* (Wach-stube, Wachs-tube)

*Druckerzeugnis* (Druck-erzeugnis, Drucker-zeugnis)

*Urheberrechtsschutz* (Urheber-Rechtsschutz, Urheberrechts-Schutz)

Phrase:

*zum Glück* (als Einwortphrase: glücklicherweise, als Zweiwortphrase: für das Glück)

Satz:

*Peter sieht den Mann mit dem Fernrohr* (Peter hat ein Fernrohr und sieht damit einen Mann, Peter sieht einen Mann, der ein Fernrohr hat)

*Meine sehr verehrten Damen und Herren* (Meine sehr verehrten (Damen und Herren), Meine sehr verehrten Damen (und Herren)) ◀

Für die sprachstatistische Betrachtung setzen wir voraus, dass für einen Text anhand der **Tokenregeln** festgelegt ist, welche Zeichenketten in einem Text als **Tokens**, also als kleinste zu verarbeitende Einheiten betrachtet werden (vgl. Abschn. 1.3.2).

Die Verteilung der Tokens in einem natürlchsprachlichen Text folgt auf allen linguistischen Ebenen für große Textmengen den Gesetzmäßigkeiten der Sprachstatistik. Die Tokens sind nicht statistisch gleichverteilt, sondern folgen einem Potenzgesetz (engl. power law), das unter dem Namen **Zipfsches Gesetz** bekannt ist: In einem natürlchsprachlichen Text kommen einige wenige Tokens sehr häufig, die meisten aber selten vor (vgl. Kap. 5). Der Linguist George Kingsley Zipf (1902–1950), nach dem das Gesetz benannt ist, sieht in dem Zusammenhang ein „Prinzip des geringsten Aufwands“, weil die Tokens, die am häufigsten verwendet werden, meist sehr kurz sind und Tokens umso seltener auftreten, je länger sie sind (vgl. Zipf 1949), was im Beispiel Tab. 1.3 gezeigt wird.

Natürliche Sprachen unterscheiden sich schließlich von formalen Sprachen durch eine evolutionären Prozessen in der Biologie vergleichbare Sprachdynamik auf Morphem-, Phrasen- und Wortebene. Auf Morphemebene führt diese Dynamik oft zu Problemen bei der Anwendung morphologischer Regeln, z. B. bei der Ableitung, ob ein Wort im Plural steht (*E-Mails*) oder in der Vergangenheitsform (*gegoogelt*). Für eine angemessene Behandlung derartiger Sonderfälle – im konkreten Beispiel der Anwendung deutscher Flektionsregeln auf Wörter englischen Ursprungs – wird linguistisches Wissen benötigt.

Nicht nur finden in einer Sprache neue Wörter Verwendung, die es vorher in dieser Sprache nicht gegeben hat, sondern es verschwinden auch Wörter wieder oder nehmen eine andere Bedeutung an, wie im Beispiel Tab. 1.4 dargestellt ist.

Zwischen dem Entstehungszeitpunkt eines Textes und der Auftretenshäufigkeit der in ihm vorkommenden Wörter besteht ein enger Zusammenhang. Ist nur ein Parameter

**Tab. 1.3** Häufigkeitsangaben aus dem Projekt Deutscher Wortschatz deu\_newscrawl-public\_2019\_10K, 10.000 Sätze (Anhang A6)

Token	Häufigkeit	Token	Häufigkeit	Token	Häufigkeit
der	4434	Jahren	184	Flüchtlinge	21
die	4208	können	180	Informationen	21
und	3166	Prozent	179	Möglichkeit	21
in	2610	habe	178	Unterstützung	21
den	1736	immer	175	tatsächlich	21

**Tab. 1.4** Sprachdynamik von Wörtern (vgl. Google Books Ngram Viewer 2020)

Wort	Verwendung	Wort	Verwendung
googeln	Seit ca. 2000	Frauenzimmer (Frau)	Bis Mitte 19. Jh
Laptop	Seit ca. 1985	Kamisol (Kleidungsstück)	Bis Mitte 19. Jh
Entschleunigung	Seit ca. 1985	Wangenstreich (Ohrfeige)	Bis Mitte 19. Jh

bekannt, aber linguistisches Wissen über die Entwicklung einer Sprache vorhanden, kann anhand des Vorkommens bestimmter Wörter der Entstehungszeitpunkt eines Textes bestimmt werden oder es können Abschätzungen darüber gemacht werden, welche Wörter wie oft in einem Text zu erwarten sind (um beispielsweise Abweichungen davon für die Überprüfung von Urheberschaften zu verwenden).

### 1.5.3 Zwei Ansätze für die Repräsentation und Verarbeitung linguistischen Wissens

Für die Beschreibung von linguistischen Gesetzmäßigkeiten können wir zwei grundlegende Ansätze unterscheiden, den **regelbasierten** und den **statistischen** Ansatz. Beide Ansätze schließen sich nicht gegenseitig aus, sondern werden bei konkreten Text-Mining-Anwendungen in der Praxis meist miteinander kombiniert.

Grundidee des regelbasierten Ansatzes ist es, die linguistischen Strukturen in den Zeichenketten eines Textes mit Hilfe von **Mustern** (engl. patterns) zu definieren. Mit Terry Winograd (1983, S. 35), dem Entwickler der sog. Blockswelt und Doktorvater des Google-Mitbegründers Larry Page, unterscheiden wir 5 Arten von Mustern:

1. Literal patterns (konkrete Zeichenfolge)
2. Open patterns (Nutzung von **Wildcards**)

3. Variable patterns (Nutzung von Variablen, gleiche Variablen bezeichnen gleiches Wort)
4. Lexical patterns (Verwendung von Variablen mit Angabe der Wortart)
5. Satzstruktur Patterns/Satzbaumuster

### Beispiele für Muster

Zur Verdeutlichung der genannten Muster wollen wir folgenden Beispielsatz (§3, Satz 1) aus der Verordnung des Sächsischen Staatsministeriums für Soziales und Gesellschaftlichen Zusammenhalt zum Schutz vor dem Coronavirus SARS-CoV-2 und COVID-19 (Sächsische Corona-Schutz-Verordnung – SächsCoronaSchVO) vom 10. Juni 2021, §2 (Grundsätze) und §3 (Sieben-Tage-Inzidenz und Bettenkapazität) (Bauer et al. 1991) betrachten:

*Im Sinne dieser Verordnung ist die Sieben-Tage-Inzidenz die durch das Robert Koch-Institut im Internet unter [www.rki.de/inzidenzen](http://www.rki.de/inzidenzen) veröffentlichte Zahl an Neuinfektionen mit dem Coronavirus SARS-CoV-2 je 100 000 Einwohner innerhalb von sieben Tagen.*

Am Beispiel dieses Satzes können wir die folgenden Muster definieren:

1. Literal patterns (konkrete Zeichenfolge)
  - z. B. Eigennamen wie *Robert Koch-Institut*, *SARS-CoV-2*, oder ganze Wortfolgen wie z. B. *veröffentlichte Zahl an Neuinfektionen*
2. Open patterns (Nutzung von **Wildcards**)
  - z. B. *die durch \_ im Internet unter \_ veröffentlichte Zahl an Neuinfektionen*
3. Variable patterns (Nutzung von Variablen, gleiche Variablen bezeichnen gleiches Wort)
  - z. B. *die X-Tage-Inzidenz ist die veröffentlichte Zahl an Neuinfektionen mit Y je 100 000 Einwohner innerhalb von X Tagen*
4. Lexical patterns (Verwendung von Variablen mit Angabe der Wortart)
  - z. B. *die durch EN veröffentlichte Zahl an Neuinfektionen P NP*
5. Satzstruktur Patterns/Satzbaumuster
  - z. B. *(Frageeinleitung) Enom V <Atemp> <Alok>: Wieviele Corona-Neuinfektionen gab es gestern in Leipzig? ◀*

Für die Definition von Mustern können verschiedene Formalismen verwendet werden, beispielsweise **reguläre Ausdrücke** (vgl. Kap. 3, Abschn. 3.1.1), kontextfreie **Chomsky-Grammatiken** (vgl. Kap. 3, Abschn. 3.2.4) oder kontextfreie Sprachen mit Attributen für die Informationsextraktion wie DIAL (vgl. Feldman und Sanger 2006) oder natürlichsprachliche Dialoge wie AIML (vgl. AIML Foundation 2019). Der regelbasierte Ansatz ist besonders geeignet, wenn von der Textanalyse eine hohe Genauigkeit erwartet wird und bereits umfangreiche, qualitätsgesicherte linguistische Ressourcen für die Repräsentation linguistischen Wissens für eine Sprache vorliegen, beispielsweise in

Form von Lexika und Grammatikregeln. Insofern sich regelbasierte Systeme immer auf konkrete Muster in einer Sprache beziehen, sind sie jedoch meist nur mit hohem Aufwand auf andere Sprachen zu übertragen.

Anders als der regelbasierte Ansatz berücksichtigt der statistische Ansatz die Häufigkeitsverteilungen von Zeichen und Zeichenketten im Text, wodurch sich auffällige Muster in Texten ableiten lassen. So können linguistische Strukturen in Texten datengetrieben und ohne menschliche Interaktion in sog. **unüberwachten Lernverfahren** entdeckt und verarbeitet werden. Besonders geeignet ist der statistische Ansatz für die Erstellung von **Wort-n-Grammen** und Wörtern, die statistisch signifikant gemeinsam auftreten (**Kookkurrenzen**, vgl. Kap. 5). Statistische Verfahren lassen sich leicht von einer Sprache auf eine andere übertragen.

Bei statistischen Verfahren ist zu unterscheiden zwischen sog. frequentistischen und Bayesschen Ansätzen (vgl. Howie 2002). Bei **frequentistischen Verfahren** wird die Wahrscheinlichkeit eines Ereignisses, also z. B. die Wahrscheinlichkeit des Auftretens eines Wortes in einem Text, als Grenzwert der relativen Häufigkeit interpretiert. Demgegenüber wird die Wahrscheinlichkeit eines Ereignisses beim **Bayesschen Ansatz** als Erwartungswert interpretiert, der sich aus einer Bewertung bisheriger Beobachtungen ableitet. Im Unterschied zu frequentistischen Verfahren wird daher bei Bayesschen Verfahren aus den gemachten Beobachtungen ein Modell abgeleitet, welches das Vorwissen und sog. A-Priori-Annahmen explizit ausdrückt. Im Text Mining bilden Bayessche Ansätze die Grundlage für das generative **Maschinelle Lernen** und das semantische Clustern von Termen, sog. **Topic-Modelle** (vgl. Kap. 6).

In der Entwicklung der Automatischen Sprachverarbeitung seit 1960 waren anfangs regelbasierte Verfahren bestimmd, wobei man sich stark an der **Automatentheorie** und Chomskys frühen Arbeiten zur Syntaxanalyse, insbesondere seinem Konzept von Transformationen (vgl. Chomsky 1957) orientiert hat. Musterbasierte Ansätze bildeten die Grundlage für erste natürlichsprachliche Dialogsysteme wie etwa Weizenbaums ELIZA, einem System, das oberflächlich das Gesprächsverhalten eines Psychotherapeuten bzw. einer Psychotherapeutin simuliert (vgl. Weizenbaum 1966, S. 36–45) oder Winograds SHRDLU, einem System zur natürlichsprachlichen Steuerung eines Roboters in einer Spielwelt von Bauklötzchen unter Verwendung der Programmiersprache Planner (vgl. Winograd 1971). In den 1970er Jahren wurde der musterbasierte Ansatz zu einem regelbasierten syntaktischen Parsen unter Verwendung von Parsergeneratoren erweitert. Dadurch war es möglich, aus einer von Linguisten und Linguistinnen geschriebenen kontextfreien Grammatik für eine Anwendung (in einer natürlichen Sprache) (vgl. Abschn. 2.3) automatisch einen Parser für diese Anwendung (in dieser Sprache) zu generieren. Erste größere Umsetzungen dieses Ansatzes waren Woods LUNAR, ein Dialogsystem für die Analyse von Gesteinsproben für Astronauten und Astronautinnen (vgl. Woods et al. 1972; Woods 1973, S. 441–450) und HAM-ANS, einem generischen natürlichsprachlichen System mit Anwendungen in der Analyse von Straßenverkehrsszenen, der Unterstützung von Hotelreservierungen und der Abfrage einer relationalen Datenbank mit Fischereidaten unter Leitung von Walther

von Hahn in Hamburg (vgl. Hoeppner et al. 1986, S. 189–258). Mit dem Aufkommen der Logikprogrammierung und der Programmiersprache PROLOG wurde dieser Ansatz für unifikationsbasierte Verfahren in der Sprachverarbeitung erweitert. Damit werden Grammatikmodelle bezeichnet, bei denen für jede linguistische Einheit eine Merkmalsstruktur im Sinne von morphologisch-syntaktischen Attribut-Wert-Paaren angenommen wird, wie beispielsweise in der Lexical Functional Grammar (vgl. Bresnan et al. 2016) oder Head Driven Phrase Structure Grammar (vgl. Pollard und Sag 1994) (vgl. auch Abschn. 6.1).

Anders als in der Sprachverarbeitung, bei der die Orientierung an menschlicher linguistischer Kompetenz lange Zeit als unhinterfragte Voraussetzung galt, wurden in der **Spracherkennung** (speech recognition) als Teilgebiet der Mustererkennung in der Informatik hauptsächlich statistische Verfahren eingesetzt (vgl. Jelinek 1997). Unter dem Einfluss der großen Erfolge der Spracherkennung, namentlich der Spracherkennungssysteme von IBM und Dragon Mitte der 1990er Jahre, sind statistische Verfahren wie Hidden-Markov-Modelle und Bayessche Netze (siehe Kap. 5 und 6) zunehmend auch in der Sprachverarbeitung eingesetzt worden. Der Paradigmenwechsel von regelbasierten zu statistischen Verfahren lässt sich besonders deutlich im Bereich maschineller Übersetzung nachvollziehen. Große regelbasierte, vollautomatische Übersetzungssysteme wie z. B. Systran (vgl. Hutchins und Somers 1992) sind in den 1990er Jahren durch **Translation Memories**, als Übersetzungsunterstützungssysteme abgelöst worden. Dabei werden aus den von menschlichen Übersetzerinnen und Übersetzern übersetzten Sätzen, Absätzen oder Textabschnitten mit statistischen Verfahren Hypothesen für die Übersetzung von noch zu übersetzenden Texten generiert (vgl. Kugler et al. 1993, 1995). Damit dieser Ansatz erfolgreich angewendet werden kann, ist eine sehr große Menge von Texten und ihren Übersetzungen erforderlich. Die zunehmende Digitalisierung von Arbeitsprozessen und die Verbreitung des Internets schaffen dabei nicht nur die Voraussetzung für eine erfolgreiche Umsetzung dieses Ansatzes, sondern tragen auch entscheidend dazu bei, die Qualität der mit Translation Memories generierten Übersetzungen zu verbessern. Aufbauend auf diesem datengetriebenen Ansatz sind in den letzten Jahren verstärkt auch Verfahren des maschinellen Lernens, insbesondere **neuronale Verfahren**, für die Übersetzungsunterstützung eingesetzt worden, beispielsweise die Systeme Google Translate oder DeepL (2020). In Anbetracht der vielen Programme und Sprachdaten, die mittlerweile für kommerzielle und nicht-kommerzielle Zwecke öffentlich verfügbar sind, finden sich zunehmend auch integrierende Plattformen für die Bündelung von sprachverarbeitenden Programmen wie wir sie von den Marktplätzen und Plattformen im Internet kennen (vgl. Parker et al. 2017). Eine der ersten Plattformen war die Plattform GATE (General Architecture for Text Engineering, vgl. Cunningham et al. 2002, S. 168). Eine gute Übersicht und Bündelung grundlegender Programmmodulen für die Sprachverarbeitung auf Basis der Programmiersprache Python findet sich im Natural Language Toolkit NLTK (vgl. Bird et al. 2009) und in den Bibliotheken von Huggingface (Wolf et al. 2019). Die Austauschbarkeit von Daten und Programmen

**Tab. 1.5** Übersicht historische Entwicklung der Sprachverarbeitung

Dekaden	Technologie	Inspiration	Beispiel-anwendungen	Implementation
1960er-1970er	Mustererkennung/ Pattern matching	Transformations-grammatik	ELIZA, SHRDLU	Planner
1970er-1980er	Syntaktisches Parson und Dialog Management	TG, Speech acts	LUNAR, HAM- ANS	(kaskadierte) ATNs
1980er-1990er	Unifikations- basierte Verfahren	Logik Programming	LILOG	Prolog, LFG, HPSG
1990er-2000er	Statistische Ansätze	Spracherkennung	Verbmobil, Trans- lation Memories	HMMs, Bayessche Netze
Seit 2010	Cognitive computing, konnektionistische Ansätze, NLP-Plattformen	Maschinelles Lernen, Workflow Management	WATSON NLTK, GATE	Dependency parsing, Neural networks, UIMA

für das Text und Data Mining wird aktuell im Zuge des Ausbaus von Forschungsinfrastrukturen vorangetrieben (vgl. Heyer et al. 2015).

Die Tab. 1.5 verdeutlicht zusammenfassend, wie die Entwicklung der Sprachverarbeitung einhergeht mit der Entwicklung neuer Verfahren in der Informatik.

---

## Literatur

- Aggarwal, C.C., Zhai, C.X. (Hrsg.): Mining Text Data. Springer, New York (2012)
- AIML Foundation. <http://www.aiml.foundation/doc.html> (2019). Zugriffen: 10. Dez 2020
- Aston, G., Burnard, L.: The BNC Handbook. Exploring the British National Corpus with SARA. Edinburgh Textbooks in Empirical Linguistics. University Press, Edinburgh (1998)
- Bauer, F.L., Goos, G., Dosch, W.: Informatik 1. Eine einführende Übersicht. Springer-Lehrbuch. Springer, Berlin (1991)
- Bauer, F.L., Goos, G.: Informatik 2. Eine einführende Übersicht. Springer-Lehrbuch. Springer, Berlin (1992)
- Berry, M.W.: Survey of Text Mining. Clustering, Classification, and Retrieval. Springer, New York (2010)
- Bird, S., Klein, E., Loper, E.: Natural Language Processing with Python. Analyzing Text with the Natural Language Toolkit. O'Reilly Media, Sebastopol, Beijing (2009)
- Bodendorf, F.: Daten- und Wissensmanagement, 2. Aufl. Springer-Lehrbuch. Springer, Berlin (2006)
- Bresnan, J., Asudeh, A., Toivonen, I., Wechsler, S.: Lexical-Functional Syntax. Second edition, Blackwell textbooks in linguistics, Bd. 16. Wiley-Blackwell, Chichester (2016)
- Burger, H.: Phraseologie. Eine Einführung am Beispiel des Deutschen, 5. Aufl. ESV basics, Bd. 36. Erich Schmidt, Berlin (2015)

- Böhler, K.: Sprachtheorie. Die Darstellungsfunktion der Sprache. G. Fischer, Jena (1934)
- Chomsky, N.: The Logical Structure of Linguistic Theory. Nachdruck bei Springer „Classic Titles in Linguistics“. Plenum Press, New York (1975), auch verfügbar als Mikrofilm unter [http://alpha-leonis.lids.mit.edu/wordpress/?page\\_id=466](http://alpha-leonis.lids.mit.edu/wordpress/?page_id=466)
- Chomsky, N.: Syntactic Structures. Mouton 1957, Nachdruck bei Mouton – de Gruyter, Berlin (2009)
- Church, K.: Empirical Estimates of Adaptation: The chance of Two Noriega is closer to p/2 than S. 2. In COLING 2000 Volume 1: The 18th International Conference on Computational Linguistics <https://aclanthology.org/C00-1027.pdf> (2000). Zugriffen: 14. Sept. 2021
- Cunningham H., Maynard D., Bontcheva K., Tablan V.: GATE. A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In: Isabelle, P. (Hrsg.) Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics – ACL '02. The 40th Annual Meeting, Philadelphia, Pennsylvania, 7/7/2002 – 7/12/2002, S. 168. Association for Computational Linguistics, Morristown, NJ, USA (2002)
- DCMI: About DCMI. <https://dublincore.org/about/> (2020). Zugriffen: 10. Dez. 2020
- DeepL Übersetzer – DeepL Translate. <https://wwwdeeplcomtranslator> (2020). Zugriffen: 10. Dez. 2020
- Dörre, J., Gerstl, P., Seiffert, R.: Volltextsuche und Text Mining. In: Carstensen, K.-U. (Hrsg.) Computerlinguistik und Sprachtechnologie. Eine Einführung. Spektrum Lehrbuch, S. 425–441. Spektrum Akademischer, Heidelberg (2001)
- Feldman, R., Sanger, J.: The Text Mining Handbook. Advanced Approaches in Analyzing Unstructured Data. Cambridge University Press, Cambridge (2006)
- Francis, W.N., Kučera, H.: Computational Analysis of Present-Day American English. Brown University Press, Providence (1967)
- Gesetz zur Angleichung des Urheberrechts an die aktuellen Erfordernisse der Wissensgesellschaft (Urheberrechts-Wissensgesellschafts-Gesetz – UrhWissG) vom 1. September 2017. BGBI I S. 3346. In: dejure.org Rechtsinformationssysteme GmbH. <https://dejure.org/gesetze/UrhG/> (2017). Zugriffen: 30. Nov. 2020
- Goldhahn, D., Eckart, T., Quasthoff, U.: Building large monolingual dictionaries at the Leipzig corpora collection: From 100 to 200 languages. In: Calzolari, N. et al. (Hrsg.) Proceedings of the 8th International Language Resources and Evaluation (LREC'12), Istanbul, S. 759–765. European Language Resources Association (ELRA) (2012)
- Google Books Ngram Viewer. <https://books.google.com/ngrams/info> (2020). Zugriffen: 10. Dez. 2020
- Graff, D., Cieri, C.: English Gigaword. LDC2003T05. Web Download. Linguistic Data Consortium, Philadelphia (2003). <https://doi.org/10.35111/0z6y-q265>
- Gründer-Fahrer, S., Schlaf, A., Wiedemann, G., Heyer, G.: Topics and topical Phases in German Social Media Communication during a Disaster. Nat. Lang. Eng. **24**(2), 221–264 (2018). <https://doi.org/10.1017/S1351324918000025>
- Henrich, A.: Information Retrieval 1 (Grundlagen, Modelle und Anwendungen). Lehrstuhl für Medieninformatik, Bamberg. [https://www.uni-bamberg.de/fileadmin/uni/fakultaeten/wiai\\_lehrstuhle/medieninformatik/Dateien/Publikationen/2008/henrich-ir1-1.2.pdf](https://www.uni-bamberg.de/fileadmin/uni/fakultaeten/wiai_lehrstuhle/medieninformatik/Dateien/Publikationen/2008/henrich-ir1-1.2.pdf) (2008). Zugriffen: 14. Dez. 2020
- Heyer, G., Eckart, T., Goldhahn, D.: Was sind IT-basierte Forschungsinfrastrukturen für die Geistes- und Sozialwissenschaften und wie können sie genutzt werden? Information – Wissenschaft & Praxis **66**(5–6) (2015). <https://doi.org/10.1515/iwp-2015-0054>
- Hoepner, W., Morik, K., Marburger, H.: Talking it over: The natural language dialog system HAM-ANS. In: Bolc L., Jarke M. (Hrsg.) Cooperative Interfaces to Information Systems. Topics in Information Systems, S. 189–258. Springer, Berlin (1986)

- Howie, D.: Interpreting Probability. Controversies and Developments in the Early Twentieth Century. Cambridge Studies in Probability, Induction, and Decision Theories. Cambridge University Press, Cambridge, MA (2002)
- Hutchins, W.J., Somers, H.: An Introduction to Machine Translation. Academic, London (1992)
- Jelinek, F.: Statistical Methods for Speech Recognition. Language, Speech, and Communication. MIT, Cambridge (1997)
- Jockers, M.L.: Macroanalysis. Digital Methods and Literary History. Topics in the Digital Humanities. University of Illinois Press, Urbana (2013)
- Kantner, C., Overbeck, M.: Exploring Soft Concepts with Hard Corpus-Analytic Methods. In: Reiter, N., Pichler, A., Kuhn, J. (Hrsg.) Reflektierte algorithmische Textanalyse: Interdisziplinäre(s) Arbeiten in der CRETA-Werkstatt, S. 169–190. De Gruyter, Berlin (2020). <https://doi.org/10.1515/9783110693973-008>
- Kao, A., Poteet, S.R.: Natural Language Processing and Text Mining. Springer, London (2007)
- Kleene, S.C.: Representation of events in nerve nets and finite automata. In: Shannon, C.E., McCarthy, J. (eds.) Automata Studies. Annals of mathematics studies, vol. 34, pp. 3–42. Princeton University Press, Princeton (1956)
- Kleene, S.C.: Introduction to Metamathematics. North-Holland Publishing Company, Amsterdam (1971)
- Kuckartz, U.: Qualitative Inhaltsanalyse. Methoden, Praxis, Computerunterstützung. 4th Aufl. Grundlagentexte Methoden. Beltz Juventa, Weinheim (2018)
- Kugler, M., Heyer, G., Kese, R., v. Kleist-Retzow, B., Winkelmann, G.: The *Translator's Workbench*. An environment for Multi-Lingual Text Processing and Translation. In: Nirenburg, S. (Hrsg.) Progress in Machine Translation. IOS Press, Amsterdam (1993)
- Kugler, M., Ahmad, K., Thurairaj, G. (Hrsg.): Translator's Workbench. Tools and Terminology for Translation and Text Processing. Research Reports ESPRIT, Project 2315 TWB, Bd. 1. Springer, Berlin (1995)
- Kupietz, M., Lüngen, H., Kamocki, P., Witt, A.: The german reference Corpus DeReKo: New developments – New opportunities. In: Calzolari, N. et al. (Hrsg.): Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), Miyazako, Japan, S. 4353–4360. European Language Resources Association (ELRA) (2018)
- Lissmann, U.: Inhaltsanalyse von Texten. Ein Lehrbuch zur computerunterstützten und konventionellen Inhaltsanalyse, 3rd edn. Forschung, Statistik & Methoden, Bd. 2. Empirische Pädagogik e. V., Landau (2008)
- Meier, A.: Werkzeuge der digitalen Wirtschaft. Big Data, NoSQL & Co. Springer Vieweg, Wiesbaden (2018)
- Meier, M., Beckh, M.: Text Mining. Wirtschaftsinf **42**(2), 165–167 (2000)
- Moon, R. (Hrsg.): Words, Grammar, Text: Revisiting the Work of John Sinclair. Benjamin Current Topics, Bd. 18. John Benjamins Publishing Company, Amsterdam (2009)
- Moretti, F.: Distant Reading. Verso, London (2013)
- Nokia: Historiamme | Nokia. [https://www.nokia.com/fi\\_fi/tietoa-nokiasta/mita-teemme/historiamme/](https://www.nokia.com/fi_fi/tietoa-nokiasta/mita-teemme/historiamme/) (2020). Zugegriffen: 10. Dez. 2020
- Parker, G., van Alstyne, M., Choudary, S.P.: Die Plattform-Revolution. Von Airbnb, Uber, PayPal und Co. lernen: wie neue Plattform-Geschäftsmodelle die Wirtschaft verändern: Methoden und Strategien für Unternehmen und Start-ups. mitp Business. mitp Verlags GmbH & Co. KG, Frechen (2017)
- Peirce, C.S.: Prolegomena to an apology for pragmatism. In: Peirce, C.S., Bisanz, E. (Hrsg.) The Logic of Interdisciplinarity. The Monist-Series. Deutsche Zeitschrift für Philosophie, Sonderband 20. Akademie, Berlin (2009). <https://doi.org/10.1524/9783050047331.307>

- Pollard, C.J., Sag, I.A.: Head-Driven Phrase Structure Grammar. Studies in Contemporary Linguistics. University of Chicago Press, Chicago (1994)
- Pompino-Marschall, B.: Redundanz. In: Helmut Glück (Hrsg.) *Metzler Lexikon Sprache*, 4. Aufl., S. 552. J.B. Metzler, Stuttgart (2010)
- Sakr, S., Zomaya, A. (Hrsg.): Encyclopedia of Big Data Technologies. Springer, Cham (2019)
- Schäfer, R., Bildhauer, F.: Building large corpora from the web using a new efficient tool chain. In: Calzolari, N. et al. (Hrsg.) Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC 2012), Istanbul, Turkey, S. 486–493. European Language Resources Association (ELRA) (2012)
- Shannon, C.E., Weaver, W.: The Mathematical Theory of Communication. University of Illinois Press, Urbana, IL (1949)
- Shapiro, S.C., Eckroth, D. (Hrsg.): Encyclopedia of Artificial Intelligence, Bd. 1 and 2. Wiley, New York (1987)
- Unicode. <https://www.unicode.org/standard/principles.html>. Zugegriffen: 26. Sept. 2021
- Weizenbaum, J.: ELIZA – A Computer Program for the Study of Natural Language Communication between Man and Machine. Commun. ACM **9**(1), 36–45 (1966). <https://doi.org/10.1145/365153.365168>
- Winograd, T.: Procedures as a representation for data in a computer program for understanding natural language. Dissertation. Massachusetts Institute of Technology, Cambridge <http://hci.stanford.edu/~winograd/shrdlu/AITR-235.pdf> (1971). Zugegriffen: 11. Dez. 2020
- Winograd, T.: Language as a Cognitive Process, Volume I: Snytax. Addison-Wesley, Reading (1983)
- Witten, I.H., Frank, E., Hall, M.A., Pal, C.: Data Mining. Practical Machine Learning Tools and Techniques, 4. Aufl. Elsevier/Morgan Kaufmann, Amsterdam (2017)
- Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., Rush, A.M.: HuggingFace’s transformers: State-of-the-art natural language processing. <https://arxiv.org/abs/1910.03771> (2019). Zugegriffen: 30. Aug. 2021
- Woods, W.A.: Progress in Natural Language Understanding. An Application to Lunar Geology. In: AFIPS Conference Proceedings, S. 441–450 (1973)
- Woods, W. A., Kaplan, R. M., Nash-Webber, B.: The lunar sciences natural language information system: Final report. BBN Report (1972)
- Zikopoulos, P., deRoos, D., Parasuraman, K., Deutsch, T., Giles, J., Corrigan, D.: Harness of the Power of Big Data The IBM Big Data Platform. McGraw-Hill Osborne Media (2012)
- Zipf, G.K.: Human Behavior and the Principle of Least Effort. Addison-Wesley, Cambridge (1949)



# Linguistische Repräsentationen

2

## Zusammenfassung

In diesem Kapitel behandeln wir die linguistischen Strukturen eines Textes, die beim Text Mining mit ausgewertet werden müssen. Grundlage ist die auf den linguistischen Strukturalismus zurückgehende Beobachtung, dass Zeichenketten immer nur in Bezug auf jeweils andere Zeichenketten einen Sinn bzw. eine Funktion haben. Hieraus ergeben sich für das Text Mining zwei Aufgaben: Ausgehend von einem Wort, die Vorhersage seiner Verwendungskontexte und, ausgehend von einem Kontext, die Vorhersage eines Wortes. Die Morphologie beschreibt, welche Morpheme, als kleinste bedeutungstragende Einheiten, in einer Sprache vorkommen und wie diese zu Wörtern kombiniert werden. Syntaktische Strukturen bilden die Grundlage für die Extraktion von Eigennamen, Fachbegriffen und Relationsbezeichnern, die für eine Anwendung charakteristisch sind. Zentrale Konzepte für die Beschreibung syntaktischer Strukturen sind Konstituenten und Dependenz. Gegenstand der Semantik ist die Bedeutung von Morphemen, Wörtern und Sätzen. Für die Repräsentation von Bedeutung werden drei gängige Paradigmen vorgestellt: das prozedurale, das referentielle und das strukturalistische. Fachtexte unterscheiden sich von allgemeinsprachlichen Texten durch das Vokabular sowie eine für den Anwendungsbereich charakteristische Syntax und Morphologie. Für alle linguistischen Ebenen lassen sich Regeln und deren Ausnahmen formulieren, wie an zahlreichen Beispielen verdeutlicht wird.

## 2.1 Theoretische Grundlage: Strukturalismus

### 2.1.1 Was ist die Grundidee des Strukturalismus?

Die Berücksichtigung der für eine Sprache typischen Gesetzmäßigkeiten auf allen ihren linguistischen Ebenen ist für die automatische Ermittlung von inhaltlichen Zusammenhängen aus Texten eine wesentliche Voraussetzung. Als theoretische Grundlage für die Entdeckung und Beschreibung linguistischer Strukturen beziehen wir uns im Folgenden auf den linguistischen **Strukturalismus**. Damit wird eine Richtung der Linguistik bezeichnet, die auf den Schweizer Linguisten Ferdinand de Saussure und seine posthum erschienene Vorlesungsreihe „Grundfragen der allgemeinen Sprachwissenschaft“ (Saussure 1916/1966) zurückgeht. (Zur Einordnung des Cours de linguistique générale und einer vertiefenden Einführung sei verwiesen auf Mehler et al. 2019). Grundgedanke dieses Ansatzes ist die Beobachtung, dass Zeichenketten, wie wir sie bereits bei der Beschreibung der linguistischen Ebenen in Texten kennengelernt haben, – also **Buchstaben, Morpheme, Wörter, Phrasen** und **Sätze** – immer nur in Bezug auf jeweils andere Zeichenketten dieser Ebene einen Sinn bzw. eine Funktion haben (vgl. de Saussure 1916/1966, S. 145).

---

#### Orthographisches Muster

Ein orthographisches Muster, wie z. B. A oder H, hat eine Funktion als Buchstabe nur in einem Alphabet. Wählen wir als Bezugsrahmen das lateinische Alphabet für das Deutsche und das kyrillische Alphabet fürs Russische, dann hat zwar das Muster A jeweils die gleiche Aussprache, nicht aber das Muster P, das im Russischen wie das deutsche R gesprochen wird. ◀

Die Beschreibung von linguistischen Strukturen in Texten benutzt dabei im Sinne des Strukturalismus zwei grundlegende Dimensionen:

1. Gemeinsames Auftreten – Welche Zeichen bzw. Zeichenketten treten gemeinsam auf?
2. Ähnliche Kontexte – Welche Zeichen bzw. Zeichenketten treten in ähnlichen Kontexten auf?

In der Tradition des Strukturalismus wird die erste Dimension auch als **syntagmatische**, die zweite als **paradigmatische** bezeichnet.

Aus der Betrachtung beider Dimensionen können wir für das Text Mining zwei Aufgaben ableiten:

1. ausgehend von einem Wort: Vorhersage eines Kontextes und
2. ausgehend von einem Kontext: Vorhersage eines Wortes.

Im Folgenden sollen diese Dimensionen aufbauend auf dem Begriff des **Kontexts** genauer ausgeführt werden.

### 2.1.2 Kontexte

Für jeden Text einer Sprache  $L$  bezeichne die **linguistische Ebene**  $e \in E$  (mit  $E = \{\text{Buchstaben, Morpheme, Wörter, Phrasen, Sätze}\}$ ) eine endliche Menge von gültigen Zeichen bzw. Zeichenketten  $A$  aus  $e$  (vgl. Abschn. 1.3 und Anhang A3 und Anhang A4). Die Elemente aus  $A$  bezeichnen wir als Beispiele für  $e$  aus  $L$ .

#### Bedeutungstragende Morpheme und Funktionsmorpheme

Die Ebene der Morpheme umfasst bedeutungstragende Morpheme wie *geh* oder *lauf*, aber auch Funktionsmorpheme wie *st* (in „gehst“). Die Beispiele für Morpheme im Deutschen können in Form einer Liste angegeben werden. Die Beispiele für die im Deutschen verwendeten Buchstaben sind die Buchstaben des deutschen Alphabets.

Für die Ebene der Wörter und Sätze finden sich Beispiele in den Korpora `deu_news_2015_10K-words.txt` sowie `deu_news_2015_10K-sentences.txt` (siehe Kap. 1, Anhang A6). ◀

Wesentlich ist die Beobachtung, dass für einen Text – also eine Abfolge von Wörtern (Abschn. 1.1) – die Instanziierung einer linguistischen Ebene immer nur die zulässigen Beispiele der gewählten Ebene umfasst, also beispielsweise Listen von Wörtern. Dabei werden an diesen Listen aber auch typische Gesetzmäßigkeiten der darunter liegenden Ebene deutlich. So lassen sich z. B. aus einer Liste von Wörtern des Deutschen durch regelbasierte oder statistische Verfahren typische Buchstabenkombinationen oder morphologische Muster fürs Deutsche erkennen. Die Idee, für die Entdeckung und Beschreibung linguistischer Strukturen den Kontext der jeweiligen Zeichenketten auszuwerten, können wir also in einem ersten Schritt insofern präzisieren, als dass stets für den Kontext der zulässigen Beispiele die jeweils darüber liegende Ebene zu berücksichtigen ist.

Die Beschreibung linguistischer Strukturen im Sinne des Strukturalismus kann nunmehr anhand des Begriffs des lokalen und globalen Kontexts von Zeichenketten erfolgen. Wir definieren zunächst den Begriff des lokalen Kontexts.

Ein **lokaler Kontext**  $K(a_i)$  eines Beispiels  $a_i \in A$  aus einer linguistischen Ebene  $e$  enthält ein Tupel von Zeichen bzw. Zeichenketten, mit denen die Komponente  $a_i$  auf der darüber liegenden Ebene  $e +$  gemeinsam auftritt, jedoch ohne die Komponente  $a_i$ :

$$K(a_i) = (a_1, a_2, \dots, a_{i-1}, a_{i+1}, \dots, a_n) \quad (2.1)$$

Die Länge des lokalen Kontextes wird durch  $n$  festgelegt und hängt ab von der linguistischen Ebene, die betrachtet wird. So wird auf der Ebene der Wörter die Länge

des lokalen Kontextes oft als Länge des Satzes bestimmt, in dem das Wort vorkommt. Es können aber auch Tupel einer festen Länge, z. B. 5-Tupel, betrachtet werden.

### Beispiel lokaler Kontext

Betrachten wir die folgenden Sätze  $A_1$  bis  $A_4$  mit den Wörtern *die, Sonne, scheint, ein, Kind, lacht, fröhlich* und *Kerze* als Beispiele für die Ebene der Wörter mit der darüber liegenden Ebene der zulässigen Sätze:

$$\begin{aligned} A_1 &= (\text{die}_{11}, \text{Sonne}_{12}, \text{scheint}_{13}) \\ A_2 &= (\text{ein}_{21}, \text{Kind}_{22}, \text{lacht}_{23}, \text{fröhlich}_{24}) \\ A_3 &= (\text{die}_{31}, \text{Kerze}_{32}, \text{scheint}_{33}) \\ A_4 &= (\text{die}_{41}, \text{Sonne}_{42}, \text{lacht}_{43}) \end{aligned}$$

Der lokale Kontext für das Wort *Sonne* im Beispiel ( $A_1$ ) ist dann  
 $K(\text{Sonne}_{12}) = (\text{die}_{11}, \text{scheint}_{13}) \blacktriangleleft$

Der **globale Kontext**  $K_G(a_i)$  eines Beispiels  $a_i \in A$  aus einer linguistischen Ebene  $e$  umfasst alle Zeichen bzw. Zeichenketten, mit denen  $a_i$  auf der darüber liegenden Ebene  $e+1$  in  $L$  gemeinsam auftritt.

### Beispiel globaler Kontext

Für die obigen Beispiele aus den Sätzen  $A_1$  bis  $A_4$  ist der globale Kontext für das Wort *scheint* die Vereinigung der Menge der lokalen Kontexte, welche das Wort *scheint* enthalten:

$$\begin{aligned} K_G(\text{scheint}) &= K(\text{scheint}_{13}) \cup K(\text{scheint}_{33}) \\ &= \{\text{die}_{11}, \text{die}_{31}, \text{Sonne}_{12}, \text{Kerze}_{32}\} \blacktriangleleft \end{aligned}$$

Für das Text Mining von besonderem Interesse sind diejenigen Zeichen bzw. Zeichenketten, die auf einer Ebene statistisch auffällig häufig gemeinsam auftreten. Verfahren für die Berechnung der statistischen Signifikanz des gemeinsamen Auftretens von Zeichen bzw. Zeichenketten stellen wir in Kap. 5 vor.

Wie das obige Beispiel verdeutlicht, lässt das statistisch signifikante gemeinsame Auftreten von Zeichen bzw. Zeichenketten auf einer Ebene erwarten, dass zwischen den Zeichen bzw. Zeichenketten ein funktionaler und inhaltlicher Zusammenhang besteht. Dieser Zusammenhang gilt für alle linguistischen Ebenen (vgl. die nachfolgenden Abschnitte zu Morphologie Abschn. 2.2, Syntax Abschn. 2.3 und Semantik Abschn. 2.4).

### Beispiele für statistisch signifikantes gemeinsames Auftreten von Zeichenketten

Auf der morphologischen Ebene finden wir etwa Gesetzmäßigkeiten der Art, dass bestimmte Pronomina in der Regel bestimmte Endungen des darauffolgenden Verbs verlangen, was in Tab. 2.1 dargestellt wird.

Auf der syntaktischen Ebene finden wir Muster, dass beispielsweise ein Nomen stets durch ein passendes Verb ergänzt werden kann und diese Kombination einen grammatisch vollständigen und korrekten Satz ergibt (*Die Sonne scheint*).

Auf der semantischen Ebene deutet das statistisch signifikante gemeinsame Auftreten von Zeichenketten auf inhaltliche Zusammenhänge hin, z. B. dass *scheinen* eine typische Eigenschaft von *Sonne* ist. ◀

Zeichen bzw. Zeichenketten, die auf einer Ebene statistisch signifikant in ähnlichen Kontexten auftreten, haben meist eine grammatisch und inhaltlich ähnliche Funktion.

### Beispiel für den Vergleich globaler Kontexte

Für die obigen Beispiele mit der darüber liegenden Ebene der zulässigen Sätze ist der globale Kontext für das Wort *Sonne* ähnlich dem globalen Kontext für das Wort *Kerze*:

$$K_G(\text{Sonne}) = \{\text{die}_{11}, \text{die}_{41}, \text{scheint}_{13}, \text{lacht}_{43}\}$$

$$K_G(\text{Kerze}) = \{\text{die}_{31}, \text{scheint}_{33}\} \blacktriangleleft$$

Verfahren für die Berechnung der Ähnlichkeit von Kontexten behandeln wir in Abschn. 5.4. Neben einfachen Verfahren, welche die gemeinsame Anzahl von Zeichenketten auswerten, spielen dabei zunehmend auch vektorbasierte Verfahren eine Rolle. Dabei werden die statistisch signifikant mit einem Beispiel gemeinsam auftretenden Beispiele einer Ebene als Merkmale für die Elemente dieser Ebene aufgefasst und als Vektoren repräsentiert. Die Ähnlichkeit von Elementen einer Ebene kann dann als Abstand in dem entsprechenden Vektorraum berechnet werden.

### Beispiel für Ähnlichkeit von Kontextvektoren

Nehmen wir an, wir haben den Verwendungskontext der folgenden Wörter aus einem großen Zeitungstextkorpus berechnet (also von denjenigen Wörtern, die mit dem Ausgangswort statistisch signifikant gemeinsam auftreten): *Berlin, Paris, Deutschland, Frankreich*. Werden die Verwendungskontexte dieser Wörter als Vektoren

**Tab. 2.1** Morphologische Abhängigkeiten

PERSON	PRONOMEN	VERB + ENDUNG
1	ich	geh+e, lach+e, ...
2	du	geh+st, lach+st, ...
3	er/sie/es	geh+t, lach+t, ...

repräsentiert, so befinden sich *Berlin* und *Paris* sowie *Deutschland* und *Frankreich* im Vektorraum nahe beieinander, was der semantischen Ähnlichkeit zwischen den beiden Hauptstädten und den Ländern entspricht. ◀

Linguisten haben Ende des 19. Jahrhunderts damit begonnen, Verfahren für die Beschreibung von Sprachen zu entwickeln, deren Strukturen sich nicht aus den bekannten Strukturen der europäischen Sprachen ableiten lassen. Methodisch war dafür der linguistische Strukturalismus eine wesentliche Grundlage. Heute, mit der Verfügbarkeit sehr umfangreicher digitaler Textmengen für sehr viele Sprachen, Softwarearchitekturen und Algorithmen für sehr große Textmengen und Computersystemen, die effizient und kostengünstig deren Verarbeitung erlauben, können einige dieser Verfahren auch als Grundlage für die automatische Analyse von Texten verwendet werden. Ein Programm, das automatisch aus einem Text wesentliche Inhaltsstrukturen ableiten soll, versteht von dem Text ja zunächst genau so wenig wie ein Linguist oder eine Linguistin, der oder die eine ihm oder ihr unbekannte Sprache erlernen möchte.

---

## 2.2 Morphologie

### 2.2.1 Grundbegriffe

Die **Morphologie** ist dasjenige Gebiet der Linguistik, das sich mit den Gesetzmäßigkeiten der Wortbildung beschäftigt.

Gegenstand der Morphologie ist das **Morphem** – die kleinste bedeutungstragende Einheit der Sprache – sowie die Art und Weise, wie Morpheme zu **Wortformen** kombiniert werden (vgl. Abschn. 1.2, Anhang A3 und Anhang A4). Eine gute Einführung in die Morphologie findet sich in Katamba (2000).

Morpheme können unter Berücksichtigung ihrer semantischen Funktion in folgende Gruppen aufgeteilt werden:

- lexikalische oder **Basismorpheme** sind solche, die Sachverhalte der außersprachlichen Welt bezeichnen. Zu ihnen gehören Nomina wie *Kind* und *Mantel*, aber auch Verbstämme wie *seh*. Basismorpheme werden auch als **Stamm** eines Wortes bezeichnet.
- grammatische Morpheme oder **Flexive** tragen nur grammatische Bedeutung. Sie werden an Wortstämme angefügt (affigiert) und haben z. B. die Form *t* in *geht* oder *er* in *Kinder*.
- Derivationsaffixe oder **Derivative** werden ebenfalls an Wortstämme affigiert. Man unterscheidet zwischen Primär- und Sekundärstamm, je nachdem, wie viele Affixe dem Stamm schon hinzugefügt wurden. Derivative sind z. B. *bar* in *lösbar* oder *un* in *unlösbar*.

Eine weitere Unterscheidung von Morphemen erfolgt danach, ob sie alleine vorkommen können oder nicht:

- Freie Morpheme wie z. B. *Wort* oder *rot* können ohne Affixe im Text auftreten,
- gebundene Morpheme hingegen nicht. Zu den gebundenen Morphemen zählen alle grammatischen Morpheme und Derivative, aber auch einige Basismorpheme, z. B. Verbstämme wie *seh.*

Morpheme als kleinste bedeutungstragende Elemente einer Sprache können in verschiedenen Formen auftreten. Die verschiedenen Varianten eines Morphems (in der gesprochenen oder geschriebenen Sprache) werden als **Allomorphe** bezeichnet. Beispielsweise taucht im Deutschen der Verbstamm *sprech* in den Varianten *sprich*, *sprach*, *spräch* und *sproch* auf. Allomorphie stellt für das Text Mining ein technisches Problem dar, wenn man eine Wortform durch das Abschneiden einer Endung auf ihren Stamm reduzieren möchte, da die Veränderung des Wortstammes berücksichtigt werden muss.

#### Beispiel zur irregulären Stammformreduktion

Bei der deutschen Wortform *sprichst* kann der Stamm *sprech* nicht allein durch Abschneiden des Endungssuffixs *st* abgeleitet werden. ◀

Bei der Bildung von Wortformen werden die Operationen **Flexion**, **Derivation** und **Komposition** verwendet.

**Flexion** dient der Ableitung grammatischer **Wortformen** aus einem Stamm und schafft syntaktische Oberflächenvarianten ein und desselben Wortstammes. Im Deutschen geschieht dies meist durch Anhängen von Flexionssuffixen, wie z. B. *er* in *Kinder*. Zusätzlich treten aber auch Um- und Ablaute auf (vgl. *Haus* und *Häuser* oder *singen* und *sang*), die zur Entstehung von Allomorphen führen (hier: *Haus* und *Häus* bzw. *sing* und *sang*).

Deutsche Nomina und Adjektive flektieren nach Numerus und Kasus (**Deklination**), deutsche Verben nach Person, Numerus, Tempus, Modus und Genus verbi (**Konjugation**), deutsche Adjektive können zudem noch gesteigert werden (Komparation).

Für die Darstellung von Flexionen werden traditionell sog. Flexionstabellen verwendet. Ein bekanntes Beispiel für Flexionsparadigmen ist das Lateinische mit seiner a-, o- und u-Konjugation. Ist für eine Sprache bekannt, wie die Wortformen aus den Stämmen gewonnen werden, lassen sich die Wörter in sog. Flexionsklassen einteilen. Im Deutschen gehören beispielsweise die Wörter *Schrank* und *Zug* derselben Deklinationsklasse an, da sämtliche Wortformen beider Wörter auf die gleiche Art gewonnen werden (Plural mit Umlaut usw.).

Die Tab. 2.2 und 2.3 geben einen Überblick über deutsche Substantiv-Flexive und ihre Flexionstypen (vgl. Schott 1978).

**Tab. 2.2** Deutsche Substantiv-Flexive und ihre Flexionstypen – Teil 1 (vgl. Schott 1978)

Wort	Dat	Akk	Gen	FTS
Ort	Ø	Ø	(E)S	1
Verhältnis	Ø	Ø	SES	2
Knabe	N	N	N	3
Mensch	EN	EN	EN	4
Zeitung	Ø	Ø	Ø	5
Buchstabe	N	N	NS	6
Interessante	N	Ø	N	7
Herz	EN	Ø	ENS	8
Ferien	Ø	Ø	Ø	0

Erläuterung: FTS = Flexionstyp Singular, FTP = Flexionstyp Plural, NAG = Nominativ/Akkusativ/Genitiv, DAT = Dativ  
Der Nominativ Singular ist stets durch das Null-Morphem (Ø) charakterisiert. Bei den Pluralflexionstypen 1, 6, 7, 9 können Umlaute (vgl. Gärten, Äpfel, Drähte, Wälder) auftreten

**Tab. 2.3** Deutsche Substantiv-Flexive und ihre Flexionstypen – Teil 2 (vgl. Schott 1978)

Wort	NAG	Dat	FTP
Garten	Ø	Ø	1
Auto	S	S	2
Muskel	N	N	3
Zeitung	EN	EN	4
Arbeiterin	NEN	NEN	5
Segel	Ø	N	6
Tag	E	EN	7
Verhältnis	SE	SEN	8
Leib	ER	ERN	9
Fossil	IEN	IEN	10
Eisen	Ø	Ø	0

Erläuterung: FTS = Flexionstyp Singular, FTP = Flexionstyp Plural, NAG = Nominativ/Akkusativ/Genitiv, DAT = Dativ  
Der Nominativ Singular ist stets durch das Null-Morphem (Ø) charakterisiert. Bei den Pluralflexionstypen 1, 6, 7, 9 können Umlaute (vgl. Gärten, Äpfel, Drähte, Wälder) auftreten

**Derivation** schafft neue Wörter durch Anfügung (Affigierung) von Derivativen an Wortstämme. Die meisten Derivative haben dabei eine charakteristische Bedeutung, die sie dem Stamm hinzufügen. Derivationssuffixe ändern zudem meist die Wortart des Stamms.

### Beispiel zur Derivation

Im Deutschen entsteht aus einem Verb durch Suffigieren von *bar* an einen Verbstamm ein Adjektiv:

*lös* (Verbstamm) -> *lösbar* (Adjektiv)  
*genieß* (Verbstamm) -> *genießbar* (Adjektiv) ◀

**Komposition** schafft ebenfalls Wortneubildungen, bei denen diesmal aber zwei Stämme (bzw. zwei freie Morpheme) aneinandergesetzt werden. Das Ergebnis wird als **Kompositum** bezeichnet. Wie bereits erwähnt, treten im Deutschen oft Fugenelemente zwischen beide Teile, die aber keinerlei bedeutungstragende Funktion haben (und somit keine Morpheme sind). Die meisten Komposita im Deutschen lassen sich durch den folgenden **regulären Ausdruck** beschreiben (vgl. Abschn. 3.1.1):

Präfix\* Stamm (Fugenelement? Stamm)\* Suffix\*

Das heißt: Ein Stamm kann mit anderen Stämmen mittels Komposition zu einer neuen Wortform kombiniert werden, wobei optional ein Fugenelement zwischen die beiden Stämme tritt (z. B. ist *es* in *Freundeskreis* ein solches Fugenelement). Ein Stamm bzw. ein Kompositum kann um Affixe erweitert werden (Präfixe bzw. Suffixe), die funktional entweder Derivative oder Flexive sind.

Die Vorgänge der Wortbildung werden zusammenfassend an dem Beispiel *Lösbarkeitsprobleme* erläutert:

Hierbei handelt es sich um den Plural eines Kompositums, welches aus dem Primärstamm *Problem* und dem Sekundärstamm *Lösbarkeit* zusammengesetzt wurde. Zwischen beiden befindet sich das Fugenelement *s*.

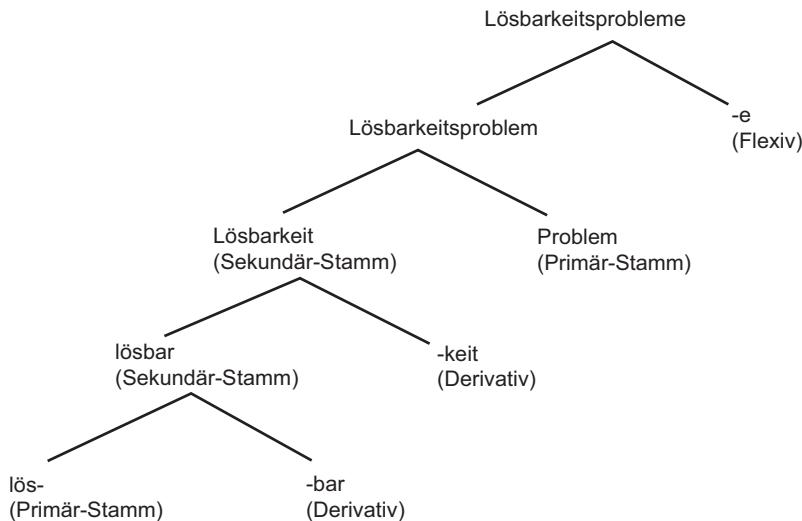
Der Sekundärstamm *Lösbarkeit* wurde durch zweimalige Derivation mithilfe der Derivative *bar* und *keit* aus dem Verbstamm *lös* gewonnen, das letzte (*keit*) bestimmt die Wortart des Ganzen (Nomen).

Der Kopf des Kompositums ist *Problem*, er ist ein Oberbegriff des Ganzen und bestimmt dessen Wortart (Nomen) und Genus (Neutrum).

Die Wortbildungsvorgänge lassen sich mit Hilfe von Strukturbäumen veranschaulichen. Der Baum für das Wort *Lösbarkeitsprobleme* ist in der Abb. 2.1 zu sehen.

## 2.2.2 Verarbeitungsparadigmen für die Morphologie

Morphologische Strukturen müssen im Text Mining immer dann verarbeitet werden, wenn entweder zwischen den verschiedenen Wortformen die morphologische Klassifikation einzelner Wortformen für die weitere Analyse, beispielsweise bei der Relationenextraktion, erforderlich ist.



**Abb. 2.1** Morphologische Zerlegung des Wortes *Lösbarkeitsprobleme*

#### Beispiel für Suche mit flektierten Formen

Werden in einem Text Vorkommen eines bestimmten Wortes, beispielsweise *Haus* gesucht, so sollten auch Vorkommen seiner flektierten Formen *Hause*, *Hause*, *Häuser* und *Häusern* gefunden werden.

Im Deutschen haben die Fälle aber auch eine semantische Funktion. So wird beispielsweise in dem Satz *Der Batteriehersteller gehört dem Automobilkonzern zu 18,5 %.* nur durch die Verwendung des Dativs deutlich gemacht, dass es der Batteriehersteller ist, der dem Automobilkonzern gehört, und nicht umgekehrt. ◀

Eine einfache Methode, bei der Auswertung von Texten keinen Unterschied zwischen den verschiedenen Wortformen eines Wortes zu machen, ist eine Reduktion der Wortformen auf ihre **Grundform** oder ihren **Stamm**. Bei der **Grundformreduktion**, auch Lemmatisierung genannt (engl. lemmatisation), wird eine Wortform auf eine im **Lexikon** ausgezeichnete Form zurückgeführt, für Nomina beispielsweise den Nominativ Singular (*Häuser* -> *Haus*) (vgl. Abschn. 1.2 und 3.2). Durch die **Stammformreduktion** (eng. stemming) werden verschiedene Wortformen auf denselben Stamm zurückgeführt.

Die Aufgabe der Stammformreduktion ist in verschiedenen Sprachen unterschiedlich schwer: Im Englischen, das nur sehr wenige Flexive besitzt (nämlich *s*, *ed* und *ing*), lässt sie sich recht einfach realisieren, im Deutschen hingegen hat man Probleme mit der großen Anzahl verschiedener Flexive und häufig auftretenden Allomorphen.

Eine automatische Stammformreduktion, die ohne ein umfangreiches Lexikon auskommen soll, erfordert in fast jedem Fall ein Abschneiden von Derivationssuffixen.

Zusätzlich muss es ein Regelwerk für Ausnahmen geben, dessen Komplexität von der morphologischen Komplexität der jeweiligen Sprache abhängt. Bei der Erstellung der Regeln ist das Wissen über Flexionsklassen hilfreich. Ein gutes – weil einfaches – Beispiel ist das Englische.

### Beispiel Stammformreduktion im Englischen

Will man im Englischen die durch Pluralbildung entstandenen Formen neutralisieren, so wird einfach bei allen Wörtern, die auf „s“ enden, dieses letzte „s“ abgeschnitten. Es reicht, zwei Ausnahmen zu beachten:

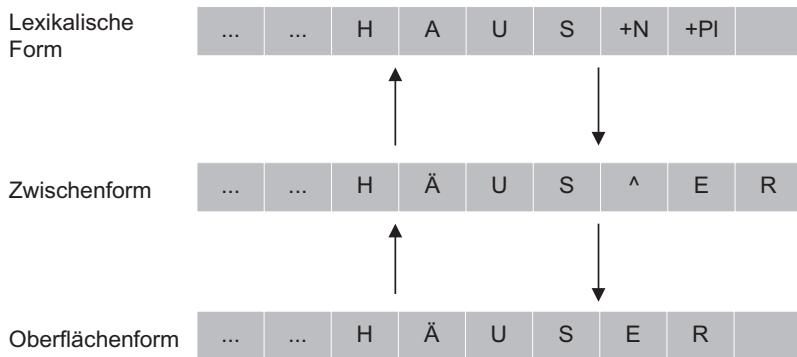
- endet das Wort auf *ss*, so soll das letzte *s* nicht abgeschnitten werden (Beispiel: *caress*)
- endet das Wort auf *ies*, so sollen diese drei Buchstaben im Ganzen abgeschnitten und durch *y* ersetzt werden (Beispiel.: *ponies* -> *pony*) ◀

Ein oft verwendetes Verfahren für die Stammformreduktion im Englischen ist der sogenannte **Porter-Stemmer** (vgl. Porter 1980). Dieser neutralisiert allerdings nicht nur Flexion sondern auch Derivation. Vergleichbare Verfahren für das Deutsche existieren, sind aber notgedrungen komplexer und weniger genau (vgl. Schott 1978), sowie ASV-Toolbox, siehe Anhang A7).

Soll eine im Text vorkommende Wortform – z. B. *Häuser* – nicht nur auf seine Stammform reduziert, sondern auch morphologisch bestimmt werden, dann sind komplexere Verfahren erforderlich, welche die morphologischen Regeln einer Sprache in Bezug auf Flexion, Derivation und Komposition verwenden. Das am häufigsten verwendete Verfahren ist die sog. 2-Ebenen Morphologie (vgl. Koskenniemi 1984; Beesley und Karttunen 2003). Grundlage dieses Verfahrens sind umfangreiche Lexika mit morpho-syntaktischen Angaben zu jedem Wort und morphologische Regeln, die in einer speziellen Regelsprache kodiert sind. Für die morphologische Verarbeitung werden das Lexikon und die Regeln in einen **endlichen Automaten** in Form eines Transduktors übersetzt. Damit kann eine bestimmte Wortform sehr schnell morphologisch bestimmt bzw. generiert werden (vgl. Zielinski und Simon 2008).

In der Abb. 2.2 ist ein Beispiel für solch eine 2-Ebenen-Morphologie zu sehen: Ein Transduktor für die Beschreibung einer morphologischen Regel in der 2-Ebenen-Morphologie enthält ein lexikalisches Band, in dem die Wortform mit ihren morphologischen Kategorien kodiert ist (im Beispiel unten N=Nominativ, Pl=Plural), ein Band für die Oberflächenform des Wortes und ein sog. Zwischenband, in dem festlegt wird, wie die morphologischen Kategorien umgesetzt werden.

Systeme wie GERTWOL (vgl. Haapalainen und Majorin 1995) oder Morphisto (vgl. Zielinski und Simon 2008) sind effiziente Implementierungen der 2-Ebenen-Morphologie fürs Deutsche. Allerdings setzen beide Systeme voraus, dass die Wörter des zu bearbeitenden Textes bereits lexikalisch erfasst sind. Unbekannte Wörter, also Wörter



**Abb. 2.2** Transduktor für die Generierung von *Häuser* aus *Haus*

ohne lexikalischen Eintrag, können damit nicht verarbeitet werden. Neuere Ansätze in der Morphologie verwenden daher Verfahren des maschinellen Lernens, um die morphologischen Regeln einer Sprache zu lernen. Damit lassen sich insbesondere auch die morphologischen Eigenschaften von neuen Wörtern, sog. **Out-of-vocabulary words**, gut vorhersagen (vgl. Janicki 2019).

## 2.3 Syntaktische Repräsentationen

### 2.3.1 Begriffsbestimmung: Was sind syntaktische Strukturen?

In seinem grundlegenden Werk „Syntactic Structures“ aus dem Jahr 1957 definiert Noam Chomsky, nach dem auch die Unterscheidung verschiedener Sprachkomplexitäten benannt ist (Chomsky-Hierarchie), **Syntax** als „[...] the study of principles and processes by which sentences are constructed in particular languages“ (Chomsky 1957, S. 1). Gegenstand syntaktischer Repräsentationen sind also die Wörter bzw. Wortformen einer Sprache und deren Kombination zu Sätzen.

#### Beispiel: Grammatische und ungrammatische Abfolge von Wörtern

Gegeben sei eine Menge von Wörtern, z. B. die Menge {Sonne, die, warm, Leipzig, in, scheint}. Welche Kombinationen daraus sind syntaktisch korrekte Sätze? Offenbar zählt (zumindest fürs Deutsche) die Reihenfolge, denn Folgen wie *die Leipzig scheint warm in Sonne* oder *warm die in Sonne scheint Leipzig* sind keine syntaktisch korrekten Sätze des Deutschen, wohl aber Folgen wie *die Sonne scheint in Leipzig warm* oder *warm scheint die Sonne in Leipzig* (sofern wir außer Acht lassen, dass ein Satz im Deutschen mit Großschreibung beginnt und einem Punkt am Satzende endet). ◀

Anders als bei einer **formalen Sprache**, z. B. einer Programmiersprache, ist allerdings für eine natürliche Sprache nicht immer eindeutig definiert, welche Kombinationen von Wörtern zulässig sind. Außerdem können Kombinationen von Wörtern strukturell mehrdeutig sein wie z. B. der Satz *Johann sieht den Mann mit dem Fernrohr* (vgl. Abschn. 1.3). Ziel einer syntaktischen Repräsentation ist daher die Angabe grammatischer Regeln, nach denen sich die Menge der syntaktisch korrekten Sätze einer Sprache bestimmen lässt. Darüber hinaus verdeutlichen syntaktische Repräsentationen aber auch die Prinzipien, nach denen Kombinationen von Wörtern eine syntaktische Repräsentation zugewiesen wird, insbesondere auch strukturell mehrdeutigen Sätzen.

Im Folgenden stellen wir die **Konstituenten-Struktur** und die **Dependenz-Struktur** als zwei Ansätze syntaktischer Repräsentationen vor sowie das sog. probabilistische **Parson** als eine korpusbasierte Methode für deren Erzeugung. Es handelt sich dabei um eine repräsentative Auswahl von Ansätzen, die beim Text Mining besonders häufig verwendet werden, vgl. auch Kap. 3. Eine gute Übersicht syntaktischer Theorien aus Sicht der Linguistik und in historischer Perspektive findet sich in Rauh (2010). Für die linguistische Theoriebildung sind dabei Fragen nach der Rolle des Lexikons bei der syntaktischen Repräsentation sowie die Abgrenzung der syntaktischen Ebene gegenüber der morphologischen und semantischen Ebene von zentraler Bedeutung. Für unsere Zwecke können wir diese Aspekte jedoch vernachlässigen und uns nur auf die zentralen Konzepte konzentrieren.

### 2.3.2 Konstituenten-Syntax

Im Sinne der sog. amerikanischen Strukturalisten Bloomfield (1984) und Harris (1968) sind Konstituenten auf der syntaktischen Ebene die unmittelbaren Bausteine von Sätzen, also Wörter und Kombinationen von Wörtern, sog. Phrasen. Empirisch lassen sich Konstituenten durch eine Reihe von Tests bestimmen, welche hinreichende Bedingungen für das Vorliegen einer Konstituente definieren, wie z. B. der Verschiebetest (vgl. Grewendorf et al. 2008):

#### Verschiebetest

Gegeben sei der grammatischen Satz *Die Sonne scheint in Leipzig warm*. Alle Wörter bzw. Gruppen von Wörtern, die sich in diesem Satz verschieben lassen, ohne dass ein ungrammatischer Satz entsteht, sind Konstituenten.

Nicht verschieben lassen sich die Präposition *in* oder das Nomen *Leipzig*, wohl aber die Kombination beider Wörter als sog. Präpositionalphrase, wie die folgenden Verschiebungen zeigen:

*Die Sonne in Leipzig scheint warm,*  
*Die Sonne scheint warm in Leipzig,*  
*In Leipzig scheint die Sonne warm.* ◀

In der sog. **Phrasen-Struktur-Grammatik** (PSG), wie sie Chomsky erstmals in dem bereits erwähnten Buch *Syntactic Structures* vorgeschlagen hat, ist eine **Konstituente** ein Wort oder eine Phrase, die von einer syntaktischen Regel als Einheit behandelt wird. Der Rahmen für die syntaktischen Regeln ist dabei eine kontextfreie Ersetzungsgrammatik (also in der Chomsky-Hierarchie eine Typ-2 Grammatik) mit allen Wörtern einer Sprache als terminalen Symbolen, den syntaktischen Kategorien und Phrasen als nicht-terminalen Symbolen und den syntaktischen Regeln als Produktionsregeln. Grundlegende Konstituenten sind dabei die syntaktischen Kategorien Nomen (N), Verb (V), Adjektiv/Adverb (A), Präposition (P) und Artikel (Art), welche als Nichtterminale über das Lexikon direkt den terminalen Wörtern zugewiesen werden, sowie die nichtterminalen Phrasen NP, VP, AP und PP. Syntaktische Strukturen werden in Form eines Syntaxbaums beschrieben, mit dem auch syntaktische Mehrdeutigkeiten dargestellt werden können.

### Syntaktische Mehrdeutigkeiten

Der Satz *Johann sieht den Mann mit dem Fernrohr* ist strukturell mehrdeutig, je nachdem, wer das Fernrohr hat (Johann oder der Mann). Nehmen wir an, den Wörtern in diesem Satz seien die folgenden syntaktischen Kategorien (KAT) zugeordnet (N=Nomen, V=Verb, Art=Artikel, P=Präposition):

KAT(Johann)=N, KAT(seht)=V, KAT(den)=Art, KAT(Mann)=N,  
 KAT(mit)=P, KAT(dem)=Art, KAT(Fernrohr)=N

Die beiden Lesarten lassen sich dann durch die beiden Bäume (mit dem Startsymbol S) in der Abb. 2.3 darstellen. ◀

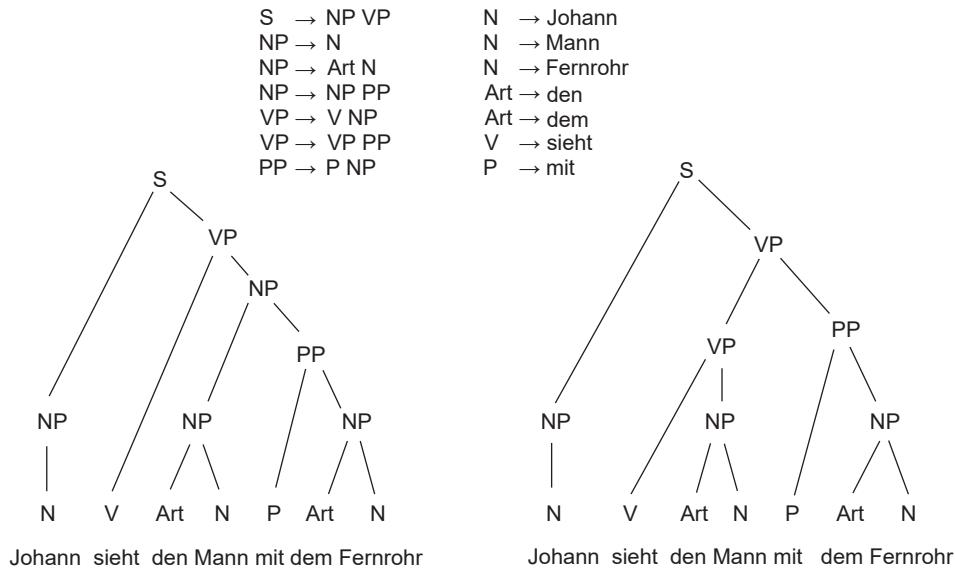
Schon einfache Verschiebungen von Phrasen lassen sich in der Phrasen-Struktur-Grammatik schwer darstellen. Chomsky hat deshalb in dem bereits erwähnten Buch *Syntactic Structures* (Chomsky 1957) sog. Transformationen als Ergänzung der kontextfreien Regeln vorgeschlagen, mit denen u. a. Probleme der freien Wortstellung und Morphologie behandelt werden sollen. **Transformationen** dienen im Rahmen der Transformationsgrammatik dazu, die beobachteten Wortfolgen (sog. Oberflächenstrukturen) durch Veränderungen und Verschiebungen von Phrasen aus sog. Tiefenstrukturen abzuleiten, welche mithilfe der kontextfreien PSG erzeugt worden sind (vgl. Chomsky 1965).

### Transformation

Der Satz der Oberflächenstruktur *Mit dem Fernrohr sieht Johann den Mann* lässt sich aus der Tiefenstruktur *Johann sieht den Mann mit dem Fernrohr* durch folgende Transformation erzeugen:

Strukturbeschreibung: 1 (Johann [N]) 2 (sieht [V]) 3 (den Mann [NP]) 4 (mit dem Fernrohr [PP])

Strukturveränderung: 1 [N] 2 [V] 3 [NP] 4 [PP] ==> 4 [PP] 2 [V] 1 [N] 3 [NP] ◀



**Abb. 2.3** Syntaktische Mehrdeutigkeit

Ein besonderes Problem für die PSG ist die Darstellung sog. thematischer Rollen, d. h. vom Verb ausgehende Anforderungen an die Anzahl und Art der ergänzenden Phrasen im Satz (vgl. Rauh 2010). Grundlage dieses Konzepts ist die Beobachtung, dass Wörter nicht beliebig miteinander kombiniert werden können, sondern dabei auch Restriktionen zu beachten sind, die syntaktische und semantische Aspekte miteinander verbinden.

#### Syntaktische Restriktionen

Im Satz *Johann sieht den Mann mit dem Fernrohr* kann das Wort *sieht* durch das Wort *sucht* ersetzt werden, wobei die strukturelle Mehrdeutigkeit erhalten bleibt. Auch kann für beide Wörter die Ergänzung *mit dem Fernrohr* weggelassen werden (*Johann sieht den Mann/Johann sucht den Mann*), anders als das Wort *sieht* kann aber das Wort *sucht* nicht nur alleine mit dem Nomen stehen (*Johann sucht* ist nicht möglich, *sucht* hat keine intransitive Lesart).

Beide Verbformen können darüber hinaus nur durch Nomina ergänzt werden, die semantisch passen, also *Informatik sieht den Mann mit dem Fernrohr* funktioniert genau so wenig wie *Johann sieht die Informatik mit dem Fernrohr*. ◀

Als syntaktische Strukturen, die mit einer kontextfreien Chomsky-Grammatik dargestellt werden können, eignen sich Konstituentenstrukturen besonders gut für die automatische Syntaxanalyse natürlichsprachlicher Sätze (in der Fachliteratur auch als Konstituenten-Parsing bezeichnet). Hierzu gibt es umfangreiche Darstellungen, auf die in diesem

Rahmen nicht weiter eingegangen werden soll (vgl. Jurafsky und Martin 2009). Standardprogramme für die automatische Syntaxanalyse von Sätzen in Konstituenten finden sich z. B. im Rahmen der Forschungsinfrastruktur CLARIN (<https://weblicht.sfs.uni-tuebingen.de/weblicht/demo/>).

Konstituenten bilden auch meist die Grundlage für **POS-Tags**. Darunter versteht man die Zuordnung von Wörtern und Satzzeichen eines Textes zu Wortarten (englisch: *part of speech*). Eine häufig verwendete Liste von POS-Tags fürs Deutsche ist das sog. Stuttgart-Tübingen Tag Set, STTS (vgl. Schiller et al. 1999).

Für das Text Mining sind Konstituentenstrukturen relevant, weil sie mit dem Phrasen-Konzept eine wichtige Möglichkeit aufzeigen, wie Wörter verschiedener POS-Tags wie Nomen, Verb, Adjektiv und Präposition miteinander kombiniert werden können. Die Bestimmung von Phrasen wird insbesondere für die automatische Extraktion von Relationen benötigt. Die Idee der Transformationen bildet darüber hinaus eine wichtige Grundlage von Dialogsystemen (vgl. Weizenbaum 1966).

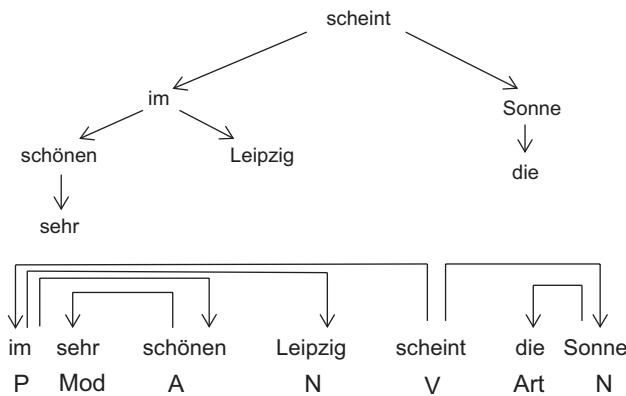
### 2.3.3 Dependenzen

Ein alternativer Ansatz zu Konstituentenstrukturen im Rahmen des regelbasierten Paradigmas sind die auf Lucien Tesnière zurückgehenden **Dependenz-Strukturen** (vgl. Tesnière 1959). Während bei der syntaktischen Repräsentation von Konstituentenstrukturen die Darstellung der Abfolge von Wörtern im Mittelpunkt steht, stellen Dependenzstrukturen die zwischen den Wörtern eines Satzes bestehenden Abhängigkeiten (Dependenzen) dar. Dabei wird angenommen, dass jedes Wort in einem Satz ein, und nur ein übergeordnetes Wort hat, von dem es abhängt. Ein wesentlicher Test für das Vorliegen einer Dependenz ist deshalb der Löschttest: Verliert ein Wort nach dem Löschen eines anderen Wortes in einem Satz seinen Sinn, dann ist das gelöschte Wort im Dependenzbaum dem anderen Wort übergeordnet.

Die Abhängigkeiten zwischen den Wörtern lassen sich ähnlich der Konstituentenstruktur in Form eines Baumes darstellen, wobei jedoch anders als in der Konstituentenstruktur-Syntax jeder Knoten im Baum einem terminalen Wort entspricht. Als Wurzel einer Dependenzbaum-Struktur wird dabei immer das Verb (und nicht eine abstrakte Kategorie Satz) angesetzt. Ähnlich der Konstituentenstruktur-Syntax können dabei auch einzelne Knoten benannt werden, etwa mit den Grundkategorien N, V, A, P, Art und Mod. Die Abb. 2.4 stellt die Dependenz-Struktur des Beispielsatzes *Im sehr schönen Leipzig scheint die Sonne* dar.

#### Dependenzstruktur

Betrachten wir den Satz *Die Sonne scheint im sehr schönen Leipzig*. Wenn wir *schönen* löschen, müssen wir auch *sehr* löschen, weil *sehr* in der Folge von Wörtern *Die Sonne scheint im sehr Leipzig* keinen Sinn ergeben. Das Wort *sehr* ist also abhängig vom übergeordneten Wort *schönen*. ◀



**Abb. 2.4** Dependenzgraph

Dependenzstrukturen erlauben eine sehr flexible und effektive Repräsentation syntaktischer Zusammenhänge. Für das Text Mining sind sie von besonderem Interesse, weil sie neben einer Benennung von Knoten auch eine Benennung von Kanten im Strukturbaum durch inhaltlich-funktionale Kategorien erlauben, wie beispielsweise Subjekt und Objekt eines Satzes. Standardprogramme für das Parsen von Sätzen in Dependenzen finden sich z. B. auf den Seiten der Initiative *Universal Dependencies* (*UD*) (<https://universaldependencies.org/>) im Rahmen der Forschungsinfrastruktur CLARIN (<https://weblicht.sfs.uni-tuebingen.de/weblicht/demo/>) oder in dem von Google vorangetriebenen Projekt *SyntaxNet* (<https://research.googleblog.com/2016/05/announcing-syntaxnet-worlds-most.html>).

### 2.3.4 Probabilistisches Parsen

Im regelbasierten Paradigma wird für die Wörter einer natürlichen Sprache  $L$  eine Grammatik angegeben, welche die – möglicherweise mehrdeutige – syntaktische Struktur einer Folge von Wörtern in  $L$  beschreibt. Damit lässt sich auch bestimmen, ob eine Folge von Wörtern in der Sprache  $L$  zulässig ist, also z. B. eine Phrase oder ein Satz ist (wobei mit der syntaktischen Korrektheit keineswegs sichergestellt ist, dass der zulässige Ausdruck auch semantisch sinnvoll ist). Demgegenüber wird beim statistischen Ansatz die Wahrscheinlichkeit bestimmt, dass eine Folge von Wörtern in der Sprache  $L$  zulässig ist. Grundlage der Berechnung ist das Auftreten von Wortkombinationen in einem Korpus. Damit werden sowohl syntaktische als auch semantische Aspekte berücksichtigt.

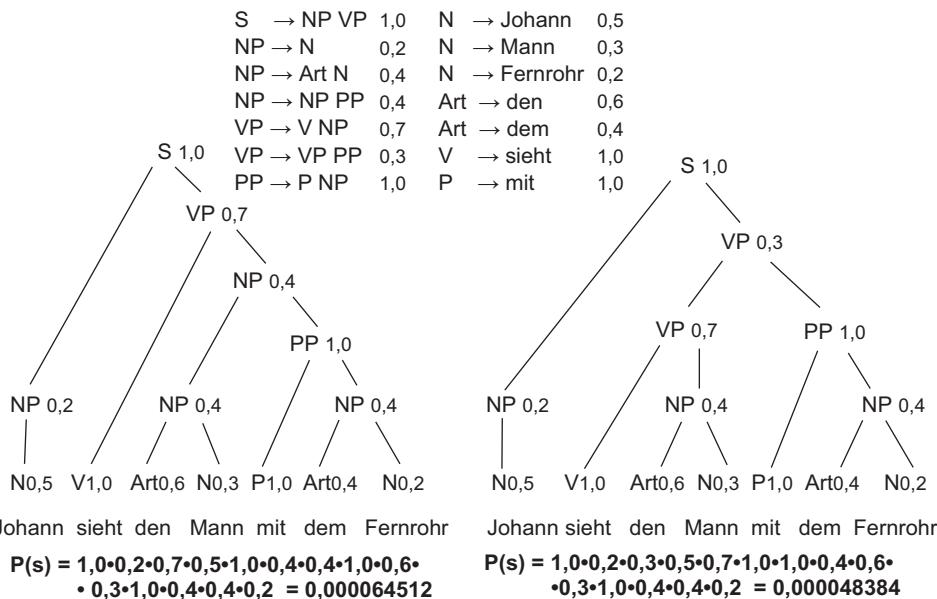
Eine **probabilistische kontextfreie Grammatik** (PCFG) ist eine Grammatik, in der jede Regel mit einer Wahrscheinlichkeit versehen ist, wobei die Summe der Wahrscheinlichkeiten aller Regeln mit demselben Symbol auf der linken Seite 1 betragen muss. Die Wahrscheinlichkeit einer Zerlegung ist dann das Produkt der Wahrscheinlichkeiten der

Regeln, die während des Parsens angewandt werden (vgl. Jelinek et al. 1992; Manning und Schütze 1999).

In der Abb. 2.5 ist eine Beispielableitung des Satzes *Johann sieht den Mann mit dem Fernrohr* dargestellt. Als Grundlage der syntaktischen Analyse verwenden wir wieder eine Phrasenstrukturgrammatik, wobei jedoch jede Regel mit einer Wahrscheinlichkeit für ihre Anwendung versehen ist.

Wie bei der Phrasenstrukturgrammatik erhalten wir mit den zwei abgeleiteten Bäumen eine Repräsentation der syntaktischen Mehrdeutigkeit dieses Satzes. Im Unterschied zur Phrasenstrukturgrammatik unterscheiden sich die beiden Ableitungen aber deutlich in den ihnen zugewiesenen Wahrscheinlichkeiten.

Für die Berechnung der Regelwahrscheinlichkeiten in einer PCFG kann ein syntaktisch analysiertes Korpus wie z. B. das TIGER-Korpus fürs Deutsche (<https://www.ims.uni-stuttgart.de/forschung/ressourcen/korpora/tiger/>) verwendet werden oder eine Baumdatenbank wie NEGRA (<http://www.coli.uni-saarland.de/projects/sfb378/negra-corpus/negra-corpus.html>), aus der für alle Regeln die Regelwahrscheinlichkeiten abgeleitet werden können.



**Abb. 2.5** Probabilistische kontextfreie Grammatik

## 2.4 Semantische Repräsentationen

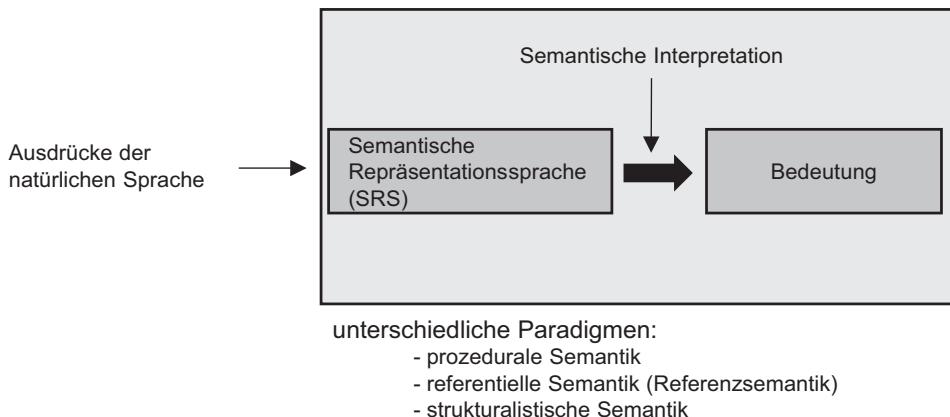
### 2.4.1 Grundbegriffe und Definition

Gegenstand der **Semantik** ist die Bedeutung von Morphemen, Wörtern und Sätzen, also Zeichen und Zeichenketten auf allen linguistischen Ebenen (vgl. Abschn. 1.3, Anhang A3 und Anhang A4). Allerdings ist der Begriff *Bedeutung* unscharf und bedarf einer genaueren Bestimmung durch eine semantische Theorie (vgl. Lyons 1977).

Einen Hinweis darauf, wie Semantik theoretisch gefasst werden kann, finden wir in der Informatik zur Semantik von Programmiersprachen. Für ein konkretes Programm in einer Programmiersprache, beispielsweise der Berechnung der Fakultät einer Zahl, muss sichergestellt sein, dass das Programm tatsächlich die mathematisch richtige Zahl berechnet. Um das zu überprüfen, ist es aber erforderlich, dass wir die Bedeutung des Programms verstehen. Und genau das leistet die Semantik eines Programms: die Abbildung des Programms auf eine Bedeutung, also z. B. die Fakultät einer Zahl, um bei dem Beispiel zu bleiben. Allgemein wird unter der Semantik einer Programmiersprache die Abbildung jedes syntaktisch korrekten Programms einer konkreten Programmiersprache auf dessen Bedeutung verstanden und die Semantik von Programmiersprachen stellt dafür die grundlegenden formalen Techniken bereit (vgl. Kindler 2005). Je nachdem, ob dabei der Prozess der Berechnung („Welche Rechenschritte sind auszuführen, um die Fakultät einer Zahl zu berechnen“), das Ergebnis der Berechnung ( $n!$ ) oder die Definition des Ergebnisses ( $n! = 1 \cdot 2 \cdot 3 \cdot \dots \cdot n$ ) im Vordergrund stehen, unterscheidet man die operationale, die denotationale und die axiomatische Semantik (vgl. Fehr 1989). Bei der **operationalen Semantik** wird die Wirkung von Programmen als schrittweise Zustandsänderung einer abstrakten oder konkreten Maschine beschrieben, indem jeder Rechenschritt in eine Folge von elementaren Maschinenoperationen übersetzt wird. Jeder **Compiler** oder Interpretierer kann dabei als operationale Semantikspezifikation verstanden werden. Die **denotationale Semantik** abstrahiert von diesen schrittweisen Zustandsänderungen einer Maschine und weist jedem Programm die Ergebnisse der zu berechnenden Funktionen als Semantik zu. Dabei wird jedem Programmabschnitt ein geeignetes mathematisches Objekt zugewiesen und das Ergebnis der Ausgabefunktion induktiv aus den Teilergebnissen berechnet. Die **axiomatische Semantik** abstrahiert noch einen Schritt weiter und repräsentiert ein Programm als Transformationen von logischen Aussagen über Zustände.

Wir wollen diese Ansätze als Rahmen für eine Einordnung der gängigen semantischen Paradigmen in der Automatischen Sprachverarbeitung verwenden und definieren die Semantik einer natürlichen Sprache als Darstellung von Bedeutung durch eine Bedeutungsrepräsentationssprache zusammen mit ihrer Interpretation.

Ausdrücke einer natürlichen Sprache werden in eine Bedeutungsrepräsentationssprache (BRS) übersetzt, die nach dem jeweiligen Paradigma interpretiert wird und dadurch in eine formale Repräsentation von Bedeutung überführt wird, wie es die Abb. 2.6 verdeutlicht.



**Abb. 2.6** Allgemeines Schema semantischer Interpretation

Die wesentlichen Paradigmen zur Festlegung der semantischen Repräsentationssprache und ihrer Interpretation sind dabei die prozedurale, referentielle und strukturalistische Semantik. Der operationalen Semantik in der Informatik entspricht die prozedurale Semantik für die natürliche Sprache. Grundidee der **prozeduralen Semantik** ist der Ansatz, dass der Gebrauch sprachlicher Ausdrücke deren Bedeutung konstituiert. Ein historischer Vorläufer ist die Sprachphilosophie Ludwig Wittgensteins: „Die Bedeutung eines Wortes ist sein Gebrauch in der Sprache“ (Philosophische Untersuchungen §43, 2001). Besonders naheliegend ist dieser Ansatz, wenn man Imperative betrachtet. Ein Kind lernt z. B. die Bedeutung des Ausrufs *Stop* im Zusammenhang mit der damit verbundenen Handlung des Stehenbleibens an einer roten Ampel.

#### Beispiel prozedurale Semantik: Blockwelt

In der Abb. 2.7 ist die sog. Blockwelt von Terry Winograd (1972) dargestellt. Ein Roboter wird über die Tastatureingabe in natürlicher Sprache so gesteuert, dass er geometrische Figuren in dieser Blockwelt hin und her bewegt und mit der nutzenden Person einen Dialog darüber führen kann, warum er wann welches Objekt bewegt. ◀

Grundlage dieses Dialogs bildet das Programm SHRDLU als semantische Repräsentationssprache. Es übersetzt elementare Ausdrücke des Englischen, die für diese Anwendung benötigt werden, in elementare Operationen im Zustandsraum der Blockwelt. Die Bedeutung komplexer Ausdrücke sind Funktionen über diese elementaren Operationen. Die Bedeutung eines Satzes ist eine komplexe Operation.

In der **referentiellen Semantik** wird unter der Bedeutung eines sprachlichen Ausdrucks das mit dem Ausdruck bezeichnete Objekt (in einem allgemeinen Sinne, also auch Sachverhalte, Eigenschaften und Ereignisse) verstanden. Ein historischer Vorläufer, der auch die Entwicklung der formalen Logik und damit die Anfänge der Informatik

Person: Hebe einen großen, roten Quader auf.

Computer: OK.

Person: Nimm die Pyramide.

Computer: Ich verstehe nicht, welche Pyramide Du meinst.

Person (ihre Absicht ändernd): Finde einen Quader, der höher ist, als der, den Du gerade hältst, und stelle ihn in die Box.

Computer: Ich vermute, dass Du mit "ihn" den Quader meinst, der höher ist, als der, den ich gerade halte.

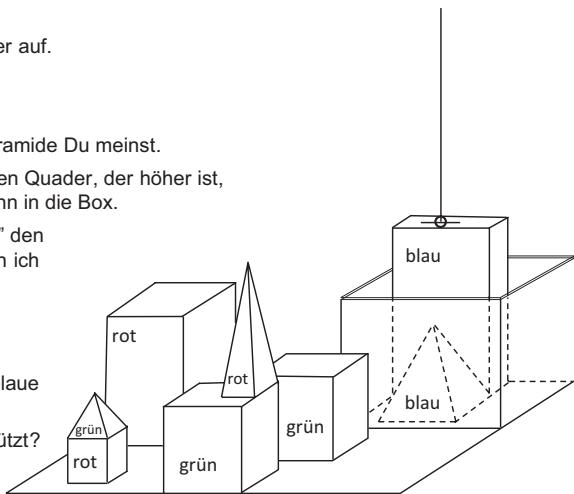
Computer: OK.

Person: Was ist in der Box?

Computer: Die blaue Pyramide und der blaue Quader.

Person: Von was wird die Pyramide gestützt?

Computer: Der Box. ...



**Abb. 2.7** Blockwelt nach Terry Winograd (1972). (Eigene Übersetzung)

wesentlich geprägt hat, ist Gottlob Frege. In seiner Schrift „Über Sinn und Bedeutung“ definiert er: „Wir drücken mit einem Zeichen dessen Sinn aus und bezeichnen mit ihm dessen Bedeutung“ (Frege 1892, S. 31). Der referentielle Ansatz ist besonders naheliegend für Eigennamen, wobei der Träger eines Eigennamens, also das von dem Namen bezeichnete Objekt, als dessen Bedeutung aufgefasst wird.

#### Beispiel referentieller Semantik: Übersetzung in Prädikatenlogik

Für die referentielle Semantik von natürlichsprachlichen Sätzen wie z. B. *Alle Menschen sind sterblich* und *Sokrates ist ein Mensch* wird zunächst eine prädikatenlogische Formel als semantische Repräsentation angegeben, z. B.  $(x) (\text{Mensch}(x) \rightarrow \text{Sterblich}(x))$  und  $\text{Sterblich}(s)$  (mit  $s$  als einer Individuenkonstante, die Sokrates denotiert). Diese prädikatenlogischen Ausdrücke werden dann mithilfe eines prädikatenlogischen Modells interpretiert, bei dem das sog. *universe of discourse* als die Grundmenge der zu referenzierenden Objekte festgelegt sowie die Interpretationsfunktion iterativ definiert werden. Für die Prädikatenlogik gilt dabei, dass jeder Variablen  $x$  ein Element aus und jedem  $k$ -stelligen Prädikat  $P$  ein  $k$ -stelliges Prädikat über dem sog. Universe of Discourse zugeordnet wird. ◀

Im Paradigma der referentiellen Semantik ist die Bedeutung eines Namens bzw. einer Individuenkonstante das von diesen bezeichnete Objekt in einem vorgegebenen Objektraum. Die Bedeutung anderer Ausdrücke sind Funktionen über Namen. Die Bedeutung eines Satzes ist ein Wahrheitswert, der aus der Bedeutung seiner Teile berechnet wird. Das dahinterstehende Prinzip, wonach die Bedeutung eines komplexen Ausdrucks stets

eine Funktion der Bedeutung seiner Teilausdrücke ist, wird als **Kompositionsprinzip** oder auch als Frege-Prinzip bezeichnet.

Tatsächlich ermöglicht dieser Ansatz auf der Grundlage der logischen Modelltheorie eine umfassende Definition extensional definierter Bedeutung, die lange Zeit das beherrschende Paradigma in der formalen Semantik natürlicher Sprache in der Linguistik und Künstlichen Intelligenzforschung war (vgl. Montague 1974; Haugeland 1993).

Das heute vorherrschende Paradigma ist die **strukturalistische Semantik**, die auch unter dem Stichwort **distributionelle Semantik** bekannt ist (vgl. Abschn. 5.4). Nach diesem Ansatz besteht die Bedeutung eines Ausdrucks in dem innersprachlichen Zusammenhang, in dem dieser Ausdruck mit anderen Ausdrücken derselben Sprache steht. In der distributionellen Semantik wird dieser Zusammenhang als der innersprachliche Verwendungskontext eines Ausdrucks gefasst, wie es etwa John Rupert Firth formuliert hat: „You shall know a word by the company it keeps“ (Firth 1957, S. 11). Ein historischer Vorläufer ist jedoch der Begründer des linguistischen Strukturalismus, Ferdinand de Saussure, der bereits in seinen posthum herausgegebenen Vorlesungen zur Allgemeinen Sprachwissenschaft als allgemeines Prinzip sprachlicher Bedeutung formuliert, dass sich die Bedeutung eines einzelnen Sprachelements nur aus dem gleichzeitigen Vorhandensein eines anderen Sprachelements derselben Sprache ergibt (vgl. Saussure 1916/1966, S. 136, vgl. Abschn. 2.1).

#### Beispiel strukturalistischer Semantik: Begriffserklärung

Ein besonders prägnantes Beispiel ist die Erklärung von Wörtern in Definitionen, wie wir sie beispielsweise in Wörterbüchern oder Fachtexten finden – die Bedeutung von Ausdrücken wird durch andere sprachliche Ausdrücke definiert, z. B. *Eine Mutter ist das mit einem Innengewinde versehene Gegenstück einer Schraube oder eines Gewindegelzens*.

Neben Definitionen und Erklärungen finden sich strukturalistische Beziehungen insbesondere aber auch bei der Bestimmung von Bedeutungskomponenten sprachlicher Ausdrücke, wie sie sich beispielsweise durch gezielte Gegenüberstellungen ergeben (semantische Komponentenanalyse, vgl. Katz und Postal (1974)):

*Mutter verhält sich zu Tochter wie Vater zu?*

*Mutter verhält sich zu Kind wie Stute zu?*

Aus diesen Oppositionen lassen sich Bedeutungskomponenten wie z. B.  $\pm$ weiblich,  $\pm$ erwachsen,  $\pm$ Mensch und  $\pm$ Pferd ableiten. Diese wiederum können in eine Merkmalshierarchie eingetragen werden, die es erlaubt, generalisierende Aussagen über die Bedeutung von Ausdrücken und ihren Bedeutungszusammenhängen zu machen (vgl. Helbig 2001) sowie die Repräsentationen FrameNet (vgl. Ruppenhofer et al. 2016) und AMR (vgl. Banarescu et al. 2013). ◀

Anders als in der referentiellen Semantik leitet sich in der strukturalistischen Semantik die Bedeutung eines Ausdrucks nicht aus seiner Verbindung mit dem Objekt ab, das von dem Ausdruck bezeichnet wird, sondern nur durch seine innersprachliche Verwendung. Im strukturalistischen Paradigma wird angenommen, dass es elementare Ausdrücke gibt, deren Bedeutung Merkmale (in einem Merkmalsraum) sind. Die Bedeutung anderer Ausdrücke sind Funktionen über diese elementaren Merkmale. Die Bedeutung eines Satzes ist ein Merkmalskomplex. Für die Analyse von Texten ist dieser Ansatz besonders naheliegend und bildet die Grundlage für kookkurrenzbasierte oder vektorbasierte Verfahren unter Verwendung von Word **Embeddings** (vgl. Abschn. 5.3 und 5.6).

## 2.4.2 Semantische Relationen

Semantische Relationen sind Sinnrelationen zwischen Wörtern, wie sie insbesondere in Bedeutungswörterbüchern oder Thesasuren verwendet werden, also z. B. Synonymie (Bedeutungsgleichheit), Unterbegriffs- oder Oberbegriffsbeziehungen. Semantische Relationen erlauben die Ableitung logischer Folgerungen, wie sie in Form von Beschreibungslogiken formalisiert werden (vgl. Baader 2003). Die Elemente von Thesasuren sind allgemein beschrieben in der ISO Norm ISO 25964 (2021).

### Beispiel Transitive Begriffsbeziehung

Wenn Schlange ein Oberbegriff zu Python ist, dann sind alle Pythons Schlangen. Und wenn alle Schlangen Reptilien sind, dann sind auch alle Pythons Reptilien.

Wenn alle Reptilien Eier legen, dann legen auch Pythons Eier. ◀

Für die Präzisierung der logischen Relationen wird vorausgesetzt, dass die Bedeutung eines Wortes durch die Menge der Objekte, Zustände oder Ereignisse angegeben werden kann, die das Wort bezeichnet. Diese Menge von Objekten, Zuständen oder Ereignissen wird **Extension** genannt. Dabei wird auch vorausgesetzt, dass eine Kenntnis der Extension eines Wortes zum Weltwissen der Sprecherinnen und Sprecher einer Sprache gehört.

### Beispiel Extension

Die Extension eines Eigennamens ist der Träger des Namens, z. B. die Extension des Namens *Bill Gates* ist der Gründer von Microsoft, Bill Gates. Die Extension eines Artbegriffs ist die Menge der zu dieser Art gehörenden Objekte, z. B. die Extension des Begriffs *Mensch* ist die Menge der Menschen, die Extension des Begriffs *Computer* die Menge der Computer, usw. Die Extension einer Eigenschaft (genauer: des Ausdrucks, der eine Eigenschaft bezeichnet) ist die Menge der Objekte, auf die diese Eigenschaft zutrifft. ◀

## Unterbegriff

Ein Wort A ist ein Unterbegriff eines Wortes B, wenn die Extension von A eine echte Teilmenge der Extension von B ist.

## Oberbegriff

Ein Wort B ist ein Oberbegriff eines Wortes A, wenn die Extension von A eine echte Teilmenge der Extension von B ist.

In Tab. 2.4 sind Beispiele zu Unter- und Oberbegriffen dargestellt.

Die Oberbegriffs- und Unterbegriffsbeziehung ist nicht symmetrisch. Wenn ein Wort B zu einem Wort A ein Oberbegriff ist, kann das Wort A nicht gleichzeitig zu diesem Wort B ein Oberbegriff sein. Das Wort A kann aber zu einem weiteren Wort C ein Oberbegriff sein. Beide Relationen sind transitive Relationen.

Der Oberbegriff eines Wortes A ist stets auch ein Oberbegriff aller Unterbegriffe von A. Hat ein Wort mehrere Unterbegriffe, werden diese auch als **Kohyponyme** bezeichnet.

### Beispiel Begriffshierarchie

In der Begriffshierarchie der Abb. 2.8 ist *Säugetiere* Oberbegriff zu *Haustiere* und *Raubtiere*. *Haustiere* wiederum ist Oberbegriff zu *Hund*, *Hauskatze* und *Meerschweinchen*.

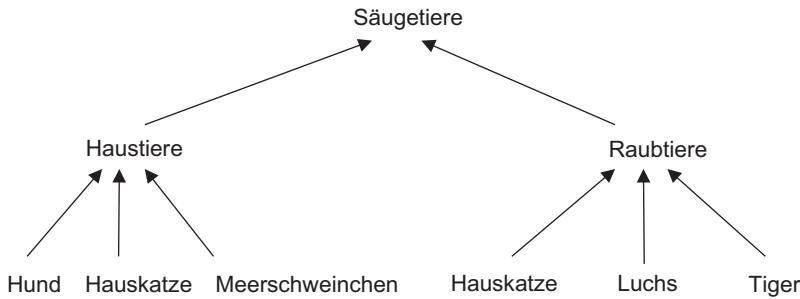
In Bezug auf den Oberbegriff *Säugetiere* sind die Wörter *Haustiere* und *Raubtiere* Kohyponyme, in Bezug auf den Oberbegriff *Haustiere* sind die Wörter *Hund*, *Hauskatze* und *Meerschweinchen* Kohyponyme.

Je nachdem, zu welchem Oberbegriff ein Wort in Beziehung gesetzt wird, kann ein Wort verschiedene Kohyponyme haben. Wenn als Oberbegriff das Wort *Haustiere* gewählt wird, so hat das Wort *Hauskatze* die Kohyponyme *Hund* und *Meerschweinchen*, wenn aber als Oberbegriff *Raubtiere* gewählt wird, hat *Hauskatze* die Kohyponyme *Luchs* und *Tiger*. ◀

Mit der Wahl des Oberbegriffs wird festgelegt, welche Merkmale der bezeichneten Dinge besonders hervorgehoben werden. In bestimmten Gebieten der Wissenschaft und

**Tab. 2.4** Unter- und Oberbegriffe

Unterbegriff	Oberbegriff
Krypta	Grab
Datsche	Haus
Eisbär	Bär
Kreuzotter	Schlange
Haschisch	Rauschgift
Karminrot	Rot
Ultramarinblau	Blau
Bechern	Trinken

**Abb. 2.8** Begriffshierarchie

Technik werden für Wörter im Zuge umfangreicher Normierungen Oberbegriffe definiert (wie z. B. in der Zoologie). Die Bezugnahme auf einen Oberbegriff hilft im Allgemeinen auch, für ein mehrdeutiges Wort eine bestimmte semantische Lesart festzulegen.

### Beispiel Disambiguierung

Das Wort *Verteidiger* ist mehrdeutig. Es kann mit Bezug auf den Oberbegriff *Fußball* ein Fußballspieler gemeint sein, oder mit Bezug auf den Oberbegriff *Recht* ein Anwalt, der vor Gericht die Interessen eines Angeklagten vertritt. ◀

### Synonyme

Zwei Wörter sind synonym, wenn sie dieselbe Extension haben.

Nach dieser Definition sind synonyme Wörter Ausdrucksvarianten, mit denen ein und dieselbe Sache auf verschiedene Weise bezeichnet wird. In der Tab. 2.5 sind Beispiele für Synonyme zu finden. Die Tatsache, dass es für eine Sache unterschiedliche Bezeichnungen gibt, hängt oft mit Dialekten (z. B. Streichholz/Zündholz, Heidelbeere/Blaubeere), unterschiedlicher Fachsprachlichkeit (z. B. Fernseher/TV-Gerät, Auto/Kfz) oder historischen Umständen (z. B. Astronaut und Astronautin für amerikanische, Kosmonaut und Kosmonautin für russische Raumfahrer und Raumfahrerinnen) zusammen.

**Tab. 2.5** Synonyme

Begriff	Synonym
Pilot	Flugzeugführer
Streichholz	Zündholz
TV-Gerät	Fernseher
Abendstern	Venus
Heidelbeere	Blaubeere
Ziegenpeter	Mumps
Auto	Kfz

**Tab. 2.6** Gegensätze

Begriff	Gegensatz	Gemeinsamer Oberbegriff
Hund	Katze	Haustiere
Hund	Computer	Freizeitbeschäftigungen
Vater	Kind	Familie
Vater	Mutter	Eltern

In der Umgangssprache erfüllen nur sehr wenige Wörter das Kriterium, dass sie die selbe Extension haben. Für die Umgangssprache sind daher nur sehr wenige Begriffspaare im strikten Sinne als Synonyme zu bezeichnen. Soweit es sich jedoch um Fachsprachen handelt (vgl. Abschn. 2.5), finden sich häufig synonyme Wörter, um die terminologische Normierung zu unterstützen.

Die Synonymie-Relation ist symmetrisch und transitiv.

### Gegensätze

Zwei Wörter A und B sind Gegensätze in Bezug auf einen Oberbegriff, wenn beide einen gemeinsamen Oberbegriff C haben und die Schnittmenge zwischen der Extension von A und der Extension von B leer ist. (Gegensätze sind Kohyponyme, deren Extensionen sich nicht überschneiden.)

In der Tab. 2.6 sind Beispiele für Gegensätze aufgeführt. Die meisten Wörter überschneiden sich hinsichtlich ihrer Extension nicht mit anderen Wörtern. Von semantischem Interesse ist dies aber meist nur, wenn beide Wörter in einer bestimmten Hinsicht einen gemeinsamen Oberbegriff haben bzw. wenn ein solcher unterstellt wird. So bilden Hund und Katze einen Gegensatz, wenn es um Haustiere geht. Steht dagegen der Gegenstand der Freizeitbeschäftigung im Vordergrund, kann der Gegensatz zum Hund auch ein Computer sein.

Sind zwei Wörter Gegensätze, lässt sich meist das Merkmal angeben, in dem sie sich unterscheiden. Dieses Merkmal variiert mit dem unterstellten Oberbegriff. So ist beim Gegensatz der Wörter *Vater* und *Mutter* das unterscheidende Merkmal *männlich* oder *weiblich*, beim Gegensatz *Vater* und *Kind* aber *erwachsen*.

Komplementärbegriffe und Antonyme sind Spezialfälle begrifflicher Gegensätze.

### Komplementärbegriffe

Zwei Wörter A und B sind Komplementärbegriffe, wenn sie Gegensätze sind und das Komplement von A bezüglich des gemeinsamen Oberbegriffs äquivalent zur Extension von B ist.

In Tab. 2.7 werden beispielhafte Komplementärbegriffe gezeigt.

### Antonyme

Antonyme sind Gegensätze, die nur in Bezug auf einen vorher definierten Begriff definiert sind. Antonyme werden in der Literatur auch als relative Gegensätze bezeichnet (vgl. Lyons 1977, Semantics 1).

**Tab. 2.7** Komplementärbegriffe

Wort A	Extension von A	Komplementärbegriff B	Gemeinsamer Oberbegriff
lebendig	lebende Wesen	tot	Körperzustand
Mann	Männer	Frau	erwachsene Menschen
ledig	unverheiratete Menschen	verheiratet	Ehestand
betrunken	Menschen, die Alkohol getrunken haben	nüchtern	Trunkenheit

**Tab. 2.8** Antonyme

Wort	Antonym	beispielhafter Bezug
groß	klein	Tiere
hell	dunkel	Licht
intelligent	dumm	Schachprogramm
schnell	langsam	Skifahrer

Tab. 2.8 zeigt Beispiele für Antonyme und macht deutlich: Eine große Fliege ist noch kein großes Tier und ein kleiner Elefant ist sicher größer als die größte Fliege. Auch ist ein intelligenter Roboter kein intelligentes Wesen. Im Unterschied zu Gegensätzen im Allgemeinen lassen sich bei Antonymen die unterscheidenden Merkmale nicht absolut angeben, sondern nur relativ in Bezug auf eine vorausgesetzte Extension.

### Konverse

Besteht zwischen begrifflichen Gegensätzen in Bezug auf einen gemeinsamen Oberbegriff eine systematische inhaltliche Relation, lässt sich diese in Form von konversen Begriffspaaren angeben, wie in Tab. 2.9 dargestellt.

Anders als bei allgemeinen Gegensatzbegriffen wie z. B. *Hund – Katze* oder *gewinnen – verlieren* besteht bei Konversen ein systematischer Zusammenhang: Wer etwas kauft, dem muss etwas verkauft werden, was links liegt, kann (aus derselben Perspektive) nicht zugleich rechts liegen, usw.

**Tab. 2.9** Konverse

Begriff	Konversbegriff	Inhaltliche Relation
kaufen	verkaufen	Eigentumswechsel
links	rechts	Lage
Mutter	Tochter	Mutter-von-Tochter

## 2.5 Fachtexte und Terminologie

### 2.5.1 Fachtexte

Dient ein Text dazu, andere Fachleute desselben Faches oder Anwendungsbereiches zu informieren oder die Kommunikation mit Vertretern und Vertreterinnen anderer Disziplinen oder Laien über fachliche Sachverhalte zu ermöglichen, so wird der Text als **Fachtext** bezeichnet (vgl. Beier 1980; Roelcke 1999). Fachtexte zeichnen sich vor allem durch die Verwendung von Wörtern aus, die für das Fach oder den Anwendungsbereich charakteristisch sind und innerhalb des Fachgebietes meist eindeutig definiert, außerhalb des Fachgebiets aber ungebräuchlich sind oder eine andere Bedeutung haben.

#### Beispiel: Die Schraubenmutter

Die Schraubenmutter ist ein hohler, in der Regel niedriger prismatischer Körper, dessen Innenfläche als Innengewinde ausgebildet ist. Die prismatische Außenkontur dient zur Verbindung mit einem Schraubenschlüssel, mit dem das Drehmoment zum Anziehen der Mutter eingeleitet wird. (Wikipedia 2021) ◀

Neben dem Vokabular werden in Fachtexten bevorzugt bestimmte syntaktische Konstruktionen verwendet, die für das Fachgebiet ebenfalls charakteristisch sind, beispielsweise die Verwendung von einfachen Aussagesätzen in Verbindung mit Relativsätzen im obigen Beispiel, tief geschachtelte Sätze mit häufigen Querbezügen in juristischen Texten oder die häufige Verwendung von Orts- und Zeitpräpositionen in Wetterberichten. In Bezug auf das eingeschränkte Vokabular einer Fachsprache und die eingeschränkte Verwendung syntaktischer Konstruktionen stellen Fachtexte in Bezug auf eine Sprache als Ganzes im Sinne von Z. Harris Subsprachen dar, also Untermengen der vom Lexikon und der Grammatik einer Sprache erzeugbaren Strukturen mit lexikalischen, morphologischen, syntaktischen und semantischen Beschränkungen (vgl. Harris 1968).

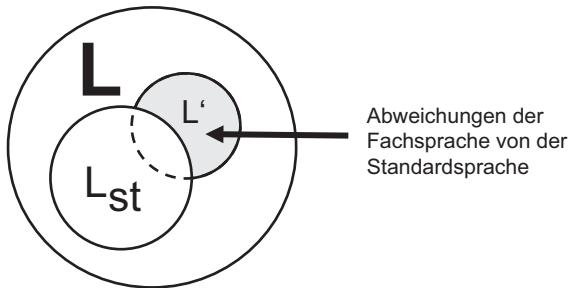
#### Beispiel für domänenspezifische Morpheme

In wissenschaftlich-technischen Fachsprachen finden sich statistisch signifikant häufig Morpheme wie *anti*, *mega*, *mikro*, *multi*, *radial*, *semi* usw. als Präfixe sowie *grad*, *heit*, *tum*, *ial*, *ik*, *tät* oder *ator* als Suffixe.

Darüber hinaus lassen sich domänenspezifische Morpheme aus zusammengesetzten Fachbegriffen identifizieren, wie z. B. *fahr*, *trieb*, *stoff*, *elektr*, *filt* oder *kanal* für den Automobilbau (vgl. Schmalenbach et al. 1998). ◀

Wie die Abb. 2.9 zum Anteil gemeinsamen Vokabulars verdeutlicht, gibt es zwischen der Standardsprache  $L_{st}$  und einzelnen Fachsprachen  $L'$  zwar eine Überlappung im Bereich des Standardvokabulars und der sog. **Stoppwörter**, also hochfrequenten Wörtern wie

**Abb. 2.9** Sprache und Fachsprache



z. B. Artikel, Präpositionen usw., aber nur wenig Überlappungen zwischen den Lexemen der Standardsprache und den Fachausdrücken der Fachsprachen.

Die linguistischen Eigenschaften von Fachsprachen sind Gegenstand umfangreicher Diskussionen (vgl. Bußmann 2002; Roelcke 1999). Als theoretisches Konzept hat der Begriff der Fachsprache die Entwicklung von anwendungsnahen, musterbasierten Verarbeitungsansätzen, wie sie beispielsweise im Linguistic String Project entwickelt worden sind (vgl. Sager et al. 1987), wesentlich beeinflusst.

## 2.5.2 Terminologie

Die Beschreibung einer Technik oder eines Produkts erfordert oft eine besondere Begrifflichkeit, die **Terminologie**. In Anlehnung an Eugen Wüster, den Begründer der modernen Terminologielehre, versteht man unter einer Terminologie „[...] das Begriffs- und Benennungssystem eines Fachgebietes, das alle Fachausdrücke umfasst, die allgemein üblich sind“ (Wüster 1991).

Die Terminologie eines Fachgebietes ist eine Sammlung von **Fachausdrücken**, die meist in Form eines **Thesaurus** beschrieben werden, also unter Nennung des Sachgebietes, der Synonyme, Übersetzungen und Übersetzungsvarianten, Oberbegriffe, Kohyponyme, Antonyme und Aufzählung von Beispielen (vgl. Wüster 1991, vgl. auch Abschn. 2.4). Sie trägt wesentlich zur inhaltlichen Strukturierung eines Fachgebietes bei. Für das Verständnis eines Fachgebietes oder einer Anwendung ist daher die Kenntnis seiner Terminologie eine wichtige Voraussetzung (vgl. Holsapple und Joshi 2002; Maedche und Staab 2001).

In der Praxis bildet sich die Terminologie eines Fachgebietes meist durch Sprachnormierung heraus. Nationale und internationale Standardisierungsorganisationen wie das DIN-Institut (Deutsches Institut für Normung) oder die ISO (International Standards Organisation) unterstützen diese Bemühungen.

Die Beschreibung der Gesetzmäßigkeiten und die Erarbeitung der Terminologie von Fachsprachen sind Gegenstand der **Terminologielehre**. Fachausdrücke werden von Terminologen und Terminologinnen gesammelt, systematisiert und in gedruckten oder elektronischen Wörterbüchern veröffentlicht.

**Tab. 2.10** Sprachliche Strukturen von Fachausdrücken

Struktur	Beispiel Fachbegriff
Substantiv	<i>Restseitenbandübertragung</i> <i>Schwefelfarbstoffe</i>
Substantiv mit Präpositionalphrase	<i>Sammelstelle für Sondermüll</i>
Substantiv mit Genitivphrase	<i>Gesetz der großen Zahlen</i>
Adjektiv + Substantiv	<i>Gefleckter Schierling</i> <i>Hydraulischer Stoßdämpfer</i>
Adjektiv + Adjektiv + Substantiv	<i>Langzeitige linksseitige Herzinsuffizienz</i>
Substantiv + Substantiv	<i>Risiko Controlling</i>

Bei Fachausdrücken handelt es sich meist um Nomina, in geringerem Maße auch um Verben, selten um Adjektive und Adverbien (vgl. Wright et al. 1997).

Fachausdrücke können einzelne Worte sein, wie z. B. *Maschine* (*de*), *machine* (*en*), aber auch Komposita oder Mehrworteinheiten, wie z. B. *Nähmaschine* (*de*), *sewing machine* (*en*) oder *machine à coudre* (*fr*). Die Schreibweise eines Fachausdrucks als Kompositum oder Mehrworteinheit hängt stark von sprachlichen Konventionen ab, die hier nicht weiter vertieft werden sollen. Darüber hinaus finden sich Fachausdrücke in Form von Phrasen, wie z. B. *Tag und Nacht* (*de*), *night and day* (*en*), die für bestimmte Fachsprachen, etwa die juristische Fachsprache, stark standardisiert sein können, wie das folgende Beispiel zeigt: *bevollmächtigen* (*de*) = *nominate, constitute, and appoint as attorney-in-fact* (*en*). Phrasen umfassen im weiteren Sinne auch typische Kombinationen von Nomina und Verben, wie z. B. *ein Patent beantragen* (*de*), *to file for a patent* (*en*). Oft finden sich Fachausdrücke auch in Form von Abkürzungen, z. B. *TCP/IP*.

Die Tab. 2.10 illustriert dazu einige sprachliche Strukturen von Fachausdrücken im Deutschen.

Im Englischen existieren teilweise ähnliche Strukturen; allerdings gibt es dort nur sehr wenige Komposita. Am folgenden Textausschnitt (aus dem Tutorial zu „Design and Behavior of Bolted Joints“ von Nutek Inc. 2014 also über verschraubte Gelenke) sind einige der wichtigsten Strukturen für das Englische zu erkennen:

**Tab. 2.11** Struktur von Fachausdrücken aus dem Beispieltext *Bolted Joints*

Struktur	Beispiel Fachbegriff
Substantiv	<i>bolt, head, nut</i>
Adjektiv + Substantiv	<i>clamping force</i> <i>solid contact, tensile stress</i>
Substantiv + Substantiv	<i>friction torque</i>
Adjektiv + Adjektiv + Substantiv	<i>localized plastic deformation</i>

### Bolted joints

When we tighten the *bolt* (by turning the *head* in above case or by turning the *nut* while the head is held fixed), we stretch the bolt slightly. The stretching does not occur, of course, until the bolt has made *solid contact* with the part. The *clamping force* is produced only when the bolt is stretched. The bolt, therefore, must be stretched to be effective as a *clamp*. The stretch creates *strain*, which then produce *tensile stress* in the bolt. The *tensile load* in the bolt causes *friction torque* between the bolt and the *joint surfaces* and between the bolt and the *joint threads*. The *torsional stress* produced due to friction torque can be as much as 1/3 to ½ the tensile stress in the bolt. When we stop tightening, a good portion of both the tensile and torsional stresses disappear. This is primarily due to relaxation (10 – 20% in first few minutes) and *localized plastic deformation*. (Nutek Inc. 2014) ◀

Die markierten Fachausdrücke lassen sich den obigen Strukturen, wie in Tab. 2.11 dargestellt, zuordnen.

Das automatische Erkennen von Fachausdrücken ist eine wichtige Anwendung des Text Mining. In Abschn. 3.2 stellen wir die hierfür erforderliche Vorverarbeitungs-pipeline vor, Verfahren für die Terminologieextraktion behandeln wir ausführlich in Abschn. 7.1.

---

## 2.6 Die Rolle von Ausnahmen in der Sprache

Wie in den vorstehenden Abschnitten beschrieben, unterliegt die natürliche Sprache vielen Regeln, und diese Regeln beziehen sich auf die verschiedenen linguistischen Ebenen. Egal wie streng die Regeln formuliert werden, ist die Anzahl der Abweichungen beträchtlich. Die umgangssprachliche Metaregel „Keine Regel ohne Ausnahme!“ kann im Bereich der natürlichen Sprache wirklich als gültig betrachtet werden. Bei den häufigsten Ausnahmen kann man zusätzliche Regeln bilden wie die Pluralbildung von Substantiven: Im Normalfall wird bei der Pluralbildung nur eine zusätzliche Endung angehängt (*Tisch*→*Tische*, *Radio*→*Radios*), aber diese Regeln reichen nicht: Es gibt Wörter mit zusätzlicher Umlautung (*Haus*→*Häuser*, *Bach*→*Bäche*), die sich wieder als Regeln formulieren lassen. Solche Regeln wurden bereits in Abschn. 2.2.1 formuliert. Schließlich gibt es weitere ungewöhnliche Pluralbildungen, bei denen Endungen ersetzt werden und von denen einige extrem selten sind: *Matrix*→*Matrizen*, *Globus*→*Globen*, *Bronchitis*→*Bronchitiden*. Diese Fälle werden durch die bisher angegebenen Regeln nicht abgedeckt.

### 2.6.1 Falsche Schreibweisen

Falsche oder unübliche Schreibweisen sollen hier auch mit als Ausnahmen behandelt werden. Solche Abweichungen von der Sprachnorm können durch Sprachvariation oder

Sprachwandel entstehen (vgl. Busse 2006), aber auch durch die Unaufmerksamkeit oder Willkür der Schreibenden. In allen Fällen gibt es wiederkehrende Abweichungen, auf die man bei der Verarbeitung vorbereitet sein sollte. Falsche Schreibweisen können sein:

- aufgelöste Umlaute (*ae* statt *ä* usw., auch *ss* statt *ß*),
- alte Rechtschreibung, speziell bei Getrennt- und Zusammenschreibung,
- Kleinschreibung in sozialen Medien,
- orthographische Fehler (z. B. *könnnen*, *garnicht*),
- Verarbeitungsfehler während der Vorverarbeitung, z. B.
  - Umlaute können verschwunden sein (*knnen*, *fr* statt *können*, *für*),
  - nicht bereinigte Silbentrennung am Zeilenumbruch erzeugen meist zwei nicht korrekte Wörter (*Umlei-* und *tung* statt *Umleitung*).

## 2.6.2 Seltene Ausnahmen

Im Folgenden wird eine Zusammenstellung von Ausnahmen im Deutschen angegeben, die für nachfolgende Verarbeitungsfehler sorgen können. Diese Liste kann fast beliebig erweitert werden und zeigt deutlich, dass in der Verarbeitung natürlicher Sprache eine hundertprozentige Qualität nicht zu erreichen ist. Andererseits sind die Effekte auch nicht allzu häufig.

### Verwendung der Zeichen

- Abweichung von der Großschreibung für Nomen und Eigennamen am Wortanfang (*pH-Wert*),
- Binnengroßschreibung (*pH-Wert*, *LehrerInnen*),
- Sonderzeichen im Wort (*Ver.di*, *E.ON*, *Lehrer\_innen*, *Lehrer:innen*, „*Bild*“-Zeitung),
- Leerzeichen im Wort (*Pro 7-Moderator*).

### Probleme bei der Grundformreduktion

- Es existiert keine Grundform (*Eltern*, *Leute*, *allerschönste*),
- Existenz der Grundform abhängig von anderen Kompositeiteilen (*Waren* → *Ware*, aber *Schreibwaren* hat keine übliche Grundform),
- bei Substantiven mit adjektivischer Flexion gibt es bei Verwendung eines unbestimmten Artikels die männliche und die weibliche Grundform (z. B. eine *Angestellte*, ein *Angestellter*),
- weitgehend unbekannte Grundform (z. B. *erkoren* → *erkiesen*),
- mehrere Bedeutungen, speziell zusätzliche Verwendung als Eigenname,
- mehrere Grundformen, speziell für flektierte Formen (*Sekten* → *Sekt/Sekte*),
- Schwierigkeiten bei der Entfernung von Flexionsmorphemen *ge-* und *zu-* bei Verben: Bekannt sei die Reduktion für *gelaufen* → *laufen*, *fortgelaufen* → *fortlaufen*, *fortzulaufen* → *fortlaufen*. Wie wird auf dieser Basis *zugelaufen* reduziert?

### Keine flektierten Formen oder mehr als üblich

- Kein Plural oder nur in Fachsprache (*Beton, Verkehr*, ...),
- mehrere Formen für Dativ (*am Tisch(e)*),
- verschiedene Pluralformen bei gleicher Bedeutung (*Klima, Konto, Aroma*),
- verschiedene Pluralformen für verschiedene Bedeutungen (*Bank → Banken/Bänke, Mutter → Mütter/Muttern*).

### Unregelmäßige flektierte Formen

- Pluralbildung: seltene Regel bei Doppelvokal -> Umlaut (*Säle → Saal, Köge → Koog*),
- unregelmäßige Pluralform (*Länderinnenminister → Landesinnenminister*),
- unregelmäßige Steigerung im Erstglied bei Adjektiven (*weitestgehend → weitgehend, besserverdienend → gutverdienend*),
- falsche Anwendung von Regeln zur Grundformreduktion möglich, weil entsprechende Wörter existieren: z. B. *Maxima → Maximum* ist korrekt, aber nicht: *Borna* (Ort) → *Bornum* (Ort).

### Derivation und Kompositabildung

- Unregelmäßige Verkleinerung mit -chen (*Bötchen → Boot, Pärchen → Paar, Härtchen → Haar*),
- für einen Algorithmus kann unklar sein, ob es sich um Kompositum oder Derivation handelt (z. B. bei -ei: das *Hühnerei*, die *Bäckerei*),
- mehrdeutige Kompositazerlegung: es gibt mehrere korrekte Formen (*Wachstube, Eistempel, ...*),
- mehrdeutige Kompositazerlegung: es gibt eine zusätzliche unübliche Zerlegung (*Bundesweh-reinheit, Rost-ocker, Urin-sekten*).

### Sonstige Ausnahmen

- Verschiedene grammatische Geschlechter erlaubt (*Blog, E-Mail*),
- abweichendes grammatisches und biologisches Geschlecht (*Mädchen, Kind*),
- nicht wohlgeformte Wörter in Eigennamen (*Beratungs GmbH*), auch alte Rechtschreibung (*Cigarettenfabrik*).

### 2.6.3 Auswirkungen dieser Sonderfälle

Für diese Abweichungen von der Regelmäßigkeit der Sprache gibt es in Abhängigkeit von ihrer Häufigkeit zwei Möglichkeiten: Ganz selten auftretende Sonderfälle sind möglicherweise allein wegen ihrer Seltenheit unproblematisch. Sie führen zwar möglicherweise zu Verarbeitungsfehlern, aber auch andere Algorithmen der Sprachverarbeitung (wie Tokenisierung oder POS-Tagging) arbeiten nicht fehlerfrei. Solange der Anteil an

der Gesamtmenge aller Fehler gering bleibt, kann man einige solche Fehler durchaus tolerieren.

Sonderfälle, die etwas häufiger auftreten, betreffen entweder nur einzelne Wörter oder haben schon systematischen Charakter und können bei Bedarf mit etwas zusätzlichen Aufwand auch korrekt behandelt werden. Im Falle eines regelbasierten Ansatzes bedeutet dies, dass für diese Sonderfälle zusätzliche Regeln aufgestellt werden müssen (und ggf. dazu ein Wörterbuch mit den wichtigsten Wörtern für diese Regeln). Mit Methoden des maschinellen Lernens kann die Bearbeitung vieler solcher Sonderfälle auch gelernt werden. Dafür helfen große Textmengen, damit die Lernverfahren auch genügend Daten zum Erkennen der Sonderfälle vorgelegt bekommen.

Damit besteht die Möglichkeit, die Anzahl der durch Sonderfälle bedingten Fehler nach und nach zu verringern und die Präzision der Algorithmen oberhalb eines möglicherweise vorgegebenen Schwellwertes zu halten.

Trotzdem sind die durch solche Sonderfälle verursachten Fehler oft unverständlich für die Anwendenden, da diese Fehler für Menschen offensichtlich sind und vermeidbar erscheinen.

Deshalb ist die Unterscheidung interessant, inwieweit die oben genannten Fehler tatsächlich sichtbar und damit störend sind. Sind sie nicht sichtbar, dann werden sie von Nutzenden nicht bemerkt und sind viel eher tolerierbar.

---

### Beispiel für tolerierbare Fehler

Bei Suchmaschinen werden oft POS-Tagging und Grundformreduktion durchgeführt, um z. B. Eigennamen als solche zu erkennen und nach flektierten Formen suchen zu können. Dies ist für den Nutzer oder die Nutzerin nicht sichtbar, dementsprechend werden Fehler kaum bemerkt.

Beispielsweise wird bei einigen Suchmaschinen zu einem eingegebenen Suchwort für manche Wörter ungefragt auch nach anderen Schreibweisen und flektierten Formen gesucht, für andere Wörter nicht. Das könnte daran liegen, dass im zweiten Fall das entsprechende Wissen nicht vorliegt oder Mehrdeutigkeiten vorliegen könnten. ◀

Schwieriger ist die Situation, wenn derartige Fehler bei der Generierung von Text, beispielsweise durch Chatbots, passieren. Hier scheint es dann, als sei der Computer der deutschen Sprache nicht ausreichend mächtig. Diese Anwendungen zählen aber nicht mehr zum Text Mining und sollen deshalb hier nicht weiter betrachtet werden.

---

## Literatur

- Baader, F.: The Description Logic Handbook. Theory, Implementation, and Applications. Cambridge University Press, Cambridge (2003)
- Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., Schneider, N.: Abstract Meaning Representation for Sembanking. In:

- Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse, 178–186 (2013). <https://amr.isi.edu/>
- Beesley, K.R., Karttunen, L.: Finite State Morphology. Studies in Computational Linguistics, Bd. 3. CSLI Publications, Stanford (2003)
- Beier, R.: Englische Fachsprache. Kohlhammer, Stuttgart (1980)
- Bloomfield, L.: Language. University of Chicago Press, Chicago (1984)
- Busse, D.: Sprachnorm, Sprachvariation, Sprachwandel – Überlegungen zu einigen Problemen der sprachwissenschaftlichen Beschreibung des Deutschen im Verhältnis zu seinen Erscheinungsformen. Deutsche Sprache 34(4), 314–333 (2006)
- Bußmann, H.: Lexikon der Sprachwissenschaft, 3. Aufl. Kröner, Stuttgart (2002)
- Chomsky, N.: Syntactic Structures. Mouton 1957, Nachdruck bei Mouton. de Gruyter, Berlin (2009)
- Chomsky, N.: Aspects of the Theory of Syntax. 2nd Aufl. Massachusetts Institute of Technology (Cambridge, MA). Research Laboratory of Electronics. Special technical report, Bd. 11. MIT Press, Cambridge (1965)
- Fehr, E.: Semantik von Programmiersprachen. Studienreihe Informatik. Springer, Berlin (1989)
- Firth, J.R.: Papers in Linguistics. 1934–1951. Oxford University Press, London (1957)
- Frege, G.: Über Sinn und Bedeutung. In: Zeitschrift für Philosophie und philosophische Kritik, N.F., (Bd. 100/1), 25–50 (1892). In: Deutsches Textarchiv [http://www.deutschestextarchiv.de/frege\\_sinn\\_1892](http://www.deutschestextarchiv.de/frege_sinn_1892). Zugegriffen: 6. Jan. 2021, auch in: Frege, G.: Funktion, Begriff, Bedeutung. Fünf logische Studien. Herausgegeben und eingeleitet von Günther Patzig, S. 38–63. Vandenhoeck & Ruprecht, Göttingen (1962)
- Grewendorf, G., Hamm, F., Sternefeld, W.: Sprachliches Wissen. Eine Einführung in moderne Theorien der grammatischen Beschreibung, 3. Aufl. Suhrkamp-Taschenbuch Wissenschaft, Bd. 695. Suhrkamp, Frankfurt a. M. (2008)
- Haapalainen, M., Majorin, A.: GERTWOL und Morphologische Disambiguierung für das Deutsche. NODALIDA-95. <http://www2.lingsoft.fi/doc/gercg/NODALIDA-poster.html> (1995). Zugegriffen: 1. Dez. 2020
- Harris, Z.S.: Mathematical Structures of Language. Wiley, New York (1968)
- Haugeland, J.: Artificial Intelligence. The Very Idea, 6. Aufl. Bradford books. MIT Press, Cambridge (1993)
- Helbig, H.: Die semantische Struktur natürlicher Sprache. Wissensrepräsentation mit MultiNet. Springer, Berlin (2001)
- Holsapple, C.W., Joshi, K.D.: A Collaborative Approach to Ontology Design. CACM 45(2), 42–47 (2002). <https://doi.org/10.1145/503124.503147>
- ISO 25964 – The International Standard for Thesauri and Interoperability with Other Vocabularies | NISO Website. <https://www.niso.org/schemas/iso25964> (2021). Zugegriffen: 3. Febr. 2021
- Janicki, M.: Statistical and Computational Models for Whole Word Morphology. Universität Leipzig (2019)
- Jelinek, F., Lafferty, J.D., Mercer, R.L.: Basic methods of probabilistic context free grammars. In: Laface, P., Di Mori, R. (Hrsg.) Speech Recognition and Understanding. Recent Advances, Trends, and Applications, volume F75 of NATO Advanced Study Institute, S. 345–360. Springer, Berlin (1992)
- Jurafsky, D., Martin, J.H.: Speech and Language Processing. An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, 2. Aufl. Pearson/Prentice Hall (2009)
- Katamba, F.: Morphology. 5. Aufl. Modern linguistics series. Palgrave Macmillan, Basingstoke (2000)

- Katz, J.J., Postal, P.M.: An Integrated Theory of Linguistic Descriptions, 6. Aufl. M.I.T. Research Monograph, Bd. 26. Massachusetts Institute of Technology, Cambridge (1974)
- Kindler, E.: Semantik. Skript zur Vorlesung – Semantik von Programmiersprachen. <https://www.yumpu.com/de/document/read/15634783/skript-zur-vorlesung-universitat-paderborn> (2005). Zugegriffen: 12. Febr. 2021
- Koskenniemi, K.: Two-Level Morphology: A General Computational Model for Word-Form Recognition and Production. @Helsinki, Uni., Diss.: 1984. Publications. Department of General Linguistics. University of Helsinki, Bd. 11, Helsinki (1983)
- Lyons, J.: Semantics, Bd. 1. Cambridge University Press, Cambridge (1977)
- Maedche, A., Staab, S.: Ontology Learning for the Semantic Web. IEEE Intell. Syst. **16**(2), 72–79 (2001). <https://doi.org/10.1109/5254.920602>
- Manning, C.D., Schütze, H.: Foundations of Statistical Natural Language Processing, 8. Aufl. MIT Press, Cambridge (1999)
- Mehler, A., Stegbauer, C., Frank-Job, B.: Ferdinand de Saussure. 1916. Cours de linguistique générale. In: Holzer B., Stegbauer C. (Hrsg.) Schlüsselwerke der Netzwerkforschung. Netzwerkforschung. Springer VS, Wiesbaden (2019). [https://doi.org/10.1007/978-3-658-21742-6\\_116](https://doi.org/10.1007/978-3-658-21742-6_116)
- Montague, R.: Formal Philosophy. Selected Papers of Richard Montague. Yale University Press, New Haven (1974)
- Nutek Inc.: Design and Behavior of Bolted Joints. <http://nutek-us.com/QITT06%20-%20Bolted%20Joints.pdf> (2014). Zugegriffen: 12. Febr. 2021
- Porter, M.F.: An algorithm for suffix stripping. Program **14**(3), 130–137 (1980). <https://doi.org/10.1108/eb046814>
- Rauh, G.: Syntactic Categories. Their Identification and Description in Linguistic Theories. Oxford Surveys in Syntax & Morphology, Bd. 7. Oxford Univ. Press, Oxford (2010)
- Roelcke, T.: Fachsprachen. Grundlagen der Germanistik, Bd. 37. Erich Schmidt Verlag, Berlin (1999)
- Ruppenhofer, J., Ellsworth, M., Petrucc, M.R.L., Johnson, C.R., Baker, C.F., Scheffczyk, J.: FrameNet II: Extended Theory and Practice. University of Berkely, San Francisco. [https://framenet.icsi.berkeley.edu/fndrupal/the\\_book](https://framenet.icsi.berkeley.edu/fndrupal/the_book) (2016). Zugegriffen: 5. Jan. 2021
- Sager, N., Friedman, C., Lyman, M.S.: MD, and members of the Linguistic String Project: Medical Language Processing: Computer Management of Narrative Data. Addison-Wesley, Reading (1987)
- de Saussure, F.: Grundfragen der allgemeinen Sprachwissenschaft, 5. Aufl. de Gruyter, Berlin (1966)
- Schiller, A., Teufel, S., Thielen, C., Stöckert, C.: Guidelines für das Tagging deutscher Textcorpora mit STTS. (Kleines und großes Tagset). <http://www.sfs.uni-tuebingen.de/resources/stts-1999.pdf> (1999). Zugegriffen: 9. Dez. 2020
- Schmalenbach, K., Freibott, G., Heid, U.: Technische Fachsprachen im Maschinen- und Anlagenbau – am Beispiel der Fördertechnik. In: Hoffmann, L., Kalverkämper, H., Wiegand, H.E. (Hrsg.) Fachsprachen – Languages for Special Purposes. Ein internationales Handbuch zur Fachsprachenforschung und Terminologiewissenschaft, Bd. 1, S. 1192–1201. De Gruyter, Berlin (1998)
- Schott, G.: Automatische Deflexion deutscher Wörter unter Verwendung eines Minimalwörterbuchs. In: Sprache und Datenverarbeitung **2**(1), 62–77 (1978)
- Tesnière, L.: Éléments de syntaxe structurale. Klincksieck (1959)

- Weizenbaum, J.: ELIZA – A Computer Program for the Study of Natural Language Communication between Man and Machine. *Commun. ACM* **9**(1), 36–45 (1966). <https://doi.org/10.1145/365153.365168>
- Wikipedia: Mutter (Technik). [https://de.wikipedia.org/w/index.php?title=Mutter\\_\(Technik\)&oldid=210127401](https://de.wikipedia.org/w/index.php?title=Mutter_(Technik)&oldid=210127401) (2021). Zugegriffen: 24. März 2021
- Winograd, T.: Understanding Natural Language. Academic, New York (1972)
- Wittgenstein, L.: Philosophische Untersuchungen. Kritisch-genetische Edition. Herausgegeben von Joachim Schulte. Wissenschaftliche Buchgesellschaft, Frankfurt a. M. (2001)
- Wright, S.E.: Term Selection: The initial phase of terminology management. In: Wright, S.E., Budin, G. (Hrsg.) *Handbook of Terminology Management*, Bd. 1, S. 13–23. Benjamins, Amsterdam (1997)
- Wüster, E.: Einführung in die allgemeine Terminologielehre und terminologische Lexikographie, 3. Aufl. Abhandlungen zur Sprache und Literatur, Bd. 20. Romanistischer, Bonn (1991)
- Zielinski, A., Simon, C.: Morphisto - An Open-source Morphological Analyzer for German. In: *FSMNLP 2008. Seventh International Workshop on Finite-State Methods and Natural Language Processing*, Ispra, Italy, S. 177–182. (2008)



# Maschinelle Verarbeitung von Text

3

## Zusammenfassung

In diesem Kapitel werden verschiedene Ansätze zum Verarbeiten von elektronisch verfügbarem Text dargestellt. Dies umfasst insbesondere die regelbasierte Verarbeitung durch Extraktionsmuster, die Verarbeitung durch statistische Verfahren des maschinellen Lernens und die Repräsentation lexikalischer Einheiten mit neuronal gelernten Methoden. Typischerweise werden durch linguistische Ebenen inspirierte Komponenten in eine linguistische Verarbeitungspipeline zusammengefasst, deren Verarbeitungsschritte aufeinander aufbauen, sodass zunächst unstrukturiert vorliegende Textdokumente mit syntaktischer und semantischer Information angereichert werden können. Dies bildet die Grundlage für Anwendungen, die auf dieser Information aufzubauen. Ferner behandelt dieses Kapitel die Verarbeitung großer Dokumentsammlungen, was insbesondere Parallelverarbeitung und die Wahl geeigneter Speicherformate umfasst, und untersucht die Auswirkung der Korpusgröße auf die Qualität der Resultate.

## 3.1 Verarbeitungsparadigmen für Text

In diesem Abschnitt werden verschiedene Paradigmen zum Verarbeiten von Text gegenübergestellt. Diese unterteilen sich in **regelbasierte Verfahren** mit Mustern, **statistische Verfahren** des feature-basierten maschinellen Lernens und **neuronale Verfahren** zur Repräsentation von textuellen Einheiten. Alle Paradigmen haben nach wie vor ihre Berechtigung und werden auch teilweise in Kombination miteinander verwendet. Unterschiede bestehen im manuellen Aufwand bei der Erstellung von Regeln oder bei der Implementierung von Merkmalen, bei der Größe der nötigen manuell annotierten

Trainingsdaten, bei der Geschwindigkeit des Trainings für die Erstellung von Modellen und bei der algorithmischen Transparenz.

Eine grundlegende Anwendung des Text Mining, welche für viele Anwendungen in der einen oder anderen Form vorausgesetzt wird, ist **Informationsextraktion** (IE, information extraction) (vgl. Jiang 2012). Anwendungsabhängig sollen lediglich die relevanten Teile der Textdokumente betrachtet werden, z. B. die darin vorkommenden Eigennamen, um diese dann im Einzelnen oder geeignet aggregiert Nutzenden der Anwendung zur Verfügung zu stellen. Hierzu werden sprachtechnologische Verarbeitungskomponenten eingesetzt.

In den letzten Jahrzehnten wurden zwei große Paradigmenwechsel in der sprachtechnologischen Forschung vollzogen: Während die durch teure Hardware mit kleinen Speicher- und Verarbeitungskapazitäten geprägte Anfangszeit der maschinellen Verarbeitung von Text bis Anfang der 1990er Jahre von direkt von linguistischen Theorien inspirierten **regelbasierten Methoden** (Abschn. 3.1.1) dominiert wurde, wurden im Anschluss durch Weiterentwicklungen im maschinellen Lernen und durch die Verfügbarkeit und Verarbeitbarkeit größerer Textsammlungen und computerlinguistischen Ressourcen (Kap. 4) vermehrt **statistische Verfahren** (Abschn. 3.1.2) entwickelt. Dies markiert auch den Übergang von explizit vorgegebenen Verarbeitungsschritten durch wohldefinierte Regeln zu statistischen Modellen, welche auf (ggf. händisch annotierten) Korpora trainiert werden: Während bei regelbasierter Verarbeitung implementiert wird, *WIE* die Verarbeitung von Text vonstattengehen soll, werden statistische Verfahren des überwachten maschinellen Lernens instruiert, *WAS* bei der Verarbeitung herauskommen soll, ohne diese notwendigerweise mit linguistischer Theorie unterfüttern zu müssen.

Für statistische Verfahren des maschinellen Lernens (Kap. 6) müssen die **Instanzen**, also für das Text Mining z. B. Wörter, Wortgruppen, Sätze oder ganze Dokumente, auf eine Weise charakterisiert werden, welche das Lernen einer Funktion ermöglicht, z. B. für die Entscheidung, ob eine Instanz bzgl. eines Problems interessant ist. Diese Charakterisierung erfolgt durch **Merkmale (Features)**, welche auf den Instanzen ermittelt werden. Beispiele für ein Merkmal für Dokumente sind z. B. die Anzahl von enthaltenen Wörtern, ein Merkmal für Wörter ist z. B. die Eigenschaft, ob das Wort groß oder klein geschrieben ist. Das **maschinelle Lernverfahren** entscheidet dann lediglich auf Basis der Merkmalswerte über die Ähnlichkeit der Instanzen (Abschn. 6.2) bzw. über deren **Klassifikation** (Abschn. 6.6), bei der eine vorgegebene Einteilung auf den in einer manuell erstellten **Trainingsmenge** enthaltenen Beispielen gelernt wird. Der manuelle Aufwand liegt folglich in der Erstellung und der Ermittlung einer geeigneten Kombination von Merkmalen, und für überwachte Verfahren im Annotieren der **Trainingsmenge**.

Der Erfolg **neuronaler Methoden** seit Anfang der 2010er Jahre beruht auf deren Fähigkeit, geeignete Feature-Repräsentationen selbst zu lernen; der Aufwand für das manuelle Definieren und Kombinieren der Merkmale entfällt (Abschn. 6.10). In neuronalen Architekturen werden Instanzen immer durch reellwertige Vektoren repräsentiert, welche bezüglich einer Zielfunktion optimiert werden – dies kann direkt die **Zielfunktion**

des überwachten Lernens sein. Es gibt aber auch Ansätze, welche die Zielfunktion auf Hilfsaufgaben wie z. B. **Sprachmodellen** (Abschn. 5.5) optimieren. Durch die Eigenschaft neuronaler Netzwerke, nichtlineare Zusammenhänge modellieren zu können, und durch die inhärente Aggregation ähnlicher Instanzen über deren Vektorrepräsentationen sind die so gelernten Merkmale oft effektiver. Ein Nachteil neuronaler Methoden ist deren großer Parameterraum und deren große Anzahl an Stellschrauben (sogenannte **Hyperparameter**) bei Architektur und Lernprozess, weswegen das Finden geeigneter Konfigurationen durch gezieltes Ausprobieren große Rechenleistung benötigt; diese steht wiederum erst seit einigen Jahren kostengünstig zur Verfügung, verursacht jedoch einen stetig steigenden Energieverbrauch.

Im Folgenden werden diese drei Verarbeitungsparadigmen weiter ausgeführt. In linguistischen Pipelines zur Textverarbeitung (Abschn. 3.2) können diese auch in **hybriden Systemen** kombiniert werden. Bei der Erwägung, welches Paradigma für eine bestehende Problemstellung eingesetzt werden sollte, fließen oftmals auch andere Einflussgrößen als die reine Qualität der Ausgabe in die Entscheidungsfindung ein. Regelbasierte Systeme sind maximal transparent, da die Funktionsweise direkt in den Regeln angegeben wird, und benötigen auch keine annotierte Trainingsmenge, sodass mit diesem Ansatz schnell funktionale Verarbeitungskomponenten erstellt werden können, welche jedoch im Allgemeinen nicht die Qualität von statistischen oder neuronalen Komponenten erreichen. Neuronale Systeme sind maximal intransparent, da weder die Merkmale, also die Repräsentationen von Instanzen, interpretierbar sind, noch die Gründe für die algorithmische Entscheidung. Dafür erreichen sie bei entsprechender Optimierung in deutlicher Mehrheit der Fälle die beste Qualität. Statistische Verarbeitung bietet hier einen Kompromiss: Die Überführung von Instanzen in deren Merkmals-Repräsentation ist transparent, die Transparenz der algorithmischen Entscheidung hängt vom eingesetzten Lernverfahren ab, ist jedoch in fast jedem Fall schwieriger nachzuvollziehen als bei regelbasierter Verarbeitung, und die Qualität bei gleicher Trainingsmenge ist oftmals höher als bei regelbasierter Verarbeitung, erreicht jedoch nicht die Qualität neuronaler Verfahren.

### 3.1.1 Regelbasierte Verarbeitung

Wir betrachten zunächst die technische Umsetzung einer intuitiven Idee: Sprache folgt regelhaften Mustern für das Ausdrücken u. a. von Sachverhalten. Wenn wir bestimmte Arten von Sachverhalten aus Text extrahieren möchten, dient Musteranalyse und deren Umsetzung in Extraktionsregeln als Instrument, aus der Vielzahl sprachlicher Muster diejenigen zu identifizieren, welche für den vorliegenden Anwendungskontext relevant sind: Regelbasierte Verarbeitung erfolgt durch das Erkennen wohldefinierter Muster, welche durch Computerlinguisten und -linguistinnen direkt erstellt werden.

Ein Vorteil dieses Ansatzes ist die schnelle Entwicklung von anwendungsrelevanten Komponenten, ein Nachteil ist die meist fehlende Abdeckung, da wegen der großen

Variabilität von Sprache und der daraus folgenden großen Anzahl von relevanten Mustern meist viele der für uns relevanten Sachverhalte durch regelbasierte Verarbeitung unerkannt bleiben.

Während in den Anfangszeiten der automatischen Sprachverarbeitung versucht wurde, alle linguistischen Ebenen (Kap. 2) durch regelbasierte Verarbeitung abzubilden, beschränkt sich deren Anwendung heutzutage auf sprachliche Subsysteme, welche sehr regelhaft funktionieren.

Betrachten wir ein Beispiel einer E-Mail an den Kundensupport eines Telekommunikationsunternehmens:

#### Beispiel – Kundensupport-E-Mail

From: [berndhunter@provider.de](mailto:berndhunter@provider.de)  
To: [kundenservice@telesmartica.de](mailto:kundenservice@telesmartica.de)  
Betreff: meine Adresse  
Sehr geehrte Damen und Herren,  
hiermit möchte ich Ihnen meine neue Adresse mitteilen: Lautenbacher Weg 15, 05667 Buntenbach, Tel. +49 (03226) 715183. Meine Kundennummer ist D17-00031148.  
mit Bitte um Eingangsbestätigung und freundlichen Grüßen,  
Bernd Hunter. ◀

Im Beispiel finden sich einige sehr regelhaft und schematisch funktionierende Elemente: E-Mail-Adressen, Telefonnummern, die Kundennummer und die Adresse folgen Schemata, welche durch **reguläre Ausdrücke** (Regular Expressions, vgl. Kleene et al. 1956) beschrieben werden können.

Reguläre Ausdrücke beschreiben eine eigene, vollständig definierte „Kunstsprache“, welche in vielen Varianten in praktisch jeder Programmiersprache, vielen Texteditorinnen und Texteditoren, vor allem aber beim Text Mining oder allgemein in der automatischen Sprachverarbeitung zum Einsatz kommt. In der Praxis ermöglicht diese Sprache, Ausdrücke zu formulieren bzw. Muster festzulegen, nach denen gesucht werden soll.

Reguläre Ausdrücke bestehen aus sog. Atomen, welche die Einheiten der Sprache sind (also einzelne Buchstaben oder ganze Wörter) und Operatoren, welche die grammatischen Regeln dieser Sprache ausdrücken. Atome stellen genau ein gesuchtes Zeichen dar. Dabei ist es gleich, ob damit eine Zeichenklasse, wie alle Kleinbuchstaben (also [a-z]), eine Auswahl davon, wie Kleinbuchstaben von a bis k (also [a-k]), oder ein konkretes Zeichen gemeint ist. Das Minus „-“ gilt jedoch nur für aufeinander folgende Bereiche von Zeichen. Soll eine spezifische Kombination von Zeichen gesucht werden, muss der Zeichenbereich explizit mit eckigen Klammern angegeben werden, wie z. B. für alle Vokale [aeiouäöü].

Mithilfe von Operatoren können Atome verknüpft werden, wobei der einfachste Operator, die Konkatenation, nicht immer durch ein Zeichen repräsentiert wird.

Auf der Buchstabenebene könnte also mit dem Ausdruck `ch` nach allen gleichzeitigen Vorkommen von `c` gefolgt von `h` gesucht werden. Werden zwei Atome mit dem Oder-Operator verknüpft, z. B. `c | h`, dann werden alle Vorkommen als Treffer gezählt, bei welchen das eine oder das andere Atom vorkommt, in diesem Fall also `c` oder `h`. Tab. 3.1 illustriert die Operatoren und ihre Syntax.

Die eigentliche Mächtigkeit der regulären Ausdrücke verbirgt sich hinter den Begriffen der **Wildcard**, welche einen Platzhalter für beliebige Zeichen darstellt, und den **Quantoren** zum Ausdruck von Wiederholungen. So lässt sich zum Beispiel mit der Kombination von Punkt und Asterisk „`*`“ in einem Text nach beliebig vielen beliebigen Zeichen suchen. Dabei steht der Punkt für genau ein beliebiges Zeichen. Der Asterisk hingegen bedeutet, dass der vorangegangene Ausdruck beliebig oft vorkommen darf, wobei auch Nullmal möglich ist. In Kombination mit anderen Atomen erlaubt dies nützliche Suchmasken, z. B. lassen sich alle Wörter, die mit `un` anfangen, mit dem Muster `^un.*$` in einer Liste finden; für das Finden im Text wird die Modellierung der Wortgrenzen notwendig, z. B. mit `(\s|^) (un.*) (\s|$)`. Tab. 3.2 erklärt die Wildcard und die Quantoren bei regulären Ausdrücken.

Soll ausnahmsweise nach den Zeichen für Wildcard oder Operatoren selbst gesucht werden, wie zum Beispiel nach allen Vorkommen von `*` (Asterisk), so muss dieses Zeichen entwertet werden mit dem sog. **Escape-Zeichen**: „`\``. Operatoren zur Bildung von Ausdrücken und Wildcards können frei kombiniert werden. Ferner existieren noch spezielle Steuerzeichen, wie `\s` für Whitespace, `\t` für Tabulator, `\d` für Ziffern u. v. m.

Für unser Beispiel können folgende reguläre Ausdrücke verwendet werden, um Information aus dieser und ähnlichen E-Mails zu extrahieren: E-Mailadressen können z. B. mit `[^\s] + @ [^\s] + \. [a-z] [a-z] [a-z]*` erkannt werden. Die Kundennummer `D17-00031148` könnte mit `[A-Z] [0-9]{2} [\-\s]? \d{8}` erkannt werden, das Format ist ein Großbuchstabe, dann zwei Ziffern, dann ein optionaler Bindestrich oder Leerzeichen, dann 8 Ziffern. Alternativ könnte ein Muster definiert werden,

**Tab. 3.1** Reguläre Ausdrücke: Operatoren

Operator	Bedeutung	Beispiel
<code>()</code>	Gruppierung als Ausdruck	<code>(ch)</code> findet alle Stellen, bei denen <code>c</code> und <code>h</code> direkt aufeinander folgen; Gruppen werden sinnvollerweise mit Iterationsoperatoren kombiniert, siehe unten
<code>[]</code>	Zeichenbereichsoperator	<code>[1-8]</code> findet alle Zahlen von 1 bis 8; <code>[aeiouäöü]</code> findet die Vokale der deutschen Sprache
<code> </code>	Oder-Operator	<code>a   b</code> findet sowohl alle <code>a</code> als auch alle <code>b</code>
<code>^</code> und <code>\$</code>	Anfangs- und Endezeichen	<code>^</code> symbolisiert den Anfang des Textes, <code>\$</code> das Ende; so kann z. B. nach Sätzen, welche mit einem Ausrufezeichen enden, mit <code>! \$</code> gesucht werden
<code>[^]</code>	Negationsoperator (nur bei Zeichenketten)	<code>[^a]</code> findet alle Zeichen außer <code>a</code>

**Tab. 3.2** Reguläre Ausdrücke: Wildcard, Quantoren, Escape-Zeichen

Zeichen	Bedeutung
.	Wildcard für genau ein beliebiges Zeichen
*	Der vorangegangene Ausdruck darf beliebig oft hintereinander wiederholt vorkommen, auch gar nicht
+	Der Ausdruck davor muss mindestens einmal vorkommen, darf aber beliebig oft wiederholt werden
?	Der Ausdruck davor ist optional, kann also 0-mal oder 1-mal vorkommen
{n}	Der Ausdruck davor muss genau $n$ (natürliche Zahl) mal vorkommen. Auch die Definition von Bereichen, Mindestanzahlen und Maximalanzahlen sind möglich
\	Escape-Zeichen zum Entwerten von Sonderzeichen (Operatoren und Wildcards)

welches den Kontext beinhaltet, z. B. „Kundennummer ist  $([\^s] + [^0-9])^*$ “. Dieser würde jedoch wegen der falschen Schreibweise „Kudennummer“ im Beispiel fehlschlagen; wichtig ist hier auch das Ende des Musters, hier limitiert durch ein Zeichen, welches keine Ziffer ist.

Bei der Erstellung der Ausdrücke ist eine Balance zu finden bei der Plausibilitätsprüfung: Beispielsweise können Datumsangaben durch die reguläre Struktur leicht erkannt werden, jedoch wird der Ausdruck schnell kompliziert, wenn z. B. der 30. Februar ausgeschlossen werden soll.

Reguläre Ausdrücke oder Muster im Allgemeinen können auch auf Ergebnissen der Vorverarbeitung definiert werden: So findet z. B. ein Muster wie NN wie (NN,) \* NN (und|oder) NN Oberbegriff-Unterbegriff-Beziehungen zwischen Nomina (Kürzel: NN), zum Beispiel in *Instrumente wie Gitarre, Geige und Gambe*, wenn die Wortarten zuvor automatisch im Kontext zugewiesen wurden, wie in (Abschn. 3.2.4) genauer erläutert wird. Ein weiteres Anwendungsgebiet für reguläre Ausdrücke ist die morphologische Verarbeitung, um den Stamm von den Flexionen von Wörtern zu trennen (Abschn. 2.2.2), in diesem Kontext werden auch sogenannte **Transduktoren** (Finite State Transducers) eingesetzt (vgl. Beesley und Karttunen 2003).

Neben regulären Ausdrücken, welche in ihrer Ausdrucksstärke beschränkt sind (Chomsky 1957), kommen in der regelbasierten Verarbeitung auch andere, komplexere Mechanismen zum Einsatz, z. B. kontextfreie Grammatikregeln beim syntaktischen Parsen, siehe Abschn. 3.2.4.

Regelbasierte Verarbeitung war bis in die 1980er Jahre hinein das einzige automatische Verarbeitungsparadigma für natürliche Sprache und befindet sich seither auf dem Rückzug, da in den meisten Fällen eine hohe Abdeckung durch die mögliche Variabilität im Sprachausdruck nur selten erreicht werden kann. Regelbasierte Verarbeitung hat weiterhin ihre Berechtigung in modernen Systemen, wenn die zu identifizierbaren Textstellen oder sprachlichen Phänomene stark regelhaft funktionieren. Auch eignet sich das regelbasierte Verarbeitungsparadigma als schnelle Lösung zum Erstellen

funktionaler, jedoch nicht besonders effektiver Sprachverarbeitungskomponenten z. B. in einem ersten Prototyp, da keine Trainingsmengen erstellt werden müssen. Sollen sprachlich diverse Phänomene mit hoher Genauigkeit und Abdeckung erkannt und für das Text Mining nutzbar gemacht werden, ist die Verwendung von statistischer oder neuronaler Verarbeitung, welche in den folgenden Abschnitten beschrieben werden, alternativlos.

### 3.1.2 Überwachte Statistische Verarbeitung

Das statistische Verarbeitungsparadigma nutzt Statistiken über das Zusammenauftreten von sprachlichen Einheiten oder zwischen sprachlichen Einheiten und manuell vergebenen Klassifikationen zur Modellierung, um sprachliche Einheiten in Beziehung zu setzen oder automatische Klassifikatoren zu lernen. Kernidee ist das Lernen durch Beispiele: Bei einer ausreichenden Größe einer manuell ausgezeichneten Trainingsmenge gelingt es, ein automatisches Modell zu lernen, welches auf neuem Text erfolgreich die Klassifikationen zuweisen kann. Hierbei werden in diesem auch ‚klassisches maschinelles Lernen‘ genannten Paradigma die sprachlichen Einheiten durch Merkmale (sogenannte Features) charakterisiert, deren Berechnung einzeln programmiert wird. Für die in diesem Kontext besprochenen überwachten Verfahren ermittelt ein **maschinelles Lernverfahren** eine Funktion, welche die Gesamtheit der Featurewerte in die in der Trainingsmenge vorgegebenen Klassen umwandelt.

Für das statistische Paradigma muss sprachliches Material zunächst also in Merkmale umgewandelt werden, um dann die quantitativen Beziehungen zwischen Featurewerten und Zielklassifikation zu nutzen. Um Statistik zu betreiben, müssen Ereignisse mehrfach, möglichst oft beobachtet werden, um **Konfidenz** in statistischen Verfahren zu erreichen. Durch die Merkmale wird sozusagen die Brille definiert, durch die das Lernverfahren das sprachliche Material betrachten kann – diese sind notwendigerweise reduktionistisch, um auch für verschiedene sprachliche Einheiten gleiche Merkmalswerte zu erreichen und so beobachtete Merkmalskonstellationen auf ungewohnte Ereignisse zu übertragen, auf Basis derer dann automatisch Klassen vergeben werden können. Die Werte der Merkmale sollten mit den Zielklassifikationen korrelieren, damit das Lernverfahren Regularitäten erkennen und nutzen kann.

Doch was genau sind diese Merkmale bzw. diese Features und wie werden deren Werte ermittelt? Für jedes Merkmal wird eine Merkmalsfunktion erstellt, welche Merkmalswerte für alle zu charakterisierenden sprachlichen Einheiten berechnen kann. Merkmalsfunktionen können hierbei auf einer Vielzahl von Eigenschaften des sprachlichen Materials definiert werden. Um die Flexibilität dieses Konzeptes zu illustrieren, werden im Folgenden Merkmalsfunktionen für verschiedene sprachliche Einheiten exemplarisch erläutert.

Eine Merkmalsfunktion  $f: X \rightarrow range(f)$  ordnet einer Instanz (hier: sprachliche Einheit) einen Wert aus dem Wertebereich aller Merkmale ( $range$ ) der zu.

Der Wertebereich kann aus numerischen (z. B. 1,5) oder nominalen Werten (z. B. aus {ROT, GRÜN, BLAU} bestehen, speziell auch aus binären Werten {TRUE, FALSE}. Die Wahl der Art der Wertebereiche bestimmt auch die Einsatzmöglichkeiten verschiedener Lernverfahren, da nicht alle solche Ansätze mit nominalen bzw. numerischen Werten umgehen können. Üblicherweise werden Instanzen durch mehrere Merkmale charakterisiert, deren Werte den Merkmalsvektor der Instanz wie in Gl. 3.1 bilden.

$$F(X) = (f_1(X), f_2(X), f_3(X), \dots, f_n(X)) \quad (3.1)$$

Merkmale für sprachliche Einheiten können auf verschiedene Arten berechnet werden:

- Merkmale auf Basis von Untereinheiten: Merkmale für die Repräsentation von Texten können die dort enthaltenen Wörter sein, Merkmale für Tokens die darin enthaltenen Buchstaben und Buchstabenkombinationen usw.
- Oberflächenmerkmale: Die Länge des Tokens/Dokuments, die Großschreibung von Tokens, die Eigenschaft eines Tokens, eine Zahl zu sein etc.
- Kontextuelle Merkmale: z. B. können benachbarte Tokens als Merkmale verwendet werden.
- Wissensbasierte Merkmale: z. B. können Städte durch eine Liste von Städten identifiziert und daraus ein binäres *ist\_stadt*-Merkmal abgeleitet werden.
- Merkmale aus Vorverarbeitungsschritten: z. B. kann die durch POS-Tagging automatisch ermittelte Wortart von Tokens verwendet werden, vgl. Abschn. 3.2.4.
- Kombinationen von Merkmalen, z. B. kontextuelle Merkmale aus Vorverarbeitungsschritten, wissensbasierte Merkmale auf Untereinheiten, musterbasierte Merkmale als Ausgabe von regelbasierter Verarbeitung etc.

Nachdem nun eine Repräsentation, quasi eine Charakterisierung einer Instanz durch Merkmale erfolgen kann, können auf Basis von Korrelationen zwischen Merkmalen und Zielklassen automatische Klassifikatoren gelernt werden.

Einzelne Merkmale oder Mengen von Merkmalen bzw. Merkmalsvektoren können dazu genutzt werden, um diese Zielklassen zu beschreiben. Zunächst entspricht jeder Merkmalswert einer atomaren Klasse, mehrere solche atomaren Klassen werden dann zu einer Zielklasse zusammengefasst. Besteht eine solche Zielklasse nur aus einer atomaren Klasse, bei der ein bestimmtes Merkmal den Wert y hat, dann können wir eine Klassenzugehörigkeitsfunktion  $K : X \times C \rightarrow \{0,1\}$  definieren durch:

$$K(x, y) = \begin{cases} 1 & \text{falls } F(x) = y \\ 0 & \text{sonst} \end{cases}$$

Der Charme dieser in Gl. 3.2 definierten Klassenzugehörigkeitsfunktion besteht darin, dass später bei sog. probabilistischen Modellen der Wertebereich auf das Intervall [0,1] erweitert werden kann, um mit Wahrscheinlichkeiten von Zugehörigkeiten zu mehreren Klassen umgehen zu können.

Zum Beispiel könnte dies ein Klassifikator sein, welcher die Zuordnung von Dokumenten aus der Menge  $X$  zu Rubriken  $C$  in der Zeitung wie Politik, Wirtschaft, Sport etc. treffen soll, indem  $K$  aufgrund der Korrelation von  $F(X)$  und den Elementen  $c \in C$  eine 1 ausgibt, falls die Rubrik als zutreffend erachtet wird, sonst 0. Dies kann z. B. erreicht werden, indem Merkmale für Inhaltswörter definiert werden, und der Klassifikator beim Vorkommen des Wortes *Fußball* im Dokument die Rubrik *Sport* favorisiert, siehe z. B. Manning und Schütze (1999), Kap. 16. Ein anderes Beispiel könnte sein, die Wortarten von Tokens im laufenden Text zuzuweisen (siehe auch Abschn. 3.2.4): Bei der Erkennung von Eigennamen und Nomen ist ein gut korreliertes Merkmal die Großschreibung des Wortes.

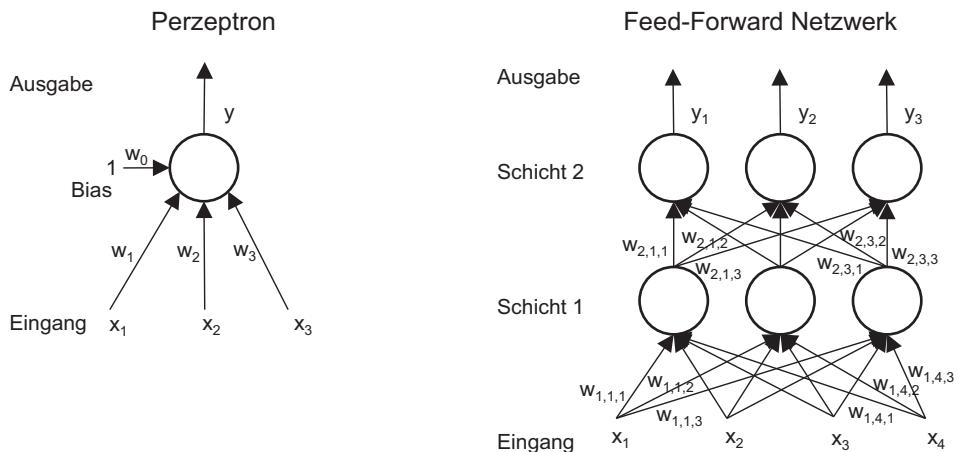
Weitere Einschränkungen können gelten, z. B. dass die Zuordnung nur genau zu einer Rubrik erfolgen soll; die Definition erlaubt aber auch die Mehrfachzuordnung. Eine weiterführende Darstellung von Klassifikation, insbesondere auch der Lernverfahren für verschiedene Arten von Klassifikatoren, erfolgt in Abschn. 6.6, im nächsten Abschnitt findet sich ein Beispiel. An dieser Stelle soll lediglich noch erwähnt werden, dass der Erfolg überwachter statistischer Verarbeitung vom Zusammenspiel der für das jeweilige Problem geeigneten Merkmale und des Klassifikationsalgorithmus abhängt. Das Hauptaugenmerk bei der Entwicklung entsprechender Komponenten und dementsprechend der höchste menschliche Arbeitsaufwand besteht in der Definition, Implementierung und Erprobung geeigneter Merkmale. Weitere signifikante Aufwände bestehen in der Erstellung geeigneter Trainingsdaten; diese können jedoch auch für die neuronale Verarbeitung genutzt werden.

### 3.1.3 Neuronale Verarbeitung

In diesem Abschnitt wird das Prinzip der neuronalen Netzwerke dargestellt. Es erfolgt eine kurze Einführung in die neuronale Sprachverarbeitung aus Anwendungssicht. Eine deutlich ausführlichere und weiterführende Darstellung leisten z. B. Goodfellow et al. (2016) und Goldberg (2017). Text und die darin enthaltenen lexikalischen Einheiten werden zunächst mithilfe sogenannter Embeddings (Abschn. 5.6) in reellwertige Vektoren umgewandelt; diese dienen dann als Eingabeschicht für neuronale Netzwerke, welche darauf trainiert werden können, bestimmte Zielgrößen zu optimieren oder Klassifikationen zu lernen. Im Gegensatz zu Modellen des ‚klassischen maschinellen Lernens‘ werden hier die Merkmale zur Charakterisierung von Trainings- und Testinstanzen nicht explizit programmiert, sondern innerhalb des Netzwerkes induziert, das heißt datengetrieben gelernt. Dieses Vorgehen ist attraktiv, da die Arbeit zur Erstellung der Merkmale entfällt. Ein Nachteil der Vorgehensweise ist die vergleichsweise große Anzahl an Trainingsdaten, welche hierfür erforderlich ist. Hier geht der Trend zum Verwenden vortrainierter Embeddings, welche lediglich auf die zu erlernende Aufgabe angepasst werden.

Kernidee bei künstlichen neuronalen Netzwerken ist eine sehr grob an die Biologie angelehnte Simulation neuronaler Verarbeitung durch **Neuronen**. Dies sind Zellen, welche Eingangsgrößen  $x_1, x_2, \dots, x_n$  mit reellwertigen Gewichten  $w_1, w_2, \dots, w_n$  multiplizieren und diese mithilfe einer Funktion in eine einzige Ausgabegröße  $y$  umwandeln (vgl. McCulloch und Pitts 1943; Rosenblatt 1958), wobei die Ein- und Ausgabegrößen jeweils durch Zahlen aus reellwertigen Intervallen (z. B. zwischen  $-1$  und  $1$ ) modelliert werden. Es gibt viele Varianten, diese Umwandlung zu realisieren, z. B. über das Erreichen eines Schwellwertes der Summe der gewichteten Eingänge und des Bias oder mithilfe einer (normalerweise nichtlinearen Funktion), oder über das Verwenden komplexerer interner Strukturen in den Zellen wie z. B. bei **Long Short-Term Memories** (LTSMS) (vgl. Hochreiter und Schmidhuber 1997) oder **Gated Recurrent Units** (GRUs) (vgl. Cho et al. 2014). Neuronen können parallel und hintereinander miteinander verschaltet werden. Bei sogenannten **Feed-Forward-Netzen** wird eine Schichtenarchitektur realisiert, bei der die Ausgänge der vorherigen Schicht in die Eingänge der Neuronen in der nächsten Schicht übergeben werden. In der Abb. 3.1 wird links ein einfaches Neuron illustriert, das **Perzeptron** mit 3 Eingängen, in der Abb. 3.1 rechts wird ein Feed-Forward-Netzwerk dargestellt: Es handelt sich um ein voll verbundenes Feed-Forward-Netz aus zwei Schichten mit je 3 Neuronen, welches vier Eingänge in drei Ausgänge überführt. Die Neuronen darin können z. B. durch Perzeptronen implementiert werden; hier wurde der Bias in der Darstellung weggelassen. Die Gewichte  $w_i$  an allen Verbindungen stellen jeweils die Parameter der einzelnen Perzeptronen bzw. des gesamten Netzwerkes dar.

Neuronale Netzwerke können für überwachte statistische Verarbeitung (Abschn. 3.1.2) als Klassifikator verwendet werden: Hierzu kann man beispielsweise eine **Netzwerktopologie** verwenden, bei der es genauso viele Ausgaben wie Klassen



**Abb. 3.1** Darstellung eines Perzeptrons und eines Feed-Forward-Netzes

C gibt, sodass jedes Neuron der höchsten Schicht einer Klasse entspricht, und bei der genauso viele Eingänge wie Merkmale vorhanden sind, sodass jedem Merkmalswert einem Eingangswert entspricht. Mithilfe von Trainingsdaten werden nun die Gewichte (auch: Parameter) des Netzwerkes iterativ angepasst (z. B. mit Gradientenverfahren und **Backpropagation**, siehe z. B. Goodfellow et al. 2016), bis das Netzwerk das gewünschte Verhalten zeigt: Das der richtigen Klasse entsprechende Neuron soll einen hohen Wert ausgeben, die anderen Klassifikationsneuronen einen niedrigen Wert. Hierbei muss auf die richtige Wahl der **Hyperparameter** geachtet werden, unter denen die Architektur des Netzwerkes, die in den Neuronen eingesetzten nichtlinearen Funktionen und verschiedene Stellschrauben, nicht aber die gelernten Gewichte und Biases subsummiert werden.

In den letzten Jahrzehnten wurden neuronale Netzwerke stark weiterentwickelt; insbesondere führten Fortschritte bei der Trainierbarkeit von **rekurrenten Netzwerken**, also Netzwerken, welche Rückwärts-Verbindungen zur Simulation von Gedächtnis beinhalten, zur sogenannten Deep-Learning-Revolution, im Zuge derer viele Anwendungen in der Künstlichen Intelligenz ein für den praktischen Einsatz ausreichendes Niveau erlangten. Beliebte Techniken im Bereich Sprachverarbeitung und Text Mining sind hierbei **Convolutional Neural Networks** (CNNs; vgl. LeCun et al. 2010) und die oben erwähnten LSTMs und GRUs. Weitere erfolgreiche Architekturen sind **Transformer-Networks** (vgl. Vaswani et al. 2017) und Abschn. 5.6 und die in der Generierung von Text eingesetzten **Seq2Seq** (vgl. Sutskever et al. 2014) und **Pointer-Generator-Networks** (vgl. See et al. 2017), welche hier zwar nicht im Einzelnen besprochen werden können, jedoch nicht unerwähnt bleiben sollen.

Der überwältigende Erfolg neuronaler Modelle in der Textverarbeitung und in anderen Bereichen der künstlichen Intelligenz beruht jedoch nicht primär auf deren Einsatz als Klassifikatoren, sondern auf der Möglichkeit, durch unüberwachtes Vortrainieren Feature-Repräsentationen automatisch zu lernen. Eine besondere Bedeutung in diesem Kontext haben sogenannte **Embeddings** für Wörter, Sätze und Dokumente. Embeddings (Abschn. 5.6) sind Vektorrepräsentationen, z. B. für Wörter, welche eine feste Anzahl von Dimensionen haben, z. B. 100. Für die Repräsentation eines Wortes (bzw. eines Satzes, eines Dokumentes) als Eingang in das neuronale Netzwerk können anstatt händisch definierter Merkmale (vgl. Abschn. 3.1.2) Embeddings verwendet werden, der Aufwand bei der Erstellung der Merkmale entfällt.

Embeddings werden auf großen, unannotierten Korpora trainiert. Hierbei ist das Prinzip die Optimierung der Embedding-Repräsentation innerhalb von **Sprachmodellen** (Abschn. 5.5): Die Repräsentation der Wörter im Kontext soll die Vorhersage der Repräsentation eines oder mehrerer Zielwörter ermöglichen. Um bei der Vorhersage erfolgreich zu sein, müssen Sprachmodelle von den einzelnen Wörtern abstrahieren; bei dieser Aufgabe erfolgreiche Embeddings sind dadurch charakterisiert, dass syntaktisch und/oder semantisch ähnliche Wörter auch ähnliche Embeddings zugewiesen

bekommen, was eine entsprechende Abstraktion ermöglicht. Die Instanziierung der Wörter als Vektoren im Embedding-Vektorraum wird auch als **distributionales semantisches Modell** bezeichnet, siehe Abschn. 5.4. Diese Modelle sind oftmals in vortrainierter Form öffentlich verfügbar. Bekannte Modelle zur Repräsentation von Wörtern sind word2vec (vgl. Mikolov et al. 2013), GloVe (vgl. Pennington et al. 2014) und fastText (vgl. Bojanowski et al. 2017) – letzteres kann auch auf Basis von Buchstaben-N-Grammen Embeddings für bisher unbekannte Wörter liefern; die Repräsentation von „Kundennummer“ im Beispiel (Abschn. 3.1.1) würde sehr ähnlich zur Repräsentation von „Kundennummer“ gewählt, was ein Extraktionsalgorithmus entsprechend ausnutzen kann. Zur Repräsentation größerer Texteinheiten existieren Erweiterungen wie z. B. doc2vec (vgl. Le und Mikolov 2014) für die Erstellung von Document Embeddings zur Textklassifikation. Während die bisher erwähnten Modelle statisch sind, d. h. für jedes Wort genau einen Embedding-Vektor bereitstellen, berechnen **kontextualisierte Embedding**-Modelle wie ELMo (vgl. Peters et al. 2018), Flair (vgl. Akbik et al. 2018) und BERT (vgl. Devlin et al. 2019) die Repräsentation von Wörtern in Abhängigkeit vom Kontext; dies führt z. B. zu einer impliziten **Disambiguierung** von mehrdeutigen Wörtern wie *Verteidiger* (Abschn. 2.4), siehe z. B. Wiedemann et al. (2019).

Der Einsatz großer vortrainierter Embedding-Modelle bietet die Möglichkeit, aus großen **Hintergrundkorpora** zu lernen und das in diesen Repräsentationen abgelegte implizite Wissen über Sprache für konkrete Anwendungen im Bereich der Sprachverarbeitung zu nutzen. Die insbesondere für den Eingang in neuronale Netzwerke vorteilhaften reellwertigen Embedding-Repräsentationen können zudem noch für konkrete Aufgaben mithilfe von Trainingsdaten feiner abgestimmt werden (sog. fine-tuning).

Neuronale Methoden sind im Begriff, sich zum Standard für die Verarbeitung natürlicher Sprache zu etablieren. Sie bieten im Allgemeinen die beste Qualität bei weniger manuellem Entwicklungsaufwand für Regeln oder Merkmale, dafür ist ein signifikanter Rechenaufwand bei der Optimierung der Hyperparameter nötig. Während der Einsatz vortrainierter Modelle oder vortrainierter Embeddings keine speziellen Anforderungen an die verarbeitende Hardware stellt, ist das Trainieren der Embeddings auf den hierfür nötigen großen Datenmengen jedoch oft nur mit Spezialhardware wie **GPUs** oder **TPUs** möglich, und übersteigt häufig die in mittleren Unternehmen oder Forschungseinrichtungen vorhandenen Rechenkapazitäten.

Der Entscheidungsmechanismus ist bei neuronalen Ansätzen im Vergleich zu den anderen Paradigmen maximal intransparent, da schon die Eingangsrepräsentation der Embeddings nicht transparent ist, was durch die Modellierung im Netzwerk mit typischerweise sehr großen Parameterräumen noch verstärkt wird. Daher wird ihr Einsatz in lebenswichtigen Bereichen kritisch gesehen; die Verbesserung der Interpretierbarkeit neuronaler Ansätze ist ein aktives Forschungsgebiet.

## 3.2 Die Linguistische Pipeline

Zum Durchführen von Text Mining müssen Texte geeignet vorverarbeitet werden, um die für die Text-Mining-Aufgabe notwendigen Elemente vorhalten zu können. Die linguistische Pipeline, welche sich grob an den **linguistischen Ebenen** (Abschn. 1.3) orientiert, ist die Zusammenfassung von Textverarbeitungsschritten, welche Textdaten inkrementell anreichern, um sie für Text-Mining-Aufgaben vorzubereiten. Dies fängt mit dem Einlesen der Daten und deren Überführung in eine geeignete interne Repräsentation an und erstreckt sich von Vorverarbeitungsschritten wie Segmentierung und das Erkennen der Sprache über die Erkennung syntaktischer Einheiten und Zusammenhänge bis hin zur Auszeichnung von semantischen Informationen wie das Erkennen von Eigennamen, Koreferenz oder der Stimmungsanalyse. Für viele sprachtechnologische Anwendungen kann dieselbe linguistische Pipeline als Basis dienen, weswegen hier die wichtigsten Schritte und Techniken zur Implementierung des Ablaufes dargestellt werden.

### 3.2.1 Pipeline-Modell

Linguistische Pipelines sind komplexe Softwaresysteme mit dezidiert für die Einzelschritte entwickelten **Komponenten**, welche mithilfe von Frameworks miteinander integriert werden können. Eine Konzeption in Einzelschritten gegenüber einer monolithischen Implementierung ist durch die Komplexität, Heterogenität und Volatilität der Komponenten motiviert. Vorverarbeitungskomponenten werden kontinuierlich entwickelt und verbessert, und sind oftmals quellenoffen verfügbar; eine Eigenentwicklung lohnt sich in der Mehrzahl der Anwendungsfälle aufgrund der teils hohen Komplexität nicht. Da die zum jeweiligen Zeitpunkt besten Komponenten sehr heterogen sind und teils in verschiedenen Programmiersprachen (z. B. Java, Python, historisch auch LISP und Prolog) implementiert sind, empfiehlt sich eine eher lose Kopplung, welche durch das Pipeline-Modell ermöglicht wird. Hier werden verschiedene Komponenten auf den zu verarbeitenden Textdaten strikt hintereinander ausgeführt; die einzelnen Komponenten in der Pipeline kommunizieren über sogenannte Annotationen, welche den Texten hinzugefügt werden.

#### Annotationen, Annotatoren, Tagset

In der Computerlinguistik sind **Annotationen** deskriptive oder analytische Anmerkungen, welche sich auf sprachliches Material beziehen. Wir unterscheiden zwischen manuell getätigten und automatischen Annotationen. Häufig sind die Annotationen durch ein **Tagset** bestimmt; Annotationen können sich auf beliebige Ausschnitte sprachlichen Materials beziehen und eine beliebige Komplexität in ihrer Struktur annehmen.

Die einfachste Art der Annotation ist die **Spannenannotation**, die sich auf konsekutive Textspannen bezieht, welche über Buchstaben, Tokens oder Token-N-Gramme (siehe Abschn. 1.2) definiert sein können. Wie in Abb. 3.2 mit den Dependenzrelationen gezeigt, können auch Annotationen definiert werden, welche sich auf andere Annotationen beziehen. Auch Anmerkungen, welche sich auf gesamte Texte beziehen, können als Annotationen aufgefasst werden, z. B. Genreinformation oder die Angabe der Quelle. Im Beispiel der Abb. 3.2 werden zwei Arten von Annotationen angezeigt: Wortarten (wie ADV, VERB, ...) als Spannenannotation über Tokens und grammatische Dependenzen (wie *advmod*, *nsubj*, ...) als Relationenannotationen zwischen Wortarten (siehe Abschn. 3.2.4) folgend der Tagsets der Universal Dependencies (vgl. McDonald et al. 2013).

Die Aufgabe der linguistischen Pipeline ist nun, diese Annotationen automatisch zu Texten hinzuzufügen, welche zunächst in unannotierter Form vorliegen und durch Komponenten in der Pipeline sukzessive mit Annotationen verschiedener Arten angereichert werden, wobei nachfolgende Komponenten die Annotationen vorangegangener Komponenten voraussetzen können. Oftmals geht dem eine manuelle Erstellung gleichartiger Annotationen voraus, um statistische oder neuronale Lernverfahren (Abschn. 3.1.2, Abschn. 3.1.3) trainieren zu können.

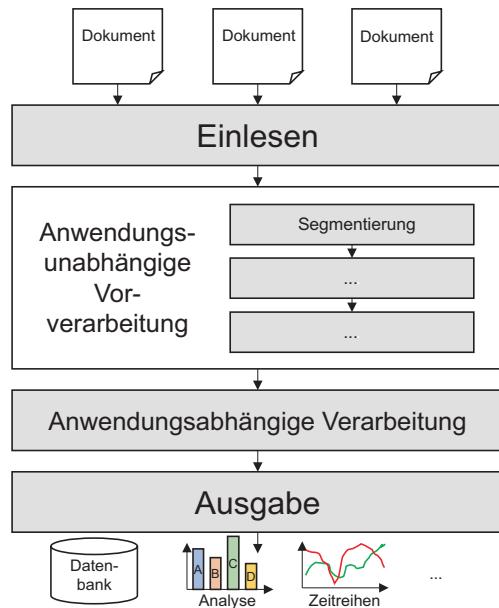
Linguistische Pipelines können in vier Teile unterteilt werden, die je nach Anwendungsfall verschiedenen instanziert werden, siehe Abb. 3.3: Dokumente werden zunächst eingelesen (Abschn. 3.2.2), dann erfolgt eine anwendungsunabhängige Vorverarbeitung (Abschn. 3.2.3 bis Abschn. 3.2.5), eine anwendungsabhängige Verarbeitung (Abschn. 3.2.6) und schließlich die Ausgabe der gewünschten Daten zur Speicherung und Analyse. Bei der Konzeption von Pipelines ist sowohl der konkrete Anwendungsfall einzubeziehen, als auch mögliche spätere Erweiterungen: Werden z. B. bereits beim Einlesen Informationen ignoriert, wie z. B. die Auszeichnung von Überschriften in den Quellen, ist ein späteres Hinzufügen zur Pipeline ggf. sehr aufwendig.

Wie sollte nun eine solche linguistische Pipeline implementiert werden? Eine einfache Möglichkeit sind voneinander unabhängige Programme oder Programm Routinen, welche entsprechend hintereinander ausgeführt werden. Auf Kommandozeilenebene (Terminal unter Unix-basierten Betriebssystemen) könnte man dies mit sogenannten Pipes realisieren, notiert im Beispiel unten mit „|“.



**Abb. 3.2** Wortarten und Dependenzrelationen, erstellt mit dem Annotationstool WebAnno (vgl. Yimam et al. 2013)

**Abb. 3.3** Linguistische Pipeline



#### Realisierung einer Pipeline auf der Kommandozeile

```
Shell> cat dokument.txt | ./satzsegment.pl | ./tokenize.
sed | ./postagger.py > tagged_dokument.txt
```

Das Eingabedokument wird mithilfe eines Perl-Skriptes in Sätze segmentiert, mit einem sed-Skript tokenisiert, anschließend mit einem Python-Skript mit Wortarten (POS) getagged, dann in eine neue Datei ausgegeben. Dieses Vorgehen zur Verarbeitung von unstrukturiertem Text ermöglicht durchaus die Kopplung heterogener Komponenten, sollte jedoch aus Software-Engineering-Sicht vermieden werden. ◀

Während im Beispiel illustriert wird, dass auf diese Weise heterogene, in verschiedenen Programmiersprachen erstellte Komponenten miteinander integriert werden können, sollte eine linguistische Pipeline auf strukturiertere Weise implementiert werden, da diese lose Kopplung einige Nachteile hat:

- Es gibt keinen Mechanismus, sicherzustellen, dass vorangegangene Komponenten die Annotationen in einer Weise speichern, wie sie spätere Komponenten benötigen.
- Es gibt keinen Mechanismus, um das richtige Encoding der Eingabedatei sicherzustellen.

- Es ist nicht trivial, dies auf Dokumentkollektionen zu erweitern und Fehler bei der Verarbeitung einzelner Dateien zu bemerken.
- Die Wiederverwendbarkeit der einzelnen Programmlogiken ist gering, da Format und Funktionalität innerhalb von Komponenten vermischt werden.

Daher wird eine standardisierte Art der Integration empfohlen, um mit Anforderungen an sprachtechnologische Software wie Flexibilität, Robustheit, Skalierbarkeit (Abschn. 3.3.1) und effiziente Weiterentwicklung umgehen zu können, siehe dazu auch Leidner (2003). In der Praxis haben sich hierzu mehrere Vorgehensweisen etabliert. Die Modellierung als Pipeline mit ihrer starren Abfolge an Schritten birgt den Nachteil, dass sich Fehler in früheren Verarbeitungsschritten, z. B. bei der falschen Entscheidung bei Mehrdeutigkeiten, fortpflanzen, sodass rein durch dieses Modell ein Abnehmen der Genauigkeit pro Schritt zu erwarten ist. Die Vorgehensweise ist jedoch zur Beherrschung der Komplexität als alternativlos anzusehen. Entsprechende Versuche, Mehrdeutigkeiten bis zu ihrer Auflösung in späteren Schritten als Alternativen vorzuhalten (z. B. Maxwell und Kaplan 1993), sind wegen der **kombinatorischen Explosion** der Möglichkeiten und den damit einhergehenden hohen Rechen- und Speicheraufwänden trotz algorithmisch effizientem Umgang mit diesen als gescheitert anzusehen.

### Frameworks für linguistische Pipelines

Das vermutlich formalisierteste Framework für die Integration von Sprachverarbeitungskomponenten in NLP-Pipelines zu ihrer Verwendung in Text-Mining-Aufgaben ist Apache UIMA (<http://uima.apache.org/>, vgl. Ferrucci und Lally 2004), welches in den Programmiersprachen Java und C++ verfügbar ist. Kern des UIMA-Frameworks ist die CAS (Common Analysis Structure), die für jedes Dokument in der Pipeline erstellt wird und welche Zugriffe auf verschiedene Annotationen eines Dokumentes erlaubt, die durch in der UIMA-Pipeline definierten **Annotatoren** hinzugefügt und weiterverarbeitet werden. Hierbei wird der Originaltext vorgehalten, Annotationen werden durch Byte-Offsets zum Originaltext verknüpft. Die Annotationen in UIMA sind durch ein Typensystem reguliert, was eine pipelinekonforme Integration ermöglicht und eine Trennung der Programmlogik vom Speicherformat realisiert. UIMA bietet ferner Möglichkeiten zur Skalierung der Verarbeitung mit Compute-Clustern.

Während UIMA die technologische Basis des Apache Projekts OpenNLP (<https://opennlp.apache.org/>) darstellt, in großen industriellen Projekten wie z. B. IBM Watson (vgl. Ferrucci 2012) und in wissenschaftlichen Projekten wie z. B. ClearTK (vgl. Bethard et al. 2014) oder DKPro (vgl. Eckart de Castilho und Gurevych 2014) eingesetzt wird, für die eine große Anzahl an Komponenten für viele Sprachen verfügbar sind, ging die Verbreitung aufgrund der hohen Anforderungen an Rechenressourcen und der für sprachtechnologische Anwendungen im Rückgang begriffenen Programmiersprache Java in den letzten Jahren deutlich zurück. Dies gilt auch für das etwas loser gekoppelte GATE-Framework (vgl. Cunningham et al. 2013). Dennoch gelten die Prinzipien der

Modularität und Interoperabilität von Komponenten weiterhin und sollten insbesondere bei größeren Projektvorhaben Berücksichtigung finden.

Insbesondere durch die große Verfügbarkeit von Deep-Learning-Programmbibliotheken für die neuronale Verarbeitung haben sich Python-Frameworks etabliert. SpaCy (<https://spacy.io/>, vgl. Honnibal und Montani 2017) bietet für etwa ein Dutzend Sprachen Komponenten für den Aufbau von Pipelines. Auch hier wird ein Dokumentobjekt angelegt; Annotationen werden in entsprechenden Objektvariablen abgelegt, welche für anwendungsunabhängige standardisierte Vorverarbeitungsstufen Namenskonventionen folgen, wodurch die Kopplung zwischen den Komponenten erreicht wird. Weitere nennenswerte Frameworks zur Modellierung von Pipelines in Python sind das sehr viele Sprachen abdeckende Polyglot (<https://pypi.org/project/polyglot/>), sowie die eher für Unterrichtszwecke als für Produktionssoftware geeigneten Toolkits Stanford CoreNLP (<https://stanfordnlp.github.io/CoreNLP/>, vgl. Manning et al. 2014) und NLTK (<https://www.nltk.org/>, vgl. Bird et al. 2009).

### **End-to-End-Systeme**

Das Pipeline-Modell mit seiner Abbildung linguistischer Ebenen auf computer-linguistische Verarbeitungsschritte ist fest etabliert. Neuere Entwicklungen bestehen in sogenannten **End-to-End-Systemen**, welche mit Deep-Learning-Technologien entwickelt werden (Abschn. 6.10). Hier wird auf die explizite Modellierung der Einzelschritte verzichtet; ein komplexes neuronales Netzwerk soll die entsprechende Aufgabe, welche oft unmittelbar der Anwendung entspricht, direkt lernen. Hierbei werden linguistische Informationen weiterhin repräsentiert, allerdings implizit (siehe z. B. Tenney et al. 2019; Jawahar et al. 2019): In den verschiedenen Schichten des Netzwerkes konnten Strukturen nachgewiesen werden, welche ungefähr den linguistischen Ebenen bzw. den klassischen Vorverarbeitungsschritten entsprechen; der Nachteil der **Fehlerfortpflanzung** wird reduziert, da in den Schichten des Netzwerkes durch die kontinuierlichen Repräsentationen keine Entscheidungen getroffen werden, vgl. Niehues et al. (2021). Für das Text Mining bedeutet dies jedoch keineswegs, dass die linguistische Pipeline sich als Konzept überholt hat: Auf linguistischen Strukturen, egal wie diese automatisch in Texten annotiert werden, lassen sich schnell und zielführend Extraktionsregeln definieren, ohne die mitunter sehr großen nötigen Trainingsdaten erstellen zu müssen, welche für End-to-End-Systeme notwendig sind.

#### **3.2.2 Einlesen und Vorverarbeitung**

In praktischen Anwendungen liegen die Eingangsdokumente für das Text Mining oft in heterogener Form vor und müssen für die Verarbeitung erst vereinheitlicht werden. Dies beinhaltet die Konvertierung aus verschiedenen Dateiformaten und Codierungen, das Entfernen bzw. Vereinheitlichen von Formatierungen und das Erkennen der Sprache. Nach dem erfolgreichen Einlesen stehen die Texte in einer Form zur Verfügung, die es

den darauffolgenden Schritten ermöglicht, von der Form des Originaldokumentes zu abstrahieren und rein auf dem daraus extrahierten Text zu arbeiten.

Jede Verarbeitung von Dokumenten oder Dokumentsammlungen für das Text Mining beginnt also mit dem Einlesen der Texte. Der Einleseprozess soll sicherstellen, dass von den zugrunde liegenden, oftmals heterogenen Dateiformaten abstrahiert wird, sodass alle weiteren Verarbeitungsschritte auf dem reinen Text erfolgen können. Da alle weiteren Verarbeitungsschritte sprachabhängig sind, wird die Erkennung der Sprache oft als Teil des Einleseprozesses modelliert, ist jedoch streng genommen ein eigener Annotator.

Während computerlinguistische Datensätze, welche z. B. zum Trainieren von Annotatoren verwendet werden können, normalerweise schon in einheitlicher Form vorliegen, geschieht Text Mining meist auf Dokumenten in verschiedenen Formaten: In innerbetrieblichen **Content-Management-Systemen** liegen Texte oft in **Markdown** und in **Office-Formaten** wie Word, Excel und PowerPoint vor, beim Web Mining müssen Texte aus HTML-Dateien verschiedener Versionen verarbeitet werden, und PDF als einheitliches Austauschformat findet sich in fast allen Kollektionen. Im Folgenden werden exemplarisch Herausforderungen beim Einlesen von HTML und PDF besprochen, die dabei auftretenden Probleme kommen jedoch in ähnlicher Form auch in anderen Formaten vor.

**HTML (Hypertext Markup Language)** ist ein Format für Webseiten zur Darstellung im Webbrowser und basiert auf Markup-Elementen, welche die Struktur der Webdokumente definieren. Herausforderungen beim Einlesen von HTML sind:

- Uneinheitliche Verwendung von Markup-Tags: Ein Großteil der Webseiten entspricht nicht dem Standard, was durch robustes Implementieren der Anzeige seitens der **Webbrowser** aufgefangen wird.
- Uneinheitliche Verwendung von Strukturinformation, z. B. verschiedene Levels von Überschriften, Verwendung von Tabellen oder Frames zur Steuerung des Layouts.
- Uneinheitliche Kodierung: **Webseiten** liegen in verschiedenen **Codepages** vor, die Kodierung ist oft, aber nicht immer Teil der im HTML-Dokument angegebenen **Metadaten**. Diese sollten einheitlich in **UTF-8** bzw. **Unicode** überführt werden.
- Uneinheitliche Kodierung von Sonderzeichen, welche im Code verschieden aussehen, jedoch gleich dargestellt werden (vgl. Abschn. 1.2.2): Z.B. kann der u-Umlaut *ü* entweder als *ü* selbst mit der Kodierung ISO 8859–1 als ASCII-Zeichen mit der Nummer 252 und in HTML mit Dezimalcode &#252; oder Hexadezimalcode &#xfc; kodiert werden, in Unicode als *ü* selbst mit der Nummer 195.188, oder innerhalb HTML hexadezimal als &#x00fc; unabhängig von der Codierung kann zudem &uuml; verwendet werden.
- Vermischung von inhaltlichen Elementen und sogenannter Boilerplate, z. B. Menüführung, Disclaimer, Strukturelemente, Werbung etc.
- Multimedia-Elemente wie Bilder, Videos und andere Interaktionen.
- Identifikation von **Spam**: ganze Spam-Seiten, Spam in Web 2.0-Elementen wie Kommentaren etc.

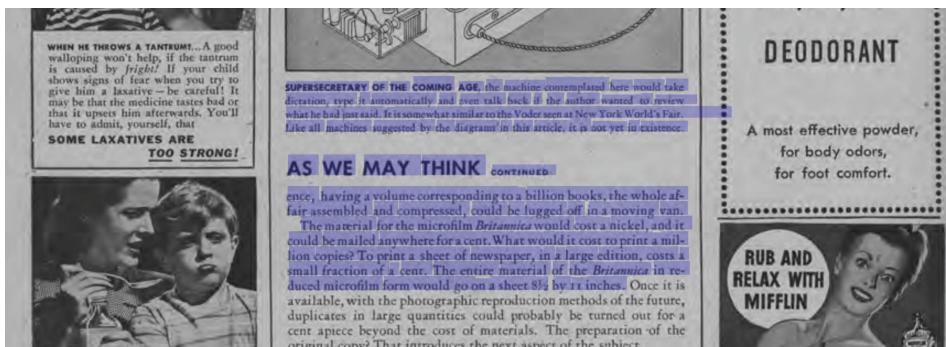
Einen Überblick über Ansätze zur Identifikation von inhaltlichen Textelementen aus Webseiten findet sich beim Cleaneval-Wettbewerb (vgl. Baroni et al. 2008).

**PDF (Portable Document Format)** wurde mit dem Ziel entwickelt, auf verschiedenen Endgeräten dieselbe Darstellung des Dokumentes zu erreichen. Ein PDF-Dokument besteht aus Objekten, welche neben Textboxen auch Bilder, Vektorgrafiken und Navigationselemente (wie z. B. verlinkte Inhaltsverzeichnisse) enthalten kann. Beim PDF-Format kann zwischen zwei für die Extraktion von Text relevanten Arten unterschieden werden, wie das sprachliche Material repräsentiert wird: Entweder als Bild, dies ist oft bei gescannten Dokumenten der Fall, oder als Text, was normalerweise bei primär elektronisch erzeugten Dokumenten vorliegt. PDFs mit Bildrepräsentation von Text können zudem von **OCR**-Software erzeugte Textdaten enthalten, welche mit den entsprechenden Stellen im Bild verknüpft sind. Während für moderne Schriftarten in hochauflösenden Scans die Qualität von OCR, z. B. mithilfe von Programmen wie Tesseract (<https://github.com/tesseract-ocr>, vgl. Smith 2007) für die Extraktion von Text aus der Bildrepräsentation ausreicht, fällt die Qualität bei historischen Dokumenten, z. B. bei Frakturschrift stark ab.

#### Beispiel zu Qualitätsproblemen bei Textextraktion von gescannten Dokumenten mit OCR

Extrahierter Text aus Markierung der Abb. 3.4 mit OCR (Ausschnitt aus Bush 1945), welcher zudem mit zu geringer Auflösung gescannt wurde:

*SUPERsEcREtARY OF THE COMING AGE, rhc mrchlnc contcmpltd hcrc  
ivould rake d~rrrrmn. rvpc~r surommc!lr and cvcn rrlk brck if rhc nurhor  
wntd ro rrr~cw hat hehad luscsud. Ir~rsomcwhar s~mil~rrochcVodrrccn%  
N~vS~rk\\'orl~'F~i~. Lik all machines suggested by the dlrgralns in rhlr zrriclc,  
ir ir not yct in cnrrcncc. AS WE MAY THINK com,ww ence, having avolume  
corresponding to a billion books, the wholcaf-fan assembled and compressed, could  
be lugged off in a moving van. Thematerial forthe microfilmBritannicawould cost a*



**Abb. 3.4** Beispiel zu Qualitätsproblemen bei Textextraktion von gescannten Dokumenten mit OCR. (Quelle: In Anlehnung an Bush 1945)

*nicke1, and it could be mailed anywhere for a cent. What would it cost to print a million copies? To print a sheet of newspaper, in a large edition, costs a small fraction of a cent. The entire material of the Briranaira in reduced microfilm form would go on a sheet 8 1/2 by 11 inches.* ◀

Folgende Probleme, viele davon illustriert im obigen Beispiel und in der Abb. 3.4, treten vornehmlich bei Bildrepräsentation in Verbindung mit OCR auf:

- Falsche Erkennung von Buchstaben, insbesondere bei Sonderzeichen und Ligaturen und bei verschiedenen Schriftarten.
- Falsche Erkennung von Leerzeichen.
- Vermischen von Texten aus verschiedenen Layoutelementen, z. B. Werbung (siehe Abb. 3.4).

Folgende Probleme bei der Konvertierung von PDF betreffen sowohl die Bildrepräsentation mit OCR als auch die Extraktion aus PDFs mit Textobjekten:

- Falsche Verarbeitung der Reihenfolge und Struktur von Spalten oder Textboxen.
- Layoutstruktur wird nicht beachtet, z. B. Überschriften, Referenzen, Tabellen, Adressfelder u.ä.
- Silbentrennung muss rückgängig gemacht werden.

Diese Probleme können je nach Homogenität der Kollektion durch **Heuristiken** bei der Konvertierung abgemildert werden; die Höhe der Vertretbarkeit des Aufwandes ist abhängig von der vorliegenden Kollektion und der Text-Mining-Anwendung. Ähnlich gelagerte Probleme treten auch mit anderen Dateiformaten auf.

Eine Programmbibliothek zum Einlesen ist Apache Tika (<https://tika.apache.org/>), welche alle gängigen Formate in eine einheitliche XML-Struktur überführt. Tika extrahiert dabei auch **Metadaten**, z. B. den Namen und das Format der Quelldatei, und vermeidet dadurch einen der häufigsten Fehler beim Einlesen: das Wegwerfen von Quelleninformation, welches zwar nicht für den Text-Mining-Prozess an sich benötigt wird, jedoch für das Transparentmachen der Quellen der extrahierten anwendungsabhängigen Informationen, was für viele Anwendungen wichtig ist.

## Erkennung der Sprache

Die Sprache der Texte im Dokument ist ein weiteres Metadatum, welches für die weiteren Verarbeitungsschritte benötigt wird. Das Identifizieren, in welcher natürlichen Sprache ein Text vorliegt, kann mit überraschend hoher Genauigkeit von 99 % oder mehr erfolgen. Hierzu können neben Sprachmodellen (Abschn. 5.5) und Textklassifikation (Abschn. 6.6) auch einfachere Mechanismen wie Buchstaben-N-Gramme oder Listen von hochfrequenten Wörtern eingesetzt werden, siehe Jauhainen et al. (2019) für einen Überblick und Abschn. 4.2.4 für den Aufbau eigener Sprachidentifizierungs-komponenten.

### 3.2.3 Segmentierung

Elektronisch verfügbare Texte sind als Zeichenketten repräsentiert. Viele Komponenten der linguistischen Pipeline nehmen an, dass diese Zeichenketten bereits in Wörter und Sätze zerlegt wurden. In diesem Abschnitt werden Vorgehensweisen zur Textsegmentierung in Einzelsätze und zur Tokenisierung in Wörter vorgestellt.

Ziel der Segmentierung ist die Erkennung von zusammengehörenden Einheiten auf Wort-, und Satzebene, da alle weiteren Verarbeitungsschritte die Zerlegung von Texten in Wörter (Abschn. 1.3.2) voraussetzen, syntaktische Verarbeitung (Abschn. 3.2.4) benötigt zudem auch die Ebene des Satzes. Das Erkennen von thematischen Abschnitten dagegen ist eine semantische Aufgabe und ist daher in späteren Schritten (Abschn. 3.2.5) verortet.

Die Zerlegung von Texten in Sätze und die Zerlegung von Sätzen in Wörter ist algorithmisch komplizierter als es auf den ersten Blick erscheint. Die von den üblichen Regeln abweichenden Fälle werden mittels musterbasierter Verfahren (Abschn. 3.1.1) und Ausnahmelisten bearbeitet. Es gibt Zeichen, die Wörter trennen, wie beispielsweise das Leerzeichen. Eine ähnliche Rolle spielen der Tabulator und ein Zeilenumbruch. Weil man alle diese Zeichen nicht direkt sieht, werden sie unter dem Begriff **Whitespace** zusammengefasst.

Während in den meisten Fällen die Segmentierung von Texten in Sätzen und von Sätzen in Wörter korrekt möglich ist, zeigt es sich, dass es auch schwierige Ausnahmefälle gibt, bei denen allgemeine Verfahren versagen. Bis zu einem gewissen Grad können diese Fälle jedoch durch zusätzliche Regeln und Ausnahmelisten erfasst werden.

Analog lassen sich die bei der Tokenisierung zu erfassenden Wörter angeben durch eine relativ einfache strukturelle Beschreibung sowie die Hinzunahme einzelner ausgewählter Wörter (beispielsweise mit weiteren Sonderzeichen), die anderenfalls zurückgewiesen würden.

#### Regeln für die Textsegmentierung

Begonnen werden soll mit der simplen Beobachtung, dass an der Trennstelle zweier Sätze der erste Satz aufhört und der zweite Satz beginnt. Deshalb werden zunächst einfache Regeln formuliert, die Satzanfänge bzw. Satzenden charakterisieren und deshalb genutzt werden können, um für eine bestimmte Stelle im Text zu entscheiden, ob dort die Trennstelle zwischen zwei Sätzen vorliegt.

Zunächst einige einfache Regeln für den Satzanfang:

- Sätze beginnen mit Großbuchstaben oder Ziffern.
- Nach einer Überschrift beginnt ein neuer Satz.
- Am Anfang eines Absatzes beginnt ein neuer Satz.
- Groß geschriebene Artikel (wie *Der, Die, Ein, ...*) sprechen für einen Satzanfang.
- Beginnt kein neuer Absatz, so steht vor dem neuen Satz ein Satzendezeichen, gefolgt von einem Leerzeichen.

Analog gibt es einige einfache Regeln für das Satzende:

- Sätze enden mit einem Satzendezeichen. Solche Zeichen sind Punkt, Fragezeichen und Ausrufezeichen. Nach dem Satzendezeichen muss zusätzlich ein Whitespace (meist ein Leerzeichen, s. u.) stehen. Punkte können jedoch auch an anderer Stelle stehen, z. B. nach Abkürzungen oder Zahlen.
- Vor einer Überschrift endet ein Satz.
- Am Ende eines Absatzes endet ein Satz.

Es bleiben einige schwierige Fälle. Bei einem Punkt nach einer Ziffer oder bei einem Punkt als Bestandteil einer Abkürzung können wir ohne einen Blick auf die folgenden Wörter nicht einfach entscheiden, ob hier ein Satzende vorliegt: Falls eine Abkürzung am Ende eines Aussagesatzes steht, werden korrekterweise nicht zwei Punkte, sondern nur ein Punkt geschrieben.

#### Beispiele für schwierige Fälle bei der Erkennung des Satzendes durch Wörter, die auf „.“ enden

*Er trägt den Titel Dr. rer. nat.*

*Seit einem halben Jahr gehört Dr. rer. nat. Stefan Schlatt dazu.*

*Sein Glückstag ist Freitag der 13.*

*Gestern war es wieder soweit: Freitag der 13. März.* ◀

Kein Satzende liegt sicher vor, wenn das folgende Wort mit Kleinbuchstaben beginnt. Analog liegt vermutlich ein Satzende vor, wenn danach ein Artikel in Großbuchstaben folgt. Diese beiden Regeln helfen aber nicht in den eben genannten Beispielen.

Weitere Schwierigkeiten bereiten Beispiele mit wörtlicher Rede, siehe folgendes Beispiel.

#### Schwierigkeiten bei der Satzsegmentierung in direkter Rede

*„Ich kann es hören! Es kommt immer näher“, rief er entsetzt.*

Handelt es sich hierbei um einen Satz oder um mehrere Sätze? Liest man dieses Textstück wie gewöhnlich von links nach rechts, so findet man nach dem Ausrufezeichen ein Satzende. Nach den Ausführungszeichen wird kein Satzende gefunden, weil es danach mit Kleinbuchstaben weitergeht.

Damit wird das Textstück in die folgenden zwei Sätze zerlegt:

*„Ich kann es hören!*

*Es kommt immer näher“, rief er entsetzt.*

Beide Sätze sind von ihrer Form her unbefriedigend, weil sie isolierte (also nicht paarweise auftretende) An- bzw. Ausführungszeichen enthalten. Im Falle des ersten Satzes lässt sich das mit einem Nachbearbeitungsschritt leicht lösen: Ist das erste

oder letzte Zeichen eines Satzes ein solches isoliertes An- oder Ausführungszeichen, so kann es einfach entfernt werden. Im Falle eines isolierten An- oder Ausführungszeichens mitten im Satz, gibt es keine vergleichbar elegante Lösung. ◀

Ein weiteres Problem besteht darin, dass der Algorithmus natürlich auch Dinge in einzelne Sätze zerlegen will, die gar keine Sätze enthalten. Bei der Konvertierung, z. B. aus Webdokumenten, entstehen gelegentlich scheinbar extrem lange Sätze dadurch, dass Listen oder Programmcode wie JavaScript nicht als solche erkannt werden. Dann werden möglicherweise alle Zeichen bis zum nächsten Punkt als sehr langer Satz interpretiert. Einfache Abhilfe ist hier möglich, indem man eine maximale Satzlänge z. B. auf 255 Zeichen festlegt und längere Objekte einfach ignoriert. Dies hat aber zur Folge, dass auch korrekte Sätze mit einer Länge oberhalb dieser Schranke zurückgewiesen werden. Bei der Wahl der maximalen Satzlänge ist also zu beachten, ob der Verlust einiger langer Sätze akzeptabel ist.

### **Umgang mit Abkürzungen**

Um Satzgrenzen nicht mit Abkürzungen zu verwechseln, müssen die Abkürzungen als solche erkannt werden. Interessant sind hier nur Abkürzungen, die mit einem Punkt enden, da sonst keine Verwechslung mit einem Satzende möglich ist.

### **Abkürzungslisten**

Die einfachste Möglichkeit ist eine Liste von Abkürzungen, die dann als solche erkannt werden.

Häufige deutsche Abkürzungen werden in Tab. 3.3 angegeben.

Mit Abkürzungslisten lassen sich häufige Abkürzungen sehr effektiv behandeln. Schwierigkeiten bereiten aber viele Abkürzungen, die nach einem festen Schema gebildet werden wie beispielsweise Straßen- oder Gesellschaftsnamen. Hier benötigen wir Muster der Art „\*str.“ oder „\*ges.“ um die Abkürzungen *Bahnhofstr.* bzw. *Handelsges. mbH* zu erkennen.

### **Segmentierung von Sätzen in Wörter: Tokenisierung**

Verglichen mit der Segmentierung von Texten erscheint die Zerlegung eines Satzes in Wörter noch einfacher. Im Deutschen sind die Wörter eines Textes durch Leerzeichen getrennt (anders als z. B. im Chinesischen). Also zerlegen wir einfach einen Satz an den Leerzeichen und erhalten die Wörter dieses Satzes. Eventuell müssen wir nach der Trennung bei den Leerzeichen noch Satzzeichen wie Punkt, Komma und Anführungszeichen entfernen.

Diese Trennung an Whitespace, wie z. B. am Leerzeichen, ist im Prinzip richtig, es gibt aber wieder einige Stellen, an denen es zu unerwünschten Effekten kommt. Dies ist eher ein linguistisches als ein informatisches Problem, denn es ist nicht so einfach, den Begriff des Tokens (vgl. Abschn. 1.2) operationalisierbar zu definieren.

**Tab. 3.3** Häufige deutsche Abkürzungen

Rang	Abkürzung
1	Mio.
2	Tel.
3	Dr.
4	bzw.
5	Nr.
6	Co.
7	Mill.
8	u.
9	Sa.
10	a.
11	Prof.
12	u. a.
13	Str.
14	Kl.
15	ca.
16	e. V.
17	z. B.
18	z. B.
19	Bekl.
20	St.

Oft wird von folgender Beschreibung ausgegangen: **Tokens** sind die in einem grammatisch und orthographisch korrekten Text stehenden Zeichenketten bestehend aus Buchstaben und Bindestrich, die im Text durch Whitespace oder Satzzeichen getrennt sind.

Dass dies nicht unbedingt so sein muss, zeigen die folgenden Beispiele:

#### Beispiele für schwierige Tokenisierung

*Solche abgelegenen Airports haben Vor- und Nachteile.*

*Die Prinz zu Hohenlohe-Jagstberg & Banghard Beratungs GmbH & Co Vermittlungs-KG, Sitz Berlin, ist als persönlich haftende Gesellschafterin eingetreten.*

*Ab Sonntag, 20.1, präsentiert der PRO 7-Moderator Aiman Abdallah sein neues Format Galileo The Game, ein Quiz über Wissen, Logik und Strategie.*

Im ersten Beispiel ist *Vor-* kein vollständiges Wort, sondern nur in der Fügung *Vor- und Nachteile* verständlich.

Im zweiten Beispiel ist *Beratungs* kein wohlgeformtes Wort, sondern nur ein Namensbestandteil.

Im dritten Beispiel schließlich ist *PRO 7-Moderator* ein Wort, welches ein Leerzeichen enthält. Bei einer weiteren Trennung an dem Leerzeichen wird der Eigenname *PRO 7* getrennt und man erhielt ungünstigerweise *7-Moderator* als Wort. ◀

### Struktur von Wörtern

Schwierigkeiten bereiten auch wortähnliche Objekte, die aber keine eigentlichen Wörter sind.

Beispiele hierfür sind Internetadressen und Dateinamen, wissenschaftlich-technische Abkürzungen wie chemische Formeln, mathematische Ausdrücke u. a.

Hier können die obigen Zerlegungsalgorithmen zu Wörtern führen, die nicht weiter berücksichtigt werden sollen. Deshalb ist es an dieser Stelle sinnvoll zu prüfen, ob die nach der Wortseparierung erhaltenen Objekte aus einer Menge von erlaubten Zeichen bestehen, d. h. auf den ersten Blick wie Wörter aussehen.

Für die deutsche Sprache sind die großen und kleinen Buchstaben von a bis z erlaubt, dazu die Umlaute Ä, Ö, Ü, ä, ö, ü und ß und je nach Art der gendergerechten Sprache der Unterstrich \_, der Doppelpunkt:, der Asterisk \* oder das i mit Tremazeichen ï. Weiterhin werden der Bindestrich und zumindest noch das é (z. B. für *Café*) gebraucht, in fremdsprachlichen Namen auch andere Diakritika. Möglicherweise sollen auch Ziffern in Eigennamen zugelassen werden (z. B. *Audi A4*). Auch Großbuchstaben folgend auf Kleinbuchstaben sind selten möglich (*pH-Wert*), ebenso wie eigentliche Satzendezeichen als Teile von Eigennamen, z. B. *Yahoo!* oder *Ver.di-Bundesvorstand*.

### Andere Sprachen

Bei anderen Sprachen als dem Deutschen gibt es ähnlich gelagerte Komplikationen bei der Tokenisierung. Neben dem offensichtlich notwendigen Austausch der Abkürzungslisten und den erlaubten Buchstaben werden hier auch andere nicht alphanumerische Zeichen innerhalb von Texten verwendet, welche bei der Tokenisierung entsprechend sprachabhängig behandelt werden müssen, insbesondere diakritische Zeichen. Während hier nicht auf alle Sprachen im Einzelnen eingegangen werden kann, illustriert das folgende Beispiel einige der zu adressierenden Phänomene.

#### Herausforderungen bei der Tokenisierung in anderen Sprachen

Spanisch: *¿Cuál fue la mejor década musical?*

Französisch: *J'attend mon avocat pour le dîner aujourd'hui.*

Englisch: *I simply can't owe you 30–40,000 bucks!*

Koreanisch: *IBM 노트북은 이제 Lenovo라고 합니다.*

Japanisch: *昼食には寿司とご飯があります。*

Türkisch: *Muvaffakiyetsizleştirileceklilerimizdenmişsinizcesine.*

Das Spanische endet Ausrufe- und Fragesätze wie das Deutsche mit den entsprechenden Zeichen, beginnt sie jedoch mit den um 180 Grad rotierten Zeichen.

Der Apostroph im Französischen ist mitunter Teil des Wortes (*aujourd’hui = heute*), mitunter ein Auslassungszeichen (*J’attend = Je attend = ich erwarte*). Im Englischen werden Kontraktionen wie *can’t* gern in *can not* oder in *can n’t* tokenisiert, Tausendertrennzeichen werden mit „“ und nicht mit „“ notiert. Das Koreanische hat wie viele andere Sprachen einen anderen Zeichensatz. Im Gegensatz zum Japanischen und Chinesischen wird hier Whitespace als Trennzeichen von Wörtern eingesetzt; Eigennamen werden im europäischen Zeichensatz wiedergegeben, welche jedoch mit koreanischen Flexionen versehen werden (*Lenovo라고합니다 = werden Lenovo genannt*), sodass der Zeichensatz mitten im Wort wechselt. Japanisch transliteriert fremdsprachliche Eigennamen mit einem eigenen Satz Zeichen, den Katakana, und nutzt andere Satzendezeichen. Im Türkischen, welches Whitespace verwendet, ist es durch Agglutinieren möglich, gesamte Sätze in einem Wort auszudrücken (im Beispiel: *Als ob Sie zufällig zu denen gehören, die wir nicht einfach zu Schöpfern von Unerfolgreichen machen könnten*). Hier verschwimmen die Grenzen von Tokenisierung und morphologischer Analyse, vgl. Abschn. 1.3.1. ◀

Die Tokenisierung ist der erste Schritt bei der Vorverarbeitung, und daher der wichtigste: Da die meisten weiteren Annotatoren auf Tokenebene operieren, schlagen Fehler in der Tokenisierung unmittelbar auf alle weiteren Schritte durch. Daher ist bei der Tokenisierung auf besondere Vorsicht zu achten: Einmal festgelegte Tokendefinitionen können nicht geändert werden, ohne alle folgenden Verarbeitungsschritte anzupassen. Beim Einbinden von Verarbeitungskomponenten, welche auf einer anderen Tokenisierung basieren, muss ggf. de-tokenisiert und re-tokenisiert werden; hier ist es von Vorteil, wenn in der Pipeline der untokenisierte Text vorgehalten wird, wie z. B. in der UIMA CAS (siehe Abschn. 3.2.1). Ein üblicher Fehler bei der Tokenisierung ist z. B. eine Normalisierung in Kleinbuchstaben und von Sonderzeichen – dies ist insbesondere im Deutschen nicht zu empfehlen – oder ein Ignorieren von Satzzeichen. Während manche Verfahren wie z. B. Topic-Modelle (Abschn. 6.4) hierunter i. Allg. nicht leiden, kann diese Normalisierung in vielen Fällen die Mehrdeutigkeit erhöhen oder die Bedeutung verändern, siehe unteres Beispiel.

#### Beispiele von Übernormalisierung, welche zu Bedeutungsveränderung bzw. Mehrdeutigkeiten führen

„Trinke in Maßen!“ vs. „Trinke in Massen!“

„Komm, wir essen, Oma!“ vs. „Komm wir essen Oma!“

„Der Junge sieht dir Ungeheuer ähnlich“ vs. „der junge sieht dir ungeheuer ähnlich“ ◀

Ein alternativer Ansatz zur Tokenisierung ist die im WordPiece bzw. SentencePiece (vgl. Kudo und Richardson 2018) und z. B. in Sprachmodellen wie BERT (siehe Abschn. 3.1.3) verfolgte Strategie, die Vokabulargröße durch einen Parameter zu

limitieren und zu einer **Subwort-Tokenisierung** überzugehen: Hier werden im Prinzip die Frequenzen von Buchstaben-N-Grammen ermittelt, die häufigsten bilden das Vokabular. Da seltene lange N-Gramme aus häufigen kürzeren N-Grammen bestehen, ergeben sich Verarbeitungseinheiten, welche ggf. nur Teile von Wörtern umfassen; hier wird die linguistische Gleichsetzung von Wörtern und Tokens technisch aufgebrochen, was ggf. bei der Interpretation der Annotationen auf diesen Einheiten Schwierigkeiten verursacht. Auf der anderen Seite wird dadurch jedoch auch das Problem der seltenen Ereignisse in der Sprache (vgl. Abschn. 5.2) verringert, da die Token-Ereignisse nicht mehr beliebig selten sind: Auch bisher ungesehene Wörter können problemlos verarbeitet werden, da sie durch mehrere Buchstaben-N-Gramme repräsentiert werden, welche oftmals Morphemen (Abschn. 2.2) entsprechen. Eine feste Vokabulargröße ist insbesondere auch für End-to-end-Systeme (siehe Abschn. 3.2.1) interessant, da nur für Vokabulareinträge Embeddings gelernt und vorgehalten werden müssen und die interne Repräsentation ohnehin nicht transparent gemacht wird.

### 3.2.4 Morphologische und Syntaktische Verarbeitung

Soll die Verarbeitung von Textinhalten über das reine Zählen von Worthäufigkeiten hinausgehen, ist es unerlässlich, die syntaktische Struktur von Sätzen zu erkennen, um mithilfe von Mustererkennung Zusammenhänge zu extrahieren. Syntaktische Verarbeitung erstreckt sich je nach Tiefe der notwendigen Verarbeitung auf das Erkennen von Wortarten und der Lemmatisierung bis hin zum Parsen des Satzes mit einer Grammatik, was ggf. morphologische Verarbeitung voraussetzt. In diesem Abschnitt werden neben Sequenz-Tagging-Ansätzen zur Wortartenerkennung auch Parsing mit Dependenz- und Konstituentengrammatik behandelt.

Während in der Linguistik die Ebenen der Morphologie und Syntax unterschieden werden und Syntax sich im Allgemeinen auf das Analysieren von Satzstrukturen bezieht, werden bei Verarbeitungskomponenten für das Text Mining gern die morphologische und syntaktische Verarbeitung zusammen erledigt, da das Eine Voraussetzung für das Andere ist. Außerdem ist das Verständnis von Syntax ein anderes: In Vorverarbeitungskomponenten finden sich oft deutlich flachere und oberflächlichere Repräsentationen von Syntax (z. B. POS-Tagging, Chunking, s. u.) als in der Linguistik z. B. bei Chomsky (1957), siehe auch Kap. 2. Der Fokus liegt im Text Mining weniger auf interessanten Modellierungsphänomenen, sondern auf anwendungsbezogener Operationalisierbarkeit.

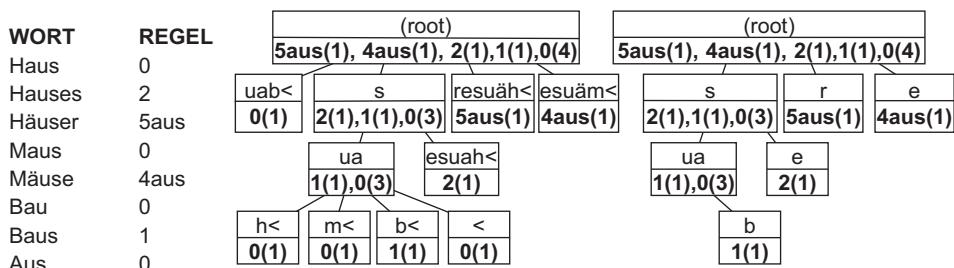
#### 3.2.4.1 Grundformreduktion und Stammformreduktion

Beim Text Mining interessieren wir uns eher für die inhaltlichen Konzepte, und nicht so sehr für die linguistische Struktur von Text. Ein erster Schritt bei dieser Abstrahierung ist die Abbildung von flektierten Wörtern, den sogenannten **Wortformen**, auf ihre **Grundform** (Ergebnis der Lemmatisierung) oder ihren **Stamm** (Ergebnis von **Stammformreduktion**), siehe auch Abschn. 2.2.2.

Bei der Lemmatisierung werden Nomen auf ihre Form im Nominativ Singular reduziert (z. B. *Baumes*, *Bäume*, *Bäumen*, ... auf *Baum*), Verben auf Infinitive (z. B. *flog*, *geflogen*, *fliegt*, ... auf *fliegen*) und Adjektive auf die prädiktive Form (z. B. *schöne*, *schönes*, *schönste*, ... auf *schön*). Dies kann mit einer morphologischen Analyse einhergehen (vgl. Abschn. 3.1.1, Beesley und Karttunen 2003). Um dies zu realisieren, ist ein rein wörterbuchbasierter Ansatz, bei dem für alle Vollformen die Grundformen in einer Liste vorgehalten werden, wegen der Produktivität des Vokabulars nicht zielführend; andererseits werden auch Neologismen regelmäßig flektiert, wie z. B. *googeln*, *googelte*, *googelnd*, *gegoogelt*. Da im Deutschen, wie in den meisten anderen Sprachen, Flektion und Derivation am Ende des Wortes durchgeführt wird, reicht es, lediglich die Wortenden zu betrachten und geeignete Transformationsregeln aufzustellen, welche die Vollformen in Grundformen überführen; hierbei ist auf ausreichende Mächtigkeit des Regelwerkes zu achten, damit auch unregelmäßige Formen korrekt behandelt werden. Für die Grundformreduktion des Deutschen können Regeln bestehend aus einer Zahl und einem ggf. leeren String folgendermaßen definiert werden: Die Zahl gibt an, wie viele Buchstaben am Ende des Wortes gelöscht werden, der String gibt an, was am Ende des resultierenden Wortes angehängt werden muss. Hiermit können sowohl regelmäßige (wie z. B. 1en für *schreibt*) als auch unregelmäßige (wie z. B. 2iegen für *lag*) Flexionen abgedeckt werden, siehe Abb. 3.5 (links) für weitere Beispiele.

Ein effizienter Ansatz zur Grundformreduktion, wie er auch in der ASV-Toolbox (vgl. Biemann et al. 2008) realisiert ist, nutzt die Datenstruktur der Compact Patricia Tries (vgl. Morrison 1968). Diese Wort-Suchbaum-Datenstruktur, illustriert in Abb. 3.5 (links), komprimiert Wortanfänge bzw. Wortenden bei umgekehrter Buchstaberenienfolge, Knoten enthalten die jeweiligen Buchstabensequenzen der abgelegten Wörter. Die Regeln werden zusammen mit ihrer Häufigkeit auf dem Pfad des Wortes von der Wurzel bis zum Wortanfangszeichen abgelegt. In der Variante in Abb. 3.5 (Mitte) gibt es für jedes eingefügte Wort einen Blattknoten, die Liste mit Regeln wird verlustfrei abgelegt.

In der Abb. 3.5 sind Beispiel-Wortliste mit Reduktionsregeln (links), Compact Patricia Trie (Mitte) und geprunter Compact Patricia Trie (rechts) für die Grundformreduktion einiger Nomen dargestellt. Für das Beispiel *Passus* wird im geprunten Baum



**Abb. 3.5** Compact Patricia Trie zur Grundformreduktion

zunächst der Knoten mit dem Inhalt *s* unter der Wurzel gewählt, für das Restwort *Passu* dann der Knoten mit dem Inhalt *ua*. Da schon das *a* als zweiter Buchstabe im Knoten nicht mit dem vorletzten Buchstaben des Restwortes übereinstimmt, wird hier mit Frequenz 3 häufigste Regel 0 zurückgeliefert. „<“ markiert hier das Wortanfangszeichen.

Bei der Grundformreduktion wird der Suchbaum von der Wurzel aus mit dem Suchwort buchstabenweise rückwärts durchlaufen, bis kein Tochterknoten mehr gefunden wird. Die in diesem Knoten am häufigsten abgelegte Regel wird anschließend für die Grundformreduktion angewendet. Für diesen Anwendungsfall muss der Suchbaum jedoch nicht die Liste mit Regeln verlustfrei speichern, es ist folgendes Pruning möglich: Tochterknoten, welche nur die im Vaterknoten häufigste Regel enthalten, können ebenso entfernt werden wie alle Inhalte der Blattknoten, welche über einen einzigen Buchstaben hinausgehen.

Vorteile dieser technischen Realisierung der Grundformreduktion ist, dass zum einen die Regeln der Eingabeliste zu 100 % memorisiert werden, zum anderen die zum Wortende passendste Regel zur Reduktion ausgewählt wird, wenn das zu reduzierende Wort nicht in der Liste enthalten ist. Ein Nachteil besteht in der Notwendigkeit der Erstellung der Eingabeliste.

**Stemming**, auch **Stammformreduktion** ist ein weiterer Ansatz zum Vereinheitlichen von Vollformen, welcher algorithmisch definiert wird; es ist keine Trainingsliste notwendig. Im Gegensatz zur Grundformreduktion ist das Ergebnis des Stemmings häufig nicht ein korrektes Wort in der Sprache, sondern lediglich der bedeutungscharakterisierende Teil des Wortstammes: z. B. reduziert der Snowball-Stemmer (vgl. Porter 2001) *Verteidiger*, *Verteidigern*, *verteidigen*, *verteidigende* sämtlich auf den Stamm *verteid*. Diese Wörter können dann nicht mehr unterschieden werden, was ein Fall von sogenanntem Overstemming darstellt. Im Gegensatz dazu wird die *Verteidigerin* auf sich selbst reduziert, während *Verteidigerinnen* in *verteidigerinn* umgeformt wird – ein Fall von sogenanntem Understemming. Stemming hat trotz der verstümmelt anmutenden Ergebnisse eine sinnvolle Verwendung im Information Retrieval: Wird Stemming lediglich hinter den Kulissen eingesetzt, um für gestemmte Anfragen gestemmte Ergebnisdokumente zurückzuliefern, wird der Recall der Ergebnismenge erhöht. Da Stemming jedoch keine Wortartinformation benötigt, ist die Reduktion rechenzeiteffizienter als die Grundformreduktion.

### 3.2.4.2 Tagging mit Wortarten

Das Zuweisen von Wortarten (engl. **Part-of-speech-Tagging (POS-Tagging)**) im Kontext ist ein wichtiger Vorverarbeitungsschritt, welcher einerseits für weiterführende syntaktische Verarbeitung wie **Parsing** (s. u.) benötigt wird, andererseits lassen sich schon auf Sequenzen von Wortarten Extraktionsregeln für die Informationsextraktion und andere Aufgaben (siehe Abschn. 4.4, Kap. 7) definieren.

Während die meisten Wörter im Vokabular unabhängig vom Kontext immer dieselbe Wortart haben, gibt es auch Wörter, deren Wortart nur in Abhängigkeit der Wortarten in ihrer Umgebung bestimmt werden kann. Abb. 3.2 (Abschn. 3.2.1) illustriert bereits das

POS-Tagging. Im Beispiel *Frau Burg verteidigte auch die verteidigte Burg* erscheint *Burg* sowohl in der Rolle als Eigenname als auch als Nomen, *verteidigte* ist sowohl Verb mit Grundform *verteidigen* als auch ein Adjektiv mit Grundform *verteidigt*, die Wortarten sind also für diese Types mehrdeutig. Frühe Ansätze zum POS-Tagging, z. B. Brill (1992) nahmen ein Wörterbuch mit Wortarteninformation als Ressource an und disambiguieren durch kontextabhängige Regeln: In unserem Beispiel kann das Adjektiv in *die verteidigte Burg* erkannt werden, da es nach einem Artikel *die* und vor einem Nomen *Burg* steht, was ein sehr ungewöhnlicher Kontext für ein Verb wäre. Moderne POS-Tagger werden mit Verfahren des Sequenztaggings (siehe Abschn. 6.8) von annotiertem Trainingstext mit Ansätzen wie Conditional Random Fields und LSTM-Architekturen gelernt. Im Trainingstext ist auch das **Tagset**, also die Menge an Wortarten, festgelegt.

Während die Existenz von Wortarten unstrittig ist, ist die Wahl des Tagsets abhängig von der intendierten Verwendung: Beispielsweise wurde das englischsprachige Brown-Corpus (vgl. Francis und Kučera 1967) manuell mit einem Tagset aus 87 Tags versehen; da dieses Tagset sich für die Verwendung im syntaktischen Parsen zu fein-granular erwiesen hat, etablierte sich für englischsprachige Baumbanken (s. u.) das Penn-Treebank-Tagset mit 45 Tags (vgl. Santorini 1995). Tagsets sind linguistisch motiviert, bei ihrer Ausprägung fließen jedoch oftmals pragmatische Überlegungen ein. So unterscheiden englische Tagsets oftmals Singular- und Pluralnomen, da diese in der englischen Morphologie einfach durch ein angehängtes Pluralmorphem „s“ unterschieden werden können; für morphologisch produktivere Sprachen werden durch Kombination von morphologischen Markern, welche in diesem Fall dem Tagset zugerechnet werden, Tagsetgrößen von deutlich über 1000 erreicht, z. B. für das Tschechische (vgl. Hajič und Hladká 1998). Für das Deutsche ist das STTS-Tagset (vgl. Schiller et al. 1999) mit 54 Tags das am weitesten verbreitete Wortartenschema; es existieren Varianten wie z. B. Erweiterungen für Social Media Texte (vgl. Beißwenger et al. 2015). Zur Vereinheitlichung der verschiedenen Tagsets wurde im Rahmen der Universal Dependencies Initiative (vgl. Nivre et al. 2016) ein POS-Tagset von 17 Tags erstellt (<https://universaldependencies.org/u/pos/all.html>), es existieren bereits annotierte Datensätze für über 90 Sprachen.

### 3.2.4.3 Chunking

**Chunking** ist das Zusammenfassen von grammatisch zusammenhängenden Phrasen, sogenannten **Chunks**, welche Konstituenten von Sätzen bilden. Im Gegensatz zu einer strukturierteren Verarbeitung mit Dependenz- oder Konstituentengrammatiken (s. u.) ist Chunking ein flacher Verarbeitungsschritt, welcher lediglich eine weitere Ebene einführt, um Phrasen für die weitere Verarbeitung erkennen zu können, daher wird es auch als *shallow parsing* bezeichnet. Die Automatisierung erfolgt wie beim POS-Tagging durch Sequenz-Tagging-Verfahren (s. o. und Abschn. 6.8). Genauso wie über POS-Tags können auch über Chunks Extraktionsregeln definiert werden. Ein im Text Mining wichtiges Konzept sind die NP-Chunks, welche Nominalphrasen repräsentieren, da in Nominalphrasen die Dinge ausgedrückt werden, welche Thema des Texts sind. Daher ist Chunking

ein sinnvoller Schritt bei Anwendungen wie Terminologieextraktion (Abschn. 7.1) oder der Extraktion von Taxonomien.

In der Abb. 3.6 sind Sätze mit POS- und Nominalphrasen-Chunkingannotation dargestellt. Mit Mustern über Chunks und Wörter, wie z. B. durch einen regulären Ausdruck definiert als `_NP ("wie" | "außer") ("z. B." | "zum Beispiel")?` `_NP (COMMA _NP)* (CONJ _NP)?`, können Ober-Unterbegriff-Relationen (*ist von Art*, engl. *is-a*) wie (*freilaufende Hühner*, *wilde Tiere*) extrahiert werden (vgl. Hearst 1992).

### 3.2.4.4 Syntaxparsing

Manche Anwendungen benötigen eine tief greifendere syntaktische Analyse als sie die bisher beschriebenen Techniken leisten, beispielsweise im Bereich Informationsextraktion (IE, Jiang 2012). Hier ist es bei der Ausfüllung von IE-Templates wie Joint Ventures, Terrorismusereignissen oder Managementwechseln (Grishman und Sundheim 1996) wichtig zu erkennen, welche Subjekt-Prädikat-Objekt-Beziehungen vorliegen und wie sich die semantischen Beziehungen in syntaktischen Relationen widerspiegeln. Im Folgenden wird Dependenzparse als beliebter Ansatz im Text Mining besprochen, danach folgt eine kurze Zusammenstellung anderer Ansätze zum syntaktischen Parsen.

Dependenzgrammatik (vgl. Tesnière 1959; Nivre 2009, siehe Abschn. 2.1.1) modelliert die funktionalen Abhängigkeiten, die **Dependenzen**, zwischen den Wörtern eines Satzes mit gerichteten, nach grammatischer Funktion getypten sog. Head-Modifier-Relationen. Hierbei wird ausgedrückt, welches Wort in welcher Rolle im Satz auftritt. Das Verb ist die bedeutungstragende Einheit für die gesamte Aussage, das (logische im Sinne der Aussagenlogik) Prädikat des Satzes. Durch das Verb als Head vorgegeben sind Rollen der Dependenzen, wie direkte (Akkusativ) und indirekte (Dativ)

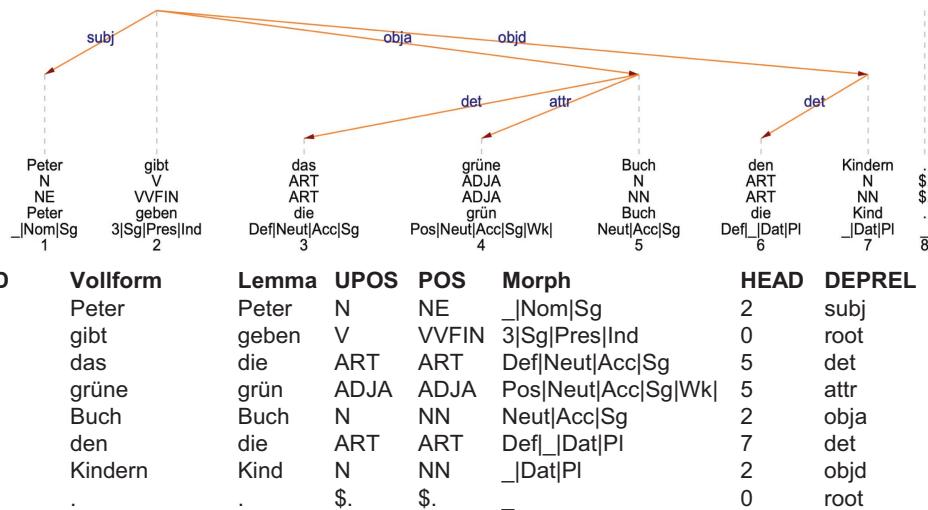


**Abb. 3.6** NP-Chunking für Taxonomieextraktion

Objekte, ausgedrückt durch Nominalphrasen, wie z. B. in *Peter gibt das grüne Buch den Kindern*. In diesen ist wiederum das hauptbedeutungstragende Nomen das Head, Modifier sind Adjektive (*grüne*) oder Artikel. Durch die Einschränkung, dass jedes Wort nur ein Head haben kann, ergibt sich eine Baumstruktur; das Head des Hauptverbes ist die Wurzel. Abb. 3.7 zeigt den Dependenzparse für unser Beispiel „*Peter gibt das grüne Buch den Kindern*.“ in graphischer Darstellung mit ParZu (<https://pub.cl.uzh.ch/demo/parzu/>) und tabuliertem CONLL-U Format (<https://universaldependencies.org/format.html>) mit Lemma, morphologischer Analyse und POS Tags für zwei Tagsets. *Peter* ist Subjekt von *gibt*, welches durch ein Akkusativobjekt *Buch* und ein Dativobjekt *Kindern* modifiziert wird.

Die Einschränkung auf einen Baumgraphen führt zu Einschränkungen bei der Repräsentation der logischen Struktur: Bei Konjunktionen wie „*Die Katze und die Kätzchen schnurrten und schliefen in dem Körbchen ein*“ müssen die beiden auf gleicher logischer Ebene befindlichen Prädikate der Verben *schnurrten* und *einschlafen* einander untergeordnet werden, ebenso sind logisch sowohl Mutterkatze als auch ihre Kinder Argumente beider Verb-Prädikate, jedoch dürfen sie im Formalismus nur von einem Verb abhängen. Die Modellierung solcher und anderer Konstruktionen wird als Konvention in der jeweiligen Grammatik festgelegt.

Um Text für die semantische Verarbeitung (Abschn. 3.2.5, Abschn. 6.8 Distributional Semantics) besser nutzbar zu machen, dienen Collapsed Dependencies (vgl. Marneffe et al. 2006 für Englisch; Ruppert et al. 2015 für Deutsch): Zum einen werden zusätzliche Dependenzrelationen eingefügt, die ggf. unter Verletzung der Baumeigenschaft die



Peter gibt das grüne Buch den Kindern.

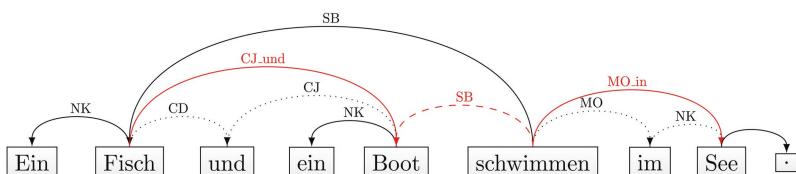
**Abb. 3.7** Dependenzparse als Baumstruktur und im tabulierten Format

logische Struktur explizit abbilden; zum anderen werden Präpositionen in der Relation modelliert, um bei Präpositionalphrasen wie *in dem Körbchen* eine direktere Verbindung zwischen Prädikat und dem Nomen der Phrase herzustellen. In der Abb. 3.8 wird verdeutlicht, wie diese Regeln die Modellierung von Repräsentationen ermöglichen, welche die Semantik des Satzes akkurater widerspiegeln: Zusätzlich zur Baumstruktur des Dependenzpareses (schwarze Linien) werden neue rote Beziehungen eingefügt, so ist Boot nun Subjekt (SB) von schwimmen, der See ist dazu in einer Relation MO\_in mit in. Dies ermöglicht es, direkt logische Prädikate wie *Schwimmt\_in(Fisch, See)* und *Schwimmt\_in(Boot, See)* abzulesen.

Auch auf dieser Ebene automatischer syntaktischer Verarbeitung vereinheitlicht das Universal Dependencies Projekt (vgl. Nivre et al. 2016) die Verarbeitung multilingualer Daten, da die nachgelagerte Verarbeitung, z. B. die Extraktion von Statements, nicht einzeln für jedes der sprachabhängig verschiedenen Tagsets definiert werden muss.

Für das Trainieren von Dependenzgrammatik existieren Baumbanken; man bindet normalerweise einen bestehenden Parser in seine Pipeline ein. Dependenzparesen ist in der Text-Mining-Praxis ein beliebter Verarbeitungsschritt für die Modellierung von Syntax großer Dokumentsammlungen, insbesondere auch wegen der effizienten Laufzeiteigenschaften, welche eine fast lineare Verarbeitung in der Satzlänge ermöglichen (siehe genauer in Nivre 2009).

**Konstituentenparsing** ist ein weiterer wichtiger Ansatz für das syntaktische Parsen (vgl. Bloomfield 1984), die Phrasenstruktur des Satzes wird hier in Anlehnung an die generative Grammatik (vgl. Chomsky 1957, siehe Abschn. 1.5) modelliert. Im Gegensatz zum Dependenzparsing, wo nur Relationen zwischen beobachtbaren Wörtern bestehen, werden hier in der Baumstruktur Knoten eingefügt, welche die Phrasenstruktur explizieren; Parsing wird hier modelliert als das Akzeptieren kontextfreier Sprachen (Abschn. 2.1.1). Während Konstituentenparsing vor einigen Jahrzehnten den Standardansatz für die Modellierung von Syntax darstellte, nahm dessen Popularität wegen der schlechten Laufzeiteigenschaften stetig ab. Andere, noch tiefer in die Analyse vordringende Parsing-Ansätze wie Head-Driven Phrase Structure Grammar (HPSG) (vgl. Müller 1999) und Lexical Functional Grammar (LFG) (vgl. Bresnan et al. 2016) sind wegen noch komplexerer Verarbeitung eher für linguistische Fragestellungen als für das Text Mining interessant, bieten jedoch deutlich genauer spezifizierte Strukturen zur formalen Repräsentation von Semantik. Eine effizientere Verarbeitung bei gleichzeitigem



**Abb. 3.8** Dependency-Parse nach Tagset-Schema der Konvertierung der Tiger-Treebank (siehe Bochnet 2010; Hajič et al. 2009). (Bild aus Ruppert et al. 2015)

Generieren von logischen Formen für die semantische Repräsentation ist Combinatory Categorical Grammar (CCG) (vgl. Steedman 2000).

### 3.2.5 Semantische Verarbeitung

Der Übergang von Syntax zu Semantik ist fließend, was bereits in den in Abschn. 3.2.4 angesprochenen Grammatikformalismen deutlich wird, welche immer darauf abzielen, eine Bedeutungsrepräsentation von Sätzen zumindest vorzubereiten. In diesem Abschnitt werden einige Verarbeitungsschritte der linguistischen Pipeline besprochen, welche der semantischen Ebene zugerechnet werden. Hierzu gehören die Erkennung von Eigennamen und deren Disambiguierung, die Erkennung der zutreffenden Bedeutung von mehrdeutigen Wörtern im Kontext, das Auflösen von Koreferenzen und das Explizieren semantischer Relationen und sogenannter semantischer Frames. Manche dieser Schritte werden anwendungsunabhängig in der Vorverarbeitungspipeline durchgeführt, andere sind wegen ihres vergleichsweise hohen Aufwands bei Erstellung und Berechnung nur dann einzusetzen, wenn die Text-Mining-Anwendung davon auch profitiert.

Ein wichtiger Schritt bei der Aufbereitung von Information aus Texten ist die Behandlung von Eigennamen, da diese direkte Referenzen zu Entitäten in der Welt darstellen. Im Verarbeitungsschritt der **Eigennamenerkennung (Named Entity Recognition, NER)** werden Typen von Namen wie Person, Organisation, Ort und manchmal auch numerische Angaben wie Daten und Geldbeträge klassifiziert. Ziel ist lediglich die Identifizierung der dazugehörigen Spannenannotation. Ein ähnlicher Verarbeitungsschritt ist das **Entity Linking**, bei dem Eigennamen gegenüber einer Wissensbasis disambiguiert werden: Hier steht insbesondere das Linking, also die eindeutige Identifikation einer in einer Ontologie hinterlegten Entität, im Vordergrund.

#### 3.2.5.1 Eigennamenerkennung

**Eigennamenerkennung (Named Entity Recognition)** wird normalerweise mit Sequenztagging-Ansätzen (siehe Abschn. 6.8) implementiert, welche neben annotiertem Trainingstext auch auf Listen bekannter Namen und Namensteile zurückgreifen können. Diese **Gazetteers** sind eine wichtige Ressource bei der Unterstützung dieser Verfahren; allein Namenslisten zum Abgleich reichen nicht aus, da diese einerseits immer unvollständig sind, andererseits viele Wörter auch als Namen verwendet werden können, z. B. *Horst* (männlicher Vorname/Storchennest), *Schuster* (Nachname/Beruf) oder als Teil von Firmennamen. In den meisten NER-Tagsets werden vier Klassen unterschieden: Personennamen (PER), Ortsnamen (LOC), Organisationsnamen (ORG) und Sonstige (MISC). Während die Erkennung für das Englische bis auf die MISC-Klasse mit sehr hoher Genauigkeit möglich ist, ist die Genauigkeit beim Deutschen etwas geringer (vgl. Tjong Kim Sang und De Meulder 2003; Akbik et al. 2019): Personennamen und Orte werden mit 90 % Genauigkeit und besser erkannt, bei Organisationen und Sonstigen ist die Qualität geringer aufgrund der höheren Variabilität in diesen Klassen. Ein möglicher

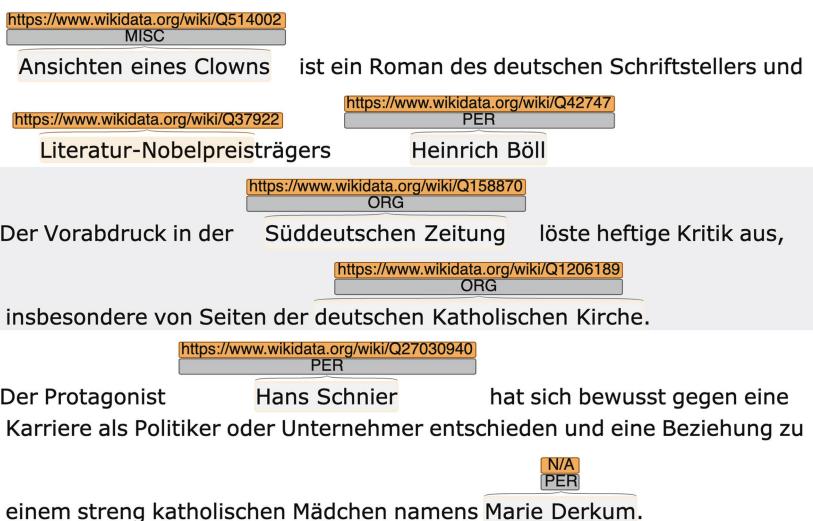
Grund für die größeren Schwierigkeiten beim Deutschen gegenüber anderen Sprachen ist die abweichende Rechtschreibung: In den meisten mit lateinischem Alphabet geschriebenen Sprachen werden nur Eigennamen mit einem Großbuchstaben begonnen, im Deutschen jedoch jedes Nomen. Für das Deutsche existieren zwei Standarddatensätze für das Trainieren und Evaluieren von NER-Systemen auf Zeitungstext (vgl. Tjong Kim Sang und De Meulder 2003; Benikova et al. 2014).

### 3.2.5.2 Entity Linking

**Entity Linking (EL)** (auch: Named Entity Recognition and Disambiguation, NERD) benötigt zunächst ein Inventar an bekannten Entitäten, wie z. B. Personen, gegeben durch eine Wissensbasis. Beliebte Wissensbasen im Entity Linking sind DBpedia (vgl. Auer et al. 2007) und Wikidata (vgl. Vrandečić und Krötzsch 2014), in welchen die in Wikipedia beschriebenen Entitäten auf strukturierte Weise gespeichert sind. Innerhalb von EL kann unterschieden werden zwischen der Erkennung von Namensnennungen (engl. Mention Detection) und der Disambiguierung der Namensnennung bzgl. des Entitäteninventars. Während EL und NER auf den ersten Blick ähnlich erscheinen, adressieren diese Schritte deutlich verschieden gelagerte Probleme: NER ignoriert die Referenz der Namen auf reelle Entitäten, findet jedoch auch Namen, welche nicht im Inventar von EL enthalten sind. Auch technisch sind die Probleme grundverschieden: NER ist eine Sequenzklassifikation in einer Handvoll Klassen, bei EL müssen Namensnennungen in sehr viele verschiedene Entitäten klassifiziert werden, die nicht alle in annotierten Trainingskorpora auftauchen. Daher stützen sich die meisten Ansätze für EL auf den Vergleich zwischen der in der Wissensbasis vorliegenden Information und des Kontextes, beide ggf. kodiert in neuronalen Repräsentationen (vgl. Sevgili et al. 2020), oft in unüberwachten Modellen. Eine beliebte Evaluationsplattform für EL ist GERBIL (vgl. Röder et al. 2018), wo Systeme automatisch anhand verschiedener Benchmarks verglichen werden können, allerdings vornehmlich für Englisch; es existieren nur wenige deutsche Benchmarks, wie z. B. Ploch et al. 2012. EL kann im Text Mining dann sinnvoll eingesetzt werden, wenn Beziehungen zwischen öffentlich bekannten Personen/Organisationen in öffentlichen Wissensbasen deutlich gemacht werden sollen; eine eigene Erstellung eines Wissensgraphen ist möglich, jedoch wegen der starken Formalisierung und der notwendigen laufenden Aktualisierung sehr arbeitsintensiv.

Das Beispiel in der Abb. 3.9 kontrastiert NER und EL-Annotationen (Named Entity Recognition in grau und Entity Linking in orange) für denselben Textabschnitt mit Wikidata. *Ansichten eines Clowns* disambiguierter korrekt zum Eintrag für das Buch, nicht für den gleichnamigen Film. Das Problem der Ressourcenunvollständigkeit wird mit der Figur *Marie Derkum* illustriert, die im Gegensatz zur Hauptfigur *Hans Schnier* nicht in Wikidata vermerkt ist [Stand: November 2020]. Unterschiedliche Definitionen von „Entität“ werden bei der Verlinkung des Literatur-Nobelpreises sichtbar.

Namenserkennung ist ein wichtiger Vorverarbeitungsschritt für die Informationsextraktion und für Anwendungen wie facettierte Suche (Abschn. 7.2).



**Abb. 3.9** Namenserkennung mit NER und EL (Beispieltext gekürzt aus Wikipedia (2020); Visualisierung der Annotation mit WebAnno (vgl. dazu Yimam et al. 2013))

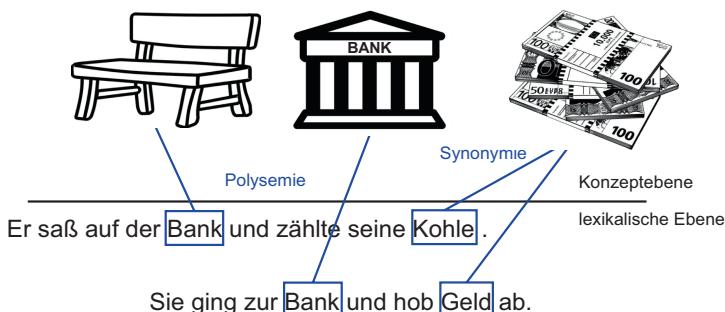
### 3.2.5.3 Koreferenzauflösung

**Koreferenzauflösung** bezeichnet den Verarbeitungsschritt zum Erkennen von Referenzen verschiedener Textsegmente auf dieselbe Entität. Während die Grenzen von Referenz fließend sind und über die Erkennung von sogenanntem Bridging (Brückebildung) wie *der Report .... das Dokument* bis hin zur Explizierung lexikalischen Ketten (vgl. Halliday und Hasan 1976) reicht, welche lediglich grob auf dasselbe Thema referieren, ist die für das Text Mining wichtigste Teilaufgabe das Auflösen von definiten Nominalphrasen (dies umfasst Pronomina), um Relationen zwischen Entitäten mit hoher Abdeckung erkennen zu können. Z. B. in *Maria und Peter trafen sich gestern. Sie schenkte dem Jungen ein Buch* ist es ohne die Auflösung der Referenz von *Sie* auf *Maria* bzw. von *dem Jungen* auf *Peter* nicht bestimmbar, wer Eigentümer oder Eigentümerin des Buches ist. Technisch wird Koreferenzauflösung derart umgesetzt, dass zunächst Anaphern (Rückreferenzen) identifiziert werden, dann wird für jede Anapher aus der Menge möglicher Antezedenzen (vorangegangene Bezugsobjekte) das passende ausgewählt (für ein Survey siehe Ng 2010). Automatische Koreferenzauflösung ist im Allgemeinen eine schwere Aufgabe, insbesondere für Bridging und das sächliche Pronomen „es“, welches sich auf viele potentielle Antezedenzen beziehen kann, oder auch auf gar keine wie in expletiven Konstruktionen, z. B. *es schneit*. Die für die meisten Anwendungen interessanten Auflösungen der männlichen und weiblichen Personalpronomina können in vielen Texten zufriedenstellend gelöst werden, indem der jeweils letzte Antezedenz mit dem passenden grammatischen Geschlecht gewählt wird. Insbesondere bei enzyklopädischen Artikeln wie z. B. aus Wikipedia ist die Heuristik erfolgreich, passende Personalpronomina auf den Titel des Artikels aufzulösen.

### 3.2.5.4 Wortbedeutungsdisambiguierung

**Wortbedeutungsdisambiguierung** bezeichnet die Zuweisung einer von mehreren möglichen Wortbedeutungen im Kontext. Ziel der Wortbedeutungsdisambiguierung ist der Übergang von der lexikalischen zur konzeptuellen Ebene, wie das Beispiel der Abb. 3.10 zeigt: Zum einen werden Mehrdeutigkeiten auf der Ebene von Wortbedeutungen aufgelöst, zum anderen werden Synonyme auf dasselbe Konzept abgebildet. Im Beispiel referiert das lexikalische Token *Bank* je nach Kontext ein Sitzmöbel bzw. ein kommerzielles Gebäude; die Tokens *Kohle* und *Geld* referieren beide in ihrem Kontext auf Zahlungsmittel; dies ist natürlich nicht die häufigste Lesart von *Kohle*.

Wortbedeutungsdisambiguierung (für ein Survey: Navigli 2009) benötigt zunächst ein Bedeutungsinventar, welches entweder durch lexikalische Ressourcen wie WordNet oder GermaNet (Abschn. 4.5) oder automatisch durch Clustering zur Wortbedeutungsinduktion (Abschn. 5.7) erstellt wurde. Bei der Methodik der Disambiguierung werden überwachte und unüberwachte Ansätze unterschieden: Überwachte, sogenannte ‚lexical sample‘-Ansätze benötigen für jedes zu disambiguierende Lemma annotierte Kontexte. Da dies für das gesamte Vokabular zu aufwendig ist, sind trotz ihrer normalerweise geringeren Genauigkeit unüberwachte ‚all-words‘-Ansätze populär, welche die Disambiguierung als Auswahl der passenden Bedeutung lediglich durch einen Vergleich des Kontextes mit den in der lexikalischen Ressource vorliegenden Bedeutungsrepräsentationen vornehmen. Die Frequenzverteilung von Bedeutungen folgt genau wie andere Frequenzverteilungen in der Sprache einem Power-Law (Abschn. 5.2): Typischerweise existiert eine dominante Wortbedeutung, z. B. „Sitzvorrichtung für Menschen auf Pferden oder Drahteseln“ für *Sattel*; die dominante Bedeutung ist jedoch sachgebietsabhängig. Im Maschinenbau z. B. bezeichnet *Sattel* die „Montageauffangvorrichtung für Kugellager“ – je nach Situation ist eine Disambiguierung nicht nötig, da ohnehin nur ein Sachgebiet im zugrunde liegenden Korpus enthalten ist, oder sie ist sehr wichtig, z. B. wenn aus allgemeinsprachlichen Korpora wie Webkorpora (Abschn. 4.2) für die Aufgabe passende Konzepte identifiziert werden müssen. Kontextualisierte Word Embeddings (Abschn. 3.1.3, Abschn. 5.6) eignen sich sehr gut für die Repräsentation



**Abb. 3.10** Beispiel für Mehrdeutigkeit (Polysemie) und Bedeutungsgleichheit (Synonymie); Bilder entnommen aus Public domain vectors (2021a, b, c)

von Wortbedeutung (vgl. Wiedemann et al. 2019) und die Modellierung kontextueller Synonymie.

Über die Repräsentation der Bedeutung von einzelnen Wörtern hinausgehend, zielt **semantisches Parsen** darauf ab, die Bedeutung ganzer Sätze, also die darin beschriebenen Aussagen, Situationen oder Ereignisse zu erfassen. Hierfür gibt es eine Reihe verschiedener Ansätze wie Semantic Role Labeling (vgl. Gildea und Jurafsky 2000), Abstract Meaning Representation (vgl. Banarescu et al. 2013), Frame-Semantic Parsing (vgl. Das et al. 2014) u. v. m., für einen Überblicksartikel siehe Kamath und Das 2019. Ziel des semantischen Parsens ist, von der syntaktischen Realisierung eines Satzes zu abstrahieren und die logische Struktur zu explizieren – hierbei werden Prädikate disambiguiert und die Argumente bezüglich ihrer Rollen in der Aussage identifiziert. Die Abb. 3.11 verdeutlicht dies anhand einer FrameNet-Annotation für das Deutsche (vgl. Burchardt et al. 2006). Es handelt sich um eine Beispielannotation des Frames REQUEST mit den Rollen (SemArg) ADDRESSEE, MESSAGE, SPEAKER und MEDIUM. Die Repräsentation abstrahiert stark von der grammatischen Satzstruktur, in welcher z. B. *Die Firma* als Subjekt von *bekommen* analysiert wird. Für die Extraktion von Beauftragungen aus Text ist hier auch Koreferenzauflösung nötig wie im Beispiel der Abb. 3.11 gezeigt.

Im Text Mining ist die Anwendung von semantischem Parsen bisher eher unüblich. Die Genauigkeit beim automatischen Zuweisen semantischer Strukturen ist begrenzt, die Ansätze zur strukturierten Klassifikation sind berechnungskomplex und für die meisten Anwendungen ist nicht die Gesamtheit aller Ereignisklassen interessant, sondern lediglich eine anwendungsspezifische Untergruppe, welche mit einfacheren Mitteln wie z. B. musterbasierten Verfahren (Abschn. 3.1.1) erfasst werden können.

### 3.2.6 Anwendungsorientierte Verarbeitung

In diesem Abschnitt wird die Konzeption von linguistischen Pipelines zur Vorverarbeitung für einige Beispieldaten des Text Mining verdeutlicht. Insbesondere werden nur diejenigen Schritte in die Vorverarbeitung aufgenommen, welche für die jeweilige Anwendung nötig sind. Die meisten der Anwendungen, welche hier in Hinblick auf ihre Verschiedenheit zu Illustrationszwecken ausgewählt wurden, werden nochmals in Kap. 7 diskutiert. Daher werden sie an dieser Stelle nur sehr kurz dargestellt. Ferner wird hier das Einlesen der Dokumente nicht diskutiert, da dies weniger von der Anwendung, sondern vielmehr von der Art des vorhandenen Textmaterials abhängig ist. Bei der



**Abb. 3.11** Semantisches Parsen mit FrameNet und Koreferenz (Visualisierung der Annotation mit WebAnno, vgl. dazu Yimam et al. 2013)

Erstellung der Pipeline aus ggf. vorhandenen oder neu zu erstellenden Komponenten sollte immer datengetrieben vorgegangen werden: Vorhandene Komponenten sind auf ihre Eignung für das vorliegende Material dahingehend zu prüfen, dass diese auf den Daten ausgeführt und ihre Ergebnisse manuell verifiziert werden. Eine sorgfältige, entwicklungsbegleitende Analyse der Funktionalität aller Schritte der Pipeline schärft das Verständnis für die Domäne und spart Zeit durch ziel- und bedarfsgerechte Entwicklung: Falls erst einmal eine unpassende Komponente in einem frühen Schritt der Pipeline verbaut wurde, müssten mitunter alle Folgeschritte bei deren Austausch angepasst werden, was es durch Analyse zu vermeiden gilt.

### 3.2.6.1 Pipeline für Terminologieextraktion

Terminologieextraktion (Abschn. 2.5.2, Abschn. 7.1) hat das Ziel, domänenspezifisches Vokabular aus einem das Sachgebiet repräsentierenden Korpus zu destillieren. Dies geschieht in Kombination des Filterns bestimmter Wortarten und der sprachlichen Struktur von Fachsprache folgenden Wortgruppen (Beispiele siehe Abschn. 2.5.2) mit statistischen Ansätzen, welche die Häufigkeit dieser Konstruktionen mit Hintergrundkorpora abgleichen, um signifikante Unterschiede in der Verwendungshäufigkeit festzustellen (genauer siehe Abschn. 5.9).

Je nach Sachgebiet ist die Notwendigkeit der Anpassung der Tokenisierung zu überprüfen. Falls z. B. chemische Verbindungen wie *1-(1,2-Dihexadecanoylphosphatidyl inositol-3,4-diphosphat* als Fachterminologie relevant sind, sollten diese auch als ein (und nicht mehrere) Token repräsentiert werden. Da Terminologie zumeist aus Nomen oder zusammenhängenden Phrasen besteht, welche in ihrer Grundform einem Lexikon von Fachausdrücken hinzugefügt werden sollen, wird Wortartentagging und Grundformreduktion benötigt. Während der Aufwand für das Parsen nicht gerechtfertigt ist, da die Analyse nicht auf hierarchische grammatische Strukturen zurückgreifen muss, könnte Chunking bei der Identifikation von Phrasen sinnvoll sein, was anhand der Daten überprüft werden sollte. Stemming als Alternative zur Lemmatisierung ist aufgrund der schlechten Lesbarkeit der Ergebnisse nicht zu empfehlen.

Die anwendungsunabhängige Vorverarbeitung der Terminologieextraktion liefert also Segmentierung, Wortarten, einige Wortgruppen und Grundformen. In einer anwendungsabhängigen Verarbeitung müssen nun die Kandidaten für Terminologie aufgrund dieser Information gefiltert werden, z. B. über Muster auf Wortarten und Chunk-Labels. Für den statistischen Vergleich mit Hintergrundkorpora muss diese Verarbeitung auf mindestens zwei Korpora durchgeführt werden; der statistische Vergleich und ggf. weitere Filterschritte für die Anwendung finden dann außerhalb der eigentlichen linguistischen Pipelines statt.

### 3.2.6.2 Pipeline für Entitätenzentriertes Retrieval und Facettierte Suche

Entitätenzentriertes Retrieval und facettierte Suche (Abschn. 7.2) reichert eine Dokumentsammlung mit Metadaten an, um ein gezieltes Filtern von Dokumenten in

Verbindung mit einer Volltextsuche zu ermöglichen und die Verbindungen zwischen Entitäten zu visualisieren. Neben Metadaten der Dokumentdateien wie Erstellungsdatum und Format umfasst dies auch die Extraktion dokumentspezifischer Merkmale wie z. B. Absender- und Empfängeradresse bei E-Mails, den Abgleich mit Wortlisten aus bekannter Terminologie, Schlüsselwortextraktion und die Erkennung von zeitlichen Referenzen und Entitäten.

Typische Dokumentsammlungen für diese Anwendung zeichnen sich durch große Heterogenität der Formate und oft durch Mehrsprachigkeit aus. Daher sollten bei der Datenanalyse auch immer Fehler beim Einlesen und bei der Spracherkennung in Betracht gezogen werden. Der hier viel zu aufwendigen Anpassung an alle Dokumenttypen, Sachgebiete und Sprachen ist eine generische Vorgehensweise bei der Segmentierung vorzuziehen, um die Anzahl sprachabhängiger Komponenten so gering wie möglich zu halten. Einige dokumentspezifische Metadaten können ohne Kenntnis der Sprache erkannt werden (wie z. B. E-Mail-Adressen), jedoch ist bereits die Annotation zeitlicher Ausdrücke und Wortlisten, welche mithilfe regulärer Ausdrücke (Abschn. 3.1.1) realisiert werden kann, genauso wie alle folgenden Schritte sprachabhängig. Die Erkennung von Schlüsselwörtern verlangt nach einer Pipeline, wie sie bereits bei der Terminologieextraktion beschrieben wurde, da auch hier bestimmte syntaktische Konstruktionen infrage kommen und daher Wortartentagging und Grundformreduktion nötig ist. Das Erkennen der Entitäten benötigt entweder Namenserkenntnung, Entity Linking oder eine Kombination aus beidem: Dies ist abhängig von der allgemeinen Bekanntheit der Entitäten und deren Abdeckung in den entsprechenden Wissensbasen. Die Wahl kann hier auf End-to-End Systeme fallen; featurebasierte Namenserkenner können die vorangegangenen Verarbeitungsschritte wie Wortartentagging nutzen. Da in dieser Anwendung das Suchen und Filtern von Dokumenten im Vordergrund steht, werden weitergehende semantische Analyseschritte wie Koreferenzauflösung, syntaktisches oder semantisches Parsen hier nicht benötigt.

Die anwendungsabhängige Verarbeitung besteht im Sammeln der jeweiligen Metadatenannotationen und der Vorbereitung für das Schreiben des Dokumentindex, was in diesem Fall die Ausgabe dieser Pipeline darstellt.

### 3.2.6.3 Pipeline für Sentimentanalyse

Sentimentanalyse (Abschn. 7.3) oder Stimmungsanalyse (Liu und Zhang 2012 für ein Survey) identifiziert positive oder negative Aussagen in Texten. Innerhalb der Sentimentanalyse werden sowohl sehr verschiedene Techniken eingesetzt, als auch verschiedene Granularitäten unterschieden: Mitunter sollen, abhängig von der Fragestellung, z. B. auch die sentimenttragenden Tokens und auf welches Ziel diese sich beziehen, extrahiert werden. Beides hat Einfluss auf die jeweils notwendigen Vorverarbeitungsschritte, weshalb keine Standardpipeline für diese Familie von Anwendungen existiert. Eine rein wortlistenbasierte Sentimentanalyse, welche aufgrund einer Liste positiver und negativer Wörter entscheidet, ob ein (kurzes) Dokument positiv oder negativ zu werten ist, benötigt neben ggf. Spracherkennung zum Filtern von fremdsprachlichem Material

eine Verarbeitung, welche die Wiedererkennung der Wortliste ermöglicht, z. B. mit regulären Ausdrücken (Abschn. 3.1.1). Neben der Tokenisierung kann dies auch ein Stemming oder eine Grundformreduktion umfassen; dies muss sowohl auf der Wortliste als auch auf den Texten durchgeführt werden. Für eine auf maschinellem Lernen basierende Sentimentanalyse auf Textebene, welche die Klassen „positiv“, „negativ“ und ggf. „neutral“ für kurze Texte wie z. B. Reviews oder Social Media Posts zuweist, sind die Schritte Segmentierung, Stemming/Grundformreduktion und Wortarttagging empfohlen, da hierdurch z. B. Namen und geschlossene Wortklassen gefiltert werden können. Falls Trainingsdaten vorhanden sind oder einfach erstellt werden können, liefert das maschinelle Lernen von Klassifikatoren meist bessere Ergebnisse als der rein wortbasierte Ansatz. Für die aspektbasierte Sentimentanalyse (auch: Stimmungsanalyse auf Objekt- und Eigenschaftsebene), welche z. B. für Restaurants Aspekte wie Service oder Essensqualität in Reviews wie „*Der Kellner war wirklich unglaublich pampig, aber die Nachspeise war spitzig!*“ getrennt bewertet, ist – unabhängig von einer wortlisten- oder lernbasierten Modellierung – zusätzlich noch Dependenzparsen empfohlen, um den Bezug der polaritätsausdrückenden Tokens zu den aspektbezogenen Tokens zu explizieren. Im Beispiel schlägt z. B. bei *pampig* die Heuristik fehl, Adjektive jeweils auf das nächste Nomen (hier: *Nachspeise*) zu beziehen.

### 3.2.6.4 Pipeline für Open Information Extraction

Open Information Extraction (OIE) (siehe Niklaus et al. (2018) für einen Überblick) extrahiert strukturierte Tripel aus (normalerweise einer sehr großen Sammlung von) Dokumenten, welche üblicherweise einer Relation und zwei Argumenten entsprechen. Im Gegensatz zur **template-basierten Informationsextraktion** (vgl. Neumann 2004), welche die Erkennung komplexer Informationsstrukturen auf eingeschränkten Domänen v. a. mit musterbasierten Verfahren erreicht, hat OIE das Ziel, allgemeines Wissen domänenübergreifend zu sammeln und zu aggregieren. Die meisten moderneren Ansätze zu OIE basieren auf Mustern über Dependenzparses, die hier zu erkennenden Tripel decken sich weitgehend mit Subjekt-Verb-Objekt-Instanziierungen, wobei bei Passivkonstruktionen das grammatische Subjekt und das grammatische Objekt die Positionen im Tripel entsprechend tauschen. Folglich muss die Vorverarbeitungspipeline einen Dependenzparser umfassen, welcher wiederum ggf. Schritte wie Segmentierung und Wortarttagging benötigt. Des Weiteren wird je nach Anwendung entweder NER oder Entity Linking durchgeführt. Je größer der angestrebte Recall und je höher der Abstraktionsgrad der Tripel, desto mehr semantische Verarbeitung ist in einem System zur OIE sinnvoll. Ein wichtiges Element ist hier die Koreferenzauflösung insbesondere für Pronomina, da Tripel wie (*kaufen, sie, es*) nicht besonders informativ sind, hierdurch aber z. B. zu (*kaufen, Petra, Motorrad*) aufgelöst werden können. Während semantisches Parsen bisher aus Effizienzgründen kaum in OIE eingesetzt wurde, implementieren viele Ansätze ähnliche Konzepte.

### 3.3 Skalierung auf große Datenmengen

#### 3.3.1 Datenparallelität

Bei der Skalierung der Verarbeitung auf sehr große Textmengen bestehen sowohl quantitative als auch qualitative Herausforderungen. Für das Text Mining sind Ansätze und Frameworks, welche die Daten in mehrere disjunkte Pakete aufteilen und damit eine datenparallele Verarbeitung ermöglichen, von besonderem Interesse.

Der Umgang mit ansteigenden Primär- und Sekundärdatenmengen (vgl. Abschn. 1.2) wäre bereits dann ein anspruchsvolles Unterfangen, wenn sich ihre Verarbeitungszeit bloß linear, d. h. unter Beibehaltung der Proportionen, verlängern würde. Und selbst in diesem Fall sähe sich Text Mining bei einer Vergrößerung des Korpus etwa von einer auf zehn Milliarden Sätze ohne spezielle Optimierung mit zehnmal längeren Verarbeitungszeiten konfrontiert. Die gegenstandstypische Datenkomplexität sowie eine verstärkte Einbeziehung auch niederfrequenter Sprachverwendungen – und damit einhergehend die Generierung zusätzlicher spezieller Annotationen – verschärfen die Problematik („Long Tail Problem“) obendrein. Hard- und Softwaresysteme für das Text Mining streben vor diesem Hintergrund eine zeitlich-räumliche Skalierbarkeit an, d. h. eine idealerweise deutlich unterproportionale Verlängerung speziell von Abfragezeiten bei gleichzeitiger Erhöhung des Datenvolumens.

**Skalierung** meint im informatischen Kontext die Leistungssteigerung technischer Infrastrukturen bei wachsenden Anforderungen. Ein historisch früher Ansatz ist die **vertikale Skalierung**. Dabei wird die Produktivität durch die Integration zusätzlicher bzw. leistungsfähigerer Hardware (CPU, Arbeitsspeicher, Festplatten etc.) erhöht („hochskaliert“). Dies erfordert keinen oder nur minimalen Programmieraufwand. In Anlehnung an die Mitte der 1960er Jahre erstmals postulierte Moore’sche Formel (vgl. Moore 1965) besteht die Strategie in der Ausnutzung elektronisch-mechanischer Innovationen, beispielsweise der regelmäßigen Erhöhung von Taktfrequenzen jeweils aktueller Prozessorgenerationen. Allerdings verhindern physikalische und wirtschaftliche Grenzen – insbesondere die Integrationsdichte von Mikroprozessoren – einen langfristig bedarfsgemäßen Leistungsanstieg.

In jüngerer Zeit erfolgt die Erweiterung klassischer Ansätze der Datenhaltung und des Information-Retrieval zur Optimierung kritischer Antwortzeiten deshalb verstärkt unter Zuhilfenahme verteilter und paralleler Architekturen (vgl. Rahm et al. 2015). Diese **horizontale Skalierung** steigert Speicher- und Rechenkapazitäten durch das Hinzufügen zusätzlicher Computer („Knoten“ oder „Nodes“) in einen vereint operierenden Rechnerverbund („Cluster“). Obgleich hardwareseitig theoretisch unbegrenzt, hängt die Effizienz horizontaler Skalierungen letztlich stark von der Implementierung geeigneter Softwarelösungen ab. Namentlich die Koordination verteilter Aufgaben und deren (Zwischen-) Ergebnisse implizieren in der Regel einen softwareseitigen Overhead, d. h. zusätzlichen Programmcode zur Sicherstellung von Synchronität, Konsistenz und Korrektheit.

Moderne Architekturen wie Multi-Core-Prozessoren, Multi-Prozessor-Systeme oder Cluster-Computing unterstützen Skalierung durch parallele (auch: nebenläufige) Arbeitsabläufe, d. h. durch die gleichzeitige Ausführung untereinander unabhängiger Aufgaben. Parallel Architekturen fokussieren je nach Skalierungsbedarf die zu verarbeitenden Daten, darauf operierende Algorithmen oder beides:

- **Parallelisierung von Daten:** Datenparallelität setzt die Möglichkeit einer Aufteilung der Gesamtdatenmenge in disjunkte Teilbereiche unter Beibehaltung der Datenstruktur voraus, was z. B. bei der Aufteilung von Korpora in einzelne Texte für den Zweck der linguistischen Vorverarbeitung (Abschn. 3.2) der Fall ist. Teilbereiche werden üblicherweise gleichmäßig auf mehrere Rechenknoten verteilt und parallel bearbeitet bzw. abgefragt. Durch das Hinzufügen zusätzlicher Knoten lässt sich der Parallelitätsgrad linear erhöhen. Die Gesamtverarbeitungszeit verkürzt sich durch die gleichzeitige Abarbeitung jeweils kleinerer Datenteilmengen.
- **Parallelisierung von Operationen:** Hier wird versucht, nicht aufeinander aufbauende Teilaufgaben zu identifizieren und nach Möglichkeit zeitgleich auszuführen. Sobald Teilaufgaben allerdings logisch miteinander interagieren, also Zwischenergebnisse austauschen um fortfahren zu können, muss der Datenfluss moderiert werden, was Programmieraufwand und Fehleranfälligkeit erhöht. Gleichwohl können auf diese Weise spürbare Geschwindigkeitsvorteile („Speedup“) erreicht werden; beispielsweise strebt die problemorientierte Algorithmisierung (vgl. Schneider 2019) eine frühzeitige Weiterverarbeitung abhängiger Operationen durch die höchstmögliche Reduzierung von Zwischenergebnisgrößen an.
- **Parallelisierung von Daten und Operationen:** Die Ausnutzung positiver Effekte von Datensegmentierung und Aufgabengranularität lässt sich kombinieren. Entsprechende Systemarchitekturen dynamisieren den Grad beider Parallelisierungen und organisieren die Verteilung von Daten und Teilaufgaben nach dem aus der Algorithmik bekannten Prinzip „Teile und Herrsche“.

Typischerweise umfassen parallele Computerarchitekturen ein ganzes Software-Portfolio, um Speicherungs-, Indexierungs- und Abfrageanforderungen aufeinander abgestimmt bedienen zu können. Als Pionier gilt das Open Source Framework Apache Hadoop (<https://hadoop.apache.org/>, vgl. White 2015), welches das ursprünglich für das effiziente Processing von Webseiten patentierte Parallelisierungsmodell MapReduce (Dean und Ghemawat 2008) implementiert. Kernmerkmale sind eine Aufteilung im Sinne der Parallelisierung von Daten und die Organisation von Batch-Arbeitsabläufen in seriellen Phasen (siehe Abb. 3.12). Nach dem initialen Einlesen der Eingabedaten werden Schlüssel-Wert-Paare auf im Netzwerk verteilten „Worker“-Knoten parallel vorverarbeitet. Eine „Shuffle Phase“ übernimmt das Sortieren und Gruppieren der Zwischenergebnisse. Daten mit identischen Schlüsselwerten werden gemeinsam in die finale „Reduce Phase“ weitergeleitet und dort aggregiert. Die Map- und Reduce-Funktionen werden in der Praxis je nach Anwendungsfall programmiert, üblicherweise

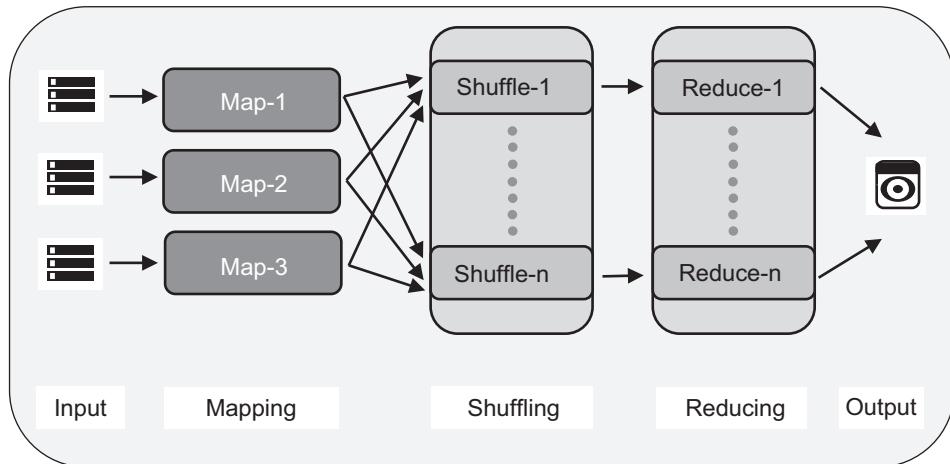


Abb. 3.12 Parallelisierung mit MapReduce

werden mehrere dieser Phasen hintereinander ausgeführt. Sämtliche übergreifenden Koordinations- und Monitoringaufgaben – beispielsweise Datensegmentierung, Parallelisierung des Datenflusses, Lastverteilung über mehrere Clusterknoten („Load Balancing“) oder der Umgang mit Hardwareproblemen – übernimmt das Framework.

Ersatzweise oder komplementär unterstützt Hadoop weitere parallele Analyse-Engines wie das speziell für die Nahe-Echtzeitverarbeitung entwickelte Spark (<https://spark.apache.org/>), vgl. Zaharia et al. 2016; Thomas 2020). Unter Beibehaltung des funktionalen Ansatzes (Operationen wie „Map“, „Reduce“ oder auch „Filter“ finden in Form sogenannter Transformationen statt) besteht der entscheidende Unterschied zu MapReduce in der physikalischen Speicherung: Während Hadoop MapReduce festplattenbasiert arbeitet und Zwischenergebnisse immer wieder schreiben und lesen muss, behält Spark sämtliche Daten im Hauptspeicher (RAM). Auf diese Weise erzielt es bis zu um den Faktor 100 kürzere Laufzeiten mit dem Preis der ggf. höheren Hardwareanschaffungskosten. Damit bietet sich Spark als Lösung für zeitkritische Analysen an, die nicht in auslastungsarme Zeitfenster verschoben werden können („Just-In-Time Delivery“), für linguistische Pipelines (vgl. Sahami et al. 2018), sowie generell für iterative Aufgaben, die bestimmte Datenverarbeitungsschritte mehrmals wiederholen.

Während Spark bereits mit sogenanntem Micro-Batching ein intervallbasiertes inkrementelles Verarbeiten von Daten möglich macht, bietet Flink (<https://flink.apache.org/>), vgl. Alexandrov et al. 2014) als drittes populäres Big Data Framework und Apache Top-Level-Projekt echtes Streaming (vgl. Hueske und Kalavri 2019). Auf diese Weise minimiert es Overhead-bedingte Verzögerungen im Ablauf, da Daten eben nicht mehr notwendigerweise in Batches aufgeteilt werden, deren Latenzen bei kürzeren Intervallzeiten tendenziell zunehmen. Datenströme lassen sich mit Flink auch fortlaufend verarbeiten, d. h. sie werden unmittelbar bei Eintreffen verarbeitet und nicht zunächst wie

bei Spark in mehr oder weniger großen Paketen gesammelt, was z. B. bei Social Media Mining von Vorteil ist.

Abgesehen von diesen architektonischen Unterschieden hängt die strategische Auswahl eines Frameworks erfahrungsgemäß von diversen situationsspezifischen Faktoren ab, etwa Anschaffungs- und Administrationskosten, der Verfügbarkeit bestimmter Programmiersprachen-Schnittstellen (APIs) oder der Integration von Bibliotheken für maschinelles Lernen.

Parallelisierungen von Daten und Operationen sind grundsätzlich für alle Text-Mining-Anwendungen praktikabel, die komplexe Berechnungen auf großen Korpora erfordern. Typische Aufgaben im Umfeld des Text Mining sind die Segmentierung und Indexierung von Textfragmenten (vgl. Berberich und Bedathur 2013), Kategorisierungen von Dokumenten, Suchen nach komplexen Mustern, Informationsextraktion oder die Datenaufbereitung für maschinelles Lernen. Das bekannteste textbasierte Beispiel für MapReduce ist die Messung von Worthäufigkeiten (vgl. Lin und Dyer 2010), andere Einsatzbereiche finden sich in der Verarbeitung von Webkorpora (Abschn. 4.2) (vgl. Biemann et al. 2013) oder der Berechnung distributioneller Ähnlichkeiten (Abschn. 5.4) (vgl. Biemann und Riedl 2013).

Auf die Grenzen der maximal erreichbaren Beschleunigung durch massive Parallelisierung weist das bereits Mitte der 1960er Jahre formulierte Amdahlsche Gesetz (Amdahl 1967) hin. Seine Kernaussage ist, dass jedes Problem nur zu einem gewissen Teil aufgeteilt und damit parallelisiert werden kann, weil bestimmte (z. B. initialisierende oder von einem Zwischenergebnis abhängige) Operationen unabänderlich sequentiell ausgeführt werden müssen. Hinzu kommt ggf. ein organisatorischer Overhead durch ansteigenden Koordinationsaufwand, etwa in Form notwendiger Kommunikation zwischen und Synchronisierung von Operationen. In der Praxis liegt es deshalb nahe, vorrangig die am häufigsten genutzten Teilprobleme zu parallelisieren.

Abgesehen von der Beschleunigung konstanter Aufgaben lohnt sich der Einsatz paralleler Ansätze gemäß Gustafsons Gesetz (Gustafson 1988) speziell bei ansteigenden Problemgrößen. Nehmen beispielsweise die Anzahl von Suchmustern einer Korpusabfrage oder von gleichzeitig auf ein System zugreifenden Anwendern bzw. Anwenderinnen zu, lassen sich die zusätzlichen Aufgaben in vielen Fällen in Form von einander unabhängiger Operationen parallelisieren.

### 3.3.2 Zusammenhang von Korpusgröße und Qualität

Bei der Arbeit mit Korpora stellt sich schnell die Frage nach der Textmenge, die verwendet werden soll. Die Antwort hängt von der zu bearbeitenden Fragestellung ab: Gibt es eine feste Korpusgröße, mit der man nichts falsch macht, oder soll man immer die maximal mögliche Größe wählen?

### 3.3.2.1 Der More-Data-Effect

Häufig erwähnt wird der sogenannte More-Data-Effect: Mit der Größe der Datenmenge steigt die Qualität der Ergebnisse (vgl. Banko und Brill 2001). Vereinfacht ausgedrückt heißt das, bei der Verwendung von mehr Daten lösen sich viele Probleme von selbst.

Natürlich ist das keine allgemeingültige Antwort, aber die folgenden Kriterien helfen bei der Entscheidungsfindung: Was könnte denn besser werden, wenn wir ein größeres Korpus verwenden? Besonders bei der Verwendung statistischer Verfahren könnte ein größeres Korpus dafür sorgen, dass die Qualität der Ergebnisdaten besser wird. Dies entspricht der Tatsache, dass eine größere Stichprobe genauere Vorhersagen von Messwerten erlaubt. Zusätzlich wird möglicherweise die Ergebnismenge größer: Wenn wir nach einer Sammlung von Wörtern, Zusammenhängen oder Sachverhalten suchen, können wir in einem größeren Korpus mehr finden. Beispiele sind Wörter aus speziellen Fachgebieten (siehe Abschn. 7.1 Terminologieextraktion) oder Listen von Eigennamen.

Aber sollten wir deshalb wirklich immer gleich mit dem größten zur Verfügung stehenden Korpus beginnen? Nein, es ist immer eine gute Idee, das eigene Vorgehen zuerst an einem kleineren Korpus zu testen. Zunächst kann man sich so versichern, dass die Korpusverarbeitung überhaupt funktioniert und die gewünschten Ergebnisse liefert. Betrachten wir ein Beispiel und nehmen dafür an, dass die Bearbeitungszeit linear von der Korpusgröße abhängt. Diese Annahme ist in den meisten Fällen sinnvoll. Benutzt man für den ersten Test beispielsweise nur 0,1 % der Daten, für den zweiten Test 1 %, dann 10 % und erst zum Schluss alle 100 %, dann benötigt man zwar insgesamt 11,1 % mehr Zeit als wenn man gleich mit dem großen Korpus gestartet hätte, aber im Falle von unangemessenem Vorgehen bemerkt man dies viel schneller. Wenn die Bearbeitungszeit für 0,1 % der Daten schon viele Stunden beträgt, sollte man über ein beschleunigtes Verfahren oder, wenn das nicht möglich ist, über Parallelisierung nachdenken (siehe Abschn. 3.3.1). Und wenn die Ergebnisse für die kleineren Korpora nicht wenigstens im Prinzip brauchbar sind, dann wird vielleicht auch das große Korpus nach viel längerer Bearbeitungszeit keine nützlichen Ergebnisse liefern.

Geht aber alles gut und die kleineren Korpora liefern auch schon prinzipiell brauchbare Daten, dann vertrauen wir auf den More-Data-Effect.

Folgende Gründe sprechen für den More-Data-Effect:

- Allgemein gibt es in größeren Korpora größere Resultatmengen pro Wort. Weil das Wort häufiger auftritt, tritt es wahrscheinlich auch häufiger in typischen Kontexten oder typischen Sätzen auf. Allerdings tritt das Wort auch häufiger in nicht-typischen Kontexten auf. Bei geordneten Resultatmengen (etwa nach statistischer Signifikanz) kann es sinnvoll sein, für ein Wort nur die besten Resultate (hier: mit der größten Signifikanz) zu betrachten.
- Größere Korpora liefern erstmalig Resultate für seltener Wörter, da viele Verfahren eine Mindesthäufigkeit des Wortes benötigen. Deshalb sind Korpora maximaler Größe nützlich bei Aufgabenstellungen, die nach Wortmengen suchen (z. B. Terminologieextraktion, Suche nach Eigennamen, Suche nach Wortpaaren in vorgegebener Relation).

### 3.3.2.2 Quantitatives Wachstum von Resultatmengen

Dürfen wir davon ausgehen, dass die Resultatmenge linear mit der Korpusgröße wächst? Dass wir also aus einem zehnmal so großen Korpus auch zehnmal so viele Daten extrahieren können?

Aus mehreren Gründen ist dies nicht immer so: Aus theoretischer Sicht muss die Resultatmenge nicht linear mit der Korpusgröße wachsen, das Wachstum kann beispielsweise auch nur logarithmisch sein. Ein schnelleres als lineares Wachstum (z. B. quadratisch) ist auch denkbar, in der Praxis aber eher selten anzutreffen. Hier können vorangestellte theoretische Überlegungen helfen.

Die Betrachtung des Wachstumsverhaltens geht davon aus, dass sich mit wachsender Korpusgröße auch immer mehr Ergebnisse erzielen lassen. In der Praxis sind aber viele Ergebnismengen endlich und von überschaubarer Größe, sodass die Gesamtmenge schnell gefunden ist und größere Korpora nichts mehr beitragen können. Oder man nähert sich der Gesamtmenge an und der Zuwachs durch den *More-Data-Effect* ist geringer als erwartet. In solchen Fällen wirkt der *More-Data-Effect* nicht oder nicht so, wie erwartet.

#### Beispiel

Die Tab. 3.4 und die dazugehörige Abb. 3.13 zeigen das Wachstum verschiedener Werte in Abhängigkeit von der Korpusgröße.

Die Größe der Korpora (gemessen in Anzahl Sätzen) wächst jeweils um den Faktor 10 von 1.000 bis 100.000.000. Dazu angegeben ist:

- Die Anzahl verschiedener Wörter (Types), wobei nur Zeichenketten als Wörter gezählt werden, die dem regulären Ausdruck  $^{\wedge}[\text{A-ZÄÖÜa-zäöüß-}]^* \$$  entsprechen und weder am Beginn oder am Ende einen Bindestrich haben.
- Die Anzahl dieser Wörter, welche signifikante Kookkurrenzen (siehe Abschn. 5.3) im Korpus besitzen, genauer: Satzkookkurrenzen, linke Nachbarn und rechte Nachbarn.

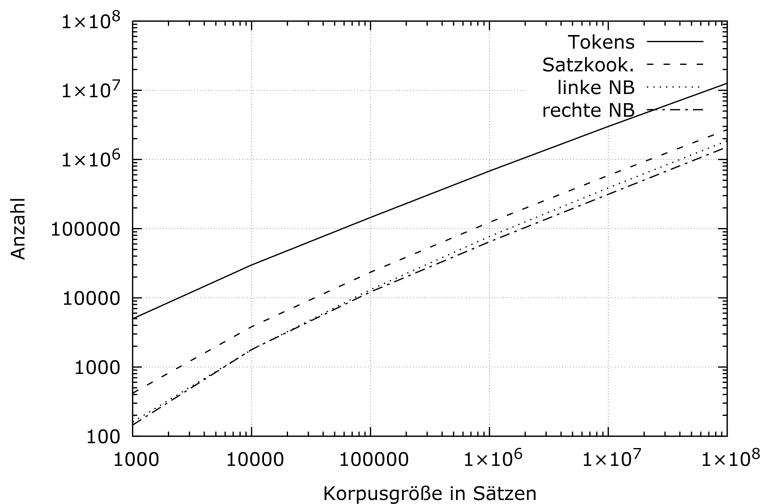
Die Skalen in Abb. 3.13 sind logarithmiert, d. h. das fast lineare Wachstum in der Abbildung entspricht einem Potenzgesetz in den realen Zahlen: Gemittelt im Bereich von  $x = 10.000$  bis  $100.000.000$  Sätzen wachsen die anderen Werte  $y$  näherungsweise zwischen  $y = x^{2/3}$  (für die Tokens) und  $y = x^{3/4}$  (für die verschiedenen Arten von Kookkurrenzen). ◀

#### Beispiel: Wörter alphabetisch vor und nach *Mustermann*

Größere Korpora liefern größere Wortlisten, wie im letzten Beispiel gezeigt. Dieses Beispiel soll zeigen, dass diese Wortlisten durchaus bekannte Wörter in ansprechender Qualität liefern, sodass diese Wortlisten beispielsweise als Grundlage für große Wörterbücher brauchbar sind.

**Tab. 3.4** Wertewachstum in Abhängigkeit von der Korpusgröße

Korpus	Tokens	Satzkookkurrenzen	linke Nachbarn	rechte Nachbarn
deu-de_web-wrt_2019_1K	4982	419	160	145
deu-de_web-wrt_2019_10K	29.838	3823	1773	1787
deu-de_web-wrt_2019_100K	145.116	23.469	13.147	12.173
deu-de_web-wrt_2019_1M	679.722	123.302	76.920	64.488
deu-de_web-wrt_2019_10M	2.999.977	587.361	388.696	313.511
deu-de_web-wrt_2019_100M	12.640.301	2.700.048	1.856.714	1.517.186

**Abb. 3.13** Korpusgröße in Sätzen

Die Abb. 3.14 zeigt jeweils die 10 Wörter alphabetisch vor und nach dem Wort *Mustermann*, je nachdem, wie viele der insgesamt häufigsten Wörter benutzt werden. Dabei werden die häufigsten 1000, 10.000, 100.000 bzw. 1.000.000 Wörter benutzt. Dabei benötigt man für die größeren Wortlisten natürlich größere Korpora. Im größten oben gewählten Korpus treten die häufigsten 1.000.000 Wörter jeweils mindestens zwanzigmal auf, dadurch wird die Liste zuverlässiger als beim nächstkleineren Korpus, wo diese Mindestzahl nur 2 beträgt. Fett gedruckt sind Wörter, die bereits in der Spalte davor auftreten. Die nicht fett gedruckten Wörter sind also jeweils neu dazugekommen. ◀

Vor <i>Mustermann</i> in den			
Top-1.000	Top-10.000	Top-100.000	Top-1M
Möglichkeiten	<b>muss</b>	<b>Muster</b>	Musterkonzept
möglichst	müsste	Muster-Widerrufsformular	Musterküche
Monat	<b>müssen</b>	Musterbeispiel	Musterküchen
Monate	müsst	Mustererkennung	Musterkunde
Monaten	musst	Musterfeststellungsklage	Musterkunden
München	müsste	mustergültig	Musterkündigung
Musik	<b>musste</b>	Musterhaus	Musterland
<b>muss</b>	müssten	Musterhäuser	Musterländle
müssen	mussten	Musterlösung	<b>Musterlösung</b>
musste	Muster	Musterlösungen	<b>Musterlösungen</b>
<b>Mustermann</b>			
Nach <i>Mustermann</i> in den			
Top-1.000	Top-10.000	Top-100.000	Top-1M
<b>nach</b>	Mut	Mustern	Mustermesse
Nach	Mutter	Musters	Mustermietvertrag
nächsten	Mütter	Musterschreiben	Mustermix
Nähe	Müttern	Musterung	<b>Mustern</b>
Namen	n	Mustervertrag	mustern
Natur	Na	Musterverträge	Musterort
natürlich	NABU	<b>Mut</b>	Musterpläne
neben	<b>nach</b>	Mutation	Musterplatten
Neben	<b>Nach</b>	Mutationen	Musterpolizeigesetz
nehmen	Nachbarn	muten	Musterportfolio

Abb. 3.14 Tabelle: Wörter alphabetisch vor und nach *Mustermann*

### 3.3.2.3 Qualitative Verbesserung von Resultatmengen

Viele Verfahren liefern Resultatmengen, die zusätzlich mit einem Zahlenwert versehen sind. Resultate mit einem höheren Zahlenwert werden in der Regel vom Algorithmus besser bewertet, sodass eine Sortierung entsprechend dem Zahlenwert sinnvoll ist. Dem entspricht auch die Sortierung der Suchergebnisse bei einer Internetsuchmaschine: Weiter oben stehende Resultate sind meist besser. Der More-Data-Effect bedeutet hier, dass dieses Ranking bei größeren Korpora besser sein sollte. Wenn man sich beim Ergebnis auf die Resultate mit dem besten Ranking beschränkt, sollten diese Resultatmengen qualitativ besser werden. Allerdings bleibt oft das Problem zu entscheiden, wie groß die Resultatmenge sein soll. Ein allgemeingültiger Schwellwert für den Zahlenwert lässt sich in der Regel nicht angeben, und eine vorher festgelegte Anzahl kann sich auch als falsch erweisen. Wie die untenstehenden Beispiele zeigen, gibt es aus algorithmischer Sicht nicht nur sieben (oder acht?) Namen für die Wochentage.

Beispiele für solche geordneten Resultatmengen sind:

- Wörter, sortiert nach Häufigkeit, auch Namen von Personen oder andere Eigennamen,
- Kookkurrenzen, geordnet nach Signifikanz,
- im Vektorraum benachbarte Wörter bei Word Embeddings, geordnet nach Abstand.

#### Beispiel

Das Beispiel der Tab. 3.5 zeigt die semantisch ähnlichen Wörter zu *Donnerstag*. Dies sollen natürlich die Namen der anderen Wochentage sein. Das Ergebnis, berechnet über die Überlappung der Satzkoorkurrenzen, zeigt semantisch ähnliche Wörter für Korpora ab 100 K Sätzen und an den vorderen Positionen eine wachsende Anzahl von Namen von Wochentagen.

Verschiedene Verfahren zur semantischen Ähnlichkeit liefern leicht abweichende Ergebnisse. Das kann bei größeren Korpora auch verblüffende und unerwünschte

**Tab. 3.5** Beispiel *Donnerstag*

Korpus	Korrektheit	Top-10 ähnliche Wörter zu <i>Donnerstag</i>
deu-de_web-wrt_2019_100K	erste 5 Wörter	Donnerstag, Dienstag, Samstag, Mittwoch, Montag, 4., Sonntag, 22., 26., 3
deu-de_web-wrt_2019_1M	erste 7 Wörter, Sonnabend fehlt	Donnerstag, Freitag, Montag, Mittwoch, Samstag, Dienstag, Sonntag, April, November, 24..
deu-de_web-wrt_2019_10M	erste 8 Wörter, korrekt	Donnerstag, Mittwoch, Dienstag, Freitag, Montag, Samstag, Sonntag, Sonnabend, 18.30, Uhr
deu-de_web-wrt_2019_100M	erste 8 Wörter, korrekt	Donnerstag, Mittwoch, Dienstag, Freitag, Montag, Samstag, Sonntag, Sonnabend, 17.30, Uhr

Effekte haben. Beispielsweise sind die Schreibfehler „Mitwoch“ und „Donnertag“ recht häufig. In einem sehr großen Korpus werden deshalb auch die fehlerhaft geschriebenen Wörter in den vielen typischen Kontexten auftreten und sich in die Ergebnismenge mischen, sogar noch vor *Samstag* und *Sonntag*.

Falls man (wie bei den Wochentagen) davon ausgehen kann, dass die gesuchten Wörter näherungsweise gleich häufig sind, dann kann diese Eigenschaft genutzt werden, um solche Schreibfehler zu identifizieren und auszusondern. ◀

---

## Literatur

- Akbik, A., Bergmann, T., Vollgraf, R.: Pooled contextualized embeddings for named entity recognition. In: Burstein, J., Doran, C., Solorio, T. (Hrsg.) Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics. Human Language Technologies. Volume 1 (Long and Short Papers), Minneapolis, MN, USA, S. 724–728 (2019). <https://doi.org/10.18653/v1/N19-1078>
- Akbik, A., Blythe, D., Vollgraf, R.: Contextual string embeddings for sequence labeling. In: Bender, E.M., Derczynski, L., Isabelle, P. (Hrsg.) Proceedings of the 27th International Conference on Computational Linguistic (COLING), Santa Fe, NM, USA, S. 1638–1649. <https://www.aclweb.org/anthology/C18-1139.pdf> (2018). Zugegriffen: 11. Jan. 2021
- Amdahl, G.M.: Validity of the single processor approach to achieving large scale computing capabilities. In: American Federation of Information Processing Societies (Hrsg.): Proceedings of the AFIPS Spring Joint Computer Conference, Atlantic City, NJ, USA, S. 483–485. Thomson Book Company, Washington, DC, USA (1967)
- Alexandrov, A., Bergmann, R., Ewen, S., Freytag, J.-C., Hueske, F., Heise, A., Kao, O., Leich, M., Leser, U., Markl, V., Naumann, F., Peters, M., Rheinländer, A., Sax, M.J., Schelter, S., Höger, M., Tzoumas, K., Warneke, D.: The stratosphere platform for big data analytics. VLDB J. 23(6), 939–964 (2014)
- Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.G.: DBpedia: a nucleus for a web of open data. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (Hrsg.) ISWC'07/ASWC'07: The Semantic Web, S. 722–735. Springer, Berlin (2007)
- Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., Schneider, N.: Abstract meaning representation for sembanking. In: Pareja-Lora, A., Liakata, M., Dipper, S. (Hrsg.) Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse. Sofia, Bulgaria, S. 178–186. <https://www.aclweb.org/anthology/W13-2322> (2013). Zugegriffen: 8. Febr. 2021
- Banko, M., Brill, E.: Scaling to Very Very Large Corpora for Natural Language Disambiguation. In: Webber, B.L. (Hrsg.) Proceedings of the 39th Annual Meeting on Association for Computational Linguistics – ACL '01, Toulouse, France, 26–33 (2001). <https://doi.org/10.3115/1073012.1073017>
- Baroni, M., Chantree, F., Kilgariff, A., Sharoff, S.: Cleaneval: a Competition for Cleaning Web Pages. In: Calzolari, N., Choukri, K., Maegaard, B. et al. (Hrsg.) Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08), Marrakech, Morocco, S. 638–643. [http://www.lrec-conf.org/proceedings/lrec2008/pdf/162\\_paper.pdf](http://www.lrec-conf.org/proceedings/lrec2008/pdf/162_paper.pdf) (2008). Zugegriffen: 20. Jan. 2020

- Biemann, C., Bildhauer, F., Evert, S., Goldhahn, D., Quasthoff, U., Schäfer, R., Simon, J., Swiezinski, L., Zesch, T.: Scalable Construction of High-Quality Web Corpora. *J. Lang. Technol. Comput. Linguist. (JLCL)* **28**(2), 23–59 (2013)
- Biemann, C., Riedl, M.: Text: Now in 2D! A Framework for Lexical Expansion with Contextual Similarity. *J. Lang. Model.* **1**(1), 55–95 (2013). <https://doi.org/10.15398/jlm.v1i1.60>
- Beesley, K.R., Karttunen, L.: Finite state morphology. *Studies in Computational Linguistics*, Bd. 3. CSLI Publications, Stanford (2003)
- Beißwenger, M., Bartz, T., Storrer, A., Westpfahl, S.: Tagset und Richtlinie für das Part-of-Speech-Tagging von Sprachdaten aus Genres internetbasierter Kommunikation. Guideline document from the Empirikom shared task on automatic linguistic annotation of internet-based communication (EmpiriST 2015), <https://sites.google.com/site/empirst2015/home/annotation-guidelines> (2015). Zugegriffen: 21. Jan. 2021
- Benikova, D., Biemann, C., Reznicek, M.: NoSta-D named entity annotation for German: Guidelines and dataset. In: Calzolari, N., Choukri, K., Declerck, T., Loftsson, H., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., Piperidis, S. (Hrsg.) *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC 2014)*, Reykjavik, Iceland, S. 2524–2531. [http://www.lrec-conf.org/proceedings/lrec2014/pdf/276\\_Paper.pdf](http://www.lrec-conf.org/proceedings/lrec2014/pdf/276_Paper.pdf) (2014). Zugegriffen: 8. Febr. 2021
- Berberich, K., Bedathur, S.: Computing N-Gram Statistics in MapReduce. In: Paton, N.W. (Hrsg.): *Proceedings of the 16th International Conference on Extending Database Technology*. Genoa, Italy, S. 101–112. ACM, New York, NY, USA (2013)
- Bethard, S., Ogren, P., Becker, L.: ClearTK 2.0: Design Patterns for Machine Learning in UIMA. In: Calzolari, N., Choukri, K., Declerck, T. et al. (Hrsg.) *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland, S. 3289–3293. [http://www.lrec-conf.org/proceedings/lrec2014/pdf/218\\_Paper.pdf](http://www.lrec-conf.org/proceedings/lrec2014/pdf/218_Paper.pdf) (2014). Zugegriffen: 19. Jan. 2021
- Biemann, C., Quasthoff, U., Heyer, G., Holz, F.: ASV Toolbox – A Modular Collection of Language Exploration Tools. In: Calzolari, N., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Tapias, D. (Hrsg.) *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, Morocco, S. 1760–1767. [http://www.lrec-conf.org/proceedings/lrec2008/pdf/447\\_paper.pdf](http://www.lrec-conf.org/proceedings/lrec2008/pdf/447_paper.pdf) (2008). Zugegriffen: 9. Febr. 2021
- Bird, S., Klein, E., Loper, E.: *Natural Language Processing with Python. Analyzing Text with the Natural Language Toolkit*. O'Reilly Media, Sebastopol, Beijing, Cambridge (2009)
- Bloomfield, L.: *Language*. University of Chicago Press, Chicago (1984)
- Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching Word Vectors with Subword Information. *TACL* **5**, 135–146 (2017). [https://doi.org/10.1162/tacl\\_a\\_00051](https://doi.org/10.1162/tacl_a_00051)
- Bohnet, B.: Top accuracy and fast dependency parsing is not a contradiction. In: *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, Beijing, China, S. 89–97 (2010)
- Bresnan, J., Asudeh, A., Toivonen, I., Wechsler, S.: *Lexical-Functional Syntax*. Second edition, Blackwell textbooks in linguistics, Bd. 16. Wiley-Blackwell, Chichester, West Sussex, Malden, MA, USA (2016)
- Brill, E.: A simple rule-based part of speech tagger. In: Bates, M., Stock, O. (Hrsg.) *Proceedings of the third conference on Applied natural language processing (ANLC '92)*, Trento, Italy, S. 152–155 (1992). <https://doi.org/10.3115/974499.974526>
- Burchardt, A., Erk, K., Frank, A., Kowalski, A., Padó, S., Pinkal, M.: The SALSA corpus: a German corpus resource for lexical semantics. In: Calzolari, N., Choukri, K., Gangemi, A., Maegaard, B., Mariani, J., Odijk, J., Tapias, D. (Hrsg.) *Proceedings of the Fifth International*

- Conference on Language Resources and Evaluation (LREC 2006), Genoa, Italy. [http://www.lrec-conf.org/proceedings/lrec2006/pdf/339\\_pdf.pdf](http://www.lrec-conf.org/proceedings/lrec2006/pdf/339_pdf.pdf) (2006). Zugriffen: 8. Febr. 2021
- Bush, V.: As We May Think. *Atlantic Monthly* **176**, 101–108 (1945)
- Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In: Moschitti, A., Pang, B., Daelemans, W. (Hrsg.) *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, S. 1724–1734 (2014). <https://doi.org/10.3115/v1/D14-1179>
- Chomsky, N.: *Syntactic Structures*. Mouton 1957, Nachdruck bei Mouton. de Gruyter, Berlin (2009)
- Cunningham, H., Tablan, V., Roberts, A., Bontcheva, K.: Getting More Out of Biomedical Documents with GATE’s Full Lifecycle Open Source Text Analytics. *PLoS computational biology* **9**(2), e1002854 (2013). <https://doi.org/10.1371/journal.pcbi.1002854>
- Das, D., Chen, D., Martins, A.F.T., Schneider, N., Smith, N.A.: Frame-semantic parsing. *Computational Linguistics* **40**(1), 9–56 (2014). [https://doi.org/10.1162/COLI\\_a\\_00163](https://doi.org/10.1162/COLI_a_00163)
- Dean, J., Ghemawat, S.: MapReduce: Simplified Data Processing on Large Clusters. *Commun. ACM* **51**(1), 107–113 (2008). <https://doi.org/10.1145/1327452.1327492>
- Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In: Burstein, J., Doran, C., Solorio, T. (Hrsg.) *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, MN, USA, S. 4171–4186 (2019). <https://doi.org/10.18653/v1/N19-1423>
- Eckart de Castilho, R., Gurevych, I.: A broad-coverage collection of portable NLP components for building shareable analysis pipelines. In: Ide, N., Grivolla, J. (Hrsg.) *Proceedings of the Workshop on Open Infrastructures and Analysis Frameworks for HLT (OIAF4HLT)*, Dublin, Ireland, S. 1–11 (2014). <https://doi.org/10.3115/v1/W14-5201>
- Ferrucci, D., Lally, A.: UIMA: an architectural approach to unstructured information processing in the corporate research environment. *Nat. Lang. Eng.* **10**(3–4), 327–348 (2004). <https://doi.org/10.1017/S1351324904003523>
- Ferrucci, D.: Introduction to “This is Watson”. *IBM J. Res. Dev.* **56**(3.4), 1:1–1:15 (2012). <https://doi.org/10.1147/JRD.2012.2184356>
- Francis, W.N., Kučera, H.: Computational analysis of present-day American english. Brown University Press, Providence (1967)
- Gildea, D., Jurafsky, D.: Automatic Labeling of Semantic Roles. In: Iida, H. (Hrsg.) *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL-00)*, Hong Kong, S. 512–520 (2000). <https://doi.org/10.3115/1075218.1075283>
- Goldberg, Y.: Neural Network Methods for Natural Language Processing. *Synthesis Lectures on Human Language Technologies* **10**(1), 1–309 (2017). <https://doi.org/10.2200/S00762ED1V01Y201703HLT037>
- Goodfellow, I., Bengio, Y., Courville, A.: *Deep Learning. Adaptive Computation and Machine Learning Series*. MIT, Cambridge (2016)
- Grishman, R., Sundheim, B.: Message Understanding Conference – 6: A Brief History. In: Tsujii, J. (Hrsg.) *Proceedings of the 16th International Conference on Computational Linguistics (COLING 1996)*, Volume 1, Copenhagen, Dänemark, S. 466–471 (1996). <https://doi.org/10.3115/992628.992709>
- Gustafson, J.L.: Reevaluating Amdahl’s Law. *Commun. ACM* **31**(5), 532–533 (1988)
- Hajič, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M.A., Márquez, L., Meyers, A., Nivre, J., Padó, S., Štěpánek, J., Straňák, P., Surdeanu, M., Xue, N., Zhang, Y.: The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages. In: Hajič,

- J. (Hrsg.) Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task. Boulder, CO, USA, S. 1–18. <https://www.aclweb.org/anthology/W09-1201> (2009). Zugegriffen: 16. Febr. 2021
- Hajič, J., Hladká, B.: Tagging Inflective Languages: Prediction of Morphological Categories for a Rich Structured Tagset. In: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1, Montreal, QC, Canada, S. 483–490 (1998). <https://doi.org/10.3115/980845.980927>
- Halliday, M.A.K., Hasan, R.: Cohesion in English. English language series, Bd. 9. Longman, London (1976)
- Hearst, M.A.: Automatic Acquisition of Hyponyms from Large Text Corpora. In: Proceedings of the 15th International Conference on Computational Linguistics (COLING 1992), Volume 2, Nantes, France, S. 539–545. <https://www.aclweb.org/anthology/C92-2082> (1992). Zugegriffen: 9. Febr. 2021
- Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997). <https://doi.org/10.1162/neco.1997.9.8.1735>
- Honnibal, M., Montani, I.: spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. (2017)
- Hueske, F., Kalavri, V.: Stream Processing with Apache Flink: Fundamentals, Implementation, and Operation of Streaming Applications. O'Reilly Media, Sebastopol, CA, USA (2019)
- Jiang, J.: Information Extraction from Text. In: Aggarwal, C.C., Zhai, C.X. (Hrsg.) Mining Text Data, S. 11–41. Springer, New York, NY, USA (2012). [https://doi.org/10.1007/978-1-4614-3223-4\\_2](https://doi.org/10.1007/978-1-4614-3223-4_2)
- Jauhainen, T., Lui, M., Zampieri, M., Baldwin, T., Lindén, K.: Automatic language identification in texts: a survey. *J. Artif. Intell. Res.* **65**(1), 675–782 (2019). <https://doi.org/10.1613/jair.1.11675>
- Jawahar, G., Sagot, B., Seddah, D.: What Does BERT Learn about the Structure of Language? In: Korhonen, A., Traum, D., Márquez, L. (Hrsg.) Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, S. 3651–3657 (2019). <https://doi.org/10.18653/v1/P19-1356>
- Kamath, A., Das, R.: A Survey on Semantic Parsing. In: Proceedings of Automated Knowledge Base Construction, Amherst, MA, USA (2019). Zugegriffen: 5. Febr. 2021
- Kleene, S.C.: Representation of events in nerve nets and finite automata. In: Shannon, C.E., McCarthy, J. (Hrsg.) Automata Studies. Annals of mathematics studies, Bd. 34, S. 3–42. Princeton University Press, Princeton, NJ, USA (1956)
- Kudo, T., Richardson, J.: SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In: Blanco, E., Lu, W. (Hrsg.) Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Brussels, Belgium, S. 66–71 (2018). <https://doi.org/10.18653/v1/D18-2012>
- Le, Q., Mikolov, T.: Distributed representations of sentences and documents. *ICML'14: Proceedings of the 31st International Conference on Machine Learning*. PMLR **32**(2), 1188–1196 (2014)
- LeCun, Y., Kavukcuoglu, K., Farabet, C.: Convolutional networks and applications in vision. In: *ISCAS 2010. 2010 IEEE International Symposium on Circuits and Systems – ISCAS 2010*, Paris, France, S. 253–256 (2010). <https://doi.org/10.1109/ISCAS.2010.5537907>
- Leidner, J.L.: Current issues in software engineering for Natural Language Processing. In: *Proceedings of the HLT-NAACL 2003 workshop on Software engineering and architecture of language technology systems – SEALTS '03*, Morristown, NJ, USA, 5/31/2003, S. 45–50 (2003). <https://doi.org/10.3115/1119226.1119233>

- Lin, J., Dyer, C.: Data-Intensive Text Processing with MapReduce (Synthesis Lectures on Human Language Technologies). Morgan and Claypool Publishers, San Rafael (2010)
- Liu B., Zhang L.: A Survey of Opinion Mining and Sentiment Analysis. In: Aggarwal C., Zhai C. (Hrsg.) Mining Text Data, S. 415–463. Springer, Boston, MA, USA (2012). [https://doi.org/10.1007/978-1-4614-3223-4\\_13](https://doi.org/10.1007/978-1-4614-3223-4_13)
- Manning, C.D., Schütze, H.: Foundations of Statistical Natural Language Processing, 8. Aufl. MIT Press, Cambridge (1999)
- Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., McClosky, D.: The Stanford CoreNLP Natural Language Processing Toolkit. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Baltimore, MA, USA, S. 55–60 (2014). <https://doi.org/10.3115/v1/S.14-5010>
- Marneffe, M.-C. de, MacCartney, B., Manning, C.D.: Generating Typed Dependency Parses from Phrase Structure Parses. In: Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06), Genoa, Italy, S. 449–454. [https://nlp.stanford.edu/pubs/LREC06\\_dependencies.pdf](https://nlp.stanford.edu/pubs/LREC06_dependencies.pdf) (2006). Zugegriffen: 20. Jan 2021
- Maxwell, J.T., Kaplan, R.M.: The Interface between Phrasal and Functional Constraints. Comput. Linguist. **19**(4), 571–590. <https://www.aclweb.org/anthology/J93-4001> (1993). Zugegriffen: 19. Jan. 2021
- McCulloch, W.S., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. Bull. Math. Biophys. **5**(4), 115–133 (1943). <https://doi.org/10.1007/BF02478259>
- McDonald, R.T., Nivre, J., Quirkbach-Brundage, Y., Goldberg, Y., Das, D., Ganchev, K., Hall, K.B., Petrov, S., Zhang, H., Täckström, O., Bedini, C., Castelló, N.B., Lee, J.: Universal Dependency Annotation for Multilingual Parsing. Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Sofia, Bulgaria, S. 92–97. <https://www.aclweb.org/anthology/S.13-2017> (2013). Zugegriffen: 11. Jan. 2021
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems. <https://arxiv.org/pdf/1310.4546.pdf> (2013). Zugegriffen: 11. Jan. 2021
- Moore, G.E.: Cramming more components onto integrated circuits. Electronics **38**(8), 114–117 (1965)
- Morrison, D.R.: PATRICIA—Practical Algorithm to Retrieve Information Coded in Alphanumeric. J. ACM **15**(4), 514–534 (1968). <https://doi.org/10.1145/321479.321481>
- Müller, S.: Deutsche Syntax deklarativ. Head-Driven Phrase Structure Grammar für das Deutsche. Linguistische Arbeiten, Bd. 394. Niemeyer, Tübingen (1999)
- Navigli, R.: Word sense disambiguation: A survey. ACM Comput. Surv. **41**(2), 1–69 (2009). <https://doi.org/10.1145/1459352.1459355>
- Neumann, G.: Informationsextraktion. In: Carstensen, K.-U., Ebert, C., Endriss, C., Jekat, S., Klabunde, R., Langer, H. (Hrsg.) Computerlinguistik und Sprachtechnologie – Eine Einführung, 2. Aufl. Elsevier, Spektrum Akad., München (2004)
- Ng, V.: Supervised noun phrase coreference research: The first fifteen years. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, S. 1396–1411. <https://www.aclweb.org/anthology/S.10-1142/> (2010). Zugegriffen: 9. Febr. 2021
- Niehues, J., Salesky, E., Turchi, M., Negri, M.: Tutorial Proposal: End-to-End Speech Translation. In: Augenstein, I., Habernal, I. (Hrsg.) Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Tutorial Abstracts. Online, S. 10–13. <https://aclanthology.org/2021.eacl-tutorials.3.pdf> (2021). Zugegriffen: 10. Sept. 2021
- Niklaus, C., Cetto, M., Freitas, A., Handschuh, S.: A Survey on Open Information Extraction. In: Proceedings of the 27th International Conference on Computational Linguistics, Santa Fe, NM,

- USA, S. 3866–3878. <https://www.aclweb.org/anthology/C18-1326> (2018). Zugegriffen: 9. Febr 2021
- Nivre, J.: Non-projective dependency parsing in expected linear time. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, Suntec, Singapore, S. 351–359. <https://www.aclweb.org/anthology/P09-1040.pdf> (2009). Zugegriffen: 20. Jan. 2021
- Nivre, J., Marneffe, M.-C. de, Ginter, F., Goldberg, Y., Hajič, j., Manning, C.D., McDonald, R.T., Petrov, S., Pyysalo, S., Silveira, N., Tsarfaty, R., Zeman, D.: Universal Dependencies v1: A Multilingual Treebank Collection. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016), Portorož, Slovenia, S. 1659–1666. [http://www.lrec-conf.org/proceedings/lrec2016/pdf/348\\_Paper.pdf](http://www.lrec-conf.org/proceedings/lrec2016/pdf/348_Paper.pdf) (2016). Zugegriffen: 21. Jan. 2021
- Pennington, J., Socher, R., Manning, C.D.: GloVe: Global Vectors for Word Representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, S. 1532–1543 (2014). <https://doi.org/10.3115/v1/D14-1162>
- Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. In: Proceedings of the 2018 conference of the North American Chapter of the Association for Computational Linguistics (NAACL), New Orleans, LA, USA, S. 2227–2237 (2018). <https://doi.org/10.18653/v1/N18-1202>
- Ploch, D., Hennig, L., Duka, A., De Luca, E.W., Albayrak, S.: GerNED: A German Corpus for Named Entity Disambiguation. In: Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC-2012), Istanbul, Turkey, S. 3886–3893. [http://www.lrec-conf.org/proceedings/lrec2012/pdf/222\\_Paper.pdf](http://www.lrec-conf.org/proceedings/lrec2012/pdf/222_Paper.pdf) (2012). Zugegriffen: 11. Jan. 2021
- Porter, M.F.: Snowball: A language for stemming algorithms. <http://snowball.tartarus.org/texts/introduction.html> (2001). Zugegriffen: 20. Jan. 2021
- Public domain vectors: Bank pictogram vector clip art. <https://publicdomainvectors.org/en/free-clipart/Bank-pictogram-vector-clip-art/18235.html> (2021a). Zugegriffen: 16. Febr. 2021
- Public domain vectors: Vector clip art of Euro notes in black and white. <https://publicdomainvectors.org/en/free-clipart/Vector-clip-art-of-Euro-notes-in-black-and-white/35490.html> (2021b). Zugegriffen: 16. Febr. 2021
- Public domain vectors: Wooden park bench vector image. <https://publicdomainvectors.org/en/free-clipart/Wooden-park-bench-vector-image/25144.html> (2021c). Zugegriffen: 16. Febr. 2021
- Rahm, E., Saake, G., Sattler, K.-U.: Verteiltes und Paralleles Datenmanagement. Von verteilten Datenbanken zu Big Data und Cloud. Springer Vieweg, Berlin (2015)
- Röder, M., Usbeck, R., Ngomo, A.: GERBIL - Benchmarking Named Entity Recognition and Linking consistently. Semantic Web **9**(5), 605–625 (2018). <https://doi.org/10.3233/SW-170286>
- Rosenblatt, F.: The perceptron: a probabilistic model for information storage and organization in the brain. Psychol. Rev. **65**(6), 386–408 (1958). <https://doi.org/10.1037/h0042519>
- Ruppert, E., Klesy, J., Riedl, M., Biemann, C.: Rule-based Dependency Parse Collapsing and Propagation for German and English. In: Proceedings of the International Conference of the German Society for Computational Linguistics and Language Technology, Duisburg-Essen, Germany, S. 58–66. German Society for Computational Linguistics and Language Technology (2015)
- Sahami, S., Eckart, T., Heyer, G.: Using Apache Spark on Hadoop Clusters as Backend for WebLicht Processing Pipelines. Selected papers from the CLARIN Annual Conference 2018, Pisa, Italy. Linköping Electronic Conference Proceedings 159, 188–195 (2018)

- Santorini, B.: Part-of-Speech Tagging Guidelines for the Penn Treebank Project (3rd Revision, 2nd Printing). <http://www.ling.helsinki.fi/kit/2010s/clt236/docs/PennTaggingGuide.pdf> (1995). Zugegriffen: 20. Jan. 2021
- Schiller, A., Teufel, S., Thielen, C., Stöckert, C.: Guidelines für das Tagging deutscher Textcorpora mit STTS. (Kleines und großes Tagset). <http://www.sfs.uni-tuebingen.de/resources/stts-1999.pdf> (1999). Zugegriffen: 9. Dez. 2020
- Schneider, R.: Mehrfach annotierte Textkorpora. Strukturierte Speicherung und Abfrage. Korpuslinguistik und interdisziplinäre Perspektiven auf Sprache (CLIP) 8). Narr Francke Attempto, Tübingen (2019)
- See, A., Liu, P.J., Manning, C.D.: Get To The Point: Summarization with Pointer-Generator Networks. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vancouver, Canada, S. 1073–1083 (2017). <https://doi.org/10.18653/v1/P17-1099>
- Sevgili, Ö., Shelmanov, A., Arkhipov, M., Panchenko, A., Biemann, C.: Neural Entity Linking: A Survey of Models based on Deep Learning. <https://arxiv.org/pdf/2006.00575> (2020). Zugegriffen: 9. Febr. 2021
- Smith, R.: An Overview of the Tesseract OCR Engine. In: Proc. Ninth International Conference on Document Analysis and Recognition (ICDAR), Band 2, September 2007, S. 629–633 (2007)
- Steedman, M.: The Syntactic Process. Language, Speech, and Communication. MIT Press, Cambridge (2000)
- Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to Sequence Learning with Neural Networks. In: Advances in Neural Information Processing Systems 27 (NIPS 2014), Montreal, Canada, S. 3104–3112 (2014)
- Tenney, I., Das, D., Pavlick, E.: BERT RedisCOVERS the Classical NLP Pipeline. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, S. 4593–4601 (2019). <https://doi.org/10.18653/v1/S.19-1452>
- Tesnière, L.: Éléments de syntaxe structurale. Klincksieck (1959)
- Thomas, A.: Natural Language Processing with Spark NLP. Learning to Understand Text at Scale, O'Reilly Media, Sebastopol (2020)
- Tjong Kim Sang, E.F., Meulder, F.de: Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In: Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, Edmonton, Canada, S. 142–147 (2003). <https://doi.org/10.31115/1119176.1119195>
- Vaswani, A., Shazeer N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems 30 (NIPS'17), Long Beach, CA, USA, S. 6000–6010 (2017)
- Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Commun. ACM **57**(10), 78–85 (2014). <https://doi.org/10.1145/2629489>
- White, T.: Hadoop. The Definitive Guide, 4. Aufl, O'Reilly Media, Sebastopol (2015)
- Wiedemann, G., Remus, S., Chawla, A., Biemann, C.: Does BERT Make Any Sense? Interpretable Word Sense Disambiguation with Contextualized Embeddings. In: Proceedings of KONVENS 2019, Erlangen, Germany, S. 161–170. <https://arxiv.org/pdf/1909.10430> (2019). Zugegriffen: 11. Jan. 2021
- Wikipedia: Ansichten eines Clowns. [https://de.wikipedia.org/wiki/Ansichten\\_eines\\_Clowns](https://de.wikipedia.org/wiki/Ansichten_eines_Clowns) (2020). Zugegriffen: 16. Febr. 2021

- Yimam, S.M., Gurevych, I., Eckart de Castilho, R., Biemann, C.: WebAnno: A Flexible, Web-based and Visually Supported System for Distributed Annotations. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Sofia, Bulgaria, S. 1–6. <https://www.aclweb.org/anthology/P13-4001> (2013). Zugegriffen: 11. Jan. 2021
- Zaharia, M., Xin, R.S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., Venkataraman, S., Franklin, M.J., Ghodsi, A., Gonzalez, J., Shenker, S., Stoica, I.: Apache Spark: a unified engine for big data processing. Commun. ACM **59**(11), 56–65 (2016). <https://doi.org/10.1145/2934664>



# Sprachdaten: Lexika und Korpora

4

## Zusammenfassung

Dieses Kapitel widmet sich der Auswahl der verwendeten Daten für das Text Mining. Hier unterscheiden wir Textdaten für die Erstellung von Korpora und lexikalische Ressourcen wie Lexika oder Wortlisten. Bei der Auswahl von Korpora ist je nach Anwendungsbereich die Art des Korpus zu beachten; falls passend, bieten vorgefertigte, generische Korpora einen schnellen Einstieg und sind bereits qualitätsge-  
sichert abrufbar und können geeignet gespeichert und indexiert verfügbar gemacht werden. Zum Aufbau eines eigenen Korpus, z. B. aus fachspezifischen Texten, sind qualitätssichernde Maßnahmen wichtig, welchen wir uns in einer gewissen Breite widmen. Diese sind insbesondere wichtig, falls korpusbasiert lexikalische Ressourcen erstellt oder erweitert werden sollen. Informationen über einzelne Wörter, wie man sie gewöhnlich in Wörterbüchern findet, werden für viele Anwendungen von der syntaktischen Analyse einzelner Sätze bis hin zur Wissensextraktion benötigt. Quellen für solche (insbesondere frei verfügbaren) lexikalischen Daten werden vorgestellt und Verfahren werden besprochen, wie lexikalische Ressourcen aus Korpora erzeugt oder erweitert werden können.

## 4.1 Korpusauswahl

Soll eine Aufgabenstellung mithilfe von Korpora gelöst werden, sprechen wir von einem **korpusbasierten Ansatz**. Aus einem großen Korpus von angemessener Qualität lassen sich oft mit einfachen Mitteln gute Ergebnisse extrahieren, die anderenfalls erheblichen Aufwand bei der Datenbeschaffung erfordern würden. Die Entscheidung für den korpusbasierten Ansatz erfordert Antworten auf zwei unterschiedliche Fragestellungen: Welches Korpus soll die Basis dafür sein? Und welche Arbeitsschritte sollen mit diesem

Korpus ausgeführt werden, d. h. welche Tools sollen auf das Korpus angewendet werden? Diese Fragestellungen sind zu einem gewissen Grade unabhängig voneinander, da Korpus und Tools einzeln ausgetauscht werden können.

Da das Korpus die Anwendungsdomäne beim Text Mining definiert, ist die Korpusauswahl ein sehr wichtiger Schritt bei der Definition einer Text-Mining-Anwendung. Im Folgenden thematisieren wir vor allem die Auswahl oder Sammlung von Hintergrundkorpora für die Erstellung oder Erweiterung lexikalischer Ressourcen (Abschn. 4.6) und dem Training von Sprachmodellen (Abschn. 5.5) und Wort-Repräsentationen (Abschn. 5.6).

Korpora existieren in unannotierter oder annotierter Form (Abschn. 6.7), wobei erstere sprachliches Wissen kodieren und letztere vor allem für das maschinelle Lernen von Sprachverarbeitungskomponenten (Abschn. 3.1, 6.6) genutzt werden. In diesem und den folgenden Abschnitten werden nur unannotierte Korpora besprochen, da diese die Basis für alle weiteren Verarbeitungs- und Verfeinerungsschritte darstellen.

### 4.1.1 Generische Korpora

Im günstigsten Fall stehen für die anvisierte Aufgabe mehrere Korpora (in unterschiedlicher Größe, aus unterschiedlichen Quellen, in verschiedenen Sprachen, etc.) zur Verfügung, aus denen ausgewählt werden kann. Das Angebot an frei verfügbaren Korpora steigt, es gibt bewährte Formate (siehe Abschn. 4.3) und Tools zu diesen Korpora. Solche frei verfügbaren und leicht wiederverwendbaren Korpora sollen als generische Korpora bezeichnet werden (vgl. Eckart et al. 2016). Diese generischen Korpora zeichnen sich durch die folgenden Eigenschaften aus:

**Verwendbar für verschiedene Anwendungen:** Bei der Erstellung der Korpora wurde darauf geachtet, dass so wenig Einschränkungen wie möglich vorgenommen wurden. Die Korpora sollen nicht auf spezielle Sachgebiete oder Genres eingeschränkt und sowohl für linguistische Fragestellungen als auch zur Wissensextraktion oder zum Training maschineller Lernverfahren geeignet sein.

**Vergleichbarkeit zwischen verschiedenen Sprachen, großer Umfang und hohe Qualität:** Sind Korpora in verschiedenen Sprachen aufgrund der Auswahlkriterien und der Verarbeitungsschritte ähnlich, so lassen sich die gleichen Verfahren auf diese Korpora anwenden und die Resultate sind vergleichbar. Besitzen die Korpora zusätzlich vergleichbare Qualität (siehe Abschn. 5.8), so sollte sich bei Austausch der Korpora auch die Qualität der Resultatmenge kaum ändern.

**Zusätzliche Daten zum Korpus, z. B. vorausberechnete Daten:** Die Extraktion von Informationen aus einem Korpus verlangt gelegentlich die Berechnung zusätzlicher Daten. Diese kann aufwendig sein, ihre Bereitstellung unterstützt die weitere Nutzung des Korpus. Solche zusätzlichen Daten sind beispielsweise:

- Annotationen wie POS-Tagging (Abschn. 3.2) oder NER (Abschn. 3.2.5),
- vorberechnete Wort-Kookkurrenzen (Abschn. 5.3),
- vorberechnete Word Embeddings (Abschn. 5.6),
- ein vorberechnetes Topic-Modell (Abschn. 6.4).

Solche vorberechneten Daten sparen die Zeit zur Berechnung vergleichbarer Daten und unterlagen bereits einer Qualitätssicherung.

**Metadaten** Ein generisches Korpus sollte über ausreichende Metadaten (siehe Abschn. 1.2) über die Herkunft der verwendeten Texte verfügen, sodass die Nutzerin oder der Nutzer Informationen über die Zusammensetzung des Korpus erhält und sich bei Bedarf auf einen selbst ausgewählten Teil der Daten beschränken kann.

**Webzugang** Komfortable Recherchemöglichkeiten im Web erlauben es, das Korpus vor der eigentlichen Anwendung testweise abzufragen. Das hilft beispielsweise sicherzustellen, dass sich die gesuchten Daten überhaupt im Korpus befinden. So kann auch die nötige Korpusgröße abgeschätzt werden, um gewünschte Ergebnismengen zu erzielen.

**Nutzung per Webservice oder API** Je nach Anwendungsfall soll das Korpus möglicherweise nicht ganz heruntergeladen und verarbeitet werden, sondern die online angebotenen Informationen sind ausreichend. In diesem Fall sollten die Daten automatisch und in großer Menge abfragbar sein. Hier bieten sich Webservices oder andere automatisierte Schnittstellen an.

**Voller Zugriff auf das komplette Korpus** Zur Bearbeitung eigener Fragestellungen ist es oft nötig, das ganze Korpus zur Verfügung zu haben. Dazu ist es erforderlich, dass die kompletten Korpora (auch mit den zusätzlichen vorberechneten Daten) zum Download zur Verfügung stehen.

### 4.1.2 Selbst erstelltes Korpus

Zusätzlich besteht immer die Option, sich ein eigenes maßgeschneidertes Korpus zu erstellen. Dies ist immer dann angezeigt, wenn die konkrete Fragestellung sich nicht mithilfe generischer Korpora beantworten lässt. Mögliche Gründe sind:

- Es handelt sich um von den Auftraggebenden zur Verfügung gestellte, eigene Daten.
- Es handelt sich um ein spezielles Genre (z. B. Social-Media-Daten eines bestimmten Anbieters) oder ein spezielles Sachgebiet; hier ist zu prüfen, ob durch Selektion von Datensätzen mit den entsprechenden Werten in den Metadaten ein passendes Korpus aus generischen Korpora hergestellt werden kann.
- Es liegen bisher keine oder zu wenig Daten in der entsprechenden Sprache als generische Korpora vor.

### 4.1.3 Dokumente oder Sätze? Nur wohlgeformte Sätze?

Dieser Abschnitt beschäftigt sich mit der Frage, in welcher Form die Texte in das Korpus aufgenommen werden sollen. Wir unterscheiden an dieser Stelle zwischen dem **Dokumentkorpus** und dem **Satzkorpus**, welche sich jeweils durch die enthaltenen Objekte unterscheiden. Vor allem aus rechtlichen Überlegungen, aber auch aus Gründen der Qualitätssicherung sind folgende Möglichkeiten denkbar:

**Dokumentkorpus – ganze Dokumente in ihrer Originalreihenfolge:** Dies erhält den Originalcharakter weitestgehend: Die Originalkontexte bleiben erhalten, sodass sowohl Wörter als auch Sätze immer in ihrem Gesamtkontext betrachtbar sind. Der Nachteil dieses Vorgehens ist vor allem rechtlicher Natur: Werden im Korpus auch urheberrechtlich geschützte Texte verwendet, so lassen sich die Originaldokumente rekonstruieren und werden mit Weitergabe des Korpus auch weitergegeben. Dieses Vorgehen ist also nur möglich, wenn man z. B. im Rahmen einer Beauftragung über die Zustimmung der Urheberrechtsinhaber oder -inhaberinnen verfügt oder gemeinfreie Dokumente benutzt.

**Satzkorpus – einzelne Sätze mit Quelle:** Zerlegt man die Dokumente in einzelne Sätze und speichert diese in zufälliger Reihenfolge ab, dann lassen sich die Originaldokumente nicht mehr rekonstruieren, was wir im Folgenden als Satzkorpus bezeichnen. Allerdings entstehen beim Zerlegen eines Textes an Satzenden auch nicht wohlgeformte Sätze (z. B. aus Überschriften, Bildunterschriften oder Tabelleninhalten). Diese wirken isoliert außerhalb ihres Kontextes wie Verarbeitungsfehler. Wenn der Kontext aber sowieso fehlt, können zusätzliche Maßnahmen sowohl die Qualität erhöhen als auch die Rekonstruierbarkeit der Originale erschweren.

**Satzkorpus – nur wohlgeformte Sätze:** Eine großzügige Entfernung nicht wohlgeformter Sätze aus einem Satzcorpus ist aus zwei Gründen hilfreich. Zum einen erhöht sich die Qualität, da Sätze von zweifelhafter Qualität entfernt werden, was wir im Abschn. 4.2 illustrieren. Außerdem ist jetzt die Rekonstruktion der Originaldokumente wirklich unmöglich, da substantielle Teile der Originale nicht mehr vorhanden sind.

Es bleibt die Frage, welche Auswahl der Sätze für ein generisches Korpus gewählt werden soll. Aus Sicht des Urheberrechts ist die letzte Variante die sicherste. Einige Anwendungen werden durch dieses Vorgehen allerdings ausgeschlossen: Ein über Satzgrenzen hinausgehender Kontext ist nicht mehr vorhanden. Deshalb können Verweise auf Nachbarsätze (z. B. mittels Pronomen) nicht mehr aufgelöst werden. Kompliziertere Zusammenhänge, die mithilfe mehrerer Sätze ausgedrückt werden, stehen nicht mehr nebeneinander und können deshalb nicht mehr extrahiert werden. Für Anwendungen des Information Retrievals sind Dokumentkorpora alternativlos, auch können Verfahren wie Topic-Modelle (Abschn. 6.4) oder transformer-basierte Sprachmodelle (Abschn. 5.6) nur auf gesamten Dokumenten sinnvoll trainiert werden.

Jedoch können durch einen großen Umfang des Korpus einige dieser Nachteile wettgemacht werden. Wenn wichtige Zusammenhänge im Korpus wiederholt genannt und an

einer Stelle durch einen längeren Satz ausgedrückt werden (z. B. in einem längeren Satz mit Nebensatz statt in zwei Sätzen), dann können sie auch aus Einzelsätzen extrahiert werden, weswegen Satzkorpora für die Erstellung oder Erweiterung von lexikalischen Ressourcen (Abschn. 4.6), insbesondere für Kookkurrenzen (Abschn. 5.3) und Wortähnlichkeiten (Abschn. 5.4) völlig adäquat sind.

#### 4.1.4 Repräsentativität und Ausgewogenheit der Zusammensetzung oder zufällige Auswahl?

Um zu vermeiden, dass eine falsche oder einseitige Zusammensetzung des Korpus falsche Messwerte liefert, sollte man sich vorher über die Zusammensetzung seines Korpus Gedanken machen. Die Zusammensetzung des aufzubauenden Korpus kann durch viele Parameter (siehe Abschn. 1.2) beschrieben werden:

- Zusammensetzung bzgl. der Textgenres (Zeitung, Literatur, wissenschaftliche Fachtexte, ...),
- Zusammensetzung bzgl. der Entstehungszeit,
- Zusammensetzung nach Fachgebiet (wieviel religiöse Texte, wieviel Chemie, ...),
- Zusammensetzung nach Herkunft (lokal nach Region, nach Sprachvarietät Deutschland/Österreich/Schweiz, wieviel übersetzter Text, ...).

Kriterien für diese Zusammensetzung sind Repräsentativität und Ausgewogenheit (vgl. Sinclair 2004). Unter Repräsentativität versteht man das Bemühen, die Anteile der Texte für das Korpus so auszuwählen wie man sie in der vollständigen Menge aller Texte findet – entweder in Bezug auf die gesamte Sprache, oder in Bezug auf die für die Aufgabenstellung relevanten Texte.

Da die genannten Parameter in der Regel kaum bekannt sind oder eine Verteilung von Texten entsprechend der vielen Parameter nicht zu realisieren ist, sind absolut repräsentative Corpora praktisch unmöglich zu erzeugen.

Als Alternative bieten sich sogenannte ausgewogene Corpora an: Bei Ausgewogenheit beschränkt man sich auf einige wenige handhabbare Parameter und erstellt sein Korpus bezüglich dieser Parameter im angestrebten Verhältnis. Verringert man diese Anzahl der zu berücksichtigenden Parameter weiter auf null, dann bleibt nur die zufällige Auswahl von Texten. Wie in anderen Bereichen der Statistik kann eine rein zufällige Auswahl einer hinreichend großen Stichprobe durchaus bestmögliche Ergebnisse liefern. Die praktische Ausführung der zufälligen Auswahl ist allerdings im Falle der Korpuskonstruktion schwierig, solange die Grundgesamtheit, aus der ausgewählt werden soll, nicht bekannt ist. Im Falle von Texten aus dem Internet kann also nur durch die **Crawlingstrategie** versucht werden, durch viele ausgewählte Texte insgesamt einen repräsentativen Teil von Dokumenten zu erreichen.

### 4.1.5 Parallele Korpora

Parallele Texte sind Texte in zwei (oder mehr) verschiedenen Sprachen mit identischem Inhalt. Meist entstehen sie durch Übersetzung. Quellen für parallele Texte sind politische Dokumente der EU (vgl. European Union 2016) in verschiedenen Mitgliedssprachen, Bedienungsanleitungen in verschiedenen Sprachen, Übersetzungen von Büchern, Filmuntertitel usw. Zu dieser Parallelität gehört zusätzlich eine Verlinkung der entsprechenden übersetzten Teile, typischerweise auf Basis von Sätzen oder Absätzen.

Während parallele Korpora von großem Interesse für die automatische Übersetzung und die automatische Erstellung von Übersetzungswörterbüchern sind, behandeln wir sie in diesem Buch nicht gesondert, da sie für das Text Mining von untergeordneter Bedeutung sind.

---

## 4.2 Satzkorpuserstellung auf Webdaten

Die folgenden Abschnitte beschreiben die Arbeitsschritte, mit deren Hilfe aus Internettexten ein Satzkorpus erstellt werden kann. Aufgrund der großen Menge an Text im Internet soll bei der Webkorpuserstellung großzügig verfahren werden im folgenden Sinne: In das endgültige Satzkorpus sollen möglichst nur **wohlgeformte Sätze** aufgenommen werden. Sätze von zweifelhafter Qualität werden nicht aufgenommen bzw. in mehreren Schritten wieder entfernt.

Je nach der geplanten Anwendung sind einige dieser Schritte optional. In Kap. 3 wurde bereits auf das allgemeine Vorgehen bei der Verarbeitung großer Textmengen eingegangen, hier soll speziell auf die Besonderheiten von Texten aus dem Web eingegangen werden.

Dies ist ein typischer Fall für die Erstellung von generischen oder fachspezifischen Hintergrundkorpora: Das Web ist die größte Ressource von elektronisch verfügbarem Text für alle Sprachen. Viele Seiten liegen im HTML-Format vor, welches sich vergleichsweise gut zur Weiterverarbeitung für die Korpuserstellung eignet. Andererseits ist die Qualität von Webdokumenten sehr heterogen, was die Qualitätssicherung besonders wichtig macht. Bei der im Folgenden beschriebenen Filterung von unerwünschtem Material werden bewusst viele Beispiele verwendet, um die bei der Korpuserstellung unbedingt notwendige qualitative und quantitative Datenanalyse zu verdeutlichen. Fallen durch oberflächliche oder gar fehlende Analyse bei der Korpuserstellung Qualitätsprobleme erst später in der Anwendung auf, müssen ggf. viele Analyseschritte nochmals durchgeführt werden. Daher gilt insbesondere auch bei der Korpuserstellung ein Grundsatz, welcher auch bei der Entwicklung von sprachtechnologischen Verfahren beachtet werden sollte: Man kann nie zu viel Zeit bei der Datenanalyse verbringen, nur gegebenenfalls zu wenig.

### 4.2.1 Crawling

Das Herunterladen von Webseiten aus dem Internet wird als Crawling bezeichnet. Dazu gibt es spezielle Programme, sogenannte **Crawler**. Für kleine Datenmengen bieten sich Programme *wget* oder *HTTrack* an, für große bis sehr große Datenmengen *Heritrix*. Diese Programme sind frei verfügbar und haben sich an vielen Stellen bewährt. *Heritrix* wurde vom Internetarchiv (Internet Archive 2021) entwickelt und kann damit auch mit gigantischen Datenmengen umgehen. In allen Fällen ist eine Ausgabe als WARC-Dateien möglich (für *HTTrack* wird noch ein Zusatzprogramm benötigt). Auf das WARC-Format wird auch noch einmal in Abschn. 4.3 eingegangen.

Bereits das Crawling kann auf Sprachen oder fachspezifische Dokumente fokussiert werden (Remus und Biemann 2016), was den Umfang des Rohmaterials reduzieren kann, jedoch keine der im Folgenden beschriebenen Schritte ersetzt.

### 4.2.2 Beschränkung auf HTML-Dokumente

Nach dem Crawling liegen die Originaldokumente im lokalen Dateisystem. Diese Originaldokumente haben in der Regel verschiedene Formate. Hier muss zunächst eine Konvertierung in ein einheitliches Textformat vorgenommen werden. Um möglichst alle Zeichen darstellen zu können, bietet sich als Zielformat der **Unicode**-Zeichensatz mit einer Kodierung wie **UTF-8** an.

Die heruntergeladenen Originaldokumente können nicht nur **HTML-Dokumente**, sondern auch PDF-Dokumente, Office-Formate, XML-Dokumente, reine Textdateien (im Format.TXT), eingescannte Texte als.JPG und viele andere Formate sein. In diesen ist die Originalinformation vollständig und unverändert abgelegt. Neben dem Text sind möglicherweise auch Layout-Informationen, Abbildungen, Tabellen usw. enthalten. Wegen dieser verschiedenen möglichen Formate ist allerdings ein einheitlicher Zugriff auf die Texte in diesen Dokumenten problematisch.

Da die überwiegende Mehrzahl aller Texte als HTML-Seiten vorliegt (siehe Schäfer und Bildhauer 2013), bietet sich eine einfache Lösung dadurch an, ausschließlich den Text aus HTML-Seiten weiterzuverarbeiten. Dadurch gehen zwar andere Dokumente für die Verarbeitung verloren, dies lässt sich aber hoffentlich dadurch ausgleichen, dass mehr HTML-Dokumente gecrawlt werden. Auch dies kann jedoch die Ausgewogenheit des Korpus beeinflussen.

### 4.2.3 Text aus HTML-Dokumenten extrahieren

Die Originaldokumente präsentieren die Informationen so, wie es ursprünglich vom Autor oder von der Autorin gedacht war, d. h. neben reinem Text enthält das Dokument möglicherweise noch Tabellen und Abbildungen, deren Information verloren geht und

welche Fragmente und falsch zugeordnete Satzteile bei der Textextraktion verursachen können.

Mögliche Schwächen bei der Konvertierung sind:

- Speziell bei Dokumenten mit anspruchsvollem Layout werden die einzelnen Textteile nicht in der gewünschten Reihenfolge exportiert. Dadurch können einzelne Sätze zerstückelt und falsch zusammengesetzt werden.
- Überschriften sind nicht mehr als solche erkennbar.
- Sonderzeichen werden möglicherweise bei der Konvertierung unleserlich oder weggelassen. Dies betrifft allerdings nur seltene Sonderzeichen und z. B. nicht die üblichen Umlaute.
- Falls in Originaldokumenten die sogenannte harte Silbentrennung verwendet wurde, sorgt sie im konvertierten Text für Bindestriche an den ehemaligen Trennstellen.

Aus diesen Gründen ist es notwendig, bei Auswahl und Konfiguration des eingesetzten Textkonverters zunächst Probeläufe mit Dokumenten aus verschiedenen Quellen vorzunehmen und die Einstellungen zu optimieren.

Wegen dieser Mängel bei der Konvertierung ist es immer angeraten, die Originaldokumente – falls rechtlich möglich – weiterhin zur Verfügung zu halten, um ggf. feststellen zu können, ob ein später merkwürdig erscheinender Satz im Original genauso stand oder erst durch einen Konvertierungsfehler entstanden ist.

Die nachfolgende Textsegmentierung und Tokenisierung verlaufen bei Webkorpora nicht anders als im allgemeinen Fall der Verarbeitung von Text und wurden bereits in Abschn. 3.2.3 besprochen.

#### 4.2.4 Qualitätssicherung

Bei der Erstellung eines generischen Satzkorpus geht es darum, eine Sammlung von Sätzen bereitzustellen, die für ganz verschiedene Zwecke genutzt werden kann. Für einige Anwendungen werden an das Korpus hohe qualitative Ansprüche gestellt werden. Das bedeutet beispielsweise:

- Die im Satzcorpus gesammelten Objekte sollen tatsächlich vollständige und wohlgeformte Sätze sein.
- Die Sätze sollen in der Sprache des Korpus formuliert sein, anderssprachige Sätze sollen während der Korpuserstellung entfernt worden sein.
- Dubletten (d. h. identische Sätze) und Quasidubletten (Sätze mit minimalen Unterschieden) sorgen bei sprachstatistischen Untersuchungen für zusätzliche Probleme und sollten ebenfalls entfernt werden.

Solche Forderungen an die Qualität kann man dadurch erreichen, dass Sätze von zweifelhafter Qualität einfach aus dem Korpus entfernt werden. Andererseits soll das Korpus so groß sein, dass die sich aus der Aufgabe abgeleiteten Informationsbedürfnisse befriedigt werden können. Deshalb dürfen bei der Qualitätssicherung nicht zu viele Sätze allein deshalb entfernt werden, weil eine gewisse Wahrscheinlichkeit für Qualitätsmängel gegeben ist. Dementsprechend wird auch nach Qualitätssicherungsmaßnahmen ein kleiner Anteil nicht wohlgeformter Sätze verbleiben.

Auch ist dabei immer das geplante Einsatzgebiet des Korpus zu beachten: Soll z. B. ein Sprachmodell für Social Media ermöglicht werden, ist es wichtig, kanalspezifische Besonderheiten wie durchgängige Kleinschreibung, Verwendung von Emojis oder Buchstabenverdopplungen nicht wegzufiltern, auch wenn diese nicht der Standardschriftsprache entsprechen, da sonst das Modell mit solchen Eingaben in den Anwendungen nicht gut zurechtkommt.

#### 4.2.4.1 Sprachseparierung auf Dokumentenebene

Bei der Erstellung eines Satzkorpus sollen nur Sätze in einer ausgewählten Sprache gesammelt werden. In den folgenden Beispielen soll das immer die Sprache Deutsch sein. Bei der Sammlung von HTML-Seiten für ein Webkorpus ist eine Beschränkung auf Seiten der **Top-Level-Domain.de** nicht ausreichend, da auch hier erfahrungsgemäß zwischen 5 % und 10 % der Seiten in englischer Sprache sind. Auch die optionale Angabe der Sprache im Header einer HTML-Datei ist nicht immer korrekt.

Die Feststellung der Sprache eines Textes ist umso einfacher, je länger ein Text ist. Aus diesem Grund ist die Sprachidentifikation für ganze Dokumente wesentlich einfacher als für einzelne Sätze oder gar Wörter. Allerdings gibt es auch mehrsprachige Dokumente, die beispielsweise eine Zusammenfassung in englischer Sprache enthalten. Solchen Dokumenten wird entweder gar keine Sprache zugewiesen oder die Sprache mit dem größten Textanteil. Dadurch entstehen Fehler bei der Sprachidentifikation, die eine zusätzliche Sprachidentifikation für einzelne Sätze nötig machen können.

Sprachidentifikation benutzt Trainingsdaten für die zu identifizierenden Sprachen und wird in der Regel auf der Basis von Substrings oder häufigen Wörtern durchgeführt: Einzelne Substrings bzw. Wörter sprechen mit bestimmten Wahrscheinlichkeiten für bestimmte Sprachen. Mithilfe eines automatischen Klassifikators werden diese Wahrscheinlichkeiten für alle Substrings (beispielsweise der Länge 1–4) aus den vorliegenden Trainingsdaten ermittelt. Zur Sprachidentifizierung eines Textes werden diese Wahrscheinlichkeiten für alle solche Substrings im Text zusammengefasst. In der Regel ergibt sich für eine Sprache eine mit Abstand größte Wahrscheinlichkeit, diese wird als Sprache des Textes angesehen. Dokumente ohne eine solch klare Entscheidung sollten verworfen werden.

Für diese Sprachidentifikation müssen zunächst Trainingsdaten bereitgestellt werden. Für die ca. 100–200 größten Sprachen (gemessen an den vorhandenen Datenmengen im Internet) ist dies kein Problem, beispielsweise können zum Training die Texte aus Wikipedia verwendet werden. Für weitere Sprachen wird die Situation schnell komplizierter.

Als einheitliche Quelle für ca. 1000 Sprachen bieten sich religiöse Texte an. So liegt das *Neue Testament* in weitgehend einheitlichem Format vor (vgl. Goldhahn 2013). Zwar unterscheiden sich religiöse Texte als Textsorte sehr von den sonst im Internet üblichen Texten, aber für einen ersten Schritt zu einem ersten Sprachidentifizierer sind diese Texte sehr nützlich, da sowohl die typischen Stoppwörter wie auch die häufigen morphologischen Strukturen enthalten sind. Hat man mit dessen Hilfe mehr Texte in der entsprechenden Sprache gesammelt, so lässt sich aus diesen eine zweite, bessere Version des Sprachidentifizierers bauen, indem man mit den neu gesammelten Texten aus dem Internet eine verbesserte Variante der Sprachidentifikation trainiert.

#### **4.2.4.2 Sprachüberprüfung auf Satzebene**

Nach der Textsegmentierung (siehe Abschn. 3.2.3) in Sätze ist eine zweite Sprachüberprüfung auf Satzebene angemessen. Viele Dokumente, die bei der Sprachidentifikation auf Dokumentebene der Sprache Deutsch zugeordnet wurden, können kleine Textmengen in anderer Sprache enthalten: Zusammenfassungen auf Englisch liefern bei der Textsegmentierung komplette englische Sätze, im Text können aber auch gemischtsprachige Sätze (z. B. mit längeren Zitaten auf Englisch) vorkommen, die entfernt werden sollen.

Dazu kann man den gleichen Sprachidentifizierer wie im vorigen Abschnitt verwenden, allerdings mit leicht verändertem Vorgehen: Es geht darum, die vorher schon einer Sprache A zugewiesenen Texte noch einmal auf Satzebene zu begutachten. Im typischen Fall werden auch die einzelnen Sätze der Sprache A mit großem Abstand vor der nächstwahrscheinlichen Sprache B zugewiesen. Sätze, für die das nicht der Fall ist, sollten verworfen werden. Sie sollten nicht der Sprache B zugeordnet werden, da sie aus einem Text stammen, der vorher bereits als Sprache A klassifiziert wurde.

#### **4.2.4.3 Umgang mit Dubletten und Quasidubletten**

Als Dubletten bezeichnen wir im Korpus mehrfach auftretende, vollkommen identische Sätze. Bei Quasidubletten wird die Forderung von Gleichheit abgeschwächt zu der Forderung nach großer Ähnlichkeit. Da diese Formulierung zunächst nicht streng ist, wird im Folgenden unterschiedliches Vorgehen in Abhängigkeit von der Ähnlichkeit vorgestellt. Die Anzahl von Varianten ähnlicher Sätze erweist sich als wesentlich besserer Indikator für die Bekanntheit von Formulierungen als die Häufigkeit kompletter Sätze im Korpus.

##### **Dubletten**

Bei der Sammlung von Sätzen werden einige Sätze wiederholt auftreten. Es stellt sich die Frage, ob dies von Interesse sein kann. Es gibt folgende grundlegend verschiedene Vorgehensweisen:

1. Die Information aus den einzelnen Dokumenten soll in maximaler Form erhalten werden. Deshalb soll so viel wie möglich in das Korpus übernommen werden. Doppelte Sätze bleiben deshalb erhalten.
2. Mit einem Satzkorpus soll die Vielfalt der Sprache dokumentiert werden. Wiederholungen verfälschen nur die Statistik. Deshalb werden keine Dubletten aufgenommen.

Je nach der geplanten Verwendung des entstehenden Korpus können beide Entscheidungen angebracht sein. Hier soll aber für die zweite Variante argumentiert werden. Man könnte zusätzlich zählen, wie oft die verschiedenen Sätze vorkommen und analog zur Frequenz bei Wörtern hoffen, dass die wiederholt vorkommenden Sätze besonders wichtig sind. Dies bestätigt sich allerdings nicht, wenn man die mehrfach vorkommenden Sätze näher betrachtet. In vielen Fällen kommen ganze Dokumentteile wiederholt vor. Werden beispielsweise Zeitungsartikel zu wichtigen Ereignissen über mehrere Tage lang aktualisiert, basieren viele Artikel auf derselben Pressemeldung. Werden solche Artikel auch noch wiederholt gecrawlt, gibt es große Überschneidungen.

Weiterhin haben die in großer Zahl wiederholt vorkommenden Sätze in der Regel nur eine technische Funktion und sind als Artefakte des Mediums und nicht als Inhalte anzusehen. Sie stehen oft am Ende eines Artikels, werden aber nicht abgetrennt.

### Beispiele für häufig wiederkehrende Sätze

*Hier können Sie den Newsletter ganz einfach und kostenlos abonnieren.*

*Wir freuen uns auf Ihre Kommentare.*

*Die Kommentarfunktion für diesen Artikel ist deaktiviert. ◀*

Aus diesem Grund ist die Häufigkeitsangabe zu Sätzen nicht sinnvoll. Auch verfälscht das wiederholte Zählen von Wörtern in solchen Dubletten die Statistik. Besonders kritisch ist dies für die Berechnung von Kookkurrenzen (Abschn. 5.3), daraus abgeleiteten Wortähnlichkeiten (Abschn. 5.4) und für Sprachmodelle (Abschn. 5.5).

Die Beseitigung von Dubletten ist vom technischen Standpunkt her einfach: Nach einer alphabetischen Sortierung der Sätze nach Textsegmentierung stehen Satz-Dubletten direkt hintereinander und mehrfach auftretende Zeilen können bis auf eine entfernt werden. Alternativ kann man mit einem **Hashverfahren** wie MD5 jedem Satz einen Hashwert zuordnen und bei Sätzen mit gleichem Hashwert davon ausgehen, dass auch die Sätze identisch waren. Dies muss zwar nicht der Fall sein, aber in Tests sind auch bei 100 Mio. Sätzen keine Kollisionen durch gleiche MD5-Hashwerte bei unterschiedlichen Sätzen aufgetreten.

### Quasidubletten

Quasidubletten sind Paare von sehr ähnlichen Sätzen, die sich jedoch ein wenig unterscheiden. Wie bei Dubletten haben wir auch bei Quasidubletten die Möglichkeit, diese

im Korpus zu belassen oder (bis auf einen der Sätze) zu löschen. Sowohl der Grad der Ähnlichkeit wie auch die Art der Unterschiede variieren stark, ebenso die Ursachen für das Entstehen solcher Quasidubletten. Dementsprechend unterschiedlich kann man bei der Behandlung vorgehen. Im Folgenden sollen diese Varianten diskutiert und Vorschläge für den Umgang gemacht werden.

### **Quasidubletten aufgrund der Verwendung verschiedener Anführungszeichen**

Im Deutschen gibt es mehrere Varianten von Anführungszeichen. Deren Nutzung ist den Autoren und Autorinnen freigestellt, und deshalb werden die verschiedenen Varianten auch sämtlich benutzt. Bei großen Quellen (etwa Tageszeitungen) ist die Verwendung zwar einheitlich, aber verschieden zu anderen Quellen. Die von Nachrichtenagenturen verbreiteten Texte werden oft mit nur kleinen Veränderungen übernommen, deshalb findet man in Webkorpora eine große Anzahl von Satzpaaren, die sich nur in der Verwendung von Anführungszeichen unterscheiden. Durch Probleme in der Verarbeitungskette kann auch der Fall eintreten, dass ein außenstehendes Anführungszeichen fälschlicherweise abgetrennt wurde und sich auf diese Weise ein weiterer Satz findet, der sich nur durch Anführungszeichen unterscheidet. In der Praxis entstehen hier meist nur Paare von Sätzen, die sich nur in der Verwendung von Anführungszeichen unterscheiden. Für sprachstatistische Auswertungen haben diese nur geringe Auswirkungen.

#### **Verteilung von Anführungszeichen**

Im Korpus deu-de\_web-wrt\_2019\_100M enthalten rund 9,5 % aller Sätze Anführungszeichen. Deren Verteilung ist in der Tab. 4.1 dargestellt.

Darunter befinden sich 1,4 % Quasidubletten, die sich nur durch Anführungszeichen unterscheiden, dies sind rund 0,13 % aller Sätze im Korpus. ◀

### **Paare von Quasidubletten aufgrund ähnlicher Formulierungen**

Meldungen von Nachrichtenagenturen werden häufig nicht wörtlich übernommen, sondern der Text wird minimal verändert. Dadurch entstehen in den meisten Fällen wieder nur Gruppen aus zwei Quasidubletten. Die folgenden Beispiele zeigen die häufigsten Unterschiede bei solchen von Redakteuren bzw. Redakteurinnen erzeugten Quasidubletten.

**Tab. 4.1** Prozentuale Verteilung von Anführungszeichen im Beispielkorpus

Anführungszeichen	Prozentsatz
„...“	53 %
"... "	41 %
«... » und »...«	6 %

**Redaktionell erzeugte Quasidubletten**

1. Unterschied: Ein Wort mehr, bei meist gleicher Aussage

BEISPIEL:

*„Das ist es, was an seiner Kampagne wirklich neu ist“, sagt Internet-Experte Malbin.*

*„Das ist es, was an seiner Kampagne wirklich neu ist“, sagt der Internet-Experte Malbin.*

2. Unterschied: Ein Wort ausgetauscht, meist durch ein Synonym oder einen Oberbegriff

BEISPIEL:

*„Bei Lufthansa sind die Folgen der Streiks insbesondere im Verkehrsgebiet Europa spürbar“, teilte die Lufthansa mit.*

*„Bei Lufthansa sind die Folgen der Streiks insbesondere im Verkehrsgebiet Europa spürbar“, teilte die Fluggesellschaft mit.*

3. Ein Satzzeichen mehr oder weniger:

BEISPIEL:

*„Das ist für Stromfirmen interessant“, sagt Zinser, der überlegt, die Anlage dann zu vermieten.*

*„Das ist für Stromfirmen interessant“, sagt Zinser, der überlegt die Anlage dann zu vermieten.*

4. Korrekt verlängerte oder verkürzte Sätze:

BEISPIEL:

*„Das ist nicht mehr rational zu erklären“, sagte ein Händler.*

*„Das ist nicht mehr rational zu erklären“, sagte ein Händler und fügte hinzu:*

*„Das hängt ausschließlich mit der Übernahme durch Porsche zusammen“.* ◀

**Quasidubletten durch Variabilität an speziellen Satzpositionen**

Vergleichsweise große Gruppen von Quasidubletten unterscheiden sich jeweils nur an einer oder mehreren festen Positionen im Satz. Häufig handelt es sich dabei um Zahlen, wie in den folgenden Beispielen.

**Beispiele für Quasidubletten mit Zahlen**

*Er war 27 Jahre alt.*

*Er war 31 Jahre alt.*

*Er war 61 Jahre alt.* ◀

### Beispiele für Quasidubletten mit Datums- und Zeitangaben

Diese Seite wurde zuletzt am 14. März 2012 um 20:08 Uhr geändert.  
 Diese Seite wurde zuletzt am 22. März 2017 um 23:38 Uhr geändert.  
 Diese Seite wurde zuletzt am 12. März 2018 um 20:03 Uhr geändert.  
 Diese Seite wurde zuletzt am 3. März 2013 um 15:58 Uhr geändert.  
 Diese Seite wurde zuletzt am 19. März 2015 um 23:49 Uhr geändert.  
 Diese Seite wurde zuletzt am 27. März 2017 um 14:27 Uhr geändert.  
 Diese Seite wurde zuletzt am 22. März 2017 um 23:11 Uhr geändert. ◀

Allein von dieser Art enthält das Korpus deu-de\_web-wrt\_2019\_100M circa 1350 weitere Beispiele.

Speziell im Falle von Ziffern lassen sich solche Quasidubletten relativ einfach identifizieren: Ersetzt man alle Ziffernfolgen durch einen Platzhalter (z. B. #), so erhält man aus den Beispielen oben die vereinheitlichten Varianten:

*Er war # Jahre alt.*

*Diese Seite wurde zuletzt am #. März # um #:# Uhr geändert.*

Löscht man jetzt Dubletten bei den vereinheitlichten Varianten, so verschwindet die Menge der Quasidubletten bis auf ein Element. Dabei sollte man natürlich den entsprechenden Originalsatz aufheben und nicht die vereinheitlichte Form.

Bei der Betrachtung des zweiten Beispiels fällt natürlich noch der Monatsname *März* auf. Völlig zu Recht vermutet man entsprechend viele Sätze auch für die anderen Monatsnamen. Wenn man dies nicht weiter beachten will, bleiben im Korpus zwölf verschiedene vereinheitlichte Varianten, also zwölf Sätze mit den verschiedenen Monatsnamen. Alternativ können wir auch einen zusätzlichen Platzhalter für die Monatsnamen einführen. Natürlich können wir auch die Wörter aus vielen weiteren Gruppen von Wörtern durch Platzhalter zusammenfassen, etwa die Wochentage, Namen von Städten oder Ländern, und vieles mehr.

### Beispiele für Quasidubletten mit Städtenamen

*Das hat das Amtsgericht Hamburg entschieden.*  
*Das hat das Amtsgericht Frankfurt entschieden.*  
*Das hat das Amtsgericht Mülheim entschieden.*  
*Das hat das Amtsgericht Steinfurt entschieden.*  
*Das hat das Amtsgericht München entschieden.*  
*Das hat das Amtsgericht Wiesbaden entschieden.* ◀

Allerdings stellt sich die Frage, ob dadurch nicht auch relevantes sprachliches Wissen aus dem Korpus entfernt wird. Typische rechte Nachbarn von *Amtsgericht* sind nicht alle deutschen Städte, sondern eben nur diejenigen, die ein Amtsgericht besitzen, was z. B. bei Word Embeddings (Abschn. 5.6) ein Signal ist, welches innerhalb der Gruppe

von deutschen Städten ab einer gewissen Größe für Ähnlichkeit in deren Repräsentation sorgt.

Einen allgemeineren Ansatz zum Filtern von Quasidubletten erlauben die in Abschn. 4.2.5 beschriebenen Satzsignaturen, da die in den Beispielen genannten Mengen von Sätzen jeweils die gleiche Satzsignatur besitzen.

### Quasidubletten, die von Interesse sein können (z. B. wegen ähnlich aus gefüllter Leerstellen)

Die folgenden zwei Beispiele sollen demonstrieren, dass solche Quasidubletten, die sich nur an einer Position unterscheiden, durchaus in ein Satzkorpus aufgenommen werden sollten. Möglich wird dadurch die Erstellung von Listen semantisch ähnlicher Wörter, welche eine solche Lücke ausfüllen können. ◀

### Beispielsätze für Muster

Die folgenden Beispielsätze folgen dem Muster [ART] [NN] *sitzt* [perfekt|tief]. Damit werden jeweils Dinge oder Sachverhalte aufgezählt, die entweder perfekt oder tief sitzen. Würde man einige davon als Quasidubletten betrachten und löschen, dann geht sprachliche Information verloren: Es wäre schwieriger zu erschließen, dass die Formulierung *sitzt perfekt* sich typischerweise auf Kleidungsstücke u. Ä. bezieht, *sitzt tief* dagegen auf Gefühle. Außerdem kann es von Interesse sein, welche Gefühle typischerweise *tief sitzen* – einerseits bei der Erstellung von Lexika, andererseits aber wie erwähnt auch bei der Berechnung von Wortähnlichkeiten.

*Die Hose sitzt perfekt.*

*Die Frisur sitzt perfekt.*

*Der Maßanzug sitzt perfekt.*

*Die Kombination sitzt perfekt.*

*Der Schock sitzt tief.*

*Der Frust sitzt tief.*

*Der Schmerz sitzt tief.*

*Der Schrecken sitzt tief.*

*Die Wut sitzt tief.* ◀

Betrachtet man zusätzlich noch eine mögliche Variabilität an anderer Stelle im Satz, so können die gefundenen semantisch ähnlichen Wörter zusätzlich mit einer Häufigkeit als Rangfolge versehen werden.

### Beispielsätze für eine Redewendung

Hier soll die Redewendung *etwas steht ihm/ihr ins Gesicht geschrieben* untersucht werden. Dazu betrachten wir das Muster [ART] [NN] [Lemma:stehen] [PPER] *ins Gesicht geschrieben*.

An der Position NN stehen die Gefühle, welche durch einen entsprechenden Gesichtsausdruck gezeigt werden. Auch hier könnte man diese Sätze als Quasidubletten betrachten und bis auf ein Beispiel löschen, aber dadurch verliert man viel Information. Neben der Liste der ausgedrückten Gefühle, ist durch die Variabilität in den anderen Satzteilen (nämlich bei Verbform und Personalpronomen) zusätzlich eine implizite Häufigkeitsangabe enthalten: *Angst* und *Entsetzen* stehen Menschen häufiger ins Gesicht geschrieben als *Ekel* und *Frust*.

*Die Angst steht ihr/ihm/ihnen/mir ins Gesicht geschrieben.*

*Die Angst stand ihr ins Gesicht geschrieben.*

*Das Entsetzen stand ihnen ins Gesicht geschrieben.*

*Das Entsetzen steht ihm/ihnen/ihr ins Gesicht geschrieben.*

*Der Ekel stand ihm ins Gesicht geschrieben.*

*Der Frust steht ihm ins Gesicht geschrieben.*

*Der Schock stand ihm ins Gesicht geschrieben.*

*Der Schock steht ihnen im Gesicht geschrieben.*

*Der Schock steht ihnen/ihr ins Gesicht geschrieben.*

*Der Schreck stand ihr ins Gesicht geschrieben.*

*Der Schreck steht ihm ins Gesicht geschrieben.* ◀

#### 4.2.4.4 Musterbasierte Entfernung nicht-wohlgeformter Sätze

Die Erkennung nicht-wohlgeformter Sätze in einem Satzcorpus zum Zweck ihrer Entfernung ist ein wichtiger Schritt bei der Qualitätssicherung. Bei den Gründen, Sätze als nicht-wohlgeformt zu erkennen, denken wir zunächst an Sätze mit orthographischen oder grammatischen Fehlern. Ohne morphologische bzw. syntaktische Analyse sind solche Fehler jedoch kaum zu erkennen, und selbst hier reicht es nicht, eine entsprechende Verarbeitungskomponente (Abschn. 3.2) einzusetzen, da diese Komponenten normalerweise robust gegenüber Fehlern sind und diese nicht explizieren. Deshalb soll der Schwerpunkt hier auf musterbasierte Verfahren gelegt werden: Mit regulären Ausdrücken oder vergleichbaren Methoden können Sätze identifiziert werden, die mit großer Wahrscheinlichkeit nicht-wohlgeformt sind und aus dem Satzcorpus entfernt werden sollen. Diese strukturellen Kriterien werden zwar manchmal auch solche Sätze aussondern, die nach linguistischen Kriterien als korrekt zu betrachten sind, aber diese sollten zumindest strukturelle Besonderheiten aufweisen.

Es gibt verschiedene Gründe für das Vorhandensein solcher Sätze im Korpus:

- Verarbeitungsfehler bei der Korpuserstellung: Bestandteile von Überschriften oder Bildunterschriften können an Sätze angefügt worden sein.
- Autorenfehler: Speziell bei Quellen ohne redaktionelle Kontrolle ist die Fehlerhäufigkeit größer. Außerdem werden bei Kommentaren und in sozialen Netzen die üblichen Regeln bewusst verletzt, beispielsweise durch Kleinschreibung aller Wörter.

Die meisten der folgenden Regeln enthalten einen Schwellwert, dessen Überschreitung den entsprechenden Satz als nicht-wohlgeformt kennzeichnet. Diese Schwellwerte sollten vor Anwendung dadurch überprüft werden, dass man sich Sätze anzeigen lässt, die entsprechend der Schwellwerte nicht wohlgeformt sind. Jede Wahl eines solchen Schwellwertes wird dazu führen, dass einige offensichtlich korrekte Sätze zurückgewiesen werden und einige Sätze trotz Mängel nicht.

Entsprechend der geplanten Nutzung können Änderungen der Schwellwerte angebracht sein.

In der Praxis zeigt sich, dass in vielen Fällen die betroffenen Sätze gleich gegen mehrere der im Folgenden aufgezählten Regeln zur Wohlgeformtheit verstößen.

Die folgenden Regeln haben sich für das Deutsche bewährt. Für andere Sprachen sind Änderungen angebracht, speziell wenn andere Zeichensätze verwendet werden oder andere Regeln für die Setzung von Satzzeichen gelten. Die Regeln können oft als reguläre Ausdrücke angegeben werden.

- **Satzlänge beschränken: Länger als 256 Zeichen.**

Natürlich gibt es keine linguistisch motivierte absolute Längenbeschränkung für deutsche Sätze. Aber für Beispielsätze zur Verwendung von Wörtern oder grammatischen Strukturen ist eine solche Beschränkung durchaus sinnvoll. Ohne Längenbeschränkung werden Sätze mit am Beginn des Satzes fehlerhaft angefügten langen Textstücken nicht entfernt.

- **Satzanfang und -ende kontrollieren:**

Im Deutschen beginnen Sätze mit Großbuchstaben oder Anführungszeichen und enden mit Satzendezeichen (Punkt, Ausrufezeichen und Fragezeichen) oder Ausführungszeichen. Hier wird der Doppelpunkt als Satzende akzeptiert; das muss mit der vorherigen Textsegmentierung in Sätze abgestimmt sein. Regulärer Ausdruck:  
"^\[A-ZÄÖÜ"'\] . \* [ . : ! ? " ' ] \\$ ``"

- **Sätze mit gesperrtem Text löschen:**

Gesperrter Text ist bei der Verarbeitung problematisch: Leerzeichen nach jedem Buchstaben im Wort erzeugen (scheinbare) Wörter bestehend aus Einzelbuchstaben, unerwünschte Wortkookkurrenzen (siehe Abschn. 5.3) usw. Außerdem: Sätze mit gesperrtem Text erfüllen oft noch weitere der hier angegebenen Kriterien. Regulärer Ausdruck: "( . ) { 8 } "

- **Sätze mit zwei aufeinanderfolgenden Leerzeichen löschen:**

Mehrere aufeinanderfolgende Leerzeichen können ein Fehler des Autors bzw. der Autorin sein, aber auch durch fehlerhaftes Zusammenziehen mehrerer Textschnipsel entstehen.

**BEISPIEL:** *Kommunale Abfallwirtschaft rechnet mit 500 DM jährlich aha München (Eigener Bericht).*

- **Sätze mit allzu vielen Kommas löschen:**

Es gibt keine Maximalzahl erlaubter Kommas in deutschen Sätzen, aber bei einer Maximallänge von 256 Zeichen (s. o.) kommen zehn Kommas eher in Aufzählungen vor als in einem wohlgeformten Satz. Regulärer Ausdruck: "( . , ) { 10 } "

Beispiel: *Krupp AG Hoesch-Krupp, Gea Vorzugsaktien, Gehe, Gerresheimer, Glunz Vorzugsaktien, Grohe Vorzugsaktien, Hornbach Vorzugsaktien, Kampa Haus, Moksel, O. Reichelt, SAP Vorzugsaktien, Spar Handel Vorzugsaktien, Strabag, Villeroy & Boch sowie Weru.*

- **Sätze mit allzu vielen Punkten löschen:** `regexp`

Punkte im Satzinneren kommen in Datumsangaben vor, bei manchen Abkürzungen usw. Sieben oder mehr Punkte sind aber kaum sinnvoll. Regulärer Ausdruck: "`(.*\.) {7}`"

- **Sätze mit mehr als 50 Leerzeichen löschen** (45 Leerzeichen sind durchaus noch sinnvoll möglich.):

Eine Maximallänge von 256 Zeichen und über 50 Leerzeichen sprechen für viele kurze Wörter im Satz; speziell bei langen Sätzen ist das ungewöhnlich.

Beispiel: *FC Köln – 29 796 Karlsruher SC 25 000 29 060 Eintracht Frankfurt 28 300 28 487 VfL Bochum – 24 274 Bayer Leverkusen 16 800 22 875 MSV Duisburg – 20 447 Bayer Uerdingen 31 415 17 349 SC Freiburg – 17 029 Dynamo Dresden – 16 588 Insgesamt 292 345*

- **Sätze mit anteilig zu vielen Leerzeichen löschen:** Statt der absoluten Zahl von Leerzeichen lässt sich auch deren relative Anzahl beschränken. Hier sind 30 % der Gesamtzeichenanzahl eine sinnvolle Obergrenze für die maximal erlaubte Anzahl an Leerzeichen.

Beispiel: *Elferwette: 2 1 2 1 1 1 2 1 1 0 2.*

- **Sätze mit mehreren Ausrufe- oder Fragezeichen löschen:** `regexp` "`[!?].*![!?]"`

Manche Autoren oder Autorinnen verwenden mehrere Ausrufezeichen oder Fragezeichen am Satzende, um deren Wirkung zu verstärken. Solche absichtlichen Regelabweichungen sind oft mit anderen Regelverletzungen im gleichen Satz verbunden.

Beispiel: *Wenn nicht dieses Tape dann gar keinz!!!!*

- **Sätze mit Sonderzeichen löschen, beispielsweise mit den folgenden Zeichen oder Zeichenfolgen:** "`>>`", "`++`", "`*"`", "`~`", "`|`" oder "`[[`"

Solche Sonderzeichen treten zwar selten in Eigennamen oder technischen Texten auf, in den meisten Fällen entstehen sie jedoch durch Probleme in der Vorverarbeitung.

- **Sätze mit vielen gleichen Sonderzeichen löschen**, z. B. mindestens vier schließende Klammern, drei Schrägstriche (/) oder drei Und-Zeichen (&).

Während jeweils wenige der genannten Sonderzeichen pro Satz erlaubt werden, wird eine größere Anzahl pro Satz nicht mehr erlaubt.

- **Sätze mit vielen Großbuchstaben oder vielen Ziffern hintereinander löschen:**

Das Schreiben in Großbuchstaben im Internet ist als schlechter Stil verpönt und wird häufig auch zusammen mit anderen problematischen Schreibweisen verwendet. Außerdem würden durch die Aufnahme ins Korpus zahllose neue Wörter von zweifelhaftem Wert entstehen. Lange Ziffernfolgen kommen in normalen Sätzen ebenfalls nicht vor. Regulärer Ausdruck: "`[A-Z-] {20}`", "`[0-9 \.,/-] {16}`"

Beispiel:

*DANKE AN EUCH ALLE, DASS IHR SO NETTE SACHEN HIER REINSCHREIBT.*

- **Sätze mit Leerzeichen vor Punkt bzw. Komma löschen:**

Im Deutschen werden Satzzeichen wie Punkt und Komma ohne Leerzeichen an das vorhergehende Wort angefügt. Manche Autoren oder Autorinnen lassen dazwischen ein Leerzeichen. Diese Sätze enthalten oft auch weitere Fehler.

- **Sätze mit Initialen am Ende löschen:**

Im Deutschen ist es fast unmöglich, dass das letzte Wort im Satz nur aus einem Buchstaben besteht. Eine der ganz wenigen Ausnahmen sind die römischen Zahlen I, V und X. Viel häufiger entstehen solche Sätze durch fehlerhafte Textsegmentierung. Regulärer Ausdruck: " [A-Za-z] .\\$"

- **Kurze Sätze mit vielen Ziffern vor Punkt löschen:**

Bei der Segmentierung entstehen häufig auch sehr kurze Sätze zweifelhafter Qualität, die relativ viele Ziffern vor dem Punkt am Ende enthalten. Das zweite Beispiel ist sicher akzeptabel, würde mit dieser Regel aber auch entfernt. Regulärer Ausdruck für Sätze mit maximal 45 Zeichen: "[0-9.,/-]{6}\\$" and length(sentence) < 45

- Beispiele:

*29. Kapitel Ev.Joh.4,27.*

*Meine Maße sind 91–60–91.*

- **Extrem kurze Sätze mit Punkt löschen:** Man könnte extrem kurze Aussagesätze von maximal 12 Zeichen löschen. Kurze Frage- oder Ausrufesätze sollten dagegen nicht gelöscht werden.

## 4.2.5 Evaluierung und Ranking von Sätzen

Viele der im letzten Abschnitt formulierten Ausschlusskriterien lassen sich umformen in eine musterbasierte Bewertung von Sätzen. Auf der Basis verschiedener Muster werden für jeden Satz Strafpunkte vergeben und diese werden pro Satz addiert. Dadurch erhält jeder Satz eine Strafbewertung und die Sätze mit der besten Bewertung werden bevorzugt ausgewählt.

Eine andere Auswahl der Sätze kann über ihre syntaktische Struktur erfolgen. Diese Struktur lässt sich einfach über die sogenannte Satzsignatur beschreiben: Dies ist die Folge der POS-Tags der Wörter im Satz. Häufige Satzsignaturen entsprechen typischen Satzstrukturen und lassen sich speziell für die automatische Weiterverarbeitung gut benutzen.

### 4.2.5.1 Ranking mittels GDEX

Ein Ranking für Sätze unter dem Namen *Good Dictionary Examples (GDEX)* wurde zuerst von A. Kilgarriff et al. (2008) entwickelt, um mit automatischen Mitteln gut geeignete Beispielsätze für Lexikographen und Lexikographinnen zur Unterstützung der

Wörterbuchherstellung anzubieten. Das Originalmaß ist eine Ordnung auf den Beispielsätzen zu einem ausgewählten Wort, da es die Position dieses Wortes im Satz berücksichtigt. Die hier vorgestellte Variante ist unabhängig von der Wortposition und ordnet die gesamte Menge aller Sätze im Korpus. Die Auswahl der Beispielsätze erfolgt entsprechend dieser Ordnung. Damit werden für häufige Wörter zuerst Beispielsätze mit guter Bewertung angezeigt, da es vermutlich viele davon gibt. Für sehr seltene Wörter muss es zwar keine Sätze mit guter Bewertung mehr geben, aber immer noch werden alle vorhandenen Beispielsätze angezeigt.

Zur Ermittlung der Bewertung werden für bestimmte Muster im Satz Strafen vergeben. Sowohl bei der Auswahl der Muster wie auch der Höhe der Strafen gibt es Gestaltungsspielraum; die folgenden Beispiele sind deshalb als Vorschläge zu verstehen. Es gibt zwei Typen von Regeln: Zum einen wird das Vorkommen bestimmter Muster bestraft, zum anderen wird für einen Messwert zum Satz (z. B. Länge in Zeichen) ein Zielwert vorgegeben und Abweichungen vom Zielwert werden bestraft.

### **Zu bestrafende Muster**

- Anzahl der Sonderzeichen außer den wichtigsten Satzzeichen [ , . ! ? ], Strafe pro Zeichen: 5 Punkte
- Anzahl der regulären Satzendezeichen [ . ! ? ] minus 1, Strafe pro solches Zeichen: 3 Punkte
- Ziffern: Strafe von 5 Punkten für jede Ziffer
- Aufeinanderfolgende Großbuchstaben, Strafe von 5 Punkten für jedes Paar
- Ungerade Anzahl Anführungszeichen, Strafe: 20 für ein isoliertes Anführungszeichen, 60 für drei Stück.

### **Abweichung von Zielwerten**

Als Zielwert wird entweder ein Zahlenwert oder ein Zielintervall vorgegeben. Abweichungen um denselben Faktor nach oben oder unten werden gleich bestraft. Die Zielwerte weichen manchmal von den Mittelwerten im Korpus ab, um für bessere Lesbarkeit zu sorgen.

- **Satzlänge in Zeichen**, Ziel: etwas kürzer als der Mittelwert von etwa 110. Zielintervall 80–110 Zeichen. Strafe bei Abweichung um Faktor 2 von Grenze = 10 Punkte.
- **Anzahl der Stoppwörter** (gemessen als Anzahl der Top-100-Wörter im Satz): Mittelwert ca. 40 % der Wörter. Strafe bei Abweichung um Faktor 2 vom Mittelwert: 10 Punkte.
- **Seltenstes Wort**: Gewünschter logarithmierter Rang zur Basis 2 für seltenstes Wort: 15. Strafe 4 für Abweichung um Faktor 2. Damit werden Sätze bestraft, deren seltenstes Wort nicht unter den Top-32.768 Wörtern ist. Speziell werden Sätze mit extrem seltenen Wörtern stark bestraft.

- **Mittlerer Rang der Wörter im Satz:** Zielwert: 8000. Strafe für Abweichung um Faktor 2: 10 Punkte. Dadurch werden Sätze mit mehreren seltenen Wörtern zusätzlich bestraft.
- **Mittlere Wortlänge,** Ziel: 6 (Mittelwert ist 6,2). Strafe für Abweichung um Faktor 2: 50 Punkte.

Insgesamt sorgt das Ranking für eine bessere Lesbarkeit der Beispielsätze, wie im folgenden Beispiel demonstriert werden soll.

#### Beispielsätze für Kanzleramt

Die Tabelle Tab. 4.2 zeigt Beispielsätze für das Wort *Kanzleramt*. Die 2572 Belegstellen wurden wie beschrieben bewertet und geordnet. Die Beispiele sind ausgewählt an der entsprechenden Position in der Rangliste (d. h. 0 % bedeutet bestbewerteter Satz mit kleinstster Strafe, 99 % bedeutet fast am Ende der Rangliste).

In der Rangliste weiter vorn stehende Sätze zeichnen sich tatsächlich durch bessere Lesbarkeit aus: geringere Satzlänge, einfachere Struktur und bekanntere und kürzere Wörter. ◀

**Tab. 4.2** Beispielsätze für das Wort *Kanzleramt*

Position	Strafe	Satz
0 %	0,2	Den Segen schreiben dann andere Sternsinger an die Wand im Kanzleramt
10 %	8	Akten aus dem Kanzleramt, den Ministerien und von Privatpersonen lagen zum Lesen bereit
20 %	10	Seine Vorschläge für ein modernes Sexualstrafrecht lagen seit dem letzten Jahr auf dem Tisch – und wurden im Kanzleramt verschleppt
40 %	24	Nach 100 Tagen Amtszeit rief die Ministerin des Bundes zum Erfahrungsaustausch im Kanzleramt auf
60 %	37	Zum Fest der Hoffnung und des Lebens laden wir Sie ein, auf der Eingangsebene der Südseite (Reichstag/Kanzleramt) im Hauptbahnhof mitzufeiern
80 %	55	War er einer der Abweichler, die 1972 den Misstrauensantrag gegen die sozialliberale Koalition scheitern ließen und somit Willi Brandt im Kanzleramt hielten?
95 %	84	Jüngste Engagements führten sie mit „Clivia“ und der Spoliansky-Revue an die Komische Oper Berlin, mit „Frau Luna“ in das Tipi am Kanzleramt oder mit „Coco“ an das Theater Bern
99 %	109	Der fast 70-seitige Entwurf eines „Gesetzes zum ordnungspolitischen Rahmen der Krankenhausfinanzierung ab dem Jahr 2009 – Krankenhausfinanzierungsrahmengesetz – KHFG“ mit Stand vom 22. Juli 2008 wurde vom Kanzleramt regelrecht „kassiert“

### 4.2.5.2 Typische Sätze

Ein zweites, völlig anderes Rankingverfahren ordnet die Sätze entsprechend ihrer syntaktischen Struktur und ermöglicht so, die Wörter in typischer Umgebung zu zeigen. Solche Sätze sollen hier typisch genannt werden. Diese typischen Sätze eignen sich besonders zur Auswahl von Sätzen zur weiteren automatischen Bearbeitung.

Zur Ermittlung der typischen Sätze wird folgendermaßen vorgegangen:

Im ersten Schritt wird für jeden Satz die sogenannte Satzsignatur ermittelt: Dazu wird der Satz mittels POS-Tagging (siehe Abschn. 3.2.4) annotiert. Danach werden die Wörter des Satzes ignoriert und die Folge der POS-Tags bildet die Satzsignatur.

Die Liste der häufigsten Satzsignaturen ist in der Tab. 4.3 dargestellt.

Das Ergebnis entspricht nicht ganz den Erwartungen, solange Quasidubletten nicht entfernt wurden: Die häufigste Satzsignatur in der Tabelle hat ihre Ursache in Quasidubletten von Sätzen wie „*Diese Seite wurde zuletzt am 17. November 2008 um 23:22 Uhr bearbeitet.*“

Solche zu Quasidubletten gehörenden Satzsignaturen lassen sich mithilfe von Kriterien an die Variabilität der Wörter an den einzelnen Positionen bestimmen: Eine Satzsignatur beschreibt vermutlich eine Menge von Quasidubletten, wenn die folgenden Bedingungen erfüllt sind:

**Tab. 4.3** Häufigste Satzsignaturen. (Quelle: deu-de\_web-wrt\_2019\_100M)

Rang	Satzsignatur	Häufigkeit
1	{PDAT}{NN}{VAFIN}{ADV}{APPRART}{ADJA}{NN}{CARD}{APPR}{CARD}{NN}{VVPP}{\$.}	23.933
2	{ART}{NN}{VAFIN}{ADJD}{\$.}	23.569
3	{ART}{NN}{VAFIN}{VVPP}{\$.}	15.346
4	{ART}{NN}{VAFIN}{ADJD}{VVPP}{\$.}	14.756
5	{PWS}{VAFIN}{ART}{NN}{\$.}	14.593
6	{ART}{NN}{VAFIN}{ADV}{ADJD}{\$.}	14.256
7	{ART}{NN}{VAFIN}{APPR}{ART}{NN}{VVPP}{\$.}	11.853
8	{ART}{ADJA}{NN}{\$.}	10.461
9	{ART}{NN}{VVFIN}{ART}{NN}{\$.}	10.403
10	{ART}{NN}{VAFIN}{ART}{ADJA}{NN}{\$.}	10.044
11	{ART}{NN}{VVFIN}{APPR}{ART}{NN}{\$.}	9.817
12	{ART}{NN}{VVFIN}{CARD}{NN}{\$.}	9.570
13	{PWS}{VVFIN}{ART}{NN}{\$.}	8.911
14	{ART}{NN}{VAFIN}{ADJD}{ADJD}{\$.}	8.194
15	{PPER}{VVFIN}{ART}{NN}{\$.}	8.125

- Mindestens die Hälfte aller Positionen im Satz müssen weitgehend unveränderlich sein (d. h. in mindestens einem Anteil von  $p=0,5$  der Sätze steht dasselbe Wort an dieser Position) oder vom Typ {CARD}. Dies darf nicht ausschließlich Stoppwörter betreffen.
- An mindestens drei Positionen muss  $p>0,9$  oder an vier Positionen  $p>0,8$  oder an fünf Positionen  $p>0,7$  erfüllt sein.
- Zusätzlich lässt sich die Entropie der Verteilung der Wörter pro Position im Satz ermitteln. Der Median dieser Entropiewerte (gemittelt über alle Positionen im Satz) ist für Mengen von Quasidublettten oft extrem klein, meist  $<0,05$ .

Interessant ist auch die Verteilung der Satzsignaturen: Die häufigsten Satzsignaturen sind nicht so häufig wie man erwarten könnte. Auf der anderen Seite treten extrem viele Satzsignaturen nur einmal auf:

Im Korpus deu-de\_web-wrt\_2019\_100M von 100 Mio. Sätzen gibt es knapp 82 Mio. verschiedene Satzsignaturen, davon treten reichlich 78 Mio. nur einmal auf. Und nur rund 290.000 haben eine Anzahl von 10 oder mehr. Diese Satzsignaturen mit einer Mindesthäufigkeit von 10 decken 12,4 % der Sätze des Korpus ab. Der Anteil der Sätze mit typischer Struktur ist damit vergleichsweise gering.

Für Anwendungen interessant sind jeweils mehrere Sätze mit gleichen oder ähnlichen Satzsignaturen und einem oder mehreren ausgewählten Wörtern an festen Positionen.

---

## 4.3 Speicherformate

Es gibt viele verschiedene Methoden, um Texte zu speichern. Da Texte von verschiedenen Tools verarbeitet werden sollen und die Textmengen sehr groß sein können, lohnt eine Betrachtung der verschiedenen Möglichkeiten. Dabei gibt es keine Lösung, die für alle Anwendungen vorzuziehen wäre, sondern eher eine Liste von verschiedenen komplexen Lösungen, die für verschiedene Szenarien jeweils Vor- und Nachteile besitzen.

- Dokumentenarchiv: Alle Dokumente werden an einem zentralen Speicherort (etwa einem Verzeichnis) abgelegt. Eine Konvertierung der Dokumente erfolgt nicht, die Dokumente liegen in ihrem Originalformat (z. B. neben HTML auch in Microsoft Word oder als PDF) vor. Sie enthalten außer reinem Text möglicherweise auch Tabellen und Bilder. Diese Dateiformate sind teilweise nicht transparent, man kann also nicht ohne weiteres den enthaltenen Text maschinell verarbeiten.
- Webarchiv: Handelt es sich bei den Dokumenten ausschließlich um HTML-Dateien, ist die Situation deutlich besser. HTML-Dokumente enthalten den Text unverschlüsselt und für Maschinen wie für Menschen lesbar, was nur eine vergleichsweise einfache Einleseprozedur zur Extraktion des Textes erfordert. Eine Eigenentwicklung ist speziell dann sinnvoll, wenn man über viele HTML-Dateien im gleichen Format

verfügt, beispielsweise aus einem **Content-Management-System**, aus denen zusätzliche Metadaten gelesen werden können. Für den generellen Fall gibt es viele fertige Programme, um HTML-Dokumente weiter zu verarbeiten. Da bei der Sammlung von HTML-Seiten typischerweise mindestens eine Datei pro Seite entsteht (und zusätzliche Dateien für Bilder verschiedenster Art), hat sich ein zusätzliches Format mit dem Namen WARC (Web ARChive) durchgesetzt, mit dem sich mehrere Millionen HTML-Seiten pro WARC-Datei speichern lassen. WARC-Dateien haben noch mehr Vorteile: Die HTML-Seiten können nicht nur wahlweise mit oder ohne die enthaltenen Bilder gespeichert werden, sondern es gibt rund um dieses Format verschiedene Tools: Mehrere **Webcrawler** erlauben das Abspeichern der heruntergeladenen Webseiten als WARC-Datei. Darunter sind populäre kleine Tools wie *wget* (<https://www.gnu.org/software/wget/>) und auch große Programme zur Webarchivierung wie *Heritrix* (<https://github.com/internetarchive/heritrix3>). Außerdem gibt es Tools zur Weiterverarbeitung von WARC-Dateien sowie ganze Suchmaschinen, die in großen Mengen von WARC-Dateien suchen können.

- XML-Format, insbesondere **TEI** (Text Encoding Initiative) (siehe <https://tei-c.org/>): Im Vergleich zu HTML erlaubt XML wegen seiner größeren Allgemeinheit, Texte mit mehr zusätzlichen Informationen auszuzeichnen. Speziell im Bereich der Geisteswissenschaften ist TEI zum Defacto-Standard geworden. Für gedruckte Werke gibt es hier Annotationsmöglichkeiten nicht nur zur Dokumentenstruktur, sondern auch für Metadaten und für besondere Strukturen wie sie z. B. in Wörterbüchern oder Theaterstücken vorkommen. Da die zusätzlichen Informationen auch auf unterschiedliche Weise mittels XML oder auch TEI kodiert werden können, ist eine wahllose Zusammenstellung solcher Dokumente aus verschiedenen Quellen zu einem Korpus problematisch. Der einheitliche Zugriff auf inhaltlich gleiche, aber verschiedenen repräsentierte Annotationen ist nur schwer zu realisieren. Stammen die XML-Dateien dagegen aus einer Quelle (etwa dem Content-Management-System eines Verlages oder einer einzigen Edition), dann kann ein Korpus aus solchen XML-Dateien durchaus sinnvoll sein.
- **JSON:** Die JavaScript Object Notation wurde als Datenaustauschformat entwickelt. JSON-Objekte sind einfacher strukturiert und kompakter als in XML, für den Menschen sind sie deshalb besser lesbar. Strukturierte Daten wie Wörterbucheinträge können damit in ihrer hierarchischen Struktur abgebildet werden; JSON wird auch als Kommunikationsformat mit NoSQL-Datenbanken verwendet, s. u.
- Zeilenformat: Ist man ausschließlich an dem Text aus den Dokumenten interessiert, so kann es sinnvoll sein, den Text zunächst in Sätze zu zerlegen und nur diese einzelnen Sätze zeilenweise zu speichern. Wenn die logische Reihenfolge der Sätze beibehalten werden soll, dann können die Sätze in der Originalreihenfolge gespeichert werden. Wenn aber Wissen über die Verwendung von Wörtern einer Sprache gesammelt werden soll, dann ist die Reihenfolge der Sätze unwichtig und auch eine zufällige Reihenfolge möglich. Von den hier genannten Formaten ist es das am besten lesbare für Menschen.

Ein Blick auf die Datei zeigt dem bzw. der Betrachtenden das Format und einen Ausschnitt aus der Textmenge, an dem zu sehen ist, wie gut der Text in Sätze zerlegt wurde. Möglicherweise erkennt man sofort Mängel an der Textsegmentierung oder Probleme mit Sonderzeichen.

- Spaltenformat, speziell die Formate ConLL und ConLL-U (<https://universaldependencies.org/format.html>, vgl. Hajič et al. 2009): In diesem weniger kompakten und schlechter lesbaren Format steht nur ein Wort oder Satzzeichen in jeder Zeile und der Text wird sozusagen von oben nach unten gelesen. Als Trenner zwischen aufeinanderfolgenden Sätzen wird eine zusätzliche Leerzeile eingeführt. Dieses Format ist vorteilhaft, wenn zu den einzelnen Wörtern noch zusätzliche Informationen hinzugefügt werden sollen. Dies kann dann auf der gleichen Zeile jeweils hinter dem Wort erfolgen. Solche Informationen können die Grundform des Wortes oder seine Wortart (POS-Tag) sein. Auch kompliziertere Annotationen sind möglich: Zusammenhängende Wortgruppen wie z. B. Eigennamen können ebenso annotiert werden wie eine Dependenzstruktur (Abschn. 3.2.4) über einem Satz.

Das Zeilen- und das Spaltenformat erlauben eine übersichtliche Darstellung von Sätzen ohne oder mit zusätzlicher Annotation, aber die Suche darin ist aufwendig und erst einmal nur sequenziell (z. B. mit regulären Ausdrücken) möglich. Dies ist in Zeiten von Internet-Suchmaschinen natürlich hochgradig unbefriedigend und für größere Datens Mengen inakzeptabel. Deshalb betrachten wir zwei verschiedene Speichertechniken, die allgemeine Anfragen schnell beantworten. In beiden Fällen ist der Aufbau der Datenstruktur mit etwas Zeitaufwand und zusätzlichem Speicherbedarf verbunden, da zusätzliche Indexe aufgebaut werden, welche die späteren Suchanfragen drastisch beschleunigen.

- Relationale Datenbanken: Sowohl die Sätze aus dem Zeilenformat wie auch die Wörter aus dem Spaltenformat werden mit den zusätzlichen Angaben in mehrere Tabellen einer relationalen Datenbank übernommen. Werden dann Daten gruppiert, lassen sich so Anzahlen zu Wörtern ermitteln und mithilfe zusätzlicher Indexe lässt sich die Suche drastisch beschleunigen. Die universelle Datenbanksprache SQL ermöglicht sehr mächtige Datenbankanfragen, auch für vorher nicht spezifizierte Fragestellungen. In diesem Sinne ist die Speicherung eines Korpus in einer relationalen Datenbank eine gute Wahl, wenn die Menge der geplanten Abfragen später erweitert werden soll. Von Nachteil sind der möglicherweise große benötigte Plattenplatz sowie die Tatsache, dass es weiterhin Anfragen an die Datenbank geben wird, die nicht so schnell wie gewünscht beantwortet werden. Dies betrifft insbesondere die Suche mit komplexen regulären Ausdrücken, die im schlimmsten Fall ein sequenzielles Durchlaufen aller Daten erfordern. Hier ist es die Aufgabe des Datenbankdesigners bzw. der -designerin, dies im Vorfeld zu berücksichtigen. Zusammengefasst ist die Nutzung von Datenbanken angebracht und die Suchzeit unschlagbar kurz, wenn auf der Basis ganzer Wörter gesucht wird oder nach anderen

Bestandteilen, die vorher indexiert worden sind. Durch sogenannte **Joins** sind auch anspruchsvolle, kombinierte Suchanfragen effektiv ausführbar.

- NoSQL Datenbanken: diese sog. schemafreien Datenbanken erlauben das Speichern und Indizieren einer variablen Anzahl von leicht erweiterbaren Feldern pro Datenobjekt, im Gegensatz zu relationalen Datenbanken, wo fehlende Felder mit Nullzeigern gefüllt werden. Eine für Korpora gern und oft eingesetzte Technologie ist Elasticsearch (Gormley und Tong 2015), basierend auf der Apache Lucene-Bibliothek, da diese Suchmaschinentechnologie primär für die Indizierung von Dokumenten entwickelt wurde. Durch eine verteilte Cloud-Architektur und redundante Speicherung können hier sehr schnelle Zugriffszeiten auch für extrem große Korpora erreicht werden, siehe z. B. (Panchenko et al. 2018).
- PAT Trees: Zusätzlich gibt es spezielle Datenstrukturen zur Suche in Texten. Dabei wird das oben beschriebene Spaltenformat in eine Baumstruktur von sog. suffix trees (vgl. Baeza-Yates und Ribeiro-Neto 1999) verwandelt, sodass eine Suche mit regulären Ausdrücken auf mehreren Spalten simultan möglich ist. Die Speicherstruktur ist speziell für diese Suchen ausgerichtet. Der benötigte Speicherplatz ist geringer als bei einer relationalen Datenbank und die Suche mit regulären Ausdrücken ist kaum langsamer als die Suche nach Wörtern. Dieser Leistungsfähigkeit steht allerdings eine etwas eingeschränkte Anfragesprache gegenüber.

---

## 4.4 Indexierung

Im Abschn. 4.3 wurden mehrere Formate vorgestellt, in denen Texte bzw. Korpora gespeichert werden können. Hier soll beschrieben werden, wie wir in diesen Daten effizient suchen können. Seit es im Internet Suchmaschinen gibt, sind wir gewöhnt, dass unsere Suchanfragen blitzschnell in Echtzeit beantwortet werden (vgl. Baeza-Yates und Ribeiro-Neto). Andererseits erlauben die Suchmaschinen auch nicht jede Suchanfrage, beispielsweise werden **Wildcards** bei den üblichen Internetsuchmaschinen nicht unterstützt: Wir können nicht nach *Super\*wagen* suchen und hoffen, Treffer für *Supersportwagen* und *Supertourenwagen* zu finden.

Die einfachste Art der Suche ist die sequenzielle Suche im Text. Dabei wird der gesamte Text Zeichen für Zeichen auf Übereinstimmung mit der Anfrage durchsucht. Diese sequenzielle Suche funktioniert bei entsprechender Programmierung auch mit Wildcards, wie in Abschn. 3.1.1 beschrieben. Aber sie dauert lange und ist bei großen Korpora praktisch nicht mehr anwendbar. Durch eine Vorverarbeitung der Daten müssen zusätzliche Informationen in einer sogenannten Indexstruktur (kurz: Index) zusammengestellt werden, welche bei der Suche zu schnellen Ergebnissen führen. Verfügt das zu durchsuchende Korpus zusätzlich über Annotationen wie POS-Tagging, so soll diese Annotation möglicherweise in die Suche einbezogen werden können.

## 4.4.1 Indexstrukturen

### 4.4.1.1 Klassisch: Einzelwort-Index, evtl. mit zusätzlichen Wortgruppen

Während der Vorverarbeitung wird als Index eine sogenannte **inverse Liste** (vgl. Baeza-Yates und Ribeiro-Neto) erzeugt, die zu jedem Wort sämtliche Vorkommen des Wortes speichert. Das können (bei Suchmaschinen) die Dokumente sein oder (bei einem Korpus) die Sätze, die das entsprechende Wort enthalten. Auf einer nach den Wörtern sortierten inversen Liste lässt sich extrem schnell suchen. Damit lassen sich für ein Suchwort nur durch einen schnellen Zugriff auf die inverse Liste die Vorkommen des Suchwortes ermitteln. Manchmal werden zusätzlich Wortgruppen indexiert, um auch diese schneller finden zu können. Dies funktioniert allerdings nur für die vor der Indexierung bereitgestellten Wortgruppen.

### 4.4.1.2 Klassisch: Einzelwort-Index mit genauer Position

Zusätzlichen Komfort bietet die Speicherung der genauen Wortposition in der inversen Liste. Dadurch kann man bei der Suche von Wortgruppen auf die vorherige Indexierung dieser Wortgruppen verzichten. Sucht man beispielsweise nach der Wortgruppe *Otto Normalverbraucher*, so kann man aus der inversen Liste alle Vorkommen von *Otto* extrahieren und alle Vorkommen von *Normalverbraucher*. Danach müssen genau die Sätze ausgewählt werden, bei denen die Position von *Otto* um eins geringer ist als die von *Normalverbraucher*, sodass beide Wörter unmittelbar nacheinander auftreten.

Suchmaschinen verwenden solche inversen Listen und erlauben damit eine sehr effiziente Suche. Allerdings ist die Verwendung von Wildcards mit einem solchen Index nicht möglich, da damit die schnelle Suche auf der inversen Liste nicht mehr ohne weiteres funktioniert. Mit anderen Datenstrukturen ist dies aber möglich.

### 4.4.1.3 Spezielle Datenstrukturen für Textsuche: PAT Trees und die Anwendung in der NoSketch-Engine

Spezielle Datenstrukturen wie PAT Trees erlauben eine komfortablere Suche. Software wie die NoSketch-Engine (<https://nlp.fi.muni.cz/trac/noske>, Rychlý 2007) nutzt diese PAT Trees, dadurch ist eine Suche mit regulären Ausdrücken auf dem im Abschn. 4.3 beschriebenen Spaltenformat möglich: In der ersten Spalte stehen die Wörter des Textes zeilenweise untereinander. In den weiteren Spalten werden zusätzliche Informationen zu dem jeweiligen Wort untergebracht, beispielsweise die Grundform, ein POS-Tag, eine Synset-ID usw. Auch kompliziertere Angaben, die sich auf mehrere unmittelbar aufeinanderfolgende Wörter beziehen, sind möglich. Damit können z. B. auch komplexe Eigennamen annotiert werden.

#### Beispiel für „wollen“ als Adjektiv

Das Wort *wollen* kommt in seltenen Fällen auch als Adjektiv vor. Die Suche nach **LEMMA: wollen, POS: ADJ** findet Beispielsätze wie:

„Er trug den *wollenen* Mantel seines Vaters.“ ◀

#### 4.4.1.4 Ranking der Suchergebnisse

Das Problem der Reihenfolge der Suchergebnisse entsteht, wenn es viel mehr Suchergebnisse gibt als angezeigt oder anders ausgeliefert werden sollen. Eine zufällige Auswahl ist möglich, aber es geht auch besser: Spätestens seit dem Siegeszug von Google als Suchmaschine weiß man, dass die Reihenfolge von Suchergebnissen für den Nutzer bzw. die Nutzerin entscheidend sein kann. Die Rankingkriterien von Internetsuchmaschinen sind jedoch für die Suche in Korpora direkt übertragbar, da sich inhaltlich bewährte Quellen nicht zusätzlich durch die Verwendung der Sprache auszeichnen müssen.

Kriterien zur Auswahl von Beispielsätzen sind aus der Lexikographie bekannt: Für Wörterbücher ausgewählte Belegstellen sollen möglicherweise ähnliche Eigenschaften erfüllen wie angezeigte Beispielsätze. Dafür wurde der Algorithmus GDEX (Good Dictionary Expressions) entwickelt (vgl. Kilgarriff et al. 2008). In Abschn. 4.2 wurde eine Variante von GDEX eingeführt, die Sätze nach Wohlgeformtheit und Lesbarkeit mit einem Zahlenwert bewertet. Die Sätze im Suchergebnis werden dann entsprechend diesem GDEX-Wert sortiert.

#### 4.4.2 Verschiedene Typen von Suchanfragen

Ein generisches Korpus hat den Anspruch, verschiedene Typen von Informationsbedürfnissen zu befriedigen. Dafür benötigt man möglicherweise verschiedene Typen von Suchanfragen. Diese basieren wiederum auf verschiedenen Angaben oder Annotationen im Korpus. In den folgenden Abschnitten sollen verschiedene Anfragetypen und ihre Realisierung betrachtet werden.

##### 4.4.2.1 Beispielsätze für Wörter, Wortgruppen oder Kookkurrenzen

Die Grundfunktion der Suche ist die Bereitstellung von Beispielsätzen für Wörter oder Wortgruppen. Da Kookkurrenzen (siehe Abschn. 5.3) auch spezielle Wortgruppen sind, ist hier die Suche nach Beispielsätzen für Kookkurrenzen mit abgedeckt. Technisch kann diese Suche über die inverse Liste realisiert werden oder über komplexere Datenstrukturen wie PAT Trees. Dann ist auch die Suche mit regulären Ausdrücken möglich.

Beispiele:

- Suche nach Sätzen mit dem Wort: *Hochhaus*.
- Suche mit mehreren benachbarten Wörtern: *Katze aus dem Sack*.
- Suche nach mehreren Wörtern mit beliebigem Abstand im Satz: Sätze mit *Hund* und *Leine*.
- Suche nach Wörtern mit regulären Ausdrücken: *Super.\*wagen*

#### 4.4.2.2 Beispielsätze für Grundformen

Sucht man nach beliebigen flektierten Formen eines Wortes, dann möchte man bei der Suche nur die Grundform angeben. Kennt das System die flektierten Formen zu allen Grundformen, so kann diese Information benutzt werden, bevor in der inversen Liste nachgeschlagen wird. Sind die Angaben zur Grundform im Korpus gespeichert (siehe Abschn. 4.3), dann kann in der entsprechenden Spalte gesucht werden.

Beispiel:

- Die Suche nach *Grundform:Hochhaus* findet zusätzlich zu Hochhaus auch die Wörter *Hochhauses*, *Hochhäuser* und *Hochhäusern*.

#### 4.4.2.3 Kombination mehrerer Suchkriterien

Komplexere Anfragen sind möglich, wenn mehrere Suchkriterien gleichzeitig berücksichtigt werden sollen. Bei der Speicherung in relationalen Datenbanken werden solche Anfragen über sogenannte **Joins** realisiert. Bei speziellen Datenstrukturen wie PAT Trees sind sie ebenfalls möglich.

Beispiele:

- Suche nach typischen Eigenschaften zu Nomen: ADJ als linke Nachbarn zum Wort
- Wichtige Inseln: typische rechte Nachbarn von *Insel* mit POS-Tag NE

#### 4.4.2.4 Extraktion von Wortlisten mit bestimmten Eigenschaften

Durch eine Weiterbearbeitung von Suchergebnissen hat man die Möglichkeit, interessante Listen aus den Sätzen zu generieren. Hat beispielsweise das Suchmuster einen Platzhalter für ein Wort, dann erhält man eine Liste von Wörtern, mit denen dieser Platzhalter gefüllt wurde. Zusätzlich von Interesse ist die Anzahl, wie oft ein Wort an dieser Stelle verwendet wurde. Damit lässt sich eine häufigkeitssortierte Liste erzeugen, sodass man zusätzlich die Wichtigkeit durch die Häufigkeit abschätzen kann. Wirklich große Korpora sind hilfreich, um entsprechend viele Vorkommen der Muster vorzufinden.

Beispiele:

- Phraseologismen: Die Suche nach „*ein.\*.\* vom Zaun brechen*“ findet nach Häufigkeit sortiert die Wörter *Streit*, *Krieg*, *Streik*, *Debatte*, *Konflikt* usw.
- Suche nach *Insel* POS:NE: Die Liste ist wesentlich länger als die Liste der Nachbarschaftskookkurrenzen, da hier keine Mindestfrequenz nötig ist. Auch weit hinten in der Liste finden sich meist korrekte Inselnamen.
- Achtung: Inselnamen können aus mehreren Wörtern bestehen, z. B. Sankt Helena, Nowaja Semlja. Außerdem machen POS-Tagger oft Fehler bei der Erkennung von Eigennamen, sodass *Insel* POS>NN ein weiteres nützliches Suchmuster ist.

#### 4.4.2.5 Extraktion von Relationen

Mithilfe einfacher Muster ist die Extraktion von Relationen zwischen Wörtern (Abschn. 2.4.2) möglich. Dies soll an den folgenden Beispielen verdeutlicht werden. Bei wirklich großen Korpora von einigen hundert Millionen Sätzen sind die Ergebnisse dieser Muster beeindruckend.

##### Ober- und Unterbegriffe

Unterbegriffe zu einem Oberbegriff werden oft in der Form *POS>NN wie POS>NN und POS>NN* formuliert. Satzteile mit diesem Muster, wie *Studienrichtungen wie Medizin und Pharmazie*, lassen sich wieder aus einem Korpus extrahieren.

Ein anderes nützliches Muster für Unterbegriffe zu einem Oberbegriff ist *POS>NN, POS>NN und andere.\* POS>NN*. Dies liefert Treffer wie *Bronchitis, Asthma und anderen Atembeschwerden*. ◀

Eine andere Möglichkeit besteht darin, die Präpositionen zwischen Nomen als Relation zu interpretieren.

##### Satzkookkurrenzen und Relationen

Kennt man zu dem Wort *Spielzeug* die Satzkookkurrenzen *Holz* und *Kinder* und möchte die Relation dieser Wörter zu *Spielzeug* wissen, so kann man nach den Mustern *Spielzeug POS:PREP Holz* und *Spielzeug POS:PREP Kinder* suchen und findet die häufigsten Varianten *Spielzeug aus Holz* und *Spielzeug für Kinder*. ◀

#### 4.4.2.6 Suche unter Verwendung von Satzsignaturen

Die in Abschn. 4.2 eingeführten Satzsignaturen können ebenfalls bei der Suche eingesetzt werden. Ähnlich wie bei der Suche mit Mustern lassen sich Mengen von Sätzen mit gleicher Satzsignatur auswählen und dann die häufigkeitssortierte Liste der Wörter an einer Position untersuchen. Solche Wörter haben oft übereinstimmende semantische Eigenschaften.

##### Beispiele nach Satzsignatur

Zur semantischen Untersuchung des Adjektivs *begeistert* werden die Beispiele mit der Satzsignatur und dem folgenden Muster {ART}{NN}{VAFIN}/{APPR}{ART}{NN} *begeistert* untersucht.

*Das Publikum war von den Filmen begeistert.*

*Die Besucher waren von den Darbietungen begeistert.*

*Die Fachabteilungen waren von der Idee begeistert.*

*Die Fans sind nach dem Konzert begeistert.*

*Die Großen waren von den Rutschen begeistert.*

*Die Klasse war von dem Film begeistert.*  
*Die Marktbesucher waren von der Musik begeistert.*  
*Die Schauspieler sind von der Ausstellung begeistert.*  
*Die Schüler waren von den Vorlesestunden begeistert.*  
*Die Trainer waren von der Ausbildung begeistert.*

An den Positionen 2, 3 und 6 erhalten wir die folgenden Listen:

Position 2: *Besucher, Fachabteilungen, Fans, Großen, Klasse, Marktbesucher, Publikum, Schauspieler, Schüler, Trainer*

Position 3: *waren, war, sind*

Position 4: *von*

Position 6: *Ausbildung, Ausstellung, Darbietungen, Film, Filmen, Idee, Konzert, Musik, Rutschen, Vorlesestunden*

In diesem Beispiel lassen sich die folgenden Eigenschaften ablesen:

- Typischerweise sind Personen(gruppen) *begeistert*.
- Sie sind begeistert *von* etwas
- Kopf der Präpositionalphrasen mit *von* sind typischerweise Ereignisse oder Darbietungen. ◀

#### Partielle Synonyme von Verben in Abhängigkeit vom Subjekt

Die Verben *suchen* und *fahnden nach* sind nicht synonym. Aber falls das dazugehörige Subjekt *Polizei* ist, dann stimmen die Listen von Objekten beider Verben fast überein, d. h. in dieser Verwendung sind die Verben synonym.

*Die Polizei sucht den Flüchtigen.*  
*Die Polizei sucht den Halter.*  
*Die Polizei sucht den Kontrahenten.*  
*Die Polizei sucht den Mann.*  
*Die Polizei sucht den Täter.*  
*Die Polizei sucht die Täter.*  
*Die Polizei fahndet nach dem Jugendlichen.*  
*Die Polizei fahndet nach dem Mann.*  
*Die Polizei fahndet nach dem Täter.*  
*Die Polizei fahndet nach den Tätern.*  
*Die Polizei fahndet nach dem Unbekannten.*  
*Die Polizei fahndet nach einem Unbekannten.*

Die zwei Listen von Objekten zeigen nicht nur partielle Übereinstimmungen bei Mann und Täter, sondern zeichnen sich zusätzlich dadurch aus, dass die weiteren Wörter oft semantisch ähnlich zu den anderen Wörtern sind. Für diese semantische Ähnlichkeit sei auch auf Abschn. 5.6 verwiesen. ◀

Das Vorgehen der vorigen Beispiele kann verallgemeinert werden, um die **Valenzstruktur** einzelner Wörter zu beschreiben. Speziell für die Valenzstruktur von Verben ergeben sich die folgenden Möglichkeiten:

Verben treten in verschiedenen typischen Satzstrukturen auf, beispielsweise kommen bestimmte Konstituenten rund um das ausgewählte Verb (wie z. B. ein Akkusativobjekt) niemals vor (*schnieien*), sind fakultativ (*rechnen*) oder obligatorisch (*kaufen*). Die häufigen Satzsignaturen zu einem Verb zeigen die folgenden Eigenschaften:

- Ist ein Dativobjekt vorhanden?
- Ist ein Akkusativobjekt vorhanden?
- Wird eine Präpositionalphrase verwendet? Wenn ja, mit welcher Präposition?
- Sind das Subjekt und die Objekte (falls vorhanden) typischerweise belebt oder unbelebt?

Die Frage nach der Belebtheit lässt sich auf einfache Weise dadurch klären, dass man die Satzbaupläne mit Personalpronomen an den entsprechenden Positionen betrachtet (*er*: belebt, *es*: unbelebt, *sie*: keine Aussage möglich).

Alternativ zu Mengen von Sätzen mit gleicher Satzsignatur lassen sich auch Sätze mit ähnlichen Satzsignaturen auswerten. Dabei muss vorher die Ähnlichkeit von Satzsignaturen ermittelt werden. Eine Möglichkeit besteht darin, die Ähnlichkeit von Sätzen, beispielsweise ermittelt durch Verfahren wie doc2vec (siehe Abschn. 3.1.3), zu nutzen. Die Anzahl solcher ähnlicher Sätze für zwei verschiedene Satzsignaturen ist ein Maß für die Ähnlichkeit der entsprechenden Satzsignaturen.

Prinzipiell lassen sich ähnliche Aussagen auch mithilfe von Wortkookkurrenzen treffen. Allerdings werden für die Ermittlung signifikanter Kookkurrenzen jeweils mehrere Vorkommen der entsprechenden Wortpaare benötigt. Bei der Verwendung von Satzsignaturen reicht oft schon ein Vorkommen. Deshalb ist die Ergebnismenge hier um ein Vielfaches höher.

---

## 4.5 Manuell erstellte lexikalische Ressourcen

Neben unstrukturiertem und annotiertem Text werden für *Text-Mining*-Verfahren oft auch rein lexikalische Ressourcen benötigt. Lexikalische Ressourcen sind Listen von Wörtern, bei denen jeder Eintrag je nach Verfügbarkeit und Art des Wörterbuchs um statistische, morphologische, syntaktische, semantische, terminologische und pragmatische Angaben ergänzt wurde. In diesem Abschnitt werden zunächst die entsprechenden Ressourcen beschrieben, im nächsten Abschnitt (Abschn. 4.6) werden die verschiedenen lexikalischen Angaben näher betrachtet und es wird diskutiert, wie sie mit automatischen Mitteln erzeugt werden können.

### 4.5.1 Klassische Wörterbücher

In Wörterbüchern werden allgemeingültige Informationen zu Wörtern zusammengetragen, die für verschiedene Nutzergruppen von Interesse sein könnten. Solche Informationen zu einem Wort entstammen also nicht einem einzelnen Korpus. Moderne Wörterbücher sind zwar in der Regel korpusbasiert; dies bedeutet aber nur, dass den Lexikographen und Lexikographinnen bei der Wörterbucherstellung ein (meist generisches) Korpus zur Verfügung stand und auf verschiedenste Arten genutzt werden konnte. Beispiele dafür sind die Stichwortauswahl, die Auswahl von Belegstellen, die Suche nach Definitionen oder anderen Beschreibungen usw.

Wörterbücher in gedruckter Form gibt es seit mehreren Jahrhunderten, viele moderne Wörterbücher liegen inzwischen auch in elektronischer Form vor. Von besonderem Interesse für das Text Mining sind darüber hinaus terminologische Wörterbücher. Sie enthalten neben den allgemeinen linguistischen Informationen insbesondere Verwaltungsinformationen, Sachgebietsangaben und Verwendungshinweise, Schreibvarianten, Definitionen, semantische Informationen wie Synonyme, Meronyme und Antonyme sowie Übersetzungsäquivalente (vgl. Melby et al. 1993). Solche Wörterbuchdaten stehen zur kommerziellen Nutzung zur Verfügung und können entweder auf der Basis einzelner Wörterbucheinträge oder kompletter Datensätze zur Nutzung erworben werden.

Zusätzlich entstanden frei verfügbare elektronische Wörterbücher, bei deren Nutzung keinerlei Kosten entstehen. Diese sollen hier den Schwerpunkt bilden und deshalb ausführlicher beschrieben werden. Wörterbücher bieten in einem Wörterbucheintrag in der Regel mehrere Angaben zu dem sog. Lemma in sehr kompakter Form an. Für die Nutzung in elektronischer Form ist es sinnvoll, die oft nur implizit gegebene Struktur der Wörterbuchstruktur aufzulösen und die einzelnen Angaben getrennt zu speichern.

Die folgenden Ressourcen enthalten frei verfügbare Wörterbuchdaten, die ggf. aber noch extrahiert werden müssen. In vielen Fällen wurde dies bereits von anderen erledigt und die unten folgende Aufstellung zeigt, welche Wörterbuchdaten aus welchen Ressourcen erhältlich sind.

**Wikipedia** Ihrer Entstehung nach ist Wikipedia (<https://de.wikipedia.org/>) eine mehrsprachige Enzyklopädie, aber sie enthält auch Wörterbuchdaten: Durch die sog. Interwiki-Links in der linken Menüspalte unter „In anderen Sprachen“ ist ein Eintrag in der Ausgangssprache mit Einträgen in anderen Sprachen verlinkt, z. B. ist *Deutsch:Buch* verlinkt mit *Afrikaans:Boek* und *Englisch:Book*. Diese Links lassen sich zur Extraktion von Übersetzungen benutzen. Dabei muss allerdings beachtet werden, dass alle Wikipedia-Titel mit Großbuchstaben beginnen und dies nicht der Schreibweise im Text entsprechen muss. Zusätzlich von Interesse sind Wikipedia-Artikel für die Extraktion von Definitionen. Autoren und Autorinnen sind durch Richtlinien dazu aufgefordert, den Artikel mit einer Begriffsdefinition zu beginnen. Der erste Satz oder der Text bis zum

Ende des ersten Absatzes eignet sich in den meisten Fällen als Definition. Dazu kann es sinnvoll sein, Satzteile in Klammern wegzulassen. Weiterhin stellen Projekte wie Wikidata (<https://www.wikidata.org/>) und DBpedia (<https://www.dbpedia.org/>) aus Wikipedia extrahierte Daten im RDF-Format zur Verwendung als Wissensgraph zur Verfügung.

**Wiktionary** Das Wörterbuchprojekt dient der „[...] Erstellung eines frei zugänglichen, vollständigen und mehrsprachigen **Wörterbuches** sowie eines entsprechenden **Thesaurus** in jeder Sprache“ (Wikipedia 2021b). Wiktionary enthält für häufige Wörter fast alle möglichen Wörterbuchangaben, für seltenere Wörter entsprechend weniger. Allerdings enthält Wiktionary auch viele automatisch erstellte Einträge oder Bearbeitungen, nach den Schätzungen von Wiktionary Statistics (vgl. Zachte 2019) betrifft dies 63 % der erstellten Einträge und 75 % der Bearbeitungen von Wiktionary. Der deutschsprachige Teil ist hier eine Ausnahme mit weniger als 10 % der erstellten Einträge und 59 % der Bearbeitungen (Stand: 2019). Die automatische Erweiterung von Wörterbüchern wird im nächsten Abschn. 4.6 genauer betrachtet.

**WordNet** Dieses **lexikalisch-semantische Netz** entstand seit 1985 zunächst für die englische Sprache. Die Wörter sind nach Wortarten geordnet und Synonyme zu sog. Synsets zusammengefasst. Diese Synsets sind entsprechend semantischer Relationen verbunden (die wichtigsten sind Ober-/Unterbegriff, Teil-Ganzes-Beziehung, Gegen teil). Die Version 3.1 enthält knapp 120.000 Synsets. WordNet ist frei verfügbar (<https://wordnet.princeton.edu/>, vgl. Fellbaum 1999).

Die Global-WordNet-Initiative bemüht sich um Versionen in anderen Sprachen, wobei die Einträge auch zwischen den Sprachen verknüpft werden. Die deutsche Version trägt den Namen GermaNet und ist nicht frei verfügbar (<https://uni-tuebingen.de/en/142806>, vgl. Hamp und Feldweg 1997). Es gibt mehrere Projekte, die an einer freien deutschsprachigen Version arbeiten, beispielsweise Open Thesaurus (<https://www.openthesaurus.de/>, vgl. Naber 2005).

## 4.5.2 Verzeichnisse

Speziell im Falle von Eigennamen sind Verzeichnisse interessant. Um beispielsweise Personen in einem Text zu markieren, ist eine Liste von Personen (Gazetteer) nützlich, welche mit dem Text abgeglichen werden kann. Allerdings klappt dies nur für die in der Liste verzeichneten Personen. Deshalb sind umfangreiche Listen hilfreich. Jedoch wächst mit dem Umfang der Listen auch die Menge der mehrdeutigen Einträge, da manche Wörter beispielsweise sowohl Orts- und Nachnamen (*Lingen, Roth*) sein können und andere Wörter nicht immer Ortsnamen, sondern auch gewöhnliche Nomen (*Halle, Essen, Hof*) sein können. Neben einem direkten Abgleich können Listen auch als Merkmale in featurebasierten Lernverfahren (Abschn. 3.1.2) genutzt werden.

Im Falle von Personennamen muss weiter unterschieden werden zwischen Listen von Namen realer Personen (bestehend aus Vornamen und Nachnamen) sowie deren Bestandteile, d. h. Vornamenlisten und Nachnamenlisten.

#### 4.5.2.1 Verzeichnisse von Personen

Umfangreiche Verzeichnisse von Personen findet man in folgenden Quellen:

- **JRC Names:** Hier (<https://ec.europa.eu/jrc/en/language-technologies/jrc-names>) findet man eine Liste mit Namen von mehr als 200.000 Personen, hauptsächlich aus den EU-Staaten. Häufig sind die Namen in mehreren Schreibweisen in verschiedenen Sprachen angegeben. Die Liste kann als Textdatei heruntergeladen und weiterverarbeitet werden. Dazu gibt es ein Java-Programm, welches diese Namen in Texten wiederfindet und markiert.
- **Wikipedia Kategorien:** Die Kategorie *Deutscher* enthält mehr als 270.000 Einträge (Stand 04/2021). Zusätzlich unterscheiden wird zwischen *Mann* und *Frau*, sodass auch Angaben zum biologischen Geschlecht der Vornamen vorhanden sind. (Wikipedia 2021a)
- **Gemeinsame Normdatei GND:** Dieser Datenbestand enthält Informationen zu Personen, hauptsächlich gesammelt aus Katalogdaten von Bibliotheken. Sie sind zum Download nur als RDF-Daten erhältlich. Dadurch sind zusätzliche Kenntnisse bei der Weiterverarbeitung nötig.

#### 4.5.2.2 Verzeichnisse von Vornamen und Nachnamen

Einzelne Verzeichnisse von Vor- und Nachnamen haben gegenüber den Personenverzeichnissen den Vorteil, dass sie auch bisher unbekannte Personennamen erkennen helfen, wenn die einzelnen Bestandteile verzeichnet sind. Man benötigt dazu lediglich Muster, wie Personennamen gebildet werden. Im Deutschen sind die wichtigsten beiden Muster [Vorname][Nachname] und [Vorname][Vorname][Nachname].

Am einfachsten lassen sich Verzeichnisse von Vor- und Nachnamen aus Personenverzeichnissen mit den eben genannten Mustern ermitteln. Solche Verzeichnisse weisen allerdings oft Probleme auf, die mit der Erstellung verbunden sind: Außer Personen enthalten Verzeichnisse möglicherweise auch Firmennamen, die dann fälschlicherweise in Vor- und Nachname zerlegt werden. Außerdem enthalten viele Personenverzeichnisse (wie Wikipedia und ältere Telefonbücher) viel mehr Einträge von Männern als von Frauen, was eventuelle Häufigkeitsangaben verfälscht und die Vollständigkeit beeinträchtigt.

#### 4.5.2.3 Geographische Eigennamen

Elektronische Verzeichnisse geographischer Eigennamen (d. h. Namen für Orte, Gewässer, Berge usw.) finden sich vor allem in geographischen Anwendungen.

Die meisten geographischen Eigennamen haben eine einfache Struktur: Sie bestehen nur aus einem Wort. Aber auch hier gibt es Ausnahmen: Auch mehrere Wörter inkl.

Abkürzungen sind möglich (*Bad Brambach, St. Petersburg*), und für Ergänzungen zum Ortsnamen sind verschiedene Muster möglich (z. B. *Frankfurt (Oder), Halle/Saale*).

- Postleitzahlverzeichnisse sind frei verfügbar und enthalten etwa 10.000 Ortsangaben in Deutschland.
- Das größte gedruckt vorliegende Verzeichnis ist Müllers Großes Deutsches Ortsbuch (2019) (Müllers Großes Deutsches Ortsbuch 2019) mit mehr als 135.000 Einträgen. Diese Daten sind jedoch nicht frei verfügbar.
- Der *NGA GEOnet Names Server (GNS)* (<http://earth-info.nga.mil/gns/html/index.html>) enthält Geonamen für fast alle Länder (außer USA) in standardisierter Form. Die Daten sind unterschiedlich umfangreich je nach Land. Für die Namen gibt es unterschiedlich viele Varianten und Schreibweisen. Auch kleine geographische Einheiten sind verzeichnet.
- Für Geonamen aus den USA gibt es eine analoge Quelle: die Website des U.S. Board on Geographic Names ([http://geonames.usgs.gov/domestic/download\\_data.htm](http://geonames.usgs.gov/domestic/download_data.htm))
- GeoNames bietet (<https://www.geonames.org/>) ein durchsuchbares Verzeichnis mit mehr als 11 Mio. geographischen Namen. Die Daten stehen auch komplett zum Download zur Verfügung.

### 4.5.3 Relationen zwischen Mengen von Wörtern

#### 4.5.3.1 Synsets

Synonyme (siehe auch Abschn. 2.4.2) lernt jeder bereits in der Schule kennen. Im strengen Sinne sind damit Paare von Wörtern gemeint, bei denen man in einem Satz stets das eine durch das andere ersetzen kann, ohne den Sinn des Satzes zu ändern. In der Praxis ist man oft auch mit einer abgeschwächten Synonymie zufrieden, da dieses strikte Kriterium der Synonymie nur selten erfüllt ist:

- *Beerdigung* und *Bestattung* sind nicht strikte Synonyme, wie man an der Zusammensetzung *Seebestattung* sieht.
- *Auto* und *PKW* sind nicht strikte Synonyme, da sie aus grammatischen Gründen nicht immer austauschbar sind: *Ich sehe das Auto./Ich sehe den PKW.*
- Wenn man *Sekt* und *Schaumwein* als Synonym betrachtet, gilt das nicht für die dazugehörigen Pluralformen: Das Wort *Sekte* hat eine zusätzliche Bedeutung als soziale Gruppierung.
- Weitere Unterschiede sind möglich durch zusätzliche Wertungen (*Hund/Köter*), regionalen Gebrauch (*Januar/Jänner*), stilistische Ebenen usw.

Da die Anzahl der wirklich strikten Synonyme gering ist, werden in praktischen Anwendungen die Anforderungen großzügig interpretiert und man spricht in den obigen

Beispielen auch von **schwachen Synonymen**. Damit lassen sich Wörter gleicher Wortart zu Synonymgruppen, den sogenannten Synsets, zusammenfassen. Mehrdeutige Wörter werden dabei in mehrere Synsets aufgenommen und ihre Bedeutung wird durch die jeweils anderen Wörter im Synset erklärt, wie das folgende Beispiel für das mehrdeutige Wort *Band* zeigt: (*Band, Gurt, Kordel, Faden, Schleife*), (*Band, Buch, Wälzer, Publikation*), (*Band, Musikgruppe, Kapelle, Combo, Ensemble*).

Quellen für solche Synsets sind die bereits in Abschn. 4.5.1 erwähnten semantischen Wortnetze:

Für die deutsche Sprache relevant sind GermaNet mit etwa 140.000 Synsets mit 175.000 lexikalischen Einheiten (Stand 2020) sowie die deutsche Version des Open Thesaurus (<https://www.openthesaurus.de/>, vgl. Naber 2005) mit knapp 42.000 Synsets, 170.000 Wörtern und 14.500 Relationen.

#### 4.5.3.2 Erweiterung von Wortnetzen: Semantische Ähnlichkeit mit Word Embeddings

Synsets und Wortnetze können dadurch erweitert werden, dass in der Nähe eines Ausgangswortes semantisch ähnliche Wörter eingefügt werden. Synonyme können natürlich im gleichen Synset hinzugefügt werden, andere semantisch ähnliche Wörter gehören wahrscheinlich zu Synsets, die mit dem Synset des Ausgangswortes verbunden sind. Für die Bereitstellung solcher semantisch ähnlichen Wörter bieten sich Verfahren wie Topic-Modelle (Abschn. 6.4) und Word Embeddings (Abschn. 5.6) an.

---

## 4.6 Automatische Erweiterung lexikalischer Ressourcen

Klassische Wörterbücher in gedruckter Form enthalten Informationen für typischerweise weniger als 200.000 Wörter, manchmal nur für einige 10.000 Wörter. Dies hat im Wesentlichen zwei Ursachen: Zum ersten ist der Platz im gedruckten Buch beschränkt, zum anderen ist die Wörterbucherstellung eine aufwendige Tätigkeit für Lexikographen und Lexikographinnen und damit kostenintensiv.

Für die automatische Verarbeitung von Korpora benötigen wir aber vergleichbare Informationen für mehr Wörter. In vielen Fällen werden jedoch nicht die kompletten Wörterbucheinträge benötigt, sondern nur Teile davon, beispielsweise grammatische Informationen. Deshalb könnte man versuchen, die in einem kleinen Wörterbuch gegebenen Daten zusätzlich für viel mehr Wörter zu erweitern. Vom Standpunkt der Informatik aus gesehen sind die gegebenen Daten sogenannte **Trainingsdaten**, aus denen die Angaben für weitere Wörter gelernt werden können. Die Bearbeitung der zusätzlichen Wörterbucheinträge ist dann eine Klassifikationsaufgabe (Abschn. 6.6), die nötigen Attribute finden sich in den Wörtern selbst (sog. **intrinsische Attribute**, beispielsweise Flexionsendungen) oder in den typischen Kontexten (sog. **extrinsische Attribute**).

Schließlich gibt es noch Wörterbuchdaten, die aufgrund ihrer Einfachheit in klassischen Wörterbüchern gar nicht explizit angegeben werden, aber von manchen Algorithmen benötigt werden. Dies betrifft beispielsweise Angaben zur Worthäufigkeit oder zur Zerlegung von Komposita. In vielen Fällen lassen sich diese Daten direkt und auf einfache Weise ermitteln: Entweder direkt aus einem Korpus (wie Häufigkeiten) oder mit spezieller Software (z. B. zur Kompositazerlegung).

Die folgende Übersicht soll die wichtigsten automatisch ermittelbaren Wörterbuchangaben kurz erläutern sowie Datenquellen zusammengestragen.

## 4.6.1 Klassische Wörterbuchangaben

### 4.6.1.1 Lemmatisierung: Vollform zu Grundform

Die Zuordnung von flektierten Formen zu den entsprechenden Grundformen (Abschn. 2.2) wird als Lemmatisierung bezeichnet, da in klassischen Wörterbüchern mit dem Wort **Lemma** die jeweiligen Wörterbucheinträge bezeichnet werden. Häufig liegen die gesuchten Angaben zu einem Wort nur zur Grundform vor und für eine flektierte Form muss man zunächst die Grundform kennen, um die Angabe nachschlagen zu können.

**Beispiel** Um herauszufinden, ob *Väter* belebt sind, muss man zunächst die Grundform *Vater* ermitteln und dann nachschlagen, ob zu *Vater* die Angabe *belebt* vorliegt.

**Datenquellen** POS-Tagger wie der TreeTagger (vgl. Schmid und Laws 2008) übernehmen in vielen Fällen zusätzlich die Lemmatisierung, zumindest für die hoch- und mittelfrequenten Wörter. Für niederfrequente und speziell längere Wörter haben sich auch maschinelle Lernverfahren auf speziellen Datenstrukturen wie Compact Patricia Tries (Abschn. 3.2.4) bewährt, solange ausreichend viele Trainingsdaten zur Verfügung stehen.

### 4.6.1.2 Flexionsklasse zu Grundform

Hat man zu den flektierten Formen die Grundform, so kann man umgekehrt jeder Grundform ihre flektierten Formen zuordnen. Diese kann man in einem Schema anordnen, für Nomen beispielsweise die vier Fälle in Singular und Plural. Dem entsprechen nicht acht verschiedene flektierte Formen, sondern einige Formen treten mehrfach auf. In klassischen Wörterbüchern werden auch nicht alle Formen angegeben, in der Regel nur Genitiv Singular und Nominativ Plural, da sich die weiteren flektierten Formen regelhaft daraus ableiten lassen.

### Beispieleinträge im Wörterbuch

*Röhre*, die: -, -n;

*Stiel*, der: -[e]s, -e.

Beschreibt man die flektierten Formen durch solche Regeln, dann fällt auf, dass es wiederkehrende Mengen von Regeln gibt, beispielsweise werden die flektierten Formen von *Stiel* und *Tisch* nach den gleichen Regeln erzeugt. Dadurch werden die Grundformen der Nomen in sogenannte Flexionsklassen eingeteilt, siehe Abschn. 2.2.1.

Ähnliche Flexionsklassen gibt es auch für Verben und Adjektive.

Für niederfrequente Wörter werden aber nicht alle möglichen flektierten Formen im Korpus gefunden, sondern vielleicht nur die häufigste Form. Hier ist es oft möglich, die Flexionsklassen mit anderen Mitteln bereitzustellen. Zwei Verfahren helfen: Kompositazerlegung und maschinelles Lernen. Mittels Kompositazerlegung kann ein zusammengesetztes Substantiv in seine Teile zerlegt werden und das bekannte Flexionsschema des letzten Teils, d. h. des Grundwortes, auf das ganze Wort übertragen werden.

**Beispiel** *Hauptleitungsrohre* hat dieselbe Flexionsklasse wie *Röhre*.

Speziell für Komposita eignen sich wieder Compact Patricia Tries (Abschn. 3.2.4), um auf der Basis vorhandener Trainingsdaten die Flexionsklassen für Grundformen zu lernen.

#### 4.6.1.3 Wortart zu Grundform: POS-Tagging und NER

Die Kenntnis der Wortart eines Wortes ist von Interesse, speziell wenn es auf den ersten Blick mehrere Möglichkeiten gibt und deshalb eine Entscheidung notwendig sein kann. Beim Verstehen von Text kann es wichtig sein, ob es sich bei einem Wort um ein normales Nomen oder einen Eigennamen handelt (*Halle, Essen, Hof, Kohl, Vogel, Fleischer, Lederhose, Waren, Steinhäuser, Zimmern*), bei Eigennamen kann unklar sein, ob es sich um Orts- oder Personennamen handelt (*Tiefensee, Meißner, Dabo, ...*).

Beim Generieren von Text sind ebenfalls Unterschiede zu beachten, beispielsweise bei der Verwendung von Adjektiven und Adverbien: Viele Adjektive können adverbiell gebraucht werden, aber nicht alle. Dieses Wissen ist nötig, um die Generierung grammatisch falscher Sätze zu verhindern bzw. zu erkennen.

##### Vergleich der Wörter abweichend und anders

Möglich sind die folgenden Formulierungen für *abweichend*:

*Seine Meinung ist abweichend./Seine abweichende Meinung...*

Nur eine davon ist möglich für *anders*: *Seine Meinung ist anders.* NICHT: *Seine anderse Meinung ...* ◀

Tools zur Bestimmung der Wortart gibt es mehrere:

Beim POS-Tagging (vgl. Schmid 1994) wird für jedes Wort im Text die Wortart automatisch bestimmt. Später kann man zusammenzählen, wie oft jedes Wort mit den möglichen Wortarten markiert wurde. Auch wenn POS-Tagger eine vergleichsweise kleine Fehlerrate von ca. 2–4 % haben (d. h. 2–4 % der Wörter im Text werden mit der falschen Wortart versehen), erhalten praktisch alle Wörter bei entsprechender Häufigkeit irgendwann einmal die falsche Wortart. Beim oben beschriebenen Zusammenzählen muss

deshalb zusätzlich mit einem Schwellwert gearbeitet und darunterliegende relative Häufigkeiten müssen ignoriert werden.

Tools für die **Eigennamenerkennung** (NER-Tools) (vgl. Schmid und Laws 2008) markieren Eigenamen in Texten automatisch. Dagegen haben POS-Tagger speziell im Deutschen Schwierigkeiten, Eigenamen von normalen Nomen zu unterscheiden. Dies liegt an der Großschreibung von sowohl Nomen und Eigenamen. In vielen anderen Sprachen werden normale Nomen im Gegensatz zu Eigenamen klein geschrieben, was die Unterscheidung viel einfacher macht. Mit NER-Tools können deshalb die Ausgaben eines POS-Taggers kontrolliert und verbessert werden. Wie oben lassen sich aus den markierten Texten Listen von erkannten Eigenamen erzeugen, die auch noch nach unterschiedlichen Typen von Eigenamen (Personennamen, Ortsnamen, Firmennamen, ...) sortiert sind.

Schließlich zeichnen sich viele Eigenamen durch typische Buchstabenfolgen aus. Ähnlich wie bei der Terminologieextraktion (Abschn. 7.1) kann man manchmal an Wortteilen die Wortart erkennen. Dies funktioniert dann auch, wenn man das entsprechende Wort nur einmal sieht und ohne seinen Kontext zu berücksichtigen:

#### Typische Buchstabenfolgen in Eigenamen

*Duderstadt* ist ein Ortsname (wegen des Wortteils *-stadt*).

*Abduschaparow* ist ein slawischer Nachname (wegen der Endung *-arow*).

*Haupitleitungsröhre* ist ein Nomen, weil *Röhre* auch ein Nomen ist. ◀

#### 4.6.1.4 Grammatisches Geschlecht zu Nomen

Geschlechtsangaben zu Nomen finden sich in praktisch jedem Wörterbuch. Für Komposita lässt sich das grammatische Geschlecht regelbasiert ermitteln: Das grammatische Geschlecht des Kompositums ist gleich dem Geschlecht des Grundwortes. Beispielsweise ist das Wort *Kleinstadtpolizist* männlich, weil *Polizist* männlich ist.

Für nicht-zusammengesetzte Nomen ist die Situation wesentlich komplizierter. Es gibt keine einfachen Regeln, weshalb auch Deutschlernende das jeweilige Geschlecht eines Wortes auswendig lernen müssen und beim Sprechen oft Fehler machen.

Mit automatischen Mitteln haben wir die folgenden Möglichkeiten:

Ist das Wort, für das wir das grammatische Geschlecht ermitteln wollen, hinreichend häufig, dann finden wir es im Text oft genug mit Artikel. Aus der Verteilung der Artikel unmittelbar vor dem Nomen lässt sich dann das Geschlecht bestimmen. Für ein selteneres Wort ist das nicht möglich, weil es im Text vielleicht nie nach einem Artikel auftaucht. Auch ohne Verwendung von Kompositazerlegung kann man Substrings am Wortende als Attribute für einen Klassifikator verwenden, siehe Abschn. 3.2.4.

#### 4.6.1.5 Kompositazerlegung

Im Deutschen finden wir sehr viele Komposita (Abschn. 2.2); diese Wörter sind aus zwei oder mehr Wörtern zusammengesetzt. In den meisten Fällen handelt es sich um

zwei Nomen, die einfach hintereinander geschrieben (*Hausboot*) oder durch ein Fugen-s verbunden werden (*Beratungsfirma*). Das hintere Wort wird oft Grundwort genannt, das davorstehende heißt Bestimmungswort, weil es das Grundwort näher erklärt. Damit erbt das zusammengesetzte Wort viele grammatische und semantische Eigenschaften vom Grundwort (Im Beispiel: Grammatisch: *Hausboot* ist genau wie *Boot* sächlich. Semantisch: Jedes *Hausboot* ist ein *Boot*.)

Wie in vielen Bereichen der Sprache gibt es auch hier Ausnahmen, bei denen das nicht gilt: Solche Komposita heißen **nicht-transparent**, sind aber selten. Ein Beispiel ist das Wort *Mauerblümchen*, welches ein schüchternes Mädchen beschreibt. Ein *Mauerblümchen* ist also kein *Blümchen* und hat auch überhaupt nichts mit einer *Mauer* zu tun.

Eine weitere Schwierigkeit besteht darin, dass Kompositazerlegung nicht eindeutig sein muss. Aber auch dies ist extrem selten und deshalb in der Praxis kein Problem. Beispiele für solche Wörter mit mehreren Zerlegungen sind *Wach-stube/Wachs-tube* und *Stau-becken/Staub-ecken*.

Einige Zerlegungen sind nur aus der algorithmischen Sicht mehrdeutig, da nur eine der Zerlegungen semantisch sinnvoll ist. So könnte das Wort *Bahnsteig* auch mit Fugen-s aus *Bahn* und *Teig* gebildet worden sein. Weitere Beispiele für nicht sinnvolle Kompositazerlegungen sind *Billigst-roman-bieter* statt *Billigstrom-anbieter* oder *Bundesweh-reinheit* statt *Bundeswehr-einheit*.

Automatische Verfahren zerlegen Komposita auf der Basis bekannter Wörter und einfacher Regeln. Einige dieser Algorithmen benutzen auch linguistisches Wissen oder distributionelle Semantik (Riedl und Biemann, 2016), um im Nachhinein nichtplausible Zerlegungen auszuschließen.

## 4.6.2 Statistische Angaben

### 4.6.2.1 Worthäufigkeiten

Die Bestimmung von Worthäufigkeiten in einem Korpus ist eine der einfachsten Aufgaben. Nach der Tokenisierung (Abschn. 3.2) stehen die Wörter zur Verfügung und können gezählt werden.

Aber warum ist die Worthäufigkeit überhaupt von Interesse? Aus dem direkten Zusammenhang von Häufigkeit im Korpus und Bekanntheit unter den Autorinnen und Autoren ergibt sich die Häufigkeit als wichtiges Auswahlkriterium. Wegen der großen Anzahl verschiedener Wörter kann das nötige Wissen nicht für alle Wörter aus vorhandenen Quellen, z. B. Wörterbüchern, beschafft werden. Deshalb ist Häufigkeit ein nützliches Kriterium. Für spezielle Anwendungen sollten die Worthäufigkeiten aus einem entsprechenden Spezialkorpus ermittelt werden: Zur Ermittlung der häufigsten Wörter im Bereich *Chemie* sollte ein Korpus von Texten aus dem Bereich *Chemie* verwendet werden. Steht kein Spezialkorpus zur Verfügung, muss mit Häufigkeiten aus anderen, generischen Korpora gearbeitet werden.

Da die genaue Anzahl der Vorkommen eines Wortes im Korpus nicht wirklich wichtig ist und auch von diesem Korpus abhängt, arbeitet man in der Praxis mit Häufigkeitsklassen statt tatsächlichen Häufigkeiten: Darin werden Wörter vergleichbarer Häufigkeit zu größeren Klassen zusammengefasst, sodass sich die **Häufigkeitsklasse** eines Wortes in verschiedenen Korpora kaum noch unterscheidet. Dazu wird die Häufigkeit des häufigsten Wortes durch die Häufigkeit des betrachteten Wortes dividiert und der Logarithmus zur Basis 2 dieses Quotienten auf die nächste ganze Zahl gerundet: Das häufigste Wort hat immer die Häufigkeitsklasse 0; ein Wort aus der Häufigkeitsklasse 1 ist näherungsweise halb so häufig. Allgemein ist ein Wort der Häufigkeitsklasse  $n+1$  etwa halb so häufig wie ein Wort aus der Häufigkeitsklasse  $n$ . In großen Korpora haben extrem seltene Wörter Häufigkeitsklassen größer als 20 und sind somit um einen Faktor von größer als eine Million seltener als die häufigsten Wörter.

Die Häufigkeitsklasse eines Wortes ist damit zwar ein an einem konkreten Korpus gemessener Wert, aber trotzdem allgemeingültig, da er kaum vom benutzten Korpus abhängt.

#### 4.6.2.2 Wortpaare: Kookkurrenzen

Als Kookkurrenzen bezeichnet man Paare von Wörtern, die statistisch auffällig oft gemeinsam auftreten, siehe Abschn. 5.3. Auch die Paare werden zwar mithilfe eines bestimmten Korpus bestimmt, aber die Ergebnisse sind trotzdem typisch für die Sprache, nicht nur für das Korpus.

**Beispiel:** Kookkurrenzen zu *Kaffee* sind *trinken* und *heiß*. Die Wortpaare *Kaffee – trinken* und *Kaffee – heiß* treten häufig und in praktisch jedem generischen Korpus gemeinsam in Sätzen auf. Aber dieser Zusammenhang ist auch in der realen Welt gegeben und in unserem Gehirn fest verankert: Denken Sie an das Wort *Kaffee*, und Ihnen werden schnell die Wörter *heiß* und *trinken* einfallen.

Von besonderem Interesse ist hier, dass diese Wortpaare oft in einem näher zu bezeichnendem semantischen Zusammenhang stehen. Solche in Relation stehenden Wortpaare zusammenzutragen ist für verschiedene Anwendungsbereiche interessant. Zusätzlich ist natürlich die Relation zwischen den beiden Wörtern interessant.

Die Eigenschaft der Kookkurrenz lässt diese Relation zunächst offen und sagt für unser Beispiel nur:

- *heiß* ist eine typische Eigenschaft von *Kaffee*,
- *trinken* ist eine typische Tätigkeit im Zusammenhang mit *Kaffee*.

Will man die Art der Relation genauer beschreiben, etwa durch *Kaffee ist typisches Objekt für trinken*, dann ist eine zusätzliche Relationserkennung nötig (vgl. Heyer et al. 2001).

### 4.6.2.3 Sentiment

Der Bereich der Sentimentanalyse (Abschn. 7.3) versucht die positive oder negative Bewertung des Autors bzw. der Autorin gegenüber dem beschriebenen Sachverhalt zu analysieren. Es soll beispielsweise herausgefunden werden, ob eine Produktbeurteilung im Internet eher positiv oder negativ ausgefallen ist. Analog können Kommentare zu Wertpapieren an der Börse als positiv oder negativ eingeschätzt werden oder ebenso Handlungen von Politikern und Politikerinnen.

Der lexikalische Ansatz bei der Sentimentanalyse beruht auf Wörtern, die im entsprechenden Text vorkommen und positive oder negative Bewertung ausdrücken. Dafür gibt es Wortlisten wie SentiWS (<https://wortschatz.uni-leipzig.de/de/download>, vgl. Remus et al. 2010), die solche Wörter zusammen mit der vermittelten Bewertung verzeichnen.

### 4.6.2.4 Sachgebietsangaben

Sachgebietsangaben zu Wörtern findet man bereits in vielen Wörterbüchern, es gibt ganze Wörterbücher, die sich einem Fachgebiet widmen (z. B. *Wörterbuch der Chemie*). Die Kenntnis des behandelten Sachgebiets eines Textes ist von Interesse, da diese Information auf verschiedene Arten genutzt werden kann:

- Klassifikation von Texten nach Sachgebiet
- Extraktion von Terminologie zu diesem Sachgebiet (siehe Abschn. 7.1)
- Weiterleitung von Dokumenten wie E-Mails an bestimmte Mitarbeitende
- Extraktion von Relationen zwischen Wörtern in einem Sachgebiet

Listen von Wörtern zu Sachgebieten finden sich in Nachschlagewerken, frei verfügbar ist beispielsweise die Zuordnung von Wörtern zu Wikipedia-Kategorien.

Eine typische Aufgabenstellung ist die Erweiterung einer vorgegebenen Liste von Wörtern zu einem Sachgebiet. Die vorhandenen Zuordnungen können dabei als Trainingsdaten verwendet werden. Mithilfe von Wortähnlichkeiten lassen sich weitere Wörter den vorhandenen Sachgebieten zuordnen. Dabei lassen sich sowohl intrinsische wie extrinsische Merkmale verwenden:

- Fachwörter sind häufig längere Wörter, die sich durch wiederkehrende Bestandteile auszeichnen. Das können Komposita mit gleichem Grundwort oder gleichem Bestimmungswort sein oder auch kleinere Wortbestandteile wie typische Endungen.
- Wenn Wörter oft im gleichen Kontext auftauchen, dann gehören sie vermutlich ebenfalls zum gleichen Sachgebiet. Wir können also Verfahren einsetzen, die semantisch ähnliche Wörter ermitteln. Solche Verfahren sind Topic-Modelle oder die Bestimmung semantisch ähnlicher Wörter mit neuronalen Netzen (z. B. word2vec) oder auf der Basis gemeinsamer Kookkurrenzen. Da die verschiedenen Verfahren

unterschiedlich starke Ähnlichkeiten berücksichtigen, lassen sich insgesamt recht umfangreiche Listen von Wörtern zu Sachgebieten erzeugen.

Solche automatisch erzeugten Listen sind natürlich nicht 100 %ig perfekt und müssen stets manuell überprüft werden.

---

## Literatur

- Baeza-Yates, R., Ribeiro-Neto, B.: Modern information retrieval. Addison-Wesley Longman Limited, Essex (1999)
- Eckart, T., Kuras, C., Quasthoff, U.: Features for generic corpus querying. In: Calzolari, N. et al. (Hrsg.) Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC), Portorož, Slovenia, S. 2794–2798 <https://aclanthology.org/L16-1444.pdf> (2016). Zugegriffen: 29. Sept. 2021
- European Union: JRC-Acquis. <https://ec.europa.eu/jrc/en/language-technologies/jrc-acquis> (2016). Zugegriffen: 13. Apr. 2021
- Fellbaum, C.: WordNet: An electronic lexical database, 2. Aufl. Language, Speech, and Communication. MIT Press, Cambridge, MA (1999)
- Goldhahn, D.: Quantitative Methoden in der Sprachtypologie: Nutzung korpusbasierter Statistiken. Universität Leipzig (2013)
- Gormley, C., Tong, Z.: Elasticsearch: the definitive guide. A distributed real-time search and analytics engine. O'Reilly, Sebastopol, CA, USA (2015)
- Hajič, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M.A., Màrquez, L., Meyers, A., Nivre, J., Padó, S., Štěpánek, J., Straňák, P., Surdeanu, M., Xue, N., Zhang, Y.: The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In: Hajič, J. (Hrsg.) Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task. Boulder, CO, USA, S. 1–18. <https://www.aclweb.org/anthology/W09-1201> (2009). Zugegriffen: 16. Febr. 2021
- Hamp, B., Feldweg, H.: GermaNet - a lexical-semantic net for German. In: Proceedings of the ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications, Madrid, Spain, S. 9–15 (1997)
- Heyer, G., Läuter, M., Quasthoff, U., Wittig, Th., Wolff, Chr.: Learning relations using collocations. In: Maedche, A., Staab, S., Nedellec, C., Hovy, E. (eds.) Proceedings of the 2nd International Conference on Ontology Learning, vol. 38, Seattle, WA, USA S. 19–24. CEUR-WS.org, Aachen, Germany (2001)
- Internet archive: Digital library of free & borrowable books, movies, music & wayback machine. <https://archive.org/> (2021). Zugegriffen: 22. Febr. 2021
- Kilgarriff, A., Husák, M., McAdam, K., Rundell, M., Rychlý, P.: GDEX: Automatically finding good dictionary examples in a corpus. In: Bernal, E., Decesaris, J. (Hrsg.) Proceedings of the XIII EURALEX International Congress, S. 425–432. Universitat Pompeu Fabra, Barcelona (2008)
- Melby, A., Budin, G., Wright, S.E.: The terminology interchange format (TIF) – a tutorial. TermNet News **40**, 9–65 (1993)
- Müllers Großes Deutsches Ortsbuch: Vollständiges Ortslexikon, 36. Aufl. De Gruyter Saur, Berlin (2019)

- Naber, D.: OpenThesaurus. Ein offenes deutsches Wortnetz. In: Sprachtechnologie. mobile Kommunikation und linguistische Ressourcen: Beiträge zur GLDV-Tagung 2005 in Bonn, S. 422–433. Peter-Lang-Verlag, Frankfurt a. M. (2005)
- Panchenko, A., Ruppert, E., Faralli, S., Ponzetto, S.P., Biemann, C.: Building a web-scale dependency-parsed corpus from common crawl. In: Calzolari, N. et al. (Hrsg.) Proceedings of the Eleventh International Conference on Language Resources Association (LREC 2018), Miyazaki, Japan, S. 1816–1823. ELRA. <https://aclanthology.org/L18-1286.pdf> (2018). Zugegriffen: 10. Sept. 2021
- Remus, S., Biemann C.: Domain-Specific corpus expansion with focused webcrawling. In: Calzolari, N. et al. (Hrsg.) Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC), Portoroz, Slovenia (LREC 2016), Portorož, Slovenia, S. 3607–3611. <https://aclanthology.org/L16-1572.pdf> (2016). Zugegriffen: 10. Sept. 2021
- Remus, R., Quasthoff, U., Heyer, G.: SentiWS - a publicly available German-language resource for sentiment analysis. In: Calzolari, N. et al. (Hrsg.) Proceedings of the 7th International Language Resources and Evaluation (LREC'10), Valletta, Malta, S. 1168–1171 (2010)
- Riedl M., Biemann, C.: Unsupervised compound splitting with distributional semantics rivals supervised methods. In: Knight, K., Nenkova, A., Rambow, O. (Hrsg.) Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2016), San Diego, CA, USA, S. 617–622. Association for Computational Linguistics, Stroudsburg, PA, USA (2016). doi: <https://doi.org/10.18653/v1/N16-1075>
- Rychlý, P.: Manatee/Bonito - a modular corpus manager. In: Sojka, P., Horák, A. (Hrsg.) RASLAN. 2007. Recent advances in slavonic natural language processing, S. 65–70 (2007)
- Schäfer, R., Bildhauer, F.: Web corpus construction. Synthesis Lectures on Human Language Technologies 6(4), 1–145 (2013). <https://doi.org/10.2200/S00508ED1V01Y201305HLT022>
- Schmid, H.: Probabilistic part-of-speech tagging using decision trees. In: Proceedings of the International Conference on New Methods in Language Processing, Manchester, UK (1994)
- Schmid, H., Laws, F.: Estimation of conditional probabilities with decision trees and an application to fine-grained POS tagging. In: Scott, D., Uszkoreit, H. (Hrsg.) Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008), Manchester, UK, S. 777–784 (2008)
- Sinclair, J.: Corpus and text: basic principles. In: Wynne, M. (Hrsg.) Developing linguistic corpora. A guide to good practice. University of Oxford (2004)
- Wikipedia: Kategorie: Deutscher. <https://de.wikipedia.org/wiki/Kategorie:Deutscher> (2021a). Zugegriffen: 28. Apr. 2021
- Wikipedia: Wiktionary. <https://de.wikipedia.org/wiki/Wiktionary> (2021b). Zugegriffen: 28. Apr. 2021
- Zachte, E.: Wiktionary Statistics. <https://stats.wikimedia.org/wiktionary/EN/BotActivityMatrixCreates.htm> (2019). Zugegriffen: 28. Apr. 2021



# Sprachstatistik

5

## Zusammenfassung

Anliegen der Sprachstatistik ist es, die Texte nur als Wortfolgen zu betrachten und aus diesen Folgen sinnvolle Zusammenhänge zu extrahieren. Dies beginnt mit Untersuchungen zu Worthäufigkeiten, speziell dem Zipfschen Gesetz. Auch bei Wortpaaren deuten auffällige Häufigkeiten, sog. Kookkurrenzen, auf inhaltliche Zusammenhänge. Schließlich kann man die mit einem Wort typischerweise auftretenden anderen Wörter als abstrakte Beschreibung im Sinne der distributionellen Semantik benutzen und Wörter in hochdimensionale Vektorräume einbetten. Diese sog. Embeddings bilden die Grundlage für neuronale Lernverfahren auf Text. In diesem Kontext wird auch die Funktionsweise der hierfür verwendeten Sprachmodelle und deren Modellierung von Wortbedeutung im Kontext dargestellt. Statistische Verfahren können zusätzlich auch genutzt werden, um Anomalien in Verteilungen festzustellen, die möglicherweise auf Qualitätsmängel im zugrunde liegenden Korpus hinweisen können. Der Vergleich zweier Korpora ermöglicht die Extraktion statistisch signifikanter Unterschiede zwischen ihnen und kann beispielsweise für die Autoren- und Quellenidentifikation oder das Monitoring und die Analyse von Nachrichtentexten verwendet werden.

---

**Ergänzende Information** Die elektronische Version dieses Kapitels enthält Zusatzmaterial, auf das über folgenden Link zugegriffen werden kann [https://doi.org/10.1007/978-3-658-35969-0\\_5](https://doi.org/10.1007/978-3-658-35969-0_5).

## 5.1 Statistische Messungen und ihre Zuverlässigkeit

### 5.1.1 Aufgaben aus der Sprachstatistik

Sprachstatistik hat die Aufgabe quantitative Informationen zur Sprache zu sammeln. Der Nutzen solcher Zahlenwerte kann in verschiedenen Bereichen liegen: Mit einzelnen Zahlenwerten können Eigenschaften einer Sprache beschrieben werden. Unterschiede für verschiedene Sprachen spiegeln sich möglicherweise in unterschiedlichen Zahlenwerten wider. Beispiele für solche Zahlenwerte sind:

- Die Anzahl der Buchstaben im Alphabet;
- die durchschnittliche Wortlänge, gemessen in Buchstaben;
- die durchschnittliche Anzahl flektierter Formen pro Grundform.

Aus entsprechenden Messwerten lassen sich dann die folgenden Aussagen ableiten:

- Das russische Alphabet hat mehr Buchstaben als das deutsche Alphabet.
- Die mittlere Wortlänge ist in der deutschen Sprache größer als im Englischen.
- Im Isländischen gibt es pro Grundform mehr flektierte Formen als im Deutschen.

Dass allerdings schon solch einfache Messungen und die damit verbundenen Aussagen missverständlich sein können, wird in diesem Abschnitt untersucht.

Neben der Ermittlung einzelner Zahlen können auch viele Zahlen gleichzeitig von Interesse sein, um eine Aussage über eine Sprache zu treffen. Ein Beispiel ist das Zipfsche Gesetz (siehe Abschn. 5.2), bei dem die Häufigkeiten aller Wörter in einen interessanten mathematischen Zusammenhang gebracht werden. Allgemeiner gesagt, können quantitative Zusammenhänge einfachen (oder auch komplizierteren) mathematischen Gesetzmäßigkeiten folgen.

Weiterhin besteht die Möglichkeit, statistisch ermittelte Zahlenwerte nicht direkt zu betrachten, sondern als Eingabe für andere Algorithmen zu benutzen. Beispiele sind die **automatische Übersetzung** oder Klassifikationsaufgaben (Abschn. 6.6) verschiedenster Art bis hin zur Verwendung von vortrainierten Wortrepräsentationen in neuronalen Netzen.

Einige der gemessenen Werte können sich auch für verschiedene Korpora in derselben Sprache unterscheiden, sodass wir statistische Eigenschaften nutzen können, um verschiedene Textgenres oder auch Texte verschiedener Autoren und Autorinnen zu unterscheiden. Auf solche Unterschiede wird in Abschn. 5.1.5 näher eingegangen. Dementsprechend handelt es sich dann nicht um sprachabhängige Messwerte, sondern diese Werte sind korpus- oder autorenabhängig.

### 5.1.2 Messgrößen der Sprachstatistik

Die folgende Liste gibt einen Überblick über verschiedene Messgrößen, die später für verschiedene Messobjekte (wie Sprachen, Korpora, Autoren bzw. Autorinnen, Sätze, Wörter) zu deren Charakterisierung verglichen werden können:

- Suche nach typischen Eigenschaften zu Nomen: ADJ als linke Nachbarn zum Wort
- Wichtige Inseln: typische rechte Nachbarn von *Insel* mit POS-Tag NE
- Umfang von Mengen wie *Alphabet* oder *Vokabular*;
- mittlere Anzahl von Teilen pro Einheit, z. B.
  - mittlere Anzahl von Buchstaben oder Silben pro Wort,
  - mittlere Anzahl von Wörtern pro Satz,
- relative Häufigkeiten von Zeichenketten wie Buchstaben, Wörtern oder Wortgruppen.

Von einigen der Messgrößen kann man erwarten, dass sie nahezu unabhängig von der Korpusgröße sind (wie die mittleren Anzahlen und die relativen Häufigkeiten oben), andere Messwerte (wie der Umfang des Vokabulars) wachsen mit der Korpusgröße. Allerdings schwächt sich das Wachstum des Vokabulars mit der Korpusgröße ab, sodass ein „durchschnittliches Wachstum“ keine sinnvolle Größe ist. Dies wird im Abschn. 5.2 (Zipfsches Gesetz) genauer betrachtet.

### 5.1.3 Präsentation der Ergebnisse

Die Ergebnisse sprachstatistischer Untersuchungen lassen sich auf verschiedene Arten präsentieren:

- Die Darstellung der Resultate in einer Grafik,
- die Beschreibung der dargestellten Zusammenhänge durch eine Formel, und
- Ausnahmen: die Suche nach den Objekten, die von der gefundenen Gesetzmäßigkeit scheinbar weit abweichen.

Liegen viele Messwerte vor, ist eine Präsentation in einem Diagramm sinnvoll. Von besonderem Interesse ist ein exakter mathematischer Zusammenhang zwischen den Messgrößen, der sich beispielsweise in einer Geraden im Diagramm zeigt: Im Falle einer Geraden kann der mathematische Zusammenhang mit nur zwei Parametern beschrieben werden. Möglicherweise ist es dazu nötig, vorher noch die Achsen des Diagramms anders zu skalieren. In vielen Fällen bietet sich eine logarithmische Skalierung an. Auch der Mittelwert einer Messung kann interessant sein. Ebenfalls von Interesse können Abweichungen von einer Regelmäßigkeit sein.

### Wortlänge und Worthäufigkeit

In Abschn. 5.1.4 wird der Zusammenhang zwischen Wortlänge und Worthäufigkeit untersucht. Neben der zu erwartenden Aussage, dass seltene Wörter tendenziell länger sind, werden wir einen linearen Zusammenhang zwischen zwei Messgrößen finden. Natürlich ist nicht jedes seltene Wort auch lang oder jedes kurze Wort häufig. Die Suche nach Abweichungen von der Regel liefert vielleicht auch interessante Treffer. Neben seltenen kurzen Wörtern (*Aal*, *Fön* und viele mehr, insbesondere Eigennamen und Abkürzungen) können wir auch nach den längsten häufigen Wörtern suchen. Unter den 200 häufigsten Wörtern finden wir *Deutschland*, *Unternehmen* und *Informationen*, während die mittlere Wortlänge unter den 200 häufigsten Wörtern nur rund 4,3 Zeichen beträgt. Diese längsten Wörter geben uns zusätzlich einen Hinweis auf die inhaltlichen Schwerpunkte im Korpus. ◀

## 5.1.4 Messungen und Messwerte

### 5.1.4.1 Beschreibung und Nachvollziehbarkeit der Messung

Nicht nur das Ergebnis einer Messung ist interessant, sondern auch der Weg, auf dem man dahin gelangt ist. Schließlich soll die Messung auch von anderen wiederholt werden können und dann das gleiche Ergebnis liefern. Erst dann kann man die Messung auch auf andere Objekte (z. B. andere Sprachen) anwenden und die Ergebnisse vergleichen. Andernfalls besteht immer die Gefahr, dass Zahlen verglichen werden, die mit völlig unterschiedlichen Verfahren ermittelt wurden und nicht vergleichbar sind. Im folgenden Abschnitt wird am **Type-Token-Verhältnis** demonstriert, wie mithilfe der gleichen Daten völlig unterschiedliche Messwerte ermittelt werden.

Die statistische Auswertung von Messergebnissen beginnt mit der Beschaffung der Messwerte. Deshalb ist zunächst die eindeutige Definition der Messaufgabe und der Messgröße nötig. Wenn beispielsweise die mittlere Wortlänge berechnet werden soll: Wie wird mit Ziffern im Text umgegangen? Werden diese vorher entfernt oder zählen Ziffern wie Buchstaben? Verbunden mit der Beschreibung der Messaufgabe sind die Festlegung des Messablaufs und die Durchführung der Messung: An welchem Korpus wird gemessen? Werden beispielsweise verschiedene Messungen für unterschiedliche Genres durchgeführt? Und wie funktioniert die Messung genau? Beispielsweise bieten Apostrophe mehrere Möglichkeiten: Sind im Englischen *it's* oder *can't* jeweils ein Wort oder zwei? Im Falle zweier Wörter: Was genau ist das jeweils zweite Wort? Unterschiedliches Vorgehen wird hier die mittlere Wortlänge verändern.

Zum vollständigen Messergebnis gehört auch eine Aussage über die Messunsicherheit. Beispielsweise werden sich für ein anderes Korpus leicht andere Zahlenwerte ergeben. Deshalb ist es wichtig, die Messgenauigkeit zu kennen und das Ergebnis mit der angemessenen Anzahl von Nachkommastellen anzugeben.

### 5.1.4.2 Abhängigkeit von der Wortdefinition am Beispiel der Type-Token-Ratio

Die Type-Token-Ratio beschreibt die lexikalische Vielfalt in einem Korpus, genauer das Verhältnis zwischen der Anzahl der verschiedenen Wörter und der Gesamtzahl aller Wörter im Text. Sein Kehrwert beschreibt die durchschnittliche Häufigkeit der Wörter im Text.

#### Type-Token Ratio

Unser Korpus soll nur aus dem folgenden Satz bestehen:

„Ich komme um 9 Uhr“, verriet ich um 7 Uhr.

Zur Berechnung der Type-Token-Ratio benötigen wir die Gesamtanzahl der Wörter. Je nachdem, ob wir Zahlen berücksichtigen, zählen wir 8 oder 10 Wörter. Viele Programme zählen aber auch Satzzeichen als einzelne Wörter und ermitteln so 14 Wörter.

Wie viele Wörter sind davon verschieden? Die Wörter *um* und *Uhr* kommen doppelt vor. Müssen wir auch *ich* und *Ich* identifizieren? Sollen wir die (verschiedenen) An- und Ausführungszeichen identifizieren, damit die gleiche Type-Token-Ratio herauskommt wie bei der Verwendung anderer Anführungszeichen im gleichen Satz (*"Ich komme um 9 Uhr", verriet ich um 7 Uhr.*)? All diese Varianten ergeben leicht unterschiedliche Anzahlen für Types und Tokens.

Für die Type-Token-Ratio ergeben sich unter anderem folgende mögliche Werte:

- $5/8=0,625$ , falls wir Zahlen und Satzzeichen ignorieren und *ich* und *Ich* identifizieren.
- $6/8=0,750$ , falls wir Zahlen und Satzzeichen ignorieren und *ich* und *Ich* als verschieden betrachten.
- $7/10=0,700$  oder  $8/10=0,800$ , falls wir nur Satzzeichen ignorieren und beide Varianten für *ich* zulassen.
- $12/14=0,857$ , falls auch verschiedene Satzzeichen als zusätzliche verschiedene Wörter betrachtet werden.

Damit schwankt unser Messergebnis um bis zu 40 %, wenn wir die Messmethode verändern. Vergleiche mit einer anderen Sprache, zu der die Abweichungen möglicherweise viel kleiner sind, sind also nur bei genauer Kenntnis der Messmethode angebracht. ◀

Die im Beispiel aufgeführten Optionen haben auch Auswirkungen auf andere Messwerte, z. B.:

- durchschnittliche Wortlänge gemessen in Zeichen,
- durchschnittliche Satzlänge gemessen in Wörtern oder Zeichen,

- Anzahl Tokens im Korpus,
- **Textabdeckung;** auch für deren Messung ist eine genaue Beschreibung der Messmethode unverzichtbar.

### 5.1.4.3 Abhängigkeit von der Korpusgröße

Bei verschiedenen Messwerten ist nicht von vornherein klar, ob oder wie der ermittelte Messwert von der Größe des Korpus (gemessen z. B. in Zahl der Zeichen) abhängt. Die durchschnittliche Satzlänge sollte beispielsweise nicht von der Korpusgröße abhängen; die Anzahl der Sätze im Korpus aber linear mit der Korpusgröße wachsen. Auch die Gesamtanzahl der Wörter im Korpus (Tokens) wächst linear mit der Korpusgröße. Bei der Zahl der Types können wir ebenfalls erwarten, dass sie mit der Korpusgröße wächst. Aber ist auch dieses Wachstum linear? Das hat wiederum Auswirkungen auf die Type-Token-Ratio. Deshalb berechnen wir die Type-Token-Ratio an Beispielkorpora, die sich nur in ihrer Größe unterscheiden.

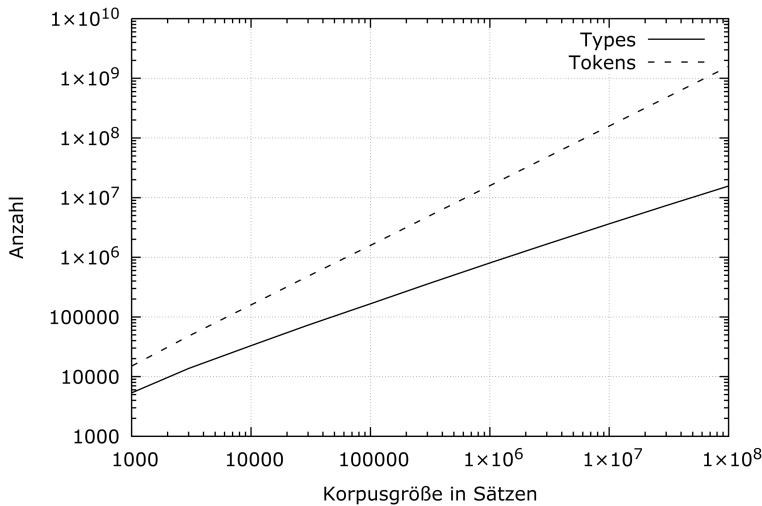
Die Tab. 5.1 zeigt die Type-Token-Ratio für gleichartige Corpora wachsender Größe. Die Anzahl der Tokens wächst linear mit der Korpusgröße. Man sieht deutlich, dass in einem zehnmal so großen Korpus diese Anzahl um den Faktor 10 größer ist. Die Zahl der Types wächst langsamer als linear, deshalb fällt die Type-Token-Ratio mit wachsender Korpusgröße.

Die Type-Token-Ratio verringert sich also mit der Korpusgröße. Wenn man für zwei verschiedene Corpora (z. B. in verschiedenen Sprachen) die Type-Token-Ratio vergleicht, muss man also unbedingt auf gleiche Korpusgröße achten.

Können wir den Zusammenhang zwischen den Anzahlen von Types (y) und Tokens (z) trotzdem noch genauer beschreiben? Dazu betrachten wir in der Abb. 5.1 eine graphische Darstellung der Werte aus Tab. 5.1: Es bietet sich eine doppelt logarithmische Darstellung von Korpusgröße (x) und Anzahlen (y bzw. z) an. Für die Zahl der Tokens erhält man eine exakte Gerade mit Anstieg 1, dies entspricht linearem Wachstum in Abhängigkeit von der Korpusgröße x. Für die Zahl der Types erhält man auch näherungsweise eine Gerade, aber mit einem kleineren Anstieg von etwa 0,67. Dies entspricht dem Wachstum nach einem Potenzgesetz mit der Potenz 0,67, wieder in Abhängigkeit von der Korpusgröße x. Daraus wiederum ergibt sich, dass zwar nicht die

**Tab. 5.1** TTR für Normgrößenkorpora

Datenbank und Größe	Anzahl Types	Anzahl Tokens	TTR
deu-de_web-wrt_2019_1K	5347	15.021	0,3560
deu-de_web-wrt_2019_10K	33.041	159.236	0,2075
deu-de_web-wrt_2019_100K	166.306	1.581.731	0,1051
1 deu-de_web-wrt_2019_5M	803.825	15.853.224	0,0507
2 deu-de_web-wrt_2019_10M	3.637.659	158.424.159	0,0230
3 deu-de_web-wrt_2019_100M	15.642.704	1.583.828.415	0,0099



**Abb. 5.1** Graphische Darstellung des Zusammenhangs von Types und Tokens.

Type-Token-Ratio  $y/z$  konstant ist, aber der Quotient  $y/z^{0,67}$  ist nahezu unabhängig von der Korpusgröße.

#### 5.1.4.4 Zählen mit oder ohne Wiederholungen: Mittlere Wortlänge

Uns soll die folgende einfache Frage beschäftigen: Aus wieviel Buchstaben besteht ein deutsches Wort im Durchschnitt? Etwas verkürzt gefragt: Wie lang ist ein durchschnittliches deutsches Wort? Nach einer kurzen Recherche im Internet findet man beim Duden-Verlag im Jahr 2021 die folgenden drei Aussagen:

- Ein Wort im Rechtschreibbuden umfasst im Schnitt 10,6 Buchstaben.
- 5,99 Buchstaben ergibt die Berechnung des Durchschnitts aller Wörter (d. h. Tokens) im Dudenkorpus.
- Nimmt man als Berechnungsgrundlage hingegen nur die unterschiedlichen Wörter (d. h. Types), also die im Dudenkorpus verzeichneten rund 18 Mio. Grundformen, so beträgt deren durchschnittliche Länge 14,43 Buchstaben. (Bibliographisches Institut 2021)

Diese drei unterschiedlichen Antworten zeigen wieder das Problem: Es muss genau definiert werden, was gemessen wird. Und vielleicht ist eine Messmethode brauchbarer als eine andere. Die Tab. 5.2 zeigt die durchschnittliche Wortlänge, gemessen für alle Wörter im Text (Tokens) bzw. nur für die verschiedenen Wörter. Es zeigt sich, dass die erste Zahl relativ konstant ist, die zweite dagegen wächst.

Die Konstanz der ersten Zahl ist nicht verwunderlich: Gezählt wird die mittlere Wortlänge aller Wörter im Text des Korpus. Wenn jetzt mehr Text von der gleichen

**Tab. 5.2** Mittlere Wortlänge für Tokens und Types bei wachsender Korpusgröße

Datenbank und Größe	Mittl. Wortlänge für Tokens	Mittl. Wortlänge für Types
deu-de_web-wrt_2019_1K	5,9201	8,8183
deu-de_web-wrt_2019_10K	6,1825	10,3570
deu-de_web-wrt_2019_100K	6,2032	11,7019
4 deu-de_web-wrt_2019_1M	6,1996	12,8293
5 deu-de_web-wrt_2019_10M	6,2011	13,8822
6 deu-de_web-wrt_2019_100M	6,2018	14,8834

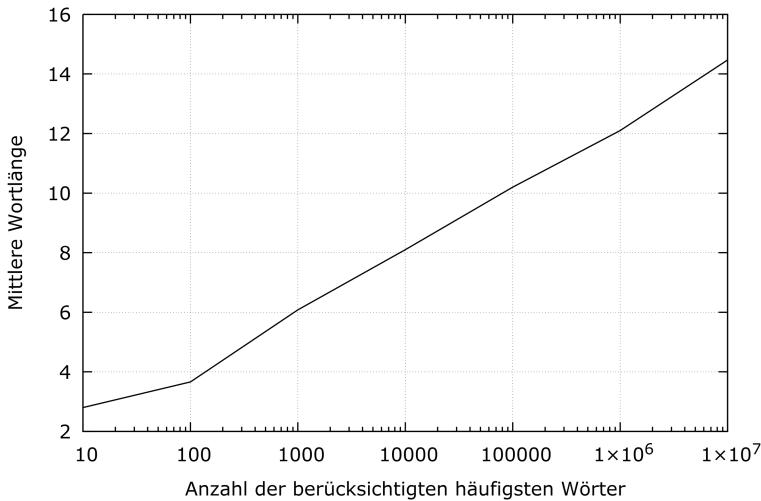
Sorte hinzukommt, gibt es keinen Grund zur Annahme, dass sich die so gemessene mittlere Wortlänge aller Wörter im Text verändert. Wie zu sehen ist, ist diese Zahl auch schon für kleine Textmengen stabil. Bei der zweiten Zahl ist das Verhalten anders: Die durchschnittliche Wortlänge aller verschiedenen Wörter wächst beständig mit der Korpusgröße. Dies lässt sich dadurch erklären, dass bei der Vergrößerung des Korpus die Anzahl der verschiedenen Wörter steigt (siehe Beispiel im Abschn. 5.1.4.3 und Tab. 5.1). Aber natürlich kommen immer seltener Wörter hinzu und seltener Wörter haben oft eine beachtliche Länge. Dadurch steigt auch die mittlere Länge für die Types langsam an. Die Graphik der Abb. 5.2 zeigt, dass das Längenwachstum sehr regelmäßig ist und noch lange anhalten kann.

Die Tab. 5.2 zeigt die mittlere Wortlänge für Tokens und Types bei wachsender Korpusgröße. Die mittlere Wortlänge der Types wächst im Gegenteil zur mittleren Wortlänge der Tokens.

Um das Wachstum in der letzten Spalte genauer zu analysieren, betrachten wir in der Abb. 5.2 eine graphische Darstellung der mittleren Wortlänge in Abhängigkeit von der Korpusgröße. Bei logarithmischer Darstellung der Korpusgröße ergibt sich für die größeren Korpora ein fast linearer Zusammenhang: Steigt die Korpusgröße um den Faktor 10, so wächst die mittlere Wortlänge ziemlich genau um zwei Buchstaben.

Diese zwei Arten, einen Messwert für alle Wörter im Text (Tokens) oder nur für die verschiedenen Wörter (Types) zu berechnen, bezeichnet man als Messung mit oder ohne Berücksichtigung der Vielfachheit, da wiederholt vorkommende Wörter auch wiederholt berücksichtigt werden oder nicht. Messwerte, gemessen mit oder ohne Berücksichtigung der Vielfachheit unterscheiden sich in der Regel und es ist wichtig anzugeben, wie gemessen wurde.

Auch müssen nicht beide Messwerte gleich sinnvoll sein: Bei der mittleren Wortlänge ist die Messung mit Berücksichtigung der Vielfachheit stabil und unabhängig von der Korpusgröße. Bei der Messung ohne Berücksichtigung der Vielfachheit muss entweder die Korpusgröße angegeben werden oder man versucht, einen mathematischen Zusammenhang zwischen Messgröße und Korpusgröße herzustellen.



**Abb. 5.2** Graphische Darstellung der mittleren Wortlänge

### 5.1.5 Abhängigkeit von der Zusammensetzung des Korpus

#### 5.1.5.1 Abhängigkeit vom Text-Genre

Die für ein Korpus verwendeten Texte können ursprünglich für ganz unterschiedliche Zwecke verfasst worden sein. Dem entsprechen unterschiedliche Textgenres wie Zeitungstext, literarische Texte oder wissenschaftliche und enzyklopädische Texte. Von ganz anderer Art sind Texte in den sozialen Netzen. Diese verschiedenen Textgenres haben Eigenschaften, welche sich auch auf sprachstatistische Messwerte auswirken können:

- In wissenschaftlichen Texten kommen viel mehr Fremdwörter und andere Fachbegriffe vor als in Zeitungstexten, einige davon sind sonst extrem selten. Der Wortschatz und die durchschnittliche Wort- und Satzlänge sind größer.
- Literarische Texte sind häufig in der Ich-Form geschrieben. Dadurch sind die Personalpronomen *ich* und *du* viel häufiger. Die Rangfolge der Wörter ist stark verändert
- Auch im Internet sind die Personalpronomen *ich* und *du* viel häufiger in Zeitungstexten.
- Zeitungstexte haben mehr Redundanz als enzyklopädische Texte, da in der Zeitung über wichtige Ereignisse wiederholt berichtet wird.
- Die Verteilung von Jahreszahlen ist je nach Quelle ganz verschieden. Bei Zeitungsartikeln aus einem Jahr sind die aktuelle Jahreszahl und die kurz davorliegenden Jahreszahlen häufig. Literarische Quellen enthalten häufig keinerlei Jahreszahlen.

- Bei Zeitungstexten oder anderen redaktionell verfassten Texten wird durch die Redaktion eine gewisse Qualität der Texte sichergestellt. Die Texte in sozialen Netzen haben dagegen oft nicht-professionelle Autoren oder Autorinnen. Es gibt dort einen laxen Umgang mit Orthographie, speziell mit der Groß-/Kleinschreibung. Sätze sind nicht immer vollständig bzw. nicht voneinander getrennt. Emoticons sind Bestandteile von Sätzen.

### 5.1.5.2 Fehlen gesprochener Sprache

Da es zum gegenwärtigen Zeitpunkt viel mehr geschriebenen Text in elektronischer Form gibt als gesprochenen Text in schriftlicher Form, beziehen sich fast alle Messungen auf geschriebene Sprache. Trotzdem ist bekannt, dass gesprochene Sprache in vielerlei Hinsicht andere Eigenschaften hat als geschriebene Sprache. Solche Eigenschaften sind:

- Gesprochene Sätze sind kürzer und syntaktisch einfacher als geschriebene.
- Gesprochene Sprache verwendet weniger lange oder seltene Wörter.
- Die Themenverteilung in gesprochener Sprache ist anders.
- Gesprochene Sprache ist öfter informell, geschriebene Sprache öfter formell.
- Einige seltene Wörter werden bevorzugt in der gesprochenen Sprache verwendet, z. B. Schimpfwörter oder seltene Wörter in Redewendungen.
- Viele Äußerungen in der gesprochenen Sprache sind keine vollständigen Sätze.

### 5.1.6 Abhängigkeit von Optionen bei der der Korpuserstellung

In Abhängigkeit von der späteren Verwendung des Korpus ist man vielleicht nicht an den Texten selbst interessiert, sondern nur an daraus ermittelten statistischen Messwerten oder anderen Daten. Deshalb kann man an dem Korpus einige Operationen vornehmen, von denen man erwartet, dass sie das Ergebnis nicht verschlechtern. Im Weiteren benutzt man nicht das Korpus selbst, sondern nur die ermittelten Zahlenwerte. Als typische Folge betrachtet man auch nicht mehr das Korpus, obwohl man dann die Ursache für unangebrachte Messergebnisse erkennen würde. Deshalb wirken sich die Details bei der Korpuserstellung möglicherweise stark auf die ermittelten Messwerte aus (vgl. Eckart et al. 2012) und speziell beim Vergleich von Messergebnissen ist eine vergleichbare Korpuserstellung wichtig.

#### 5.1.6.1 Verzerrte Ergebnisse durch mangelnde Qualität der Rohdaten

##### Dubletten

Beim zufälligen Crawling im Internet werden einige Sätze mehrfach gesammelt. Das liegt daran, dass einige Sätze in verschiedenen Texten vorkommen („Ich liebe dich!“), dass einige Dokumente mehrfach an verschiedenen Stellen im Internet auffindbar sind (wie das Grundgesetz oder die Bibel), oder dass technisch bedingt manche Sätze auf

vielen Seiten auftauchen („Sie müssen sich anmelden, um einen Beitrag schreiben zu können.“).

Dubletten verzerren die Statistiken und weisen für manche Wörter unerwartete Häufigkeiten oder Kontexte aus. Für statistische Betrachtungen ist es sinnvoll, mehrfach vorkommende Sätze bis auf ein Exemplar zu löschen. Steht jeder Satz auf einer Zeile und kommt es nicht auf die Reihenfolge der Sätze an, lässt sich das ohne jede Programmierung mit dem Unix-Befehl `sort -u` erledigen.

### Quasidubletten

Quasidubletten, also Sätze, die sich nur unwesentlich unterscheiden, wurden bereits in Abschn. 4.2 als problematisch erkannt. Ob solch ein Unterschied für die spätere Anwendung wirklich unwesentlich ist, kann von der betrachteten Fragestellung abhängen.

Es stellt sich die Frage, wie möglichst viele dieser Quasidubletten automatisch erkannt werden können. Hier bietet sich wieder die alphabetische Sortierung aller Sätze an: Wenn, wie in vielen Beispielen aus Abschn. 4.2, zwei solche Quasidubletten einen genügend langen übereinstimmenden Satzanfang haben, stehen sie wahrscheinlich bei der alphabetischen Sortierung unmittelbar hintereinander. Es reicht also, die alphabetisch sortierte Satzliste einmal zu durchlaufen und bei aufeinanderfolgenden Sätzen mit genügend langem übereinstimmenden Satzanfang zu prüfen, ob es sich um Quasidubletten handelt.

Dieses Verfahren funktioniert nicht, wenn sich die Quasidubletten weit vorn im Satz unterscheiden. Dann kann man dasselbe Verfahren auf die von hinten zu lesenden Sätze anwenden oder man muss auf gänzlich andere Verfahren ausweichen. In Abschn. 5.6.4 wird die semantische Satzähnlichkeit auf der Basis von doc2vec betrachtet.

### Fremdsprachen, speziell Englisch

Die Sprachseparierung bei der Korpuserstellung kann in ungünstigen Fällen fremdsprachige Teile akzeptieren statt ablehnen. Das größte Problem stellt hier die englische Sprache dar, da in vielen Texten englische Zitate, Film- oder Buchtitel oder Eigennamen vorkommen. Diese englischen Bestandteile wiederum enthalten auch Stopwörter, sodass praktisch alle häufigen englischen Wörter auch in einem deutschsprachigen Korpus gefunden werden können. Beispiele für solche Sätze sind:

*Als Zugabe werden „What a Wonderful World“ und „In the Mood“ gegeben, dafür gibt es stehende Ovationen.*

*Das Skript kann praktisch ohne Konfiguration „out of the box“ benutzt werden.*

Dadurch erscheinen englische Wörter in der Wortliste des deutschen Korpus, was zu Verwirrung führen kann.

Dieses Problem des Auftretens einer zweiten Sprache tritt nur dann auf, wenn diese den gleichen Zeichensatz benutzt. In anderen Fällen wie der Erstellung eines russischen Korpus lassen sich Teile in englischer Sprache dadurch sicher entfernen, dass man keine lateinischen Zeichen in Sätzen in russischer Sprache zulässt.

### 5.1.6.2 Fragwürdige Optionen

Sicher wird durch das Weglassen mancher optionaler Schritte bei der Korpuserstellung die Verarbeitung einfacher oder schneller, es kann Speicherplatz gespart werden oder es entstehen andere Vorteile. Vielleicht besteht aber auch die Gefahr, dass die Ergebnisse schlechter oder verfälscht werden. Die folgenden Optionen sollten nur nach reiflicher Überlegung angewendet werden. Alle haben die Eigenschaft, dass sich ihre Anwendung später nicht mehr rückgängig machen lässt.

**Stoppwörter weglassen** Stoppwörter allein tragen keine Information. Dies führt immer wieder zu der Idee, die Stoppwörter wegzulassen oder wenigstens nicht zu indexieren. Dadurch tritt ein merklicher Vorteil bei der Bearbeitung ein: Die Datenmenge wird etwa um die Hälfte reduziert, und die Wörter mit der größten Häufigkeit sind verschwunden. Nachteile sind, dass dann plötzlich Wörter als unmittelbare Nachbarn erscheinen, die wegen dazwischenliegender Stoppwörter niemals nebeneinander stehen. Damit wird auch die Suche nach Wortgruppen schwierig, wenn diese Wortgruppen Stoppwörter enthalten.

**Umlaute auflösen** Vor der Einführung der Personalcomputer waren Großrechner häufig nicht in der Lage, mit Umlauten und Sonderzeichen umzugehen. Aus dieser Zeit stammt das Vorgehen, Umlaute aufzulösen (*ä* -> *ae* usw.). Diese Umwandlung lässt sich später nicht mehr vollständig umkehren und führt zu zusätzlichen Schwierigkeiten. Spätestens seit praktisch alle Computer mit der Zeichencodierung durch **UTF-8** umgehen können, ist von der Auflösung von Umlauten konsequent abzuraten.

**Alles in Kleinschreibung** In der Anfangszeit der Großrechner waren diese nur zur Verarbeitung von Großbuchstaben fähig. Die moderne Version dieser Selbstbeschränkung ist die Transformation aller Texte in Kleinbuchstaben. Damit wird zwar die Großschreibung am Satzanfang beseitigt, aber es werden auch viele sonst sinnvollerweise unterschiedene Wörter vereinheitlicht. Dies ist insbesondere bei der Verarbeitung von deutschen Korpora problematisch, da Großschreibung Bedeutungen und Wortarten unterscheidet. Dadurch überwiegen in der Regel auch bei dieser Option die Nachteile.

### 5.1.6.3 Umgang mit seltenen Ereignissen

Eines der großen Probleme in der Sprachstatistik kommt daher, dass es viele seltene Ereignisse gibt. Im Abschn. 5.2 werden wir sehen: Etwa die Hälfte der Wörter (Types) tritt nur einmal im Korpus auf. Betrachten wir das folgende Gedankenexperiment: Ein großes Korpus  $K$  wird zerlegt in zwei immer noch recht große Korpora  $K_1$  und  $K_2$ . Deren Worthäufigkeiten werden zur Ermittlung von Wortwahrscheinlichkeiten genutzt. Wenn ein extrem seltes Wort  $w$  nur einmal in  $K$  vorkommt, dann kommt es nach der Teilung entweder in  $K_1$  oder in  $K_2$  vor. In einem Korpus wird eine kleine, positive Auftretenswahrscheinlichkeit für  $w$  ermittelt, im anderen die Wahrscheinlichkeit null.

Das Beispiel zeigt, dass eine ermittelte Wahrscheinlichkeit null nicht „unmöglich“ bedeutet. Da in mathematischen Formeln solche Wahrscheinlichkeiten aber häufig multipliziert werden, ersetzt man in der Praxis die ermittelte Wahrscheinlichkeit null durch einen kleinen positiven Wert. Dadurch wird aus „unmöglich“ die schwächere Hypothese „extrem unwahrscheinlich“. Durch dieses Vorgehen verbessern sich viele mathematische Vorhersagen, siehe auch Smoothing im Abschn. 5.5. Allerdings gibt es dadurch aus statistischer Sicht keine unmöglichen Wörter oder unmöglichen Wortkombinationen mehr, was auch nicht der Realität entspricht.

---

## 5.2 Zipfsches Gesetz

Für das Text Mining spielt die Häufigkeit von Wörtern eine wesentliche Rolle. Von besonderem Interesse ist das sog. Zipfsche Gesetz, das wir nachfolgend vorstellen. Es bildet die Grundlage für strukturelle Analysen des Vokabulars von Textkollektionen sowie den Korpusvergleich und die Terminologie-Extraktion. Auch wenn die Vorhersagen des Zipfschen Gesetzes für reale Korpora nur annähernd gelten, ermöglicht es zahlreiche praktische Anwendungen wie z. B. Abschätzungen über den Umfang des Vokabulars eines Textes oder über die Anzahl von Wörtern, die genau  $n$ -mal im Text auftreten. Das Wissen über die Universalität des Zipfschen Gesetzes auf Sprache ist auch ein wichtiger Baustein bei der Entwicklung erfolgreicher Verfahren in der Sprachtechnologie, da es dabei hilft falsche Annahmen über die Häufigkeit von Wörtern zu vermeiden, z. B. dass die meisten Wörter ungefähr gleichhäufig auftreten würden, oder dass seltene Wörter etwas Besonderes seien.

### 5.2.1 Zusammenhang zwischen Rang, Häufigkeit und Wortlänge

In seinem Buch „Human Behaviour and the Principle of Least Effort“ (Zipf 1949) skizziert George Kingsley Zipf eine Theorie menschlicher Sprache und Gesellschaft, die sich am Effizienzbegriff der Ökonomie orientiert und ein Kosten-Nutzen-optimierendes „Prinzip des geringsten Aufwands“ in den Mittelpunkt stellt. Ausgangspunkt seiner Überlegung ist die Frage, wie das **Vokabular** einer Sprache strukturiert sein muss, damit das Verhältnis zwischen der Anzahl von Wörtern und der Anzahl von Bedeutungen, die jedes Wort hat, aus Sicht des oder der Sprechenden und aus Sicht des oder der Hörenden einer Sprache optimal ist. Seinem Ansatz nach wäre für einen Sprecher bzw. eine Sprecherin der Kodierungsaufwand dann am geringsten, wenn er so wenig Wörter wie möglich verwenden würde, auch wenn diese Wörter viele Bedeutungen haben, während für einen Hörer bzw. eine Hörerin der Aufwand für die Dekodierung dann am geringsten ist, wenn er bzw. sie nur mit eindeutigen Wörtern konfrontiert würde, auch wenn dies wiederum sehr viele wären. Zipf spricht von diesen beiden Aspekten als von zwei im Widerstreit stehenden Kräften, der *Force of Unification* und der *Force of Diversification*. Seine zentrale

Annahme ist nun, dass das Vokabular der gesprochenen Sprache eine (optimale) Balance zwischen genau diesen zwei Kräften darstellt (Zipf 1949, S. 22). Die Kraft der Unifikation bewirkt dabei eine Reduzierung der Anzahl der Wörter verbunden mit einer Erhöhung der Häufigkeit ihrer Verwendung, wogegen die Kraft der Diversifikation in umgekehrter Richtung eine Erhöhung der Anzahl der Wörter verbunden mit einer Verminderung der Häufigkeit ihrer Verwendung bewirkt. Es folgt daraus, dass die Anzahl von Wörtern und die Häufigkeit ihrer Verwendung die zentralen Parameter für die Beschreibung der Vokabularstruktur einer Sprache sind: „[...] number and frequency will be the parameters of vocabulary balance“ (Zipf 1949, S. 23).

Seine Überlegung zum Zusammenhang zwischen der Anzahl von Wörtern und der Häufigkeit ihrer Verwendung illustriert Zipf am Beispiel des Vokabulars von James Joyce *Ulysses* (unter Verwendung des sogenannten *Hanly Index*). Dabei sind alle Wörter (im Sinne von Token), die im Text vorkommen, ihrer Häufigkeit nach in eine geordnete Liste geschrieben, die in Tab. 5.3 dargestellt ist.

Wie man erkennt, ist das Produkt aus dem **Rang** eines Wortes (innerhalb der häufigkeitssortierten Liste) mit seiner Häufigkeit für alle Wörter in etwa derselbe Wert. Dieser Zusammenhang wird allgemein als das **Zipfsche Gesetz** bezeichnet:

$$r \cdot f \approx k \text{ (mit Rang } r, \text{ Häufigkeit } f \text{ und korpuspezifischer Konstante } k) \quad (\text{Gl. 5.1})$$

Um den vom Zipfschen Gesetz formulierten Zusammenhang berechnen und für die Analyse realer Korpora nutzen zu können, setzen wir im Folgenden ein ideales Zipf-Korpus voraus, in dem die formulierten Zusammenhänge nicht nur näherungsweise, sondern exakt gelten. Wir stellen zunächst die Berechnungsansätze vor und zeigen dann an ausgewählten Korpora, wie gut die vorhergesagten mit den tatsächlichen Werten übereinstimmen.

**Tab. 5.3** Hanly Index von James Joyce Ulysses (aus Zipf 1949, S. 24), Auszug

I <b>Rang r</b>	II <b>Frequenz f</b>	III <b>Produkt aus I und II r · f = k</b>
10	2653	26.530
20	1211	26.220
30	926	27.780
50	556	27.800
100	265	26.500
500	50	25.500
1000	26	26.000
2000	12	24.000
5000	5	25.000
10.000	2	20.000
20.000	1	20.000
29.899	1	29.899

Wird der vom Zipfschen Gesetz beschriebene Zusammenhang in einem Funktionsgraph mit Rang  $r$  als x-Achse und Häufigkeit  $f$  als y-Achse dargestellt, erhält man eine Hyperbel.

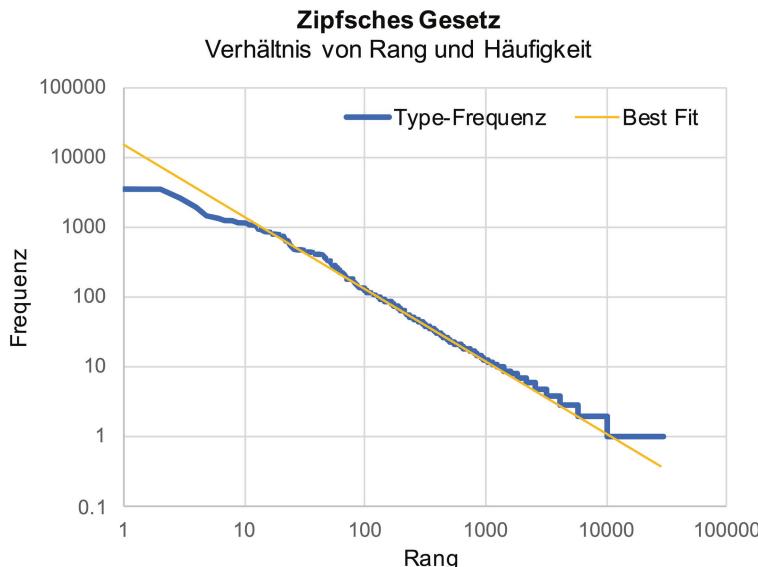
Wählt man für die x- und y-Achse eine doppelt logarithmische Darstellung, also  $\log(r)$  und  $\log(f)$ , dann erhält man mit  $\log(f) = -\log(r) + \log(k)$  eine Gerade mit negativer Steigung  $-1$ .

Werden reale Textdaten verwendet, dann entspricht dieses ideale Verhältnis von Rang und Häufigkeit jedoch nicht dem tatsächlichen Funktionsverlauf, wie es Abb. 5.3 verdeutlicht. (Für die Berechnung der Geraden sind dabei die ersten 10 häufigsten Types ebenso wie der ‚long tail‘ von Types mit Häufigkeit 1 nicht berücksichtigt worden). Der Zusammenhang zwischen Rang und Häufigkeit wird für Wörter mit sehr kleinem oder sehr großem Rang nur unzureichend durch die einfache Formel  $f = k/r$  wiedergegeben. Im Diagramm mit logarithmisch skalierten Achsen weichen diese Wörter stärker von der vorausgesagten Geraden ab.

Eine bessere Beschreibung liefert folgende Formel nach Benoît B. Mandelbrot (vgl. Mandelbrot 1953):

$$(r + c_1)^{1+c_2} \cdot f \approx k \quad (\text{Gl. 5.2})$$

Die beiden Konstanten  $c_1$  und  $c_2$  dienen hier als Parameter. Sie ermöglichen eine Anpassung an die konkreten Daten. Hierbei hat  $c_1$  großen Einfluss auf die Krümmung im Bereich der niederen Ränge, während  $c_2$  die Anpassung im Bereich der hohen Ränge vornimmt. Setzt man  $c_1 = c_2 = 0$ , ergibt sich die ursprüngliche Formel von Zipf.



**Abb. 5.3** Rang gegen Häufigkeit, Referenzkorpus Deutscher Wortschatz 2019, 10.000 Sätze, best fit:  $x^{-1.0339134398416199} + 15.446$

Das „Prinzip des geringsten Aufwands“ findet sich insbesondere auch in der Kodierung der Wörter wieder. So sind zum Beispiel die am häufigsten gebrauchten Wörter meist sehr kurze Funktionswörter, siehe Tab. 5.4, und seltener verwendete Wörter sind länger. Fürs Deutsche gilt dabei, dass ein Wort um nahezu zwei Buchstaben länger wird, wenn sich der Rang um den Faktor 10 erhöht (Eckart et.al. 2011), wie wir es schon in Abb. 5.2 gesehen haben.

## 5.2.2 Vorhersagen und Anwendungen

Nehmen wir an, dass für ein Textkorpus bereits eine Häufigkeitssortierte Liste erzeugt worden ist. Am Anfang der Liste stehen diejenigen Types, die am häufigsten vorkommen und deshalb den niedrigsten Rang haben; Types, die nur einmal im Text vorkommen, stehen am Ende der Liste. Das letzte Element der Liste bekommt den **höchsten Rang**. Jeder Type kommt zwar  $f$ -mal im Textkorpus vor, für ein bestimmtes  $r$  gibt es aber möglicherweise mehrere Types, die genauso oft im Korpus vorkommen. Sind für ein Textkorpus die korpuspezifische Konstante  $k$  und die Gesamtzahl der Tokens gegeben, dann können aus dem Zipfschen Gesetz eine Reihe von Abschätzungen zum Verhältnis von Rang und Häufigkeit von Types in diesem Korpus abgeleitet werden.

Betrachten wir als erstes die Abschätzung der Häufigkeit  $f$  für einen Type mit Rang  $r$ . Wir schreiben dafür  $f_r$  und wenden das Zipfsche Gesetz an. Es gilt demnach  $r \cdot f_r = k$  und für die Häufigkeit eines Types auf Rang  $r$ :

$$f_r = k/r \quad (\text{Gl. 5.3})$$

Insbesondere lässt sich die Häufigkeit der häufigsten Wörter, die in der Häufigkeitssortierten Liste aller Types auf Rang 1 steht, abschätzen:

$$f_1 = k \quad (\text{Gl. 5.4})$$

**Tab. 5.4** 10 häufigste Wörter in einem Korpus des Deutschen. (Quelle: Deutscher Wortschatz, deu\_newscrawl-public\_2019\_100K, 1.498.643 Tokens, 145.971 Types)

Rang r	Wort	Häufigkeit f	$r \cdot f$
1	der	44.066	44.066
2	die	42.150	84.300
3	und	32.366	97.098
4	in	25.799	103.196
5	den	17.083	85.415
6	das	13.627	81.762
7	mit	13.227	92.589
8	ist	13.147	105.176
9	zu	13.083	117.747
10	von	12.816	128.160

Betrachten wir nun den höchsten Rang von Types, die genau f-mal in der frequenzsortierten Liste eines Korpus vorkommen. Wir schreiben dafür  $r_f$  und wenden das Zipfsche Gesetz an. Es gilt demnach  $r_f \cdot f = k$ . Der höchste Rang von n Types, die genau f-mal vorkommen, ist also:

$$r_f = k/f \quad (\text{Gl. 5.5})$$

### Höchster Rang von Wörtern, die genau 50-mal in einem Text vorkommen

Gegeben sei ein deutschsprachiges Korpus mit einer Gesamtzahl von  $N = 150.000$  Tokens und einer korpuspezifischen Konstante  $k = 12.000$ . Nach der Vorhersage des Zipfschen Gesetzes befindet sich dann ein Wort, das genau 50-mal im Text vorkommt, in der Häufigkeitssortierten Wortliste ab Rangposition  $r_{50} = 12.000/50 = 240$ . ◀

Für die Abschätzung des höchsten Rangs derjenigen Types, die nur einmal vorkommen, setzen wir  $f = 1$  und erhalten.

$$r_1 = k \quad (\text{Gl. 5.6})$$

Da der höchste Rang derjenigen Types, die nur einmal vorkommen, am Ende der Häufigkeitssortierten Liste aller Types eines Korpus steht, ist die Abschätzung für  $r_1$  auch eine Abschätzung des **Umfangs v des Vokabulars** eines Korpus,  $v = r_1 = k$ .

Wir können den Rang eines Types verwenden, um abzuschätzen, wie oft ein Type in einem Korpus vorkommt (vgl. Salton 1989). Wir schreiben für die Anzahl n von Tokens von einem Type  $n_f$ . Diese Anzahl entspricht genau der Differenz zwischen dem höchsten Rang derjenigen Types, die in der frequenzsortierten Liste genau f-mal vorkommen, und dem höchsten Rang derjenigen Types, die genau  $(f+1)$ -mal vorkommen. Daher gilt:

$$\begin{aligned} n_f &= r_f - r_{f+1} = k/f - k/(f+1) \\ &= k/(f(f+1)) \end{aligned} \quad (\text{Gl. 5.7})$$

Für Types mit Häufigkeit 1 liefert das Zipfsche Gesetz die Abschätzung, dass ihr Anteil die Hälfte aller Types eines Korpus ausmacht:

$$n_1 = k/2 \quad (\text{Gl. 5.8})$$

Die aus dem Zipfschen Gesetz ableitbaren Abschätzungen für ein reales Korpus hängen entscheidend von der Wahl der korpuspezifischen Konstante  $k$  ab. Die korpuspezifische Konstante  $k$  ist abhängig von der Korpusgröße. Im idealen Zipf-Korpus ist die korpuspezifische Konstante  $k_i$  für alle Ränge gleich. In realen Textdaten ist diese Annahme, dass  $k_i = k$ , jedoch meist nicht erfüllt. Wir können  $k$  auf verschiedene Weise bestimmen, z. B. als Mittelwert der korpuspezifischen Konstanten aller Types oder als höchster Rang derjenigen Types, die nur einmal im Korpus vorkommen. Die Abweichungen der vorhergesagten von den tatsächlichen Werten unterscheiden sich deutlich. Die Genauig-

keit der Vorhersagen hängt darüber hinaus ab vom Rang der Types und der Größe des untersuchten Korpus.

### Beispiel zu korpuspezifischen Konstante

Betrachten wir zur Erläuterung einige Abschätzungen für das Korpus deu\_de-news-wrt\_2019\_10K-sentences.txt, einem Korpus mit 10.000 Sätzen, 131.245 Tokens und 25.926 Types (wie wir es bereits für Abb. 5.3 kennengelernt haben).

In der einen Variante berechnet sich für jeden Type die korpuspezifische Konstante  $k_i$  als Produkt des Ranges  $r_i$  und der Häufigkeit  $f_i$ :  $k_i = r_i \cdot f_i$ . Die korpuspezifische Konstante  $k$  erhalten wir dann als Mittelwert der korpuspezifischen Konstanten aller Types:

$$k = \frac{1}{|Types|} \sum_{i=1}^{|Types|} k_i \quad (\text{Gl. 5.9})$$

Als Mittelwert der korpuspezifischen Konstanten für das genannte Korpus hat  $k$  den Wert  $k = 12.276$ .

In der anderen Variante bestimmen wir  $k$  nach Gl. 5.6 als den höchsten Rang derjenigen Types, die genau einmal vorkommen. Für das genannte Korpus erhalten wir so  $k = 25.926$ .

Abhängig davon, wie  $k$  bestimmt wird, unterscheiden sich die Abschätzungen deutlich, wie die Tab. 5.5 verdeutlicht (Abweichungen der Vorhersage von der tatsächlichen Anzahl der Types jeweils bezogen auf die Vorhersage). ◀

Empirische Untersuchungen zeigen, dass die Abweichungen bei der Vorhersage der Vorkommen eines Types mit der Korpusgröße zunehmen (vgl. Anhang A9). Dies liegt vermutlich daran, dass die Anzahl der Types und Tokens in einem Korpus mit zunehmender Korpusgröße nicht in gleichem Maße wächst (vgl. Abschn. 5.1). Trotz aller Einschränkungen ist dennoch festzuhalten, dass die aus dem Zipfschen Gesetz ableitbaren Vorhersagen für das Text Mining nützliche Hinweise auf den Umfang und die Strukturierung des Vokabulars eines Korpus geben.

**Tab. 5.5** Abschätzungen für das Korpus deu\_de-news-wrt\_2019\_10K-sentences.txt

Häufigkeit	Rangbereich	Anzahl Types	Vorhersage $k = 12.276$	Abweichung in %	Vorhersage $k = 25.926$	Abweichung in %
1	9.673–25.926	16.253	6.138	164,79 %	12.963	25,38 %
5	2.585–3.191	606	409	48,16 %	864	-29,88 %
10	1.298–1.445	147	112	31,25 %	236	-37,63 %
50	254–261	7	5	40,00 %	10	-30,00 %
100	130–132	3	1	200,00 %	3	0,00 %

## 5.3 Kookkurrenzen

Das Erkennen von semantischen Zusammenhängen zwischen einzelnen Wörtern, die in einem Text vorkommen, ist eine zentrale Herausforderung an das Text Mining. Die hier verfolgte Herangehensweise geht davon aus, dass häufiges gemeinsames Auftreten zweier Wörter in enger textueller Nachbarschaft ein starkes Indiz für einen semantischen Zusammenhang ist.

### 5.3.1 Struktur von signifikanten Kookkurrenzen

Zunächst sollen Beispiele für häufig gemeinsam auftretende Wörter auf ihren Zusammenhang hin untersucht werden.

#### Häufig gemeinsam auftretende Wörter

Für jedes der folgenden Wortpaare gilt offensichtlich: Enthält ein Satz eines der beiden Wörter, so ist das Auftreten des anderen Wortes recht wahrscheinlich.

1. *Los – Angeles*
2. *schwere – Krankheit*
3. *Romeo – Julia*
4. *Polizei – verhaftet*
5. *der – die*
6. *Paris – London*

Die Gründe für dieses gemeinsame Auftreten sind unterschiedlich.

1. Das erste Paar bildet zusammen den häufig gefundenen Städtenamen *Los Angeles*.
2. Beim zweiten Beispiel beschreibt das Adjektiv *schwere* eine typische Eigenschaft von *Krankheit*.
3. Das dritte Paar begegnet uns meist in der Fügung *Romeo und Julia*.
4. Dagegen treten die Wörter *Polizei* und *verhaftet* zwar häufig gemeinsam in Sätzen aus Zeitungsberichten über Polizeieinsätze auf, aber nicht in einer festen Fügung. Trotzdem besteht ein semantischer Zusammenhang, da Verhaftungen in der Regel Erfolgsmeldungen der Polizei sind.
5. Das gemeinsame häufige Auftreten der Artikel *der* und *die* hingegen hat keinen semantischen Hintergrund. Beide Artikel sind jeder für sich allein genommen schon so häufig, dass sie allein aus diesem Grund auch häufig gemeinsam in einem Satz auftreten.

6. Das Wortpaar *Paris* – *London* hat nicht die gleiche starke Eigenschaft wie die Wortpaare (1)–(3) oben, dass man beim Auftreten eines Wortes mit großer Wahrscheinlichkeit auch das zweite vorfindet. Trotzdem hat sich gezeigt, dass sie oft zusammen auftreten. ◀

Das statistisch auffällige gemeinsame Vorkommen zweier Wörter bezeichnen wir als **signifikante Kookkurrenz**.

Für zwei Wörter vergleicht man die Anzahl der gemeinsamen Vorkommen mit derjenigen Anzahl, die man im Falle von zufällig gewählten Wörtern entsprechender Häufigkeit erwarten würde. Die mathematische Beschreibung dieser statistischen Auffälligkeit erfolgt im Abschn. 5.3.2. Unter dem gemeinsamen Vorkommen soll dabei einerseits ein Vorkommen als unmittelbare Nachbarn, andererseits aber auch ein Vorkommen in größeren Abständen verstanden werden, wie die Beispiele von oben zeigen.

Um diese Anordnung von Kookkurenzen zu berücksichtigen, werden solche Wortpaare bezeichnet als:

- *Nachbarschaftskookkurrenzen*, falls sie unmittelbar benachbart auftreten, bzw.
- *Satzkookkurrenzen*, falls das gemeinsame Auftreten der zwei Wörter in einem Satz betrachtet wird.

Statt dieser zwei Nachbarschaftstypen sind auch sogenannte Fenster üblich: Um ein Wort herum denkt man sich jeweils ein Textfenster von beispielsweise fünf Wörtern nach rechts und links. Die gemeinsamen Vorkommen werden dann in solchen Fenstern gezählt.

Hier übernehmen die Sätze die Rolle von Fenstern, weil man die semantischen Grenzen an Satzenden nicht ignorieren will. Mit der bereits besprochenen Textsegmentierung in Sätze (vgl. Abschn. 3.2.3) bereitet dieses Vorgehen auch keinen großen zusätzlichen Arbeitsaufwand.

Die obigen Beispiele liefern erste Muster für signifikante Beispiele für Kookkurenzen:

- **Eigennamen** und **feste Fügungen** liefern signifikante Kookkurenzen, weil die beteiligten Wörter
- immer wieder zusammen auftreten. Hierbei handelt es sich um signifikante Nachbarschaftskookkurenzen.
- Substantive mit Adjektiven, die typische Eigenschaften beschreiben, bilden signifikante Nachbarschaftskookkurenzen.
- Handlungen mit typischen Subjekten oder Objekten bilden signifikante Satzkookkurenzen.
- Wörter, die häufig zusammen in Aufzählungen stehen, bilden signifikante Satzkookkurenzen.

Weiter beobachtet man:

- Signifikante Nachbarschaftskookkurrenzen sind auch signifikante Satzkookkurrenzen, weil unmittelbar benachbarte Wörter natürlich im gleichen Satz stehen.

Als Beispiele für signifikante Satzkookkurrenzen werden deshalb immer solche gewählt, die nicht gleichzeitig signifikante Nachbarschaftskookkurrenzen sind.

### 5.3.2 Maße für die statistische Signifikanz einer Kookkurrenz

Aufgabe eines Maßes für die statistische Signifikanz einer Kookkurrenz ist es, einem Paar von Wörtern (wie z. B. *Polizei – verhaftet*) eine Zahl zuzuordnen, welche die Signifikanz dieser Kookkurrenz beschreibt. Signifikantere Kookkurrenzen sollen solche sein, deren Zusammengehörigkeit dem Betrachtenden relativ stark erscheint. Damit soll die berechnete Signifikanz dem intuitiven Gefühl der Zusammengehörigkeit von Wörtern entsprechen (vgl. Evert und Krenn 2001). Dabei ist zunächst nicht klar, wie z. B. diese Zusammengehörigkeit des Paares *Romeo – Julia* mit der Zusammengehörigkeit von *Polizei – verhaftet* verglichen werden soll.

Bleibt aber ein Wort des Paares unverändert, ist ein Vergleich intuitiv möglich.

Es erscheint beispielsweise der Zusammenhang des Wortes *Polizei* mit dem Wort *verhaftet* stärker als mit dem Wort *berittene*. Damit besteht zumindest prinzipiell die Möglichkeit, die Signifikanz von Kookkurrenzen empirisch zu überprüfen. Ein Beispiel eines solchen Vergleiches findet sich im Abschn. 5.3.3.

Jedem Paar von Wörtern wird durch das **Signifikanzmaß** ein Signifikanzwert zugeordnet. Paare mit einem Signifikanzwert oberhalb einer gewählten Schwelle werden als signifikant betrachtet. Leider gibt es nicht einen Schwellenwert, der die „guten“ Paare, denen man also intuitiv eine Zusammengehörigkeit zuspricht, von den „schlechten“, die auf den ersten Blick nicht zusammengehören, trennt. In der Praxis findet sich eher ein Übergangsbereich von einer bestimmten Breite, in dem die Paare vermischt auftreten. Setzt man die Signifikanzschwelle hoch an, sind die ausgewählten Paare zwar von guter Qualität, aber nur wenige solcher Paare werden gefunden. Bei niedrigerer Schwelle werden außer weiteren guten sofort auch ungewünschte Paare mitgeliefert.

Bevor konkrete Signifikanzwerte betrachtet werden sollen, werden zunächst die zur Verfügung stehenden Parameter untersucht: Welche Zahlenwerte können verwendet werden, um zu berechnen, wie auffällig das gemeinsame Auftreten zweier Wörter A und B (wie *Polizei* und *verhaftet*) ist?

Um die Anzahl der gemeinsamen Vorkommen von A und B richtig beurteilen zu können, benötigt man neben der Anzahl dieser gemeinsamen Vorkommen zusätzlich die Anzahlen für die Wörter A und B einzeln. Möglicherweise ist auch die Gesamtanzahl der Sätze wichtig.

Damit hat man die folgenden vier Größen:

$n_A, n_B$ : Anzahl der Sätze, die A bzw. B enthalten.

$n_{AB}$ : Anzahl der Sätze, die A und B gemeinsam enthalten.

$n$ : Gesamtzahl der Sätze.

Um aus diesen vier Zahlen ein Signifikanzmaß zu berechnen, kann auf verschiedene Weise vorgegangen werden. Am häufigsten verwendet wird das im Folgenden beschriebene Log-Likelihood-Maß. Weitere Maße werden im Abschn. 5.3.4 betrachtet.

Ausgangspunkt ist die (völlig unangemessene) Hypothese, dass die Vorkommen der zwei Wörter A und B nichts miteinander zu tun hätten, also statistisch unabhängig wären. Unter dieser Annahme kann aus diesen vier Zahlen nun die Wahrscheinlichkeit dafür berechnet werden, dass unter diesen Umständen in  $n$  Sätzen beide Wörter (mindestens)  $n_{AB}$ -mal zusammen angetroffen werden. Von dieser sehr kleinen Wahrscheinlichkeit wird der negative Logarithmus genommen und alles noch mit einer Konstante multipliziert, um auf anschauliche Werte zu kommen.

Das Ergebnis ist das sogenannte **Log-Likelihood-Maß** (vgl. Dunning 1993): Falls das gemeinsame Auftreten der Wörter häufiger ist als bei Unabhängigkeit zu erwarten wäre (also falls  $n \cdot n_{AB} > n_A \cdot n_B$ ), dann definieren wir die Signifikanz als.

$$\text{sig}(A, B) = 2 \left( \begin{array}{l} n \log n - n_A \log n_A - n_B \log n_B + n_{AB} \log n_{AB} + (n - n_A - n_B + n_{AB}) \\ \log(n - n_A - n_B + n_{AB}) + (n_A - n_{AB}) \log(n_A - n_{AB}) + (n_B - n_{AB}) \\ \log(n_B - n_{AB}) - (n - n_A) \log(n - n_A) - (n - n_B) \log(n - n_B) \end{array} \right) \quad (\text{Gl. 5.10})$$

Die Auswirkung der einzelnen Anzahlen auf das Ergebnis ist diesen Formeln nicht mehr ohne weiteres anzusehen. Selbstverständlich erwartet man, dass  $\text{sig}(A, B)$  monoton steigend in  $n_{AB}$  ist, also ein häufigeres gemeinsames Auftreten zu einer höheren Signifikanz führt. Unklar ist jedoch zunächst, wie sich die Signifikanz für Wortpaare aus verschiedenen Häufigkeitsbereichen unterscheidet. Wünschenswert wäre es, wenn für ein fest gewähltes Wort A die verschiedenen Wörter B, die durch fallende Signifikanz in eine geordnete Reihenfolge gebracht werden, in einer Reihenfolge erscheinen, sodass die von uns erwarteten Wörter in diesem Ranking möglichst weit vorn stehen. Wie im folgenden Abschnitt exemplarisch gezeigt wird, kommt das Ranking der Assoziationsstärke im menschlichen Gehirn recht nahe.

Das Bemerkenswerte an diesem Maß ist, dass das Log-Likelihood-Maß nicht nur dann große Werte liefert, wenn extrem seltene Wörter wiederholt miteinander auftreten. Auch im Falle mittel- oder hochfrequenter Wörter, bei denen die relative Häufigkeit der gemeinsamen Vorkommen nur wenig über dem bei zufälliger Verteilung erwarteten Wert liegt, können aufgrund der großen absoluten Zahlen eine hohe Signifikanz erhalten.

Beispiele dafür sind in der Tab. 5.6 angegeben. Die Tabelle enthält drei Gruppen von Satzkookkurrenzen, die jeweils näherungsweise eine Signifikanz von 20.000, 2000 bzw. 200 haben. Obwohl sich sowohl die einzelnen Worthäufigkeiten und die Anzahlen der

**Tab. 5.6** Satzkookkurrenzen mit annähernd gleicher Signifikanz

Wort A	Wort B	n <sub>A</sub>	n <sub>B</sub>	n <sub>AB</sub>	sig(A,B)
Kinder	Jugendliche	70.894	12.653	3566	20.178,4
Cristiano	Ronaldo	1400	2605	1259	20.486,3
über	hinaus	337.054	15.974	5967	19.632,7
grünes	Licht	1889	10.076	1547	19.603,7
rechnen	mit	10.606	1.247.973	7632	19.800,0
Grün	Rot	2334	2204	200	2001,2
Twente	Enschede	128	192	96	2000,6
bei	Temperaturen	509.257	6702	1416	2003,0
Innenminister	Bayerns	5665	2269	237	2003,1
Säbener	Straße	206	39.886	192	2000,4
eigenen	Regeln	41.438	7548	144	202,1
operativ	entfernt	377	10.865	30	196,4
ein	30-Jähriger	919.157	133	79	204,5
täuschend	echten	161	3810	21	203,0
in	Guatemala	2.401.624	347	200	202,1

gemeinsamen Vorkommen stark unterscheiden, ist intuitiv klar, dass die größeren Signifikanzen einer stärkeren Zusammengehörigkeit entsprechen. Die Berechnungen wurden für einen Beispielkorpus mit 10.000.000 Sätzen vorgenommen.

Um die Abhängigkeit der Signifikanz von der Korpusgröße zu verdeutlichen, betrachten wir mehrere Korpora bestehend aus Zeitungstext mit einem Größenunterschied jeweils von einem Faktor 10. Dazu soll jeweils das Auftreten der Wörter *Polizei* und *Autofahrer* einzeln sowie gemeinsam im Satz gezählt werden. Die Signifikanz steigt genau wie die Anzahlen näherungsweise linear mit der Korpusgröße. Dies entspricht der Tatsache, dass wir zwei Wörter immer mehr als zusammengehörig empfinden, je größer die Anzahl der gemeinsamen Vorkommen auch bei konstanter Ereignisdichte ist, siehe Tab. 5.7.

**Tab. 5.7** Signifikanz in Abhängigkeit von der Korpusgröße

Anzahl Sätze	n <sub>A</sub>	n <sub>B</sub>	n <sub>AB</sub>	sig(A,B)
1K	3	0	0	--
10K	53	2	0	--
100K	631	81	6	17,87
1M	7.235	1.185	89	243,04
10M	77.530	11.072	801	2105,12
100M	838.758	118.372	8846	23.805,50

### 5.3.3 Plausibilität der Ergebnisse

Zwischen der Signifikanzstärke von Kookkurrenzen und der Stärke von Assoziationen bei Menschen, wie sie in **Stimulus–Response-Experimenten** ermittelt wird, besteht eine erstaunliche Übereinstimmung, wie bereits von Rapp (1996) nachgewiesen wurde.

In solchen Experimenten wird den Versuchspersonen (VPn) jeweils ein Stimuluswort genannt. Darauf sollen sie schnell und spontan mit dem ersten Wort, welches ihnen dazu einfällt, antworten.

Die Tab. 5.8 zeigt für den Stimulus *Butter* die am Beispiel häufigsten genannten Reaktionen von Versuchspersonen (Spalten 2 und 3) sowie die stärksten im Text ermittelten Kookkurrenzen mit ihren Signifikanzen (Spalten 4 und 5) nach Schmidt (1999). Die Übereinstimmung der Resultate wird deutlich, wenn man betrachtet, dass die Wörter *Brot*, *Käse*, *Milch*, *Margarine* in beiden Versuchen unter den ersten zehn Ergebnissen waren und beide Ergebnislisten von *Brot* angeführt werden.

### 5.3.4 Andere Signifikanzmaße

Neben dem in Abschn. 5.3.2 eingeführten Log-Likelihood-Maß gibt es noch andere Möglichkeiten, aus den vier dort genannten Anzahlen  $n_{AB}$  (Anzahl der Sätze, die A und B gemeinsam enthalten),  $n_A$ ,  $n_B$  (Anzahl der Sätze, die A bzw. B enthalten) sowie  $n$  (Gesamtzahl aller Sätze) andere Signifikanzmaße zu wählen.

**Tab. 5.8** Stimulus *Butter*

Stimulus	Antworten von Versuchspersonen mit Anzahl		Stärkste Satzkookkurrenzen mit Signifikanz	
<b>Butter</b>	Brot	60	Brot	51
	weich	40	Käse	49
	Milch	32	Zucker	29
	Margarine	27	Milch	23
	Käse	20	Margarine	22
	Fett(e)	16	Mehl	18
	gelb	14	Eier	16
	Butterbrot	8	Pfund	14
	Dose	6	zerlassener	13
	essen	6	Fleisch	13

**Anzahl der gemeinsamen Auftreten  $n_{AB}$  als Signifikanzwert:** Ausschließlich die Anzahl  $n_{AB}$  zu betrachten, ist nur dann sinnvoll, wenn die betrachteten Wörter näherungsweise die gleiche Häufigkeit besitzen. In allen anderen Fällen verzerrt die unterschiedlichen Wörthäufigkeiten das Ergebnis zuungunsten seltenerer Wörter.

**Tanimoto-Ähnlichkeit:** Hier betrachtet man die Anzahl der Doppeltreffer  $n_{AB}$  im Vergleich zur Anzahl der Einzeltreffer. Sie ist definiert als:

$$\text{sim}_T(A, B) = n_{AB}/(n_A + n_B - n_{AB}) \quad (\text{Gl. 5.11})$$

und nimmt Werte zwischen null (keine gemeinsamen Auftreten) und eins (beide Wörter treten stets gemeinsam auf) an. Der Wert ist asymptotisch unabhängig von der Korpusgröße: Bei Vergrößerung eines Korpus steigen alle in der Formel verwendeten Anzahlen linear an, was die Tanimoto-Ähnlichkeit unverändert lässt.

**Pointwise Mutual Information (PMI):** Hier wird die Abweichung von der statistischen Unabhängigkeit der Wörter A und B gemessen. Bezeichnen  $p_A$  und  $p_B$  die Auftretenswahrscheinlichkeiten der Wörter A bzw. B in einem Satz und  $p_{AB}$  die Wahrscheinlichkeit des gemeinsamen Auftretens, so wäre im Falle statistischer Unabhängigkeit  $p_{AB} = p_A p_B$ . Die Abweichung von dieser Unabhängigkeit lässt sich ebenfalls als Maß für die Zusammengehörigkeit interpretieren. Pointwise Mutual Information ist definiert als:

$$\text{PMI}(A, B) = \log(p_{AB}/(p_A p_B)) = \log(n_{AB} n_{ges}/(n_A n_B)) \quad (\text{Gl. 5.12})$$

Für statistisch unabhängige Wörter ist die PMI gleich null. In der Variante der Positive Pointwise Mutual Information (PPMI) werden auch negative PMI-Werte auf null gesetzt. In dem uns interessierenden Fall, dass Wörter häufiger als erwartet gemeinsam auftreten, ist die PMI positiv und kann für typischerweise gemeinsam auftretende, aber trotzdem seltene Wörter (wie *Sri* und *Lanka*) beliebig groß werden. PMI ist wie die Tanimoto-Ähnlichkeit asymptotisch unabhängig von der Korpusgröße.

### 5.3.5 Signifikante Kookkurrenzen – Beispiele und Anwendungen

Signifikante Kookkurrenzen beschreiben in vielen Fällen semantische Zusammenhänge wie Ober- und Unterbegriffe, typische Eigenschaften oder typische Tätigkeiten. Ebenso finden sich häufig mehrere Wörter aus einem Sachgebiet oder einem sprachlichen Bereich.

Signifikante Kookkurrenzen beschreiben das statistisch auffällige gemeinsame Auftreten von Wörtern. Zunächst werden die stärksten signifikanten Kookkurrenzen für einige typische Wörter näher betrachtet.

Die Tab. 5.9 zeigt typische signifikante Satzkookkurrenzen für ausgewählte Wörter. In den ersten drei Zeilen finden sich unter den stärksten signifikanten Satzkookkurrenzen jeweils Wörter verschiedener Wortart, die möglicherweise auch zu verschiedenen Themen gehören.

**Tab. 5.9** Beispiele Satzkookkurrenzen

Ausgangswort	Stärkste Satzkookkurrenzen
Silber	Gold, Bronze, Kupfer, Platin, in, gewann, und, Schwarz, Farben, Rot, ausgezeichnet, Blau, holte, Palladium, Blei, Edelmetalle, ...
grünes	Licht, gegeben, gab, ein, für, Klassenzimmer, rotes, gibt, Paradies, vorzeitigen, Blattgemüse, blaues, hat, geben, Gemüse, ...
Blei	Cadmium, Quecksilber, Kupfer, Arsen, Nickel, Zink, Silber, Kadmium, Chrom, Schwermetalle, Schwermetallen, wie, enthalten, Gold, Stoffe, Zinn, ...

Im Falle von *Silber* sind dies zunächst verschiedene Wörter aus dem Bereich Sport (*Gold, Bronze, gewann, holte*), aber auch als Farbe (*Schwarz, Farben, Rot, Blau*) sowie aus dem Bereich Metalle (*Gold, Kupfer, Platin, Blei, Edelmetalle*).

Bei *grünes* sind die signifikanten Satzkookkurrenzen noch inhomogener, man erkennt beispielsweise die Redewendung *grünes Licht geben für*. Bei *Blei* hingegen ist die Menge der signifikanten Kookkurrenzen recht homogen, es finden sich viele weitere Metalle.

Die Tab. 5.10 zeigt typische Mengen von linken Nachbarn. Hier gehören die Kookkurrenzen viel stärker zu einer Wortart als bei Satzkookkurrenzen. Der Grund dafür ist, dass beispielsweise in **Nominalphrasen** unmittelbar vor dem Substantiv ein Adjektiv steht, welches wiederum eine typische Eigenschaft des Substantivs beschreibt, deshalb treten viele Farbadjektive als linke Nachbarn für *T-Shirt* auf.

Weil Personennamen aus Vor- und Nachnamen bestehen, treten als linke Nachbarn von Nachnamen typischerweise Vornamen auf. Diese Vermutung wird bei den linken Nachbarn des Nachnamens *Mayer* bestätigt. Im letzten Beispiel finden sich als linke Nachbarn des Partizips II *getrunken* hauptsächlich typische Objekte zum Verb *trinken*, dies sind die wichtigen Getränke. Außerdem finden sich Adverbien und Adverbialkonstruktionen wie *über den Durst*.

Die Tab. 5.11 zeigt typische rechte Nachbarn für ausgewählte Wörter. Nach den bisherigen Beispielen ist klar, dass rechts von *Insel* natürlich Inselnamen zu finden sind und rechts von *Kubikmeter* Dinge, die in Kubikmetern gemessen werden. Als rechte Nachbarn von *mutmaßliche* findet sich eine Ansammlung von Verbrechensbeteiligten, hauptsächlich Täter bzw. Täterinnen, aber auch Opfer und Tatwerkzeuge.

**Tab. 5.10** Typische Mengen linker Nachbarschaftskookkurrenzen

Ausgangswort	Stärkste linke Nachbarschaftskookkurrenzen
T-Shirt	ein, weißes, das, schwarzes, weißem, schllichtes, weißen, einem, Ein, Das, Damen, rotes, rosa, schlichten, eigenes, neues, ...
Mayer	Thomas, Stephan, Sally, Pill, Marissa, Frau, Hans, Eric, Achim, Steffen, Rupert, Wolfgang, Albrecht, Gina, Leopold, Markus, ...
getrunken	Alkohol, Kaffee, Bier, viel, Tee, Wasser, Wein, kalt, bedenkenlos, in Maßen, pur, Zielwasser, wenig, genug, Cola, über den Durst, ...

**Tab. 5.11** Typische Mengen rechter Nachbarschaftskookkurrenzen

Ausgangswort	Stärkste rechte Nachbarschaftskookkurrenzen
Insel	Rügen, Usedom, Mainau, Sylt, Hiddensee, Reichenau, Föhr, Lesbos, Poel, Borkum, Sulawesi, Neuwerk, Vilm, Rhodos, Capri, Fehmarn, ...
Kubikmeter	Wasser, Luft, Holz, pro, Abwasser, Beton, Erdgas, Müll, Trinkwasser, Gas, Erde, Erdreich, Erdmassen, Torf, großen, ...
mutmaßliche	Täter, Tatwaffe, Einwilligung, Unfallverursacher, Mörder, Mitglieder, Gruppenvergewaltigung, Drahtzieher, Opfer, Schütze, Einbrecher, Täterin, Brandstifter, Attentäter, ...

### 5.3.6 Erste Anwendungen für signifikante Kookkurrenzen

#### 5.3.6.1 Fremdsprachliche Daten im Text

Trotz aller Bemühungen in einem Korpus beispielsweise nur deutschsprachige Text zu sammeln, kommen auch englische Wörter durch Film- und Musiktitel, Zitate usw. hinein.

Wird nun aus dem Korpus beispielsweise eine nach Häufigkeit sortierte Wortliste erstellt, so sollen in der Regel die englischen Wörter entfernt werden. Beim sequenziellen Durchlesen der Wortliste sind die englischen Wörter sehr verstreut und viel Zeit wird mit dem Lesen korrekter deutscher Wörter verschwendet.

Auffällig ist, dass die gesuchten englischen Wörter im Text nicht zufällig verstreut sind, sondern in der Regel mehrere englische Wörter beieinander stehen. Diese statistische Auffälligkeit lässt eine Untersuchung der signifikanten Satzkookkurrenzen nützlich erscheinen. In der Tat bestehen solche Mengen gemeinsam auftretender englischer Wörter in deutschem Text fast ausschließlich aus anderen englischen Wörtern.

#### Kookkurrenzen in deutschem Korpus: Englisch

Beispielsweise umfasst die Menge der Kookkurrenzen für *the*:

*of, and, on, Year, to, for, The, is, world, with, by, way, from, in, out, We, at, over, a, are, Out, Over, around, ...* ◀

Damit können mit einem Schlag viele englische Wörter in einem deutschen Text identifiziert werden.

#### 5.3.6.2 Mundart

Nach kurzem Nachdenken fällt vielleicht auf, dass sich eine **Mundart** in einem großen Korpus ähnlich verhält wie die eben betrachteten englischen Textteile: Die entsprechenden Textstellen sind verstreut in vielen Teilen des Korpus, aber wieder geht es nicht um isolierte Wörter im Korpus. Deshalb kann die gleiche Methode benutzt werden, um Wörter einer Mundart zu identifizieren.

**Kookkurrenzen in deutschem Korpus: Berliner Mundart**

Für ein typisches Wort wie z. B. *ick* erhält man die folgende Menge der Kookkurrenzen:

nich, Ick, wat, hör, hab, dat, dir, det, och, weeß, ne, jut, janz, jesagt, einjestiegen, is, Dat, Wat, och, se, Na, Det, find, nu, Nee, ollen, en, mal, wa, ... ◀

**5.3.7 Signifikante Kookkurrenzen und Polysemie**

An den bisherigen Beispielen fällt Folgendes auf: Durch signifikante Satzkookkurrenzen werden mithilfe eines einzigen Startwortes oft sachbezogene Wortfelder bzw. große Teile einer Subsprache identifiziert.

Viele Wörter der natürlichen Sprache zeichnen sich dadurch aus, dass sie mehrere Bedeutungen haben. Was passiert, wenn solche Wörter als Startwörter benutzt werden?

Beispiele für diese Mehrdeutigkeit finden sich bei *Bank* (als Geldinstitut oder Sitzgelegenheit) oder *Band* (*das Band* z. B. als Geschenkband, *der Band* als Buch oder *die Band* als Musikgruppe). Aber auch Eigennamen sorgen für Mehrdeutigkeit.

Bei solch mehrdeutigen Wörtern besteht die Menge von Kookkurrenzen in der Regel aus Wörtern, die sich den verschiedenen Bedeutungen zuordnen lassen. Deshalb sollte die Menge von Kookkurrenzen entsprechend der verschiedenen Bedeutungen zerlegt werden (vgl. Abschn. 2.4.2 „Disambiguierung“). Dies ist beispielsweise dadurch möglich, dass man die gemeinsamen Kookkurrenzen für zwei Wörter betrachtet, nämlich die Kookkurrenzen zum Ausgangswort sowie die Kookkurrenzen zu einem zweiten Wort, welches die entsprechende Bedeutung gut charakterisiert. Interessant sind jeweils solche Wörter, die starke Kookkurrenzen sowohl zum Ausgangswort wie auch zum zweiten Wort sind.

**Die Polysemie von Stich**

Näher analysiert werden soll die Liste der signifikanten Kookkurrenzen des Wortes *Stich*: Tab. 5.12 enthält jeweils die gemeinsamen Kookkurrenzen zu Stich und einem weiteren Wort.

Dabei werden folgende unterschiedliche Verwendungen des Wortes Stich sichtbar:

- Die Redewendung: jemdn. im Stich lassen;
- Stich, kurz für Insektstich;
- Verletzung mit einer Stichwaffe;
- Verwendung als Nachname, z. B. Michael Stich.

**Tab. 5.12** Kookkurrenzen zu Stich und einem weiteren Wort

Wortpaar	gemeinsame Kookkurrenzen
Stich+gelassen	fühlen, im, fühlt, sich, fühlten, habe, hat, sie, ihn, haben, uns, er, fühlte, fühle, sieht...
Stich+Insekten	Bienen, Mücken, Wespen, übertragen, Biene, Wespe, Malaria, Spinnentiere, Zecken, töten, Larven, Honigbiene, gestochen, Parasiten, Gift, tödlich, Stachel, stechen, Hornissen, Stechmücken, ...
Stich+Messer	Hals, verletzt, Opfer, Mann, Bauch, Oberkörper, Rücken, stach, gestochen, Brust, lebensgefährlich, verletzte, Frau, Herz, tödlich, Klinge, zückte, getötet, schwer, Oberschenkel, Streit, Täter, zugefügt, ...
Stich+Michael	Becker, Trainer, Boris, Patrick, Turnierdirektor, Wimbledonsieger, Graf, Haas, Rothenbaum, ...

Es gibt noch andere Verwendungen des Wortes Stich, die in Korpora aber nur selten auftreten und deshalb kaum prominente Kookkurrenzen haben und sich schlechter korpusbasiert identifizieren lassen, z. B.:

- Stich, kurz für Kupferstich;
- Stich im Kartenspiel;
- die Redewendung *Stich ins Wespennest*;
- Abweichung in Farbe oder Geschmack. ◀

### 5.3.8 Semantische Relationen zwischen signifikanten Kookkurrenzen

Wie in den Beispielen deutlich wurde, können zwischen einem Wort und ihren signifikanten Kookkurrenzen verschiedene semantische Relationen bestehen. Im Folgenden werden die wichtigsten Relationen mit dazugehörigen Beispielen angegeben (vgl. Steele 1990 und Abschn. 2.4.2).

Bei gewöhnlichen Wörtern (d. h. nicht als Eigename oder Bestandteil von Wendungen) findet man bei signifikanten Nachbarschaftskookkurrenzen:

- Über- oder Unterordnung: *Schwermetalle* zu *Blei*; *Medaillen* zu *Silber*;
- typische Eigenschaften: *grünes* bei *Ampellicht*, *weißes* bei *T-Shirt*, *verschlossen* bei *Tür*;
- Handlungstragende und Handlungen: *tötet* und *erschießt* zu *Amokläufer*;
- Handlungen und dazugehörige Objekte: *Bier* und *Kaffee* zu *trinken*;
- Maßangaben zu Stoffen: *Tonnen* zu *Getreide*, *Kubikmeter* zu *Trinkwasser*.

Als signifikante Satzkookkurrenzen findet man zusätzlich:

- Unterbegriffe zum gleichen Oberbegriff: *Zink, Kupfer* usw. bei *Blei*; *rotes, gelbes* usw. bei *grünes*;
- Teil-Ganzes-Beziehung: *Seiten* und *Kapitel* bei *Buch*;
- Wirkung zur Ursache: *Feuer* bei *Kurzschluss*, *Stromschlag* bei *Verbrennungen*;
- Hilfsmittel und Werkzeuge (Instrument): *Mikroskop* und *Lupe* zu *Vergrößerung*;
- Gegenteil: *geöffnet* bei *verschlossen*, *heiß* bei *kalt*;
- Ortsangabe: *Schmiede* zu *Amboss*, *Rathaus* zu *Bürgermeister*;
- Personenangaben: *Hufschmied* zu *Amboss*, *Heimwerker* zu *Bohrmaschine*;
- Synonyme: *Schreibkraft* bei *Sekretärin*, *Fahrer* bei *Chauffeur*.

Lässt man zusätzlich Eigennamen zu, erhält man weitere Relationen:

- Titel und Berufe als linke Nachbarn von Personen: *Nobelpreisträger* zu *Krugman* und *Fermi*;
- Ortsnamen zu Personennamen: *Bayreuth* zu *Wagner*;
- Produktnamen als rechte Nachbarn zu Firmennamen: *Fiesta* zu *Ford*;
- Organisationsnamen zu Personennamen: *Apple* zu *Steve Jobs*.

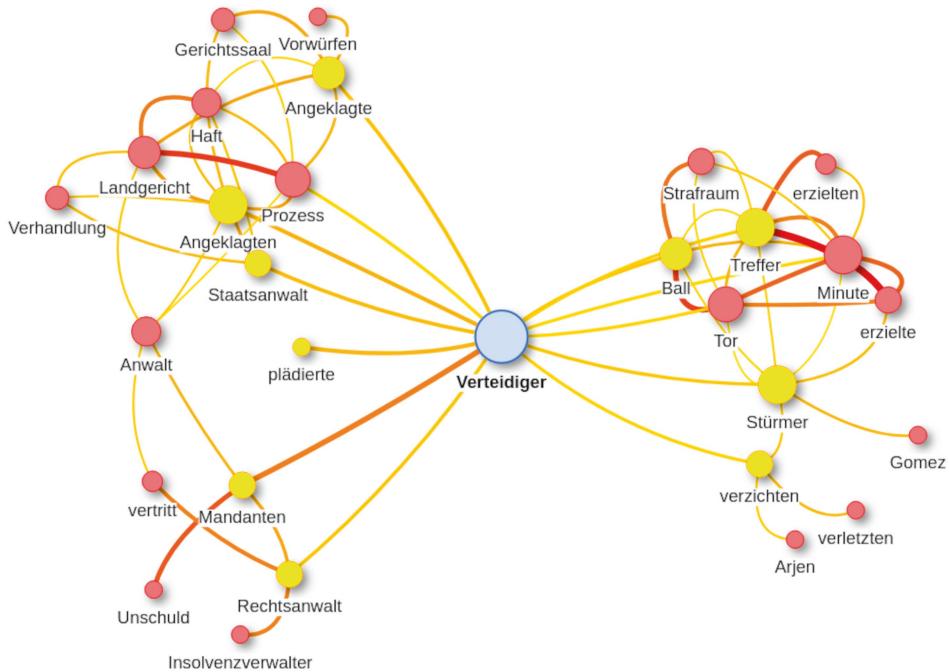
### 5.3.9 Visualisierung von signifikanten Kookkurrenzen

Die graphische Darstellung mehrerer Kookkurrenzen zu einem Wort vermittelt einen umfassenderen Überblick über die im Text gefundenen Zusammenhänge. Besonders im Falle von Mehrdeutigkeiten zerfällt der Graph der signifikanten Kookkurrenzen in verschiedene Teile, die diese Mehrdeutigkeit deutlich sichtbar machen.

Das Beispiel in der Abb. 5.4 zeigt eine Visualisierung der Kookkurrenzen von *Verteidiger* (erzeugt mit dem LeipzigCorpusMiner, Beschreibung im Anhang A8).

Wenn man zwei zu einem Wort A gehörige signifikante Satzkookkurrenzen B und C betrachtet, so sind die letzteren möglicherweise wieder untereinander signifikante Satzkookkurrenzen. Man hat dann drei (oder auch mehr) Wörter, von denen jedes Paar aus signifikanten Satzkookkurrenzen besteht. Denkt man sich die Wörter als Knoten und signifikante Satzkookkurrenzen jeweils durch eine Kante verbunden, so entsteht ein Wortnetz, das sogenannte Netz signifikanter Kookkurrenzen.

Insgesamt ergibt sich so ein sehr großes Netz mit vielen, auf den ersten Blick recht unübersichtlichen Kanten. Deshalb betrachtet man zunächst nur Ausschnitte, die jeweils einen zentralen Begriff enthalten. Neben diesem Ausgangsbegriff werden all die Begriffe aus der dazugehörigen Menge der signifikanten Kookkurrenzen mit aufgenommen, zwischen denen untereinander mindestens noch eine Verbindung besteht. In diesem Graphen sind zwei Begriffe A und B durch eine Kante verbunden, falls die Signifikanz  $\text{sig}(A,B)$  einen Schwellwert übersteigt. Die Stärke der Verbindungsstrecke entspricht dabei



**Abb. 5.4** Visualisierung der Kookkurrenten von *Verteidiger*

der Größe dieses Signifikanzwertes. Zu schwache Verbindungen werden der Übersichtlichkeit wegen nicht eingezeichnet. Ist die Menge der signifikanten Kookkurrenten des Ausgangswortes sehr groß, werden ggf. nur die stärksten signifikanten Kookkurrenten berücksichtigt. Soll die Menge der so ausgewählten Kookkurrenten dagegen weiter vergrößert werden, können auch die stärksten Kookkurrenten im Abstand zwei vom Ausgangswort hinzugenommen werden. Bei diesen Wörtern im Abstand zwei zeigen sich zusätzlich Synonyme zum Ausgangswort (hier z. B. *Anwalt* und *Rechtsanwalt*), da diese viele Kookkurrenten mit dem Ausgangswort gemeinsam haben.

Nachdem so das Netz beschrieben und ausgewählt wurde, muss es noch graphisch dargestellt werden. Als Visualisierungsverfahren wird hier das sogenannte kraftbasierte Verfahren verwendet.

Dabei erfolgt die Anordnung der Punkte nach dem folgenden Schema:

Zunächst werden die Punkte zufällig in der Bildebene verteilt. Zwischen je zwei Punkten herrscht eine konstante Abstoßungskraft wie zwischen elektrisch gleich geladenen Teilchen. Zusätzlich herrscht zwischen zwei Punkten eine Anziehungskraft, wie wenn diese Teilchen zusätzlich durch Federn verbunden wären. Die dazugehörige Federkonstante ist proportional zur Signifikanz  $\text{sig}(A,B)$ . Ermöglicht man nun vorsichtig Bewegungen der Punkte entsprechend dieser Kräfte, so ergibt sich nach einiger Zeit eine

stabile Anordnung. Diese entspricht einem lokalen energetischen Minimum und zeigt die inhaltlichen Zusammenhänge zwischen den Wörtern relativ gut. Statt der geradlinigen Verbindung durch Federn wird in den Bildern eine optisch ansprechendere Verbindung durch Bézierkurven gewählt.

---

## 5.4 Distributionelle Semantik

Die auf der Theorie des Strukturalismus aufbauende distributionelle Semantik repräsentiert die Bedeutung von Wörtern auf Basis ihrer Verwendung in Korpora. Nach einer Einführung in die zugrundeliegende Theorie werden in diesem Abschnitt zunächst sogenannte Sparse-Modelle besprochen, welche die Funktionsweise der distributionellen Semantik illustrieren; danach wird die Idee der Dimensionalitätsreduktion für sogenannte Dense-Modelle skizziert. Dies bildet die Basis für die in Abschn. 5.6 besprochenen Word Embeddings.

Für Text Mining und andere Anwendungen der Sprachtechnologie muss die Bedeutung von Wörtern repräsentiert werden. Was passiert, wenn wir Menschen Sprache ‚verstehen‘ in dem Sinne, dass wir ihren Elementen Bedeutung zuweisen? Nach De Saussure (1966), siehe Abschn. 2.1, erfolgt die Analyse auf zwei Ebenen: der syntagmatischen Ebene, welche die Beziehung der Wörter zueinander regelt, und der paradigmatischen Ebene, welche die Bedeutung der Wörter dadurch charakterisiert, was diese mit Wörtern mit ähnlicher Bedeutung gemeinsam haben und wie sie von diesen abgegrenzt werden können. Dies wurde in der **Distributionellen Hypothese** (vgl. Harris 1951) präzisiert, welche diesem Vorgehen zugrunde liegt: Definiert man die Distribution (Verteilung) sprachlicher Elemente als die Summe aller Kontexte, in denen das jeweilige Element auftritt, dann können Elemente als distributionell äquivalent angesehen werden, wenn ihre Distribution gleich ist. Während Harris die Hypothese mit Blick auf Äquivalenzklassen von Phonemen formulierte, operationalisierten sie Miller und Charles (1991) für Wortbedeutung: Je ähnlicher die Kontexte sind, in denen zwei Wörter auftreten, desto ähnlicher ist deren Bedeutung. Dies gibt uns nun die Möglichkeit, die graduelle Bedeutungsähnlichkeit von Symbolen (z. B. Wörtern) zu berechnen, indem wir sie anhand der in einem Korpus beobachteten Kontexte vergleichen. Hierbei kann Kontext verschieden definiert werden, z. B. als gemeinsames Vorkommen in einem Dokument, in einem Satz oder als direkte Nachbarn; die Menge der signifikanten Kookkurrenzen (Abschn. 5.3) kann auch als Kontextdefinition dienen.

Distributionelle Semantische Modelle sind konzeptuell verwandt mit dem **Vektorraummodell** im Information Retrieval (vgl. Salton et al. 1975): Hier werden Dokumente anhand der Wörter charakterisiert, welche in ihnen enthalten sind. Diese Dokumentvektoren werden in Suchanwendungen zum Ranking bzgl. einer Suchanfrage genutzt. Ebenso können Wörter anhand der Dokumentkontakte charakterisiert werden, in denen sie auftreten. In der distributionellen Semantik sind jedoch kleinere Kontexte üblich, wie z. B. das gemeinsame Auftreten von Wörtern im Satz, welches durch die Satzkookkurrenz-Matrix

Dokument\Term	der	Hund	Katze	Auto	...	klein
Doc1.txt	3	0	0	2	...	0
Doc2.txt	5	2	1	0	...	1
Doc3.txt	7	1	0	0	...	1
...						
Doc21983.txt	11	0	0	0	...	0
Doc21984.txt	2	0	0	4	...	3

Wortvektor  
„Hund“

Dokumentvektor  
„Doc3.txt“

**Abb. 5.5** Term-Dokument-Matrix mit Dokument- und Wortvektoren zur Bedeutungsrepräsentation

Satzkook.	Kontext						
	Leine	gehen	läuft	Besitzer	...	Tier	bellt
Hund	3	5	2	5	...	3	2
Katze	0	3	3	2	...	3	0
Löwe	0	3	2	0	...	1	0
leicht	0	0	0	0	...	0	0
...							
bellt	1	0	0	2	...	1	0
Auto	0	0	1	3	...	0	0

Wortvektor  
„Katze“

**Abb. 5.6** Satzkookurrenzmatrix mit Wortvektor zur Bedeutungsrepräsentation

modelliert wird. Abb. 5.5 illustriert die sich daraus ergebenden Vektorrepräsentationen von Wörtern anhand einer sog. **Term-Dokument-Matrix** und Abb. 5.6 die sich daraus ergebenden Vektorrepräsentationen anhand einer Satzkookurrenzmatrix.

Als direkte Konsequenz aus der Frequenzverteilung von Wörtern (Abschn. 5.2) sind diese Matrizen dünn besetzt („sparse“): die meisten Einträge sind Nullen. Die Größe des Vokabulars erlaubt es auch nicht, diese Matrizen explizit zu repräsentieren. Im Information Retrieval wird die Term-Dokument-Matrix durch eine invertierte Liste gespeichert, Kookurenzen werden als Liste von Wortpaaren mit Frequenz bzw. Signifikanz gespeichert. Trotzdem erfolgt formal die Repräsentation von Wörtern mit reellwertigen Vektoren – in der distributionellen Semantik wird der Wortvektor als die Repräsentation der Bedeutung des Wortes aufgefasst – und dies ermöglicht nun die

Quantifizierung der Bedeutungsähnlichkeit von Wörtern mithilfe der Berechnung von Vektorähnlichkeit. Hier kommt normalerweise der Kosinus des Winkels (Abschn. 5.6.2) zwischen den Vektoren zum Einsatz; dieser ist maximal 1 bei Vektoren gleicher Richtung und 0 bei orthogonal stehenden Vektoren.

Für den praktischen Einsatz ist es von Vorteil, wenn die Länge der Repräsentation deutlich kleiner ist als der Umfang des Vokabulars. Dies kann durch Dimensionsreduktionstechniken erreicht werden, welche die hochdimensionalen, dünn besetzten Vektoren in niedrigerdimensionale, dicht besetzte „dense“ Vektoren transformiert. Alternativ kann die Liste der Kontexte pro Wort auf die  $n$  wichtigsten/signifikantesten beschränkt werden, z. B. auf die 200 signifikantesten Satzkoorkurrenten, was interpretierbare „sparse“ distributionelle Modelle ermöglicht.

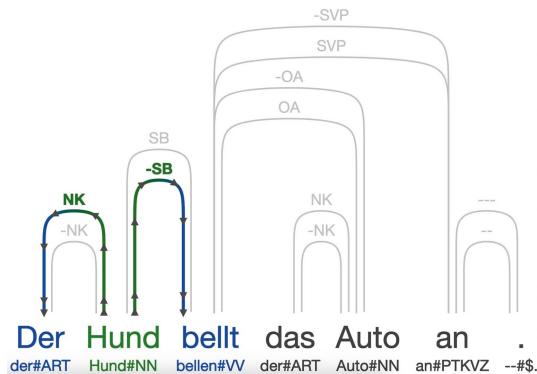
Im Folgenden wird der „sparse“ JoBimText-Ansatz (vgl. Biemann und Riedl 2013) beschrieben, welcher semantische Ähnlichkeit anhand der signifikantesten Kontexte berechnet; danach folgt eine Darstellung einer frühen Dimensionalitätsreduktionstechnik namens Latent Semantic Analysis (LSA, Deerwester et al. 1990), modernere Ansätze zu „Dense“-Wortvektoren werden genauer in Abschn. 5.6. beschrieben.

Der JoBimText-Ansatz benötigt zunächst eine Festlegung von lexikalischer Einheit (meist: ein Wort) und eine Kontextdefinition, z. B. benachbarte lexikalische Einheiten oder in Dependenzrelation (Abschn. 3.4.) stehende andere lexikalische Einheiten. Für jede lexikalische Einheit werden die signifikantesten  $n$  (z. B. 1000) Kontexte berechnet, auf dieselbe Weise wie in Abschn. 5.3 beschrieben. Danach werden für alle Paare von lexikalischen Einheiten gemeinsame Kontexte gezählt, was deren Ähnlichkeitswert ergibt. Die Berechnung ist für große Korpora aufwendig, jedoch parallelisierbar, vgl. Abschn. 3.3.1, hierbei werden zu häufige und zu seltene Kontexte ignoriert. Das Ergebnis kann als gewichteter Graph interpretiert werden, in dem Knoten die lexikalischen Einheiten sind und die Kantengewichte die Anzahl gemeinsamer signifikanter Kontexte repräsentieren. Im Folgenden wird die Kontextdefinition und die Ähnlichkeiten für ein dependenzgeparstes deutsches Nachrichtenkorpus illustriert, vgl. auch Padó und Lapata (2007).

In der Abb. 5.7 sind lexikalische Einheiten zu sehen. Diese sind hier Lemmata mit POS, Kontexte sind durch gerichtete Dependenzrelationen verbundene Dependensen, z. B. ist hier „bellen#VV#-SB“ (lies: steht in Subjektposition des Verbes „bellen“) ein Kontext von „Hund#NN“. In der Tab. 5.13 sind die zehn signifikantesten Kontexte für „Hund#NN“ zu sehen. Z.B. wurde „Hund#NN“ 4519-mal in einer Konjunktion (CJ) mit „Katze#NN“ beobachtet (in beiden Richtungen), 709-mal mit „angeleint#ADJA#NK“. In der Tab. 5.14 stehen die zehn ähnlichsten lexikalischen Einheiten für „Hund#NN“. Z. B. hat „Katze#NN“ 151 Kontexte gemeinsam, davon die vier signifikantesten wie hier angegeben. Katzen sind also auch streunend und freilaufend, bellen aber nicht, werden nicht angeleint oder geprügelt und auch nicht Gassi geführt.

Der Vorteil von „sparse“-Modellen wie JoBimText ist deren Interpretierbarkeit: Die Gründe für die Ähnlichkeit zweier lexikalischer Einheiten sind direkt ablesbar, die Repräsentation der signifikantesten Kontexte ist transparent und menschenlesbar,

**Abb. 5.7** Symbolisches distributionelles JoBimText Modell (Visualisierung mit JoBimText (Language Technology Group 2017), Daten aus Ruppert et al. (2015)).



**Tab. 5.13** Kontexte für „Hund#NN“

Kontext zu „Hund“	Signifikanz	Frequenz
Katze#NN#CJ	45.543,99	3541
streunend#ADJA#NK	19.982,37	1403
Katze#NN#-CJ	12.583,57	978
freilaufend#ADJA#NK	12.214,60	895
bellen#VV#-OA	11.398,29	832
angeleint#ADJA#NK	10.233,01	709
bellen#VV#-SB	9.645,39	755
geprügelt#ADJA#NK	9.501,14	685
bellend#ADJA#NK	8.770,86	621
Gassi#NE#NK	7.835,48	555

**Tab. 5.14** Ähnliche lexikalische Einheiten für „Hund#NN“

Ähnlich zu „Hund“	Anz. Kontexte
Hund#NN	1000
Tier#NN	154
Katze#NN	151
Vierbeiner#NN	127
Kaninchen#NN	116
Kampfhund#NN	111
Schwein#NN	108
Haustier#NN	97
Hühner#NN	97
Schäferhund#NN	93

die Berechnung ist deterministisch und liefert bei ähnlichen Korpora ähnliche Repräsentationen. Ein Nachteil besteht in der mangelnden Fähigkeit zur Abstraktion: Ähnlichkeiten zwischen Kontexten werden nicht ausgenutzt, was dazu führt, dass „sparse“-Modelle im Allgemeinen für ein Modell gleicher Qualität größere Korpora benötigen. Ein weiterer Nachteil ist die Schwierigkeit, die Berechnung dieser Modelle zu skalieren, hierzu werden entsprechende Parallelisierungsframeworks (Abschn. 3.3.1) genutzt.

Ein früher Ansatz zur Dimensionalitätsreduktion ist Latent Semantic Analysis (LSA, Deerwester et al. 1990). Diese Technik wurde für Term-Dokument-Matrizen entwickelt, kann jedoch auch für Wort-Kookkurrenz-Matrizen angewendet werden (vgl. Abb. 5.5, und 5.6). Bei dieser Methode, die hier nur intuitiv skizziert werden kann, wird eine sogenannte Singulärwertzerlegung der Matrix durchgeführt: Gesucht wird eine Darstellung der Matrix mit alternativen, zueinander orthogonalen, latenten Dimensionen, was für jede Matrix möglich ist. Werden diese Dimensionen nach ihrer Wichtigkeit für die Darstellung der Originalmatrix geordnet, dann liefern die  $n$  wichtigsten Dimensionen die mit dieser Dimensionalität  $n$  bestmögliche Approximation der Originalmatrix. Mit dieser Methode kann z. B. in der Wort-Kookkurrenz-Matrix die Anzahl von Dimensionen vom Umfang des Vokabulars auf eine festgelegte Zahl  $n$  (üblich sind Werte zwischen 100 und 1000) reduziert werden, jedes Wort wird nun mit einem reellwertigen Vektor der Länge  $n$  repräsentiert. Die  $n$  Dimensionen entsprechen den orthogonalen, latenten Dimensionen und sind nicht mehr interpretierbar; die Darstellung beinhaltet keine systematischen Nullwerte, die Wortvektoren sind „dense“. Neben der kürzeren, daher handhabbareren Repräsentation ist ein weiterer Vorteil dieser Darstellung die geringere Redundanz: Ähnliche Wörter werden auf ähnliche Dimensionen abgebildet, was den Vergleich von Wörtern erlaubt, welche kaum exakte, jedoch viele ähnliche Kontexte gemeinsam haben. Ein Nachteil exakter Berechnungsmethoden wie der Singulärwertzerlegung oder der verwandten Hauptkomponentenanalyse ist neben der fehlenden Interpretierbarkeit des Ergebnisses deren Berechnungskomplexität; letztere wird durch approximative Verfahren verbessert, siehe Levy und Goldberg (2014) und Abschn. 5.6, welche auf Sprachmodellen beruhen, die Gegenstand des nächsten Abschnittes sind.

---

## 5.5 Sprachmodelle

Sprachmodelle werden dazu verwendet, die Wahrscheinlichkeit von Wörtern im Kontext vorherzusagen und längeren Abschnitten wie Sätzen eine Wahrscheinlichkeit zuzuweisen. Diese Modelle werden auf großen Korpora trainiert. Neben den probabilistischen N-Gramm-Modellen haben sich in den letzten Jahren neuronale Sprachmodelle etabliert. Anwendungen von Sprachmodellen im Text Mining sind vielfältig und reichen von Information Retrieval über die Textextraktion auf Webseiten (Abschn. 4.2) bis hin zum Vortrainieren von Embedding-Repräsentationen (Abschn. 5.6) und zum Generieren von Text.

Nicht alles, was sprachlich möglich ist, ist gleich wahrscheinlich. In einem Satz, welcher mit „*Die Kaninchen sprangen über die Wiese und*“ beginnt, kann durchaus noch das Wort *Sozialraumanalyse* vorkommen, dies ist jedoch ohne Kenntnis des Äußerungskontextes als sehr unwahrscheinlich anzusehen – nicht nur, weil *Sozialraumanalyse* ein relativ seltes Wort ist, sondern auch, weil es thematisch nicht zu einer Naturbeschreibung passt. Im Gegensatz dazu wäre der vergleichsweise ähnlich seltene *Mümmelmann* – beide haben Häufigkeitsklasse 19 (Abschn. 4.6.2) im Deutschen Wortschatz (Anhang A5) – hier weniger überraschend.

Sprachmodelle können diese Wahrscheinlichkeit quantifizieren. Ein Sprachmodell wird auf einem Korpus (Abschn. 4.1) trainiert und approximiert im Modell die Sprache dieses Korpus. Bei Anwendung des Sprachmodells wird das Modell mit vorliegendem Textmaterial abgeglichen: Es kann zum einen messen, wie gut ein Wort in einen Kontext passt oder wie erwartbar ein Satz ist. Zum anderen kann es erwartbare Texte Wort für Wort generieren. Diese Erwartbarkeit wird immer als Wahrscheinlichkeit bzgl. der Sprache des Trainingskorpus modelliert – insbesondere gibt es keine Sprachmodelle für gesamte Sprachen wie Deutsch (oder Englisch), sondern lediglich Sprachmodelle bzgl. bestimmter Korpora für Deutsch (oder Englisch). Da Modelle auf größeren Korpora durch eine bessere statistische Approximation im Allgemeinen bessere Ergebnisse erzielen, stellen qualitativ hochwertige (Web-)Korpora eine wichtige Voraussetzung für die Erstellung von Sprachmodellen dar. (Abschn. 4.2 Webkorpora; Abschn. 5.8 Qualitätsmaße).

Formal kann die Aufgabe des Sprachmodells angegeben werden als die Modellierung der Wahrscheinlichkeit des nächsten Wortes zum Zeitpunkt  $t$ , gegeben eine (ggf. leere, ggf. sehr lange) Liste an vorangegangenen Wörtern zu den Zeitpunkten 1 bis  $t-1$ :

$$P(w_t | w_1 w_2 .. w_{t-1}) \quad (\text{Gl. 5.13})$$

Dies lässt sich auch als Wahrscheinlichkeitsverteilung für das Generieren verwenden, ferner auch durch wiederholte Anwendung (Kettenregel für bedingte Wahrscheinlichkeiten) für die Berechnung der Wahrscheinlichkeit eines Satzes  $S = w_1 w_2 .. w_T$  der Länge  $T$  bzgl. des Modells:

$$P(S) = P(w_1 w_2 .. w_T) = P(w_1) \cdot P(w_2 | w_1) \cdot P(w_3 | w_1 w_2) \cdot \dots \cdot P(w_T | w_1 w_2 .. w_{T-1}) \quad (\text{Gl. 5.14})$$

Ein einfaches, jedoch in Anwendungen sehr effektives Sprachmodell ist das **N-Gramm-Modell**: Hier werden in starker Vereinfachung der Sprache lediglich  $N$  konsekutive Wörter (**N-Gramm**) für die Modellierung betrachtet. Das  $N$  ist hierbei die Ordnung des N-Gramm-Modells. Im Folgenden werden wir diese Art von Sprachmodell mit einem Bigramm-Modell veranschaulichen. Das Bigramm-Modell ( $N=2$ ) betrachtet nur zwei aufeinanderfolgende Tokens, d. h. die Vorhersage des nächsten Tokens erfolgt nur auf Basis des vorangegangenen Tokens:

$$P_{\text{BIGRAMM}}(w_t | w_1 w_2 .. w_{t-1}) \approx P(w_t | w_{t-1}) \quad (\text{Gl. 5.15})$$

Bei einem Trigramm-Modell ( $N=3$ ) würde die Vorhersage auf Basis zweier vorangegangener Tokens erfolgen, also  $P_{\text{TRIGRAMM}}(w_t | w_{t-2} w_{t-1})$ .

Die Wahrscheinlichkeit wird in erster Näherung durch die **relative Häufigkeit** geschätzt. Bezeichnen wir mit  $C(\cdot)$  die Anzahl der Vorkommen eines N-Gramms im Trainingskorpus, dann kann die Bigramm-Wahrscheinlichkeit angesetzt werden als:

$$P(w_t | w_{t-1}) = \frac{C(w_{t-1} w_t)}{\sum_{v \in V} C(w_{t-1} v)} = \frac{C(w_{t-1} w_t)}{C(w_{t-1})} \quad (\text{Gl. 5.16})$$

Die Häufigkeit des Bigramms  $w_{t-1} w_t$  wird hier durch die Summe der Häufigkeiten aller mit  $w_{t-1}$  beginnenden Bigramme geteilt, letzteres entspricht der Frequenz, oder Unigrammhäufigkeit, von  $w_{t-1}$ . Die Trigramm-Wahrscheinlichkeit wird analog aus der Frequenz des Trigramms geteilt durch die Frequenz des Bigramms berechnet.

Das eben beschriebene Vorgehen wird Maximum-Likelihood-Schätzung genannt, da hier die Wahrscheinlichkeit des Trainingskorpus, aus dem die Frequenzen herangezogen werden, maximiert wird. Ein schwerwiegendes Problem dieses Vorgehens, welches die insgesamt zur Verfügung stehende Wahrscheinlichkeitsmasse komplett für die im Trainingskorpus beobachteten Ereignisse verwendet, ist der Umgang mit nicht beobachteten Ereignissen. Das Problem des ungesehenen Vokabulars (**out of vocabulary, OOV**), welches direkt aus dem mit Korpusgröße wachsenden Vokabularumfang folgt (vgl. Abschn. 5.2), kann noch dadurch behoben werden, dass seltene Wörter unter einer Mindestfrequenz für den Zweck der Modellierung auf ein spezielles Token „unbekanntes Wort“ (üblicherweise wird <UNK> verwendet) abgebildet werden, was sowohl bei N-Gramm-Modellen als auch bei neuronalen Modellen gern eingesetzt wird, um die Vokabulargröße zu beschränken. Jedoch löst dies nicht das sogenannte „Nullenproblem“, welches im Beispiel unten (Bigramm-Modell mit Maximum-Likelihood-Schätzung) illustriert wird: Viele der mit dem Vokabular möglichen N-Gramme treten im Trainingskorpus nicht auf, können aber bei Anwendung des Modells auf neuen Text auftreten. Bei Maximum-Likelihood-Schätzung wird in diesen Fällen die Wahrscheinlichkeit mit Null geschätzt, woraus auch eine Satzwahrscheinlichkeit von Null folgt. Dies ist weder konzeptuell mit der Produktivität von natürlicher Sprache zu vereinbaren, welche sich ja gerade dadurch auszeichnet, auch neue Sachverhalte beschreiben zu können, noch ist dies praxistauglich: Insbesondere können nullwahrscheinliche Sätze nicht sinnvoll anhand ihrer Wahrscheinlichkeit verglichen werden, z. B. um eine bessere Übersetzungsalternative auszuwählen. Dieses Problem tritt verschärft für große Werte von  $N$  auf, was neben der Seltenheit langer N-Gramme auch ein Grund dafür ist, dass größere  $N$  nicht notwendigerweise zu besseren N-Gramm-Sprachmodellen führen.

### Bigramm-Modell mit Maximum-Likelihood-Schätzung

Gegeben sei ein Trainingskorpus aus 2 Sätzen für ein Bigrammmodell:

< BOS > Johann gibt Marie das Manuskript.

*<BOS> Peter sieht das Auto, und dass Johann Marie das Buch gibt.*

Aus technischen Gründen wird für ein Bigrammodell das *<BOS>*-Symbol (Begin-of-Sentence) am Satzanfang benötigt.

Die Wahrscheinlichkeit eines neuen Satzes „Johann gibt Marie das Buch.“ bezüglich des auf dem Trainingskorpus erstellten Bigrammodells berechnet sich als:

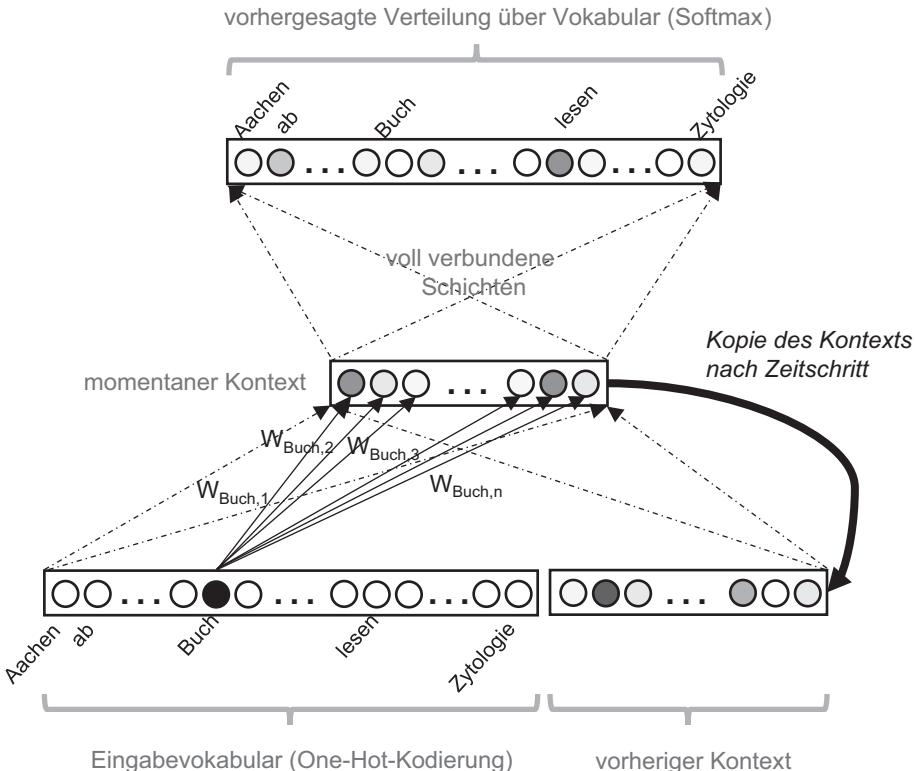
$$\begin{aligned}
 & P(\text{Johann gibt Marie das Buch.}) \\
 & = P(\text{Johann}|\text{<BOS>}) \cdot P(\text{gibt}|\text{Johann}) \cdot P(\text{Marie}|\text{gibt}) \cdot P(\text{das}|\text{Marie}) \cdot \\
 & P(\text{Buch}|\text{das}) \cdot P(\text{.}|\text{Buch}). \\
 & = C(\text{<BOS>}|\text{Johann})/C(\text{<BOS>}) \cdot C(\text{Johann gibt})/C(\text{Johann}) \cdot \dots \cdot C(\text{Buch.})/ \\
 & C(\text{Buch}). \\
 & = 1/2 \cdot 1/2 \cdot 1/2 \cdot 1/1 \cdot 1/3 \cdot 0/2. \\
 & = 0.
 \end{aligned}$$

Der Satz erhält die Wahrscheinlichkeit Null, da das Bigramm „Buch.“ nicht im Trainingskorpus enthalten ist, was die Unzulänglichkeit der Maximum-Likelihood-Schätzung illustriert und den Einsatz von Smoothing für N-Gramm-Modelle motiviert. ◀

Abhilfe schaffen Smoothing-Verfahren, welche die Wahrscheinlichkeit beobachteter Ereignisse verringern, um die freiwerdende Wahrscheinlichkeitsmasse ungesesehenen aber möglichen Ereignissen zuzuweisen. In der Literatur wird hierzu eine Reihe an Strategien aufgeführt; einen sehr guten Überblick bieten Manning und Schütze (1999, Kap. 6). Sprachmodelle werden anhand ihrer **Perplexität** von Testkorpora bzgl. des Modells verglichen. Intuitiv beschrieben wird hier ermittelt, wie ‚überrascht‘ das Modell vom Testkorpus ist. Mit Kneser–Ney-Smoothing (vgl. Kneser & Ney 1995) auf sehr großen Korpora (z. B. Web1T, Brants und Franz 2006) trainierte 5-Gramm-Modelle gehören hier zu den erfolgreichsten Ansätzen, benötigen jedoch sehr viel Speicherplatz. Für manche Anwendungsfälle, wie z. B. das Erkennen der Sprache (Abschn. 3.2.2), sind bereits Unigramm-Modelle ( $N=1$ ) ausreichend, welche auch als **Bag-of-Words-Modelle** bezeichnet werden, da hier nur Einzelwortwahrscheinlichkeiten unabhängig von der Wortreihenfolge betrachtet werden. Hier wird je ein Unigramm-Modell pro Sprache trainiert, das Modell mit der höchsten Wahrscheinlichkeit für einen zu bestimmenden Text weist die Sprache zu.

Defizite der N-Gramm-Modelle bei der Modellierung von Sprache ist zum einen deren beschränkter Horizont, welcher eben nur  $N-1$  vorangegangene Token umfasst. Zum anderen betrachten N-Gramm-Modelle verschiedene Wörter ungeachtet ihrer Ähnlichkeit als verschiedene Symbole – im Beispiel oben könnte die Nullwahrscheinlichkeit vermieden werden, wenn das Modell nur ‚wüsste‘, dass *Buch* und *Manuskript* sehr ähnliche Wörter sind. Diese beiden Defizite werden durch rekurrente neuronale Sprachmodelle adressiert, welche im Folgenden beschrieben werden.

Die Abb. 5.8 illustriert das vergleichsweise einfache rekurrente neuronale Sprachmodell nach Mikolov et al. (2010). In der Darstellung werden Aktivierungen von Neuronen mit Grauwerten dargestellt. Die hier dargestellten drei Schichten sind jeweils



**Abb. 5.8** Sprachmodell mit rekurrentem neuronalem Netz nach Mikolov et al. (2010)

voll verbunden, wie in Abschn. 3.1.3, Abb. 3.1. Die Kernidee dieses Modells besteht darin, dass der momentane Kontext, also die bisherigen Beobachtungen, welche beim Sprachmodell für die Vorhersage des nächsten Tokens verwendet werden, in einem reellwertigen Kontextvektor repräsentiert werden, welcher eine deutlich geringere Dimensionalität besitzt (z. B. 50 oder 300) als die Anzahl der Types im Vokabular. Diese Repräsentation, welche als innerer Zustand des Modells aufgefasst werden kann, speist sich einerseits aus der Beobachtung des letzten Wortes, andererseits aus dem vorangegangenen inneren Zustand, welcher durch Rekurrenz (zeitlich verzögerte Verbindungen, hier realisiert durch Kopieren der Repräsentation, vgl. Abschn. 3.1.3) verfügbar gemacht wird. Zur Vorhersage einer Wahrscheinlichkeitsverteilung über das Vokabular wird der momentane Kontext mit einer Ausgabeschicht verbunden, in der aufgrund des Kontextes erwartbarere Wörter höhere Aktivierungen erhalten sollen. Durch Normalisieren dieser Aktivierungen mit der sogenannten **Softmax-Funktion** wird eine Verteilung über das Vokabular erreicht, welche als Wahrscheinlichkeitsverteilung aufgefasst werden kann, wie für die Sprachmodellierung nötig. Bei der Verarbeitung des nächsten Wortes, im Beispiel „Buch“, wird dieses in der Eingabeschicht mit Wert 1 aktiviert, alle anderen Wörter erhalten den Wert 0 (One-Hot-Kodierung).

Zur Berechnung der Repräsentation des momentanen Kontexts (Mitte) wird neben dem Eingabevokularvektor auch die Repräsentation des vorherigen Kontexts herangezogen, welche zu diesem Zweck kopiert wurde.

Der momentane Kontext, in dem Information über das letztbeobachtete Wort „Buch“ als auch Information über alle vorangegangenen Beobachtungen durch die zeitlich verzögerte, rekurrente Verwendung der vorherigen Kontexte vorhanden ist, wird dann zur Berechnung der Ausgabeschicht verwendet, in der jedes Neuron einem Wort im Vokabular entspricht. Das hier im Beispiel am höchsten aktivierte Wort ist „lesen“.

Die Dimensionalität der Ausgabeschicht ist genauso groß wie die Dimensionalität des Eingabevokabulars, also typischerweise 100.000 und mehr. Die letzte Beobachtung wird mit einer One-Hot-Kodierung (auch: 1-aus-n-Code) in das Netzwerk eingebracht, die zu einem Wort gehörenden Gewichte (ein Vektor mit geringer Dimensionalität, z. B. 50 oder 300) bezeichnet man als dessen Word **Embeddings** (Wortebettung). Zum Beispiel in Abb. 5.8 entsprechen die Gewichte zwischen „Buch“ und dem momentanen Kontext, in der Abbildung mit  $w_{\text{Buch},i}$  bezeichnet, dem Embedding von „Buch“. Embeddings werden im folgenden Abschnitt (Abschn. 5.6) noch genauer beleuchtet.

Das Training des rekurrenten Sprachmodells erfolgt genauso wie bei Feed-Forward-Netzwerken über Backpropagation mit Stochastic Gradient Descent mithilfe von tokenisierten Korpora: Typischerweise wird das Korpus mehrfach durchlaufen. Die zunächst zufällig initialisierten Gewichte werden für eine Vorhersage herangezogen, welche mit der tatsächlichen nächsten Beobachtung verglichen wird. Die Abweichung zwischen Vorhersage und Beobachtung bestimmt dann die Veränderung der Gewichte. Dieses Trainingsschema bezeichnet man als prädiktives Training, im Gegensatz zu zählbasierten Verfahren wie dem N-Gramm-Modell (vgl. Baroni et al. 2014). Auf diese Weise werden insbesondere auch die Word Embeddings trainiert. Ähnliche Wörter wie *Buch* und *Manuskript*, welche ja auch in ähnlichen Kontexten auftreten (Abschn. 5.4) und ähnliche Vorhersagen machen sollen, erhalten ähnliche Embeddings, worüber eine höhere Generalisierung erreicht wird als bei N-Gramm-Modellen. Das hier vorgestellte einfache rekurrente Sprachmodell ist zudem deutlich kompakter als z. B. ein 5-Gramm-Modell. Die inzwischen verwendeten, deutlich komplexeren Modellarchitekturen zur Sprachmodellierung wie GPT (z. B. Brown et al. 2020) oder Transformer-Varianten (z. B. BERT, Devlin et al. 2019) übersteigen jedoch die Komplexität von N-Gramm-Modellen bei Weitem.

Neben den bereits im Vorangegangenen (Abschn. 3.2.2, Abschn. 4.2) und im Folgenden besprochenen Anwendungen von Sprachmodellen zum Trainieren von Embeddings werden diese auch für die Erkennung gesprochener Sprache eingesetzt, um z. B. bei unklarer Aussprache das wahrscheinlichste Wort nicht nur aufgrund des Sprachsignals, sondern auch bzgl. des Kontextes auszuwählen. Ein anderer Anwendungsfall ist die maschinelle Übersetzung, wo ein Sprachmodell dafür sorgen kann, dass von mehreren möglichen Übersetzungen die sprachlich flüssigste ausgewählt werden kann.

## 5.6 Dense Vector Embeddings

In diesem Abschnitt soll ein Verständnis von Embeddings vermittelt werden. Embeddings sind Repräsentationen von lexikalischen Einheiten wie Wörtern, Sätzen oder Texten, welche diese Einheiten in einen hochdimensionalen Vektorraum einbetten: Ein Embedding besteht aus einem reellwertigen Vektor, welcher als Punkt in diesem Raum interpretiert werden kann. Neben direkten Anwendungen von Embeddings für die Berechnung von distributioneller Ähnlichkeit (Abschn. 5.4) sind Embeddings vor allem wichtig bei der neuronalen Verarbeitung (Abschn. 3.1.3) von Text: Embeddings „übersetzen“ symbolische Einheiten wie Wörter in die für neuronale Verarbeitung nötigen kontinuierlichen Zahlenwerte. Gute Embeddings bieten nützliche Merkmale für die Klassifikation und können z. B. durch Sprachmodelle (Abschn. 5.5) unüberwacht auf großen Korpora vortrainiert werden: Das Ziel des Trainings wird meist dahingehend formuliert, dass ein Wort aufgrund seines Kontextes vorhergesagt werden kann, was auch der Modellierung in der strukturalistischen Semantik (Abschn. 2.4) entspricht.

Das **Bag-of-Words-Modell**, bei dem ein Text anhand der Anzahl der in ihm vorkommenden Wörter beschrieben wird, unabhängig von der exakten Position ihres Vorkommens im Text, ist einfach zu verstehen und gleichzeitig sehr effektiv (vgl. Abschn. 5.5). Für eine geometrische Interpretation, also eine Betrachtung von Wörtern angeordnet in einem Vektorraum, die das Auftreten einzelner Wörter als Merkmale benutzt, zieht diese Repräsentation aber Probleme nach sich, wie die Abb. 5.9 verdeutlicht.

In dieser exemplarischen binären **Wort-Kontext-Matrix** der Abb. 5.9 werden den zwei Komponisten (links) gewisse Assoziationen (Kontext; oben) zugeordnet. Trotz unseres Vorwissens, dass es sich um zwei Komponisten handelt, ist sofort ersichtlich, dass basierend auf der Matrix keinerlei Ähnlichkeit bestünde. Diese Tatsache ist eine Schwäche der dünnbesetzten Matrix, die zwar mit großen Vokabular-Dimensionen umgehen kann, bei der aber Vergleiche nur im Falle von genauer Übereinstimmung der Merkmale (Abschn. 6.1) funktionieren. In diesem Beispiel wurde das mit Wortpaaren demonstriert, die trotz lexikalischer Verschiedenheit eine hohe semantische Ähnlichkeit aufweisen; dasselbe Phänomen tritt aber ebenso bei Synonymie in den Kontexten oder in der Praxis auch bei leicht unterschiedlicher Tokenisierung oder Schreibweise auf. In den folgenden Abschnitten zeigen wir einen Ansatz, die sogenannten Embeddings, die

**Abb. 5.9** Exemplarische binäre Wort-Kontext-Matrix

	Italien	Deutschland	Violinist	Pianist
$W_1 = \text{Beethoven}$	0	1	0	1
$W_2 = \text{Paganini}$	1	0	1	0

Wörter (oder auch andere sprachliche Einheiten) in einen Vektorraum einbetten und damit eine Alternative zum oben vorgestellten Ansatz bieten. Hierbei unterscheiden wir zwischen statischen Word Embeddings, welche jedem Wort aus dem Vokabular einen Vektor zuordnen, kontextualisierten Word Embeddings, welche dies für jedes Wort im Text leisten, und sogenannten Document Embeddings, welche Vektorrepräsentationen fester Länge für gesamte Texte realisieren.

► **Definition** Ein statisches Word Embedding über einem Vokabular  $V$  definieren wir als:

$$e : V \rightarrow \mathbb{R}^n$$

Dabei wird jedem Wort-Type im Vokabular  $V$  ein reellwertiger Vektor mit Dimension  $n$  zugeordnet.

Ein kontextualisiertes Word Embedding über einem Text  $T = (w_1, w_2, \dots, w_k)$  aus  $k$  Tokens definieren wir als:

$$e : w_i \rightarrow \mathbb{R}^n$$

Dabei wird jedem Wort-Token im Text ein reellwertiger Vektor mit Dimension  $n$  zugeordnet.

Ein Document Embedding über ein Korpus  $K$  bestehend aus  $m$  Texten  $T_1, T_2, \dots, T_m$  definieren wir als:

$$e : T_i \rightarrow \mathbb{R}^n$$

Dabei wird jedem Text ein reellwertiger Vektor mit Dimension  $n$  zugeordnet. Dieser wird häufig unter Verwendung der Word Embeddings der enthaltenen Wörter berechnet.

### 5.6.1 Statische Word Embeddings

Word Embeddings ordnen Wörter aus dem Vokabular im  $n$ -dimensionalen Vektorraum an und stellen damit einen Zusammenhang zwischen Operationen im Vektorraum und semantischen/syntaktischen Eigenschaften der Wörter her. Dieses schon früh von Rumelhart et al. (1988) prinzipiell beschriebene Vorgehen wurde durch word2vec von Mikolov et al. (2013) populär, in welchem die besagte Anordnung im Vektorraum durch ein neuronales Netzwerk vorgenommen wird, welches auch auf sehr großen Korpora effizient trainiert werden kann. Solche Ansätze werden auch als selbstüberwachte Verfahren bezeichnet, da hier Training betrieben wird, dies jedoch nicht auf manuellen Annotationen geschieht.

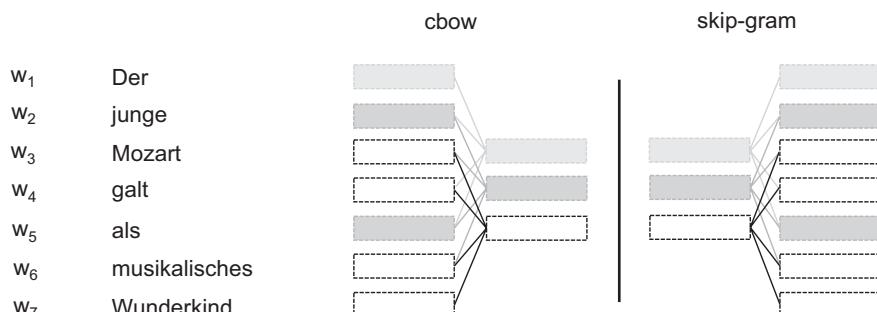
Neben word2vec gibt es noch einige weitere Ansätze, die sich durch die Art der Berechnung der Vektoren unterscheiden.

Im Unterschied zum Bag-of-Words Modell werden für Embeddings in der Regel dichtbesetzte Vektoren (vgl. Abschn. 5.4) verwendet. Dichtbesetzte Vektoren haben den Vorteil, dass man bei Multiplikation der Elemente, z. B. in den Produkten der Summe

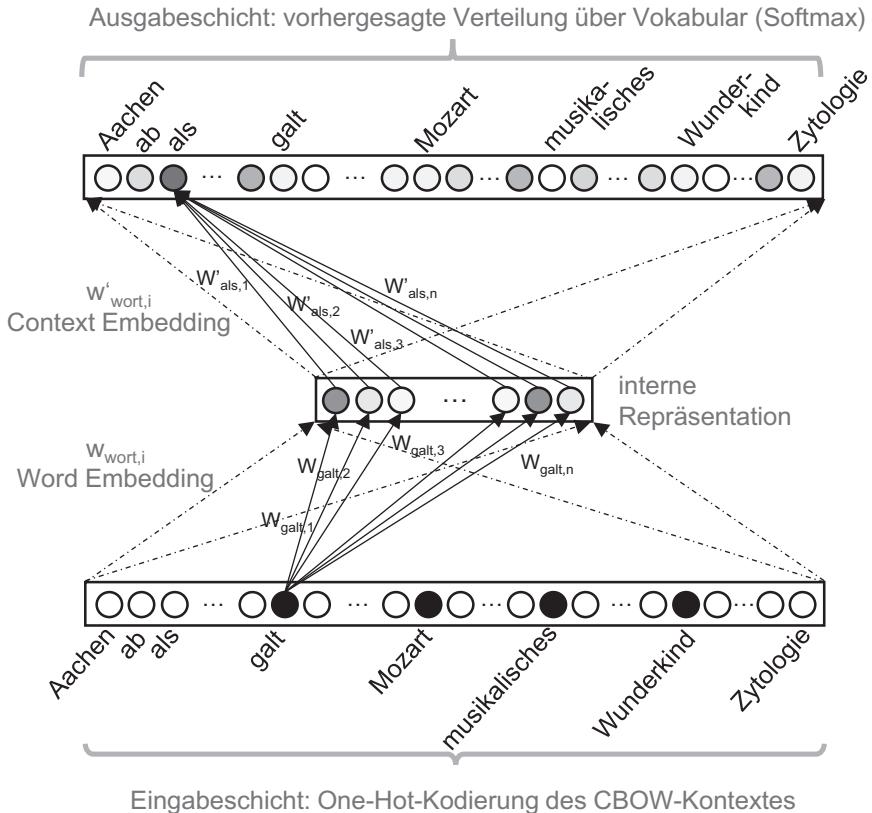
im Skalarprodukt, nicht mit den vielen Nullen umgehen muss, die bei dünnbesetzten Vektoren, z. B. durch fehlende Übereinstimmung trotz eines potentiell ähnlichen Vokabulars, zu dem Problem wie eingangs skizziert führen.

Eine Möglichkeit zur Erzeugung dieser Embeddings ist, diese durch ein neuronales Netzwerk lernen zu lassen. Mikolov et al. (2013) stellen hierfür zwei Architekturen vor: Continuous Bag-of-Words und Skip-Gram. Beim **Continuous Bag-of-Words (CBOW) Modell** wird versucht ein Wort vorherzusagen, wobei nur die Kontextwörter in einem Fenster (Sliding Window) links und rechts des Wortes betrachtet werden, typischerweise zwei links und zwei rechts. Da die Eingabewörter in einer Summe verrechnet werden, spielt die Reihenfolge der Wörter innerhalb des Fensters keine Rolle, weswegen der Ansatz analog zum Bag-of-Words Modell als CBOW bezeichnet wurde, jedoch mit dem Unterschied, dass statt Wörtern reelle Vektoren benutzt werden (vgl. Mikolov et al. 2013). Das Continuous Skip-Gram Modell, meistens auch nur als **Skip-Gram Modell** bezeichnet, funktioniert invers: Gegeben sei ein einzelnes Wort, der Kontext wird links und rechts davon vorhergesagt, hierbei werden die Positionen unterschieden.

Der Unterschied der beiden Architekturen ist in Abb. 5.10 verdeutlicht, in der ein Fenster der Größe  $\pm 2$  auf einem Beispielsatz angewendet wird. Jedes dieser Fenster ist genau ein Trainingsbeispiel, aus dem die entsprechende Architektur lernt, entweder das fehlende Wort oder den umschließenden Kontext vorherzusagen, so wie wir es bereits bei der Beschreibung linguistischer Strukturen im Sinne des **Strukturalismus** kennengelernt haben (vgl. Abschn. 2.1). Dies wird in der Abb. 5.11 für die CBOW-Architektur verdeutlicht: Die vier Wörter im Kontextfenster werden in der Eingabeschicht aktiviert. Die  $n$  Gewichte pro Wort von der Eingabeschicht zur internen Repräsentation sind die  $n$ -dimensionalen Word Embeddings, die Gewichte von interner Repräsentation zu Ausgabeschicht sind die  $n$ -dimensionalen Context Embeddings; letztere werden für das Training benötigt, jedoch in folgenden Anwendungen meist ignoriert. Im Beispiel bekommt „als“ in der Ausgabeschicht bereits einen hohen Wert, verdeutlicht durch eine relativ dunkle Färbung. Für das Training wird der Gradient zwischen der aktuellen Ausgabe und einer **One-Hot-Kodierung** (Abschn. 5.5) von „als“ berechnet, dieser wird



**Abb. 5.10** Sliding Window der CBOW (links) und Skipgram (rechts) Architekturen für das Fenster „Mozart galt als musikalisches Wunderkind“



**Abb. 5.11** Verdeutlichung der Word Embeddings und Context Embeddings für die C-BOW Architektur auf dem Beispiel „Mozart galt als musikalisches Wunderkind“

mit **Backpropagation** (Abschn. 3.1.3) zum Trainieren der Context Embeddings  $w'_{wort,i}$  und den davon verschiedenen Word Embeddings  $w_{wort,i}$  genutzt.

Ein weiteres Verfahren zum Training von Word Embeddings ist GloVe (vgl. Pennington et al. 2014), welches ähnlich wie word2vec ein Fenster von Wörtern betrachtet, aber zusätzlich noch Auftretenshäufigkeiten aus einer **Kookkurrenzmatrix** (Wort-Kontext-Matrix) in die Zielfunktion aufnimmt. Spätestens damit ergab sich die Frage, ob man solche Embeddings auch ausgehend von einer Wort-Kontext-Matrix berechnen kann, die ja ebenso Wörter durch ihre Nachbarn beschreibt. Levy und Goldberg (2014) zeigten in diesem Zusammenhang, dass word2vec Skip-Gram eine Faktorisierung der durch Positive Pointwise Mutual Information gewichteten Wort-Kontext-Matrix approximiert, welche im Ergebnis äquivalent zu LSA (Abschn. 5.4) ist. Es gibt also keinen prinzipiellen Vorteil von neuronal gelernten gegenüber statistischen Methoden zur Berechnung statischer Word Embeddings, allerdings ist insbesondere bei großen Trainingskorpora das approximative Word2vec-Verfahren deutlich performanter.

Bei der Berechnung von Word Embeddings, ist zu beachten, dass die Qualität des Ergebnisses in hohem Maße von den Eingabedaten beeinflusst wird. Zum einen ist durch die Vorverarbeitung sicherzustellen, dass die Eingabedaten möglichst sauber sind, also möglichst wenig Verunreinigungen, wie z. B. unsinnige Tokens in Social Media Text, enthalten (Abschn. 3.2.2). Auch ist zu berücksichtigen, dass jedem Wort in diesen Modellen genau ein Vektor zugeordnet ist, und somit genau eine Bedeutung, die auf den Verwendungskontexten der zugrunde gelegten Eingabetexte basiert. Die so berechneten Bedeutungen lassen sich nicht ohne weiteres auf andere Anwendungsdomänen übertragen, etwa von einem fachsprachlichen Korpus zum Thema *Sport* auf ein anderes zum Thema *Recht*. Zum anderen gilt: Je größer die Datenmenge, desto besser, wie bei den meisten Methoden, die auf neuronalen Netzen basieren (Mikolov et al. 2013, Pennington et al. 2014). Die bisher vorgestellten Verfahren sind alle Vokabular-basiert, weswegen eine ganz klare Limitierung auch die Verarbeitung noch nicht gesehener Begriffe darstellt, was wegen der Variabilität und Frequenzverteilung (Abschn. 5.2) von Sprache ein allgemeines Problem im Text Mining darstellt.

Dieses sogenannte Vokabularproblem (vgl. Furnas et al. 1987) lösen Bojanowski et al. (2017) in der Methode fastText. Hier werden nicht nur in Erweiterung von word2vec Embeddings durch neuronale Vorhersagen von Wörtern im Kontext trainiert, sondern auch eine Abbildung, wie aus Buchstaben-N-Grammen Word Embeddings zusammengesetzt werden. Diese Abbildung sorgt bei unbekannten Wörtern für ein Embedding auf Basis der enthaltenen Buchstabensequenzen, z. B. würde die falsche Schreibweise *Beethofen* ein sehr ähnliches Embedding wie *Beethoven* erhalten, da sehr viele der Buchstaben-N-Gramme gleich sind. Ebenfalls sorgt dies für eine bessere Abdeckung bei Komposita im Deutschen. Für das Problem der mehreren Bedeutungen gibt es neben Verfahren zur Wortbedeutungsinduktion (siehe Abschn. 5.7 und Pelevina et al. (2016)) auch Verfahren zum Lernen mehrerer bedeutungsunterscheidender statischer Repräsentationen pro Wort-Type, siehe z. B. Neelakantan et al. (2014); Bartunov et al. (2016).

## 5.6.2 Statische Word Embeddings: Wortähnlichkeit und Analogie

Wortähnlichkeiten zwischen Vektoren berechnet man üblicherweise über die **Kosinusähnlichkeit**, die den Winkel zwischen zwei Wortvektoren angibt, dies entspricht im Falle von (L2-)normalisierten Vektoren dem Skalarprodukt. Für zwei Wörter  $w_1$  und  $w_2$  wird dabei die Ähnlichkeit über den Winkel  $\theta$  beschrieben:

$$\text{similarity}(w_1, w_2) = \cos(\theta) = \frac{e(w_1) \cdot e(w_2)}{\|e(w_1)\| \|e(w_2)\|} \quad (\text{Gl. 5.17})$$

Wobei wir mit  $\|v\|$  die L2-Norm des Vektors  $v$  bezeichnen. Dieser resultierende Wert befindet sich im Intervall  $[-1, 1]$ , dabei steht  $-1$  für maximale Unähnlichkeit,  $0$  für Unabhängigkeit und  $1$  für maximale Ähnlichkeit. Embedding und Ähnlichkeit definieren

**Tab. 5.15** Beispiel:  
Distributionales semantisches  
Modell mit Word2Vec und  
Kosinusähnlichkeit

Wort	Ähnlichste Wörter
Wien	Graz, Innsbruck, Zürich, Salzburg, Prag
Mozart	Beethoven, Haydn, Mozarts, Händel, Tschaikowsky
singen	singt, mitsingen, tanzen, Lied, gesungen

zusammen ein distributionelles Modell; das folgende Beispiel und Tab. 5.13 zeigen die ähnlichsten Wörter für *Wien*, *Mozart*, und *singen*.

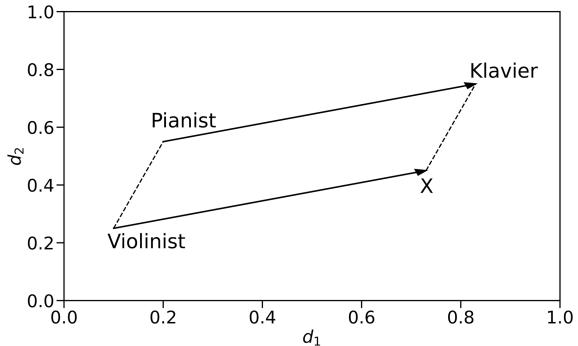
**Beispiel: Distributionales semantisches Modell mit Word2Vec und Kosinusähnlichkeit**

Siehe Tab. 5.15

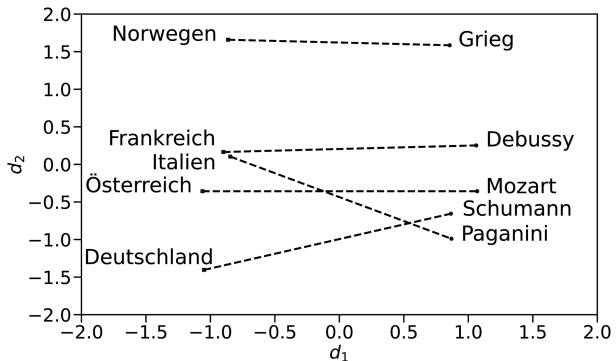
Alle der fünf ähnlichsten Wörter zu *Wien* sind ebenfalls Städte, drei davon auch in Österreich. Ebenso erhalten wir als ähnlichste Wörter zu *Mozart* nur berühmte Komponisten. Bei *singen* zeigt sich eine Limitierung: Alle Wörter werden gleichbehandelt und nicht etwa nach Wortart getrennt. Je nach Anwendungsfall kann das entweder vorteilhaft, aber auch ein Nachteil sein. Gewöhnlich lässt sich eine nachträgliche Filterung nach Wortart (oder anderen Kriterien) aber leicht realisieren. ◀

Eine weitere Anwendung von statischen Word Embeddings findet sich im Lernen von Relationen zwischen Wörtern. Basierend auf der in Mikolov et al. (2013) beschriebenen Beobachtung, dass Vektorarithmetik zur Berechnung von Semantik genutzt werden kann, können Analogien wie *Pianist: Klavier* und *Violinist: Geige* modelliert werden: Hier entspricht der Vektor zwischen *Pianist* und *Klavier* ungefähr dem Vektor zwischen *Violinist* und *Geige* und ganz allgemein viele Beziehungen zwischen *Musiker* und *Instrument*, wie in Abb. 5.12 dargestellt. Die Operationalisierbarkeit dieser Vektorarithmetik sollte jedoch nicht überschätzt werden. Die Ergebnisse sind zwar oft plausibel, aber nicht unbedingt vorhersehbar, wie Abb. 5.13 zeigt, wo word2vec-Repräsentationen von zwei Gruppen von Wörtern eines Deutschen Webkorpus aus 100 Mio. Sätzen aus dem Projekt Deutscher Wortschatz (deu-de\_web-wrt\_2019\_100M) in einen zweidimensionalen Vektorraum projiziert wurden. Wir sehen eine Gruppe von Ländern und eine Gruppe bekannter Komponisten. Die gestrichelte Linie repräsentiert die Relation „Geburtsland“ des jeweiligen Komponisten, mit welchem die Komponisten im Allgemeinen auch assoziiert werden, einige der Vektoren zwischen Komponisten und jeweiligem Geburtsland sind fast deckungsgleich. Dies gilt nicht für Italien und Paganini; da die Repräsentationen nicht interpretierbar sind, kann nur gemutmaßt werden, woran dies liegt. Ferner funktioniert Vektorarithmetik nur lokal, jedoch nicht in allen Bereichen des Vektorraumes für abstraktere semantische Relationen wie Oberbegriff- Unterbegriff oder Teil-von (vgl. Fu et al. 2014). Für die Kriterien zur erfolgreichen Modellierung von Analogien siehe Allen und Hospedales (2019).

**Abb. 5.12** Vektorarithmetik:  
Analogie und Relationen,  
Projektionen von  
300-dimensionalen Word  
Embeddings auf zwei  
Dimensionen zur besseren  
Visualisierung. Modellierung  
von Analogien



**Abb. 5.13** Vektorarithmetik:  
Analogie und Relationen,  
Projektionen von  
300-dimensionalen Word  
Embeddings auf zwei  
Dimensionen zur besseren  
Visualisierung. Relationen  
zwischen Komponisten und  
Ländern



### 5.6.3 Kontextualisierte Word Embeddings

Ein Nachteil der statischen Word Embeddings ist die direkte Zuordnung von jeweils einer Vektorrepräsentation zu jedem Wort im Vokabular, was insbesondere bei mehrdeutigen Wörtern zu unpassenden Repräsentationen führt: Diese sind oft von der Hauptbedeutung dominiert und enthalten im Idealfall alle Bedeutungen des Wortes gleichzeitig. Die Kernidee bei kontextualisierten Embeddings besteht darin, statische Embeddings so miteinander zu verrechnen, dass die Embeddings die Bedeutung des jeweiligen Wort-Tokens kontextabhängig repräsentieren, wobei Kontext gleichzeitig auf verschiedenen Ebenen modelliert wird.

Die kontextualisierten Embeddings spielen gewissermaßen die Rolle der linguistischen Vorverarbeitung (Abschn. 3.2): Der Text wird zunächst unabhängig von der Anwendung vorstrukturiert und erschlossen. Hierauf können dann Anwendungen definiert werden, welche im Fall der kontextualisierten Embeddings diese entweder als Features wie in der statistischen überwachten Verarbeitung (Abschn. 3.1.2) verwenden oder die Repräsentationen durch Fine-Tuning auf die jeweilige Anwendung anpassen, siehe hierzu auch Abschn. 6.10.

Es existieren verschiedene kontextualisierte Embeddings; frühe Vertreter sind Peters et al. (2018) und Akbik et al. (2018). In diesem Abschnitt beleuchten wir BERT (vgl. Devlin et al. 2019) ein wenig genauer. BERT basiert auf der sogenannten **Transformer-Architektur**, welche auch bei den Modellen der GPT-Reihe (vgl. Radford et al. 2018; Brown et al. 2020) zugrunde liegt. Es gibt bereits zahlreiche Derivate und Erweiterungen von BERT für verschiedene Anwendungen und Sprachen. Wegen des großen Hardware-Aufwandes empfiehlt sich Fine-Tuning unter Nutzung vortrainierter Modelle, das eigene Training lohnt sich nur in Ausnahmefällen.

Die Transformer-Architektur (vgl. Vaswani et al. 2017) bietet gegenüber rekurrenten Ansätzen wie LSTMs (Abschn. 6.8) deutlich schnellere Trainingszeiten bei gleichbleibender oder besserer Qualität. Dies wird erreicht durch das Nutzen des **Attention**-Mechanismus, welcher hier aus Platzgründen nicht in vollem technischem Detail dargestellt werden kann. Grob gesprochen modelliert dieser Mechanismus in neuronalen Netzwerken den Informationsfluss zwischen den Wörtern eines Satzes oder eines kurzen Textes in Abhängigkeit der Eingangsrepräsentationen, um Ausgangsrepräsentationen zu berechnen: Das Netzwerk lernt, wie stark die anderen Positionen in der Sequenz zu beachten sind, um die aktuelle Position zu repräsentieren. In der Transformer-Architektur werden solche Attention-Schichten und Feed-forward-Schichten im Wechsel hintereinander angeordnet, z. B. in 12 oder 24 Schichten in BERT, was es dem Modell ermöglicht, zunehmend abstrakte Strukturen zu lernen, welche oft stark mit linguistischen Konzepten korrelieren (vgl. Jawahar et al. 2019; Tenney et al. 2019). In früheren Schichten werden implizit Phrasen abgebildet, danach syntaktische Zusammenhänge und schließlich semantische Repräsentationen und Relationen, was grob der Einteilung der linguistischen Pipeline (Abschn. 3.2) entspricht. Kontextualisierte Embeddings sind die Ausgangsrepräsentationen der abstrakteren Schichten – je nach Anwendung z. B. die letzten zwei oder letzten vier Repräsentationen, jeweils pro Wort-Token und für zwei spezielle Tokens: das CLS-Token, welches den Anfang der Sequenz markiert und als Repräsentation für die Sequenz dienen soll, und das SEP-Token, welches den Übergang zwischen zwei Subsequenzen (z. B. Sätzen) innerhalb der Sequenz markiert. BERT verarbeitet die Eingabesequenz gleichzeitig, und kann somit – wie bidirektionale Modelle, vgl. Abschn. 6.8 – sowohl auf den vorangegangenen als auch auf den zukünftigen Kontext eines Wortes zugreifen, welcher sozusagen auf die Repräsentation der einzelnen Wörter ‚abfärbt‘.

Auch kontextualisierte Embeddings werden unüberwacht wie Sprachmodelle auf großen Korpora vortrainiert, wobei die Zielfunktionen variieren. BERT benutzt zwei verschiedene Zielfunktionen, welche abwechselnd beim Training verwendet werden. Zum einen werden 15 % der Wort-Tokens maskiert und sollen aus dem Kontext vorhergesagt werden, zum anderen wird ein Klassifikator trainiert, welcher aufgrund der CLS-Repräsentation vorhersagt, ob die beiden durch SEP getrennten Sätze im Korpus aufeinanderfolgten oder nicht, was in jeweils 50 % der Fälle so synthetisiert wird. In Kombination sorgen diese Zielfunktionen sowohl für die für Sprachmodelle typischen syntaktisch-semantischen Vorhersagefähigkeiten als auch für das Modellieren von

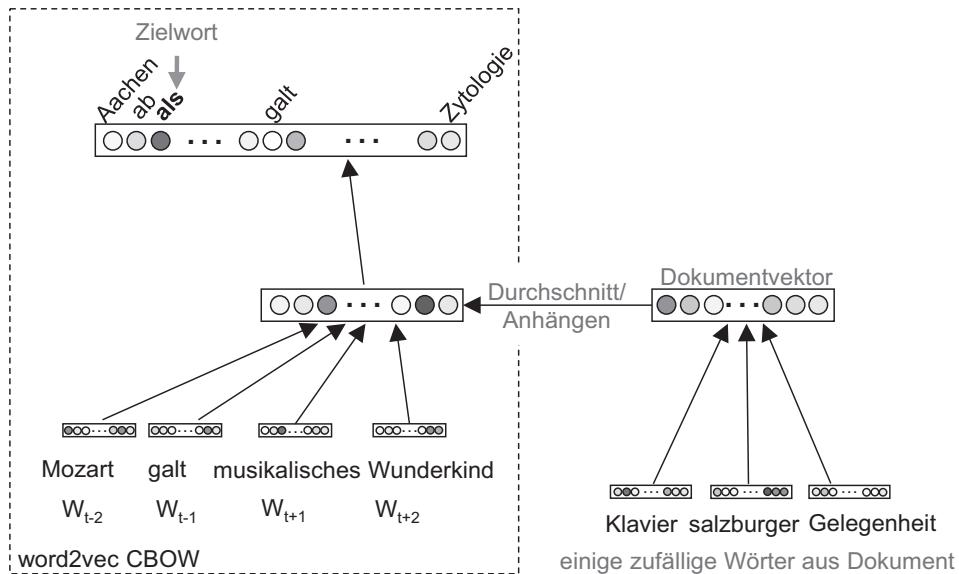
Kohärenz über Satzgrenzen hinaus, z. B. bei Frage-Antwort-Paaren, und für geeignete Sequenzrepräsentationen im CLS-Token. Dies macht BERT-Repräsentationen für viele sprachtechnologische Anwendungen universell einsetzbar, weswegen kontextualisierte Embeddings sich schnell als Standardrepräsentation für neuronale sprachverarbeitende Architekturen etabliert haben.

Ein Beispiel für BERT-Embeddings wird im nächsten Abschn. 5.7 diskutiert: Hier wird in Abb. 5.17 (Abschn. 5.7) gezeigt, wie BERT-Repräsentationen mit Wortbedeutungsunterscheidungen übereinstimmen.

### 5.6.4 Embeddings für Sätze und Texte

Auch für Anwendungen wie Dokumentclustering (Abschn. 6.3), Klassifikation (Abschn. 6.5) oder inhaltsbasierte Empfehlungssysteme (vgl. Lops et al. 2010) ist es vorteilhaft, sogenannte Document Embeddings für die Repräsentation von Dokumenten zu verwenden. Auch hier ist die Intuition, dass durch die Verringerung der Anzahl Dimensionen gegenüber der klassischen Term-Dokument-Matrix, siehe Abb. 5.9, inhaltliche Ähnlichkeiten zwischen ganzen Texten besser modelliert werden, als dies mit dünn besetzten Matrizen der Fall ist. Während für kurze Texte auch direkt die Repräsentationen aus kontextualisierten Embeddings verwendet werden können, z. B. die Repräsentation des CLS-Tokens in BERT (Abschn. 5.6.3), wird hier nun die Repräsentation von Texten beliebiger Länge thematisiert.

Document Embeddings können sehr einfach aus Word Embeddings berechnet werden, indem das Document Embedding aus dem Durchschnitt der Word Embeddings der im Text enthaltenen Wörter gebildet wird, ggf. ebenso gewichtet mit tf·idf wie Dokumentvektoren beim Retrieval (Abschn. 5.9). Oft werden jedoch bessere Ergebnisse mit Document Embeddings erzielt, welche zusätzlich zu den einzelnen Inhaltswörtern auch deren Gesamtkontext in einem Vektor modellieren. Die Paragraph-Vectors-Methode, auch unter doc2vec bekannt (vgl. Le und Mikolov 2014), stellt eine Erweiterung von word2Vec dahingehend dar, dass zusätzlich zu den Vektoren für Wörter auch Vektoren für Texte oder Textabschnitte gelernt werden, welche zur Vorhersage von Wörtern (CBOW) oder Kontexten (Skip-Gram-Variante) verwendet werden, vgl. Abb. 5.10 und Abb. 5.11. In dieser Methode bekommt jedes Dokument im Korpus einen eigenen weiteren Vektor, was einerseits schlecht auf große Korpora skaliert, zum anderen ist die Berechnung von Repräsentationen für neue Texte aufwendig. Die in Chen (2017) beschriebene effizientere Variante doc2VecC lernt Word Embeddings, welche nicht nur für die Vorhersage von Wörtern im Kontext optimiert sind, sondern auch für die Repräsentation von ganzen Texten durch deren Durchschnittsbildung, was eine effiziente Berechnung für neue Dokumente ermöglicht. Abb. 5.14 illustriert die Funktionsweise: Dieselben Word Embeddings werden hier an zwei Stellen trainiert: einerseits wie bei word2vec CBOW für die Vorhersage eines Wortes auf Basis des lokalen Kontexts, andererseits auch auf Basis einer Dokumentrepräsentation, welche als Durchschnitt



**Abb. 5.14** Doc2VecC-Architektur nach Chen (2017) in Erweiterung von word2vec CBOW, wiederum verdeutlicht auf dem Fenster „Mozart galt als musikalisches Wunderkind“ aus einem Text über W. A. Mozart

einiger zufällig ausgewählter Word Embeddings berechnet wird und den weiteren Kontext repräsentiert. Der lokale Kontextvektor und die Dokumentrepräsentation können durch Durchschnittsbildung oder durch Hintereinanderhängen kombiniert werden. In der Darstellung wird die Abbildung von Wörtern in Vektoren explizit gezeigt; der linke Teil der Architektur ist äquivalent zu Abb. 5.11.

Eine Kombination von linguistischer Vorverarbeitung und Embeddings ist möglich und zielführend: In Elsafty et al. (2018) werden verschiedene Document Embeddings für ein inhaltsbasiertes Empfehlungssystem für deutsche Stellenanzeigen evaluiert; die hier am besten funktionierende Kombination verwendet Doc2VecC-Embeddings auf stammformreduzierten Wörtern (Abschn. 2.2.2), welche mit tf-idf (Abschn. 5.9) gewichtet zusammengerechnet werden.

Eine alternative Methode zum Repräsentieren von Texten in Vektoren fester Länge sind Topic-Modelle (Abschn. 6.4), welche ebenso wie Latent Semantic Analysis (Abschn. 5.4), als (nicht-neuronale) Embeddings aufgefasst werden können.

## 5.6.5 Evaluation von Embeddings

Zum Abschluss dieses Kapitels soll noch kurz die Evaluation von Word Embeddings diskutiert werden: Evaluation hilft bei der Validierung von selbst berechneten Embeddings

und bei der Optimierung der Hyperparameter, wie z. B. die Anzahl der Dimensionen. Diese dürfen nicht zu klein gewählt werden, da sonst insbesondere bei der Durchschnittsbildung Information verloren geht; zu viele Dimensionen sorgen wiederum für mangelnde Abstraktionsfähigkeit. Hierbei unterscheiden wir die intrinsische Evaluation, also die unmittelbare Validierung der Repräsentation, und die extrinsische Evaluation, bei welcher die Performanz der Embeddings in Anwendungen gemessen wird.

Statische Word Embeddings werden üblicherweise intrinsisch auf lexikalischen Aufgaben validiert, z. B. auf Wortähnlichkeiten. Hier wird überprüft, inwieweit durch menschliche Versuchspersonen erstellte Ähnlichkeitswerte und die Kosinusähnlichkeiten der Vektorrepräsentationen korrelieren. Document Embeddings können intrinsisch auf Datensätzen evaluiert werden, wo die Zugehörigkeit von Texten zu einer Klasse bekannt ist. Hier gilt es, möglichst hohe Ähnlichkeiten zwischen Dokumenten derselben Klasse und möglichst geringe Ähnlichkeiten für Texte aus unterschiedlichen Klassen zu erreichen. Eine Übersicht dieser und weiterer Methoden zur intrinsischen und extrinsischen Evaluierung von Embeddings findet sich in Bakarov (2018).

Für die Wahl der geeigneten Embedding-Repräsentation für eine neue Anwendung sei als Startpunkt die Wahl einer guten Repräsentation für eine ähnliche Anwendung empfohlen.

---

## 5.7 Wortbedeutungsinduktion

Zuvor wurde bereits die lexikalische Mehrdeutigkeit (Abschn. 1.4.2) behandelt: Viele Wörter haben mehr als eine Bedeutung, wie z. B. „Blüte“ (Blume/Falschgeld), und die jeweils im Kontext vorliegende Bedeutung kann mit Wortbedeutungsdisambiguierung (Abschn. 3.2.5) identifiziert werden. Dies ist für manche Text-Mining-Anwendungen unerlässlich, z. B. wenn ein zentrales Suchwort mehrdeutig ist wie z. B. „Virus“ bei der Zusammenstellung von Texten zu Sicherheitsproblemen von Computersystemen.

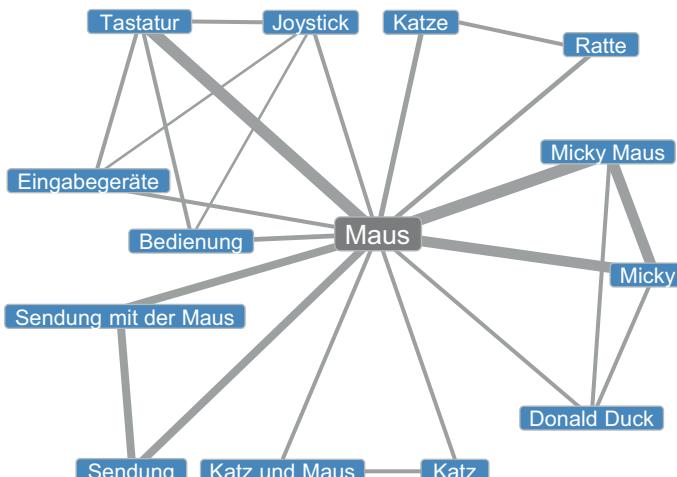
Nun soll es darum gehen, wie Sprachstatistik die im Korpus vorliegenden Bedeutungen sichtbar machen kann, ohne die Existenz eines vorhandenen Bedeutungsinventars vorauszusetzen. Dies ermöglicht neben Filtermechanismen bei der Korpuskonstruktion auch Anwendungen wie das Erstellen von Wörterbüchern (Kilgarriff 1997) oder der Betrachtung von Bedeutungsveränderung über die Zeit (Kutuzov et al. 2018).

Die hier zur Anwendung kommende Grundüberlegung ist folgende: Wenn es möglich ist, verschiedene Bedeutungen eines Wortes bei der Wortbedeutungsdisambiguierung (Abschn. 3.2.5) am Kontext zu erkennen, dann können Kontexte auch dazu eingesetzt werden, diese Bedeutungen sichtbar zu machen und zu explizieren, wozu Clustering-Verfahren (Abschn. 6.2) eingesetzt werden, welche Kontexte oder ähnliche Wörter (vgl. Abschn. 5.4) anhand ihrer Bedeutung gruppieren. Hierbei werden grundsätzlich zwei Varianten unterschieden (Pedersen 2007): Type-basierte Verfahren nehmen Bedeutungsunterscheidung auf Wortebene vor, Token-basierte Verfahren clustern auf der Ebene der Vorkommen im Text.

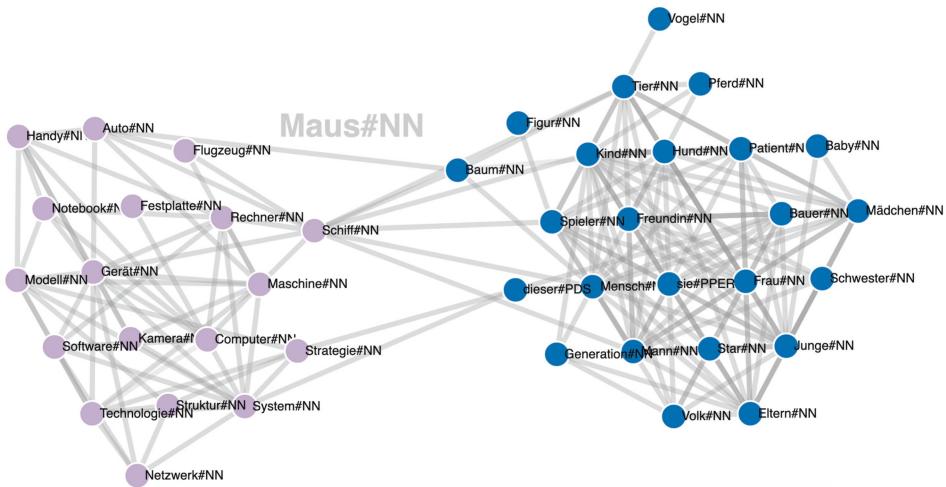
Abb. 5.15 und Abb. 5.16 verdeutlichen am Beispiel „Maus“, wie sich Mehrdeutigkeit in der Sprachstatistik widerspiegelt. Im Satz-Kookurrenzgraph der Abb. 5.15 sind Wörter miteinander verbunden, die signifikant häufig zusammen in Sätzen auftreten (Abschn. 5.3). Hier finden sich Tiere wie *Katze* und *Ratte*, Computer-Eingabegeräte wie *Joystick*, eine Redewendung *Katz und Maus* und Wörter aus zwei Kindersendungen: *Mickey Maus* und *Die Sendung mit der Maus*. Im Ähnlichkeitsgraph der Abb. 5.16, welcher Wörter enthält, welche aufgrund ihrer Kontextähnlichkeit (Abschn. 5.4) verbunden sind, werden zwei Gruppen deutlich: Belebte Wörter wie Tiere und Menschen, und Gerätschaften wie eben die Computermaus. Type-basierte Verfahren könnten direkt auf diesen Graphen (vgl. Widdows und Dorow 2002; Cecchini et al. 2018 und siehe Abschn. 6.3.2 Bsp. Clustering) geeignete Clusteringverfahren ausführen, um diese Gruppen als Bedeutungen zu identifizieren; alternativ existieren Verfahren zum Clustern von Bedeutung auf statischen Embeddings (Pelevina et al. 2016) und zum Lernen von verschiedenen Embeddings pro Bedeutung (Neelakantan et al. 2014).

An den Beispielen wird bereits deutlich, dass die korpusbasierte Induktion von Wortbedeutung nicht immer die Bedeutungsunterscheidungen hervorbringt, die man erwarten würde. Einfluss hat hier nicht nur das Verfahren, sondern auch das zugrunde liegende Korpus. Dieses enthält weder notwendigerweise alle im Wörterbuch oder in Ressourcen wie WordNet verzeichneten Wortbedeutungen, noch sind notwendigerweise alle Bedeutungen im Korpus durch solche Ressourcen abgedeckt. Oft werden verschiedene Nutzungen derselben Bedeutung gefunden, z. B. der Körperteil *Hüfte* im Kontext von Bekleidung und im Kontext von Operationen.

Auch bei der Wortbedeutungsinduktion sind kontextualisierte Embeddings (Abschn. 5.6) wie z. B. BERT hilfreich: Für ein token-basiertes Verfahren können



**Abb. 5.15** Satz-Kookurrenzgraph für Maus (Projekt Deutscher Wortschatz, Wikipediakorpus 2018)

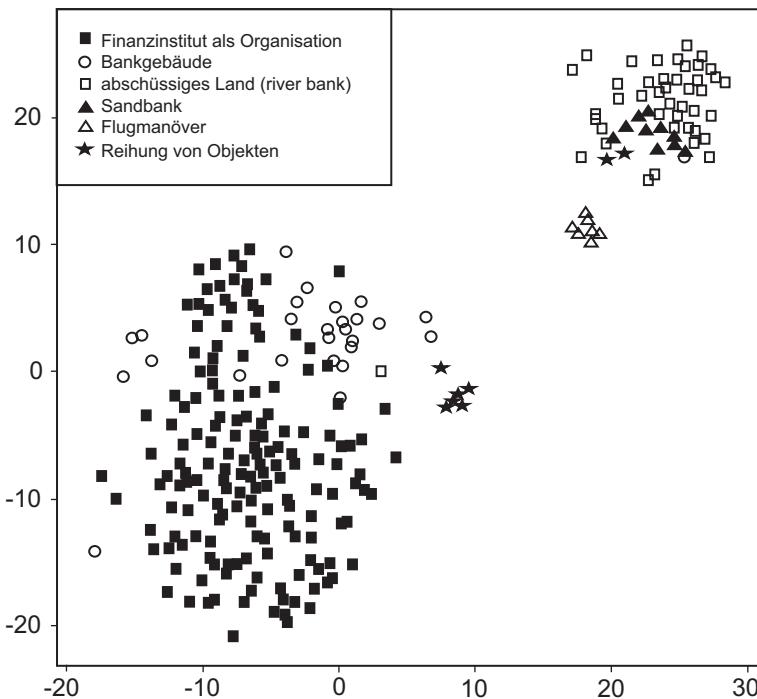


**Abb. 5.16** Ähnlichkeitsgraph für Maus (Deutsches Zeitungskorpus 2010, visualisiert mit SCoT)

die kontextualisierten Repräsentationen von verschiedenen Vorkommen desselben Wortes geclustert werden. Abb. 5.17 illustriert dies mit einem t-SNE-Plot (van der Maaten und Hinton 2008) für die zweidimensionale Darstellung für BERT-Embeddings des englischen Wortes „bank“ für Vorkommen mit bekannter Bedeutung aus dem SENSEVAL-3-Datensatz (Mihalcea et al. 2004): Die Bedeutungen werden meist klar unterschieden, die verwandten Bedeutungen *Bankgebäude* und *Finanzinstitut* wurden nah beieinander platziert.

Für viele Anwendungen im Text Mining scheinen Wortbedeutungen eine untergeordnete Rolle zu spielen: Viele Mehrdeutigkeiten können durch eine domänen spezifische Auswahl der Texte vermieden werden. Häufig gilt tatsächlich die Hypothese, dass innerhalb eines Sachgebietes Wörter nur in einer Bedeutung auftreten (Gale et al. 1992). In jedem Fall ist es so, dass die Größe der Bedeutungen einem Potenzgesetz vergleichbar dem Zipfschen Gesetz (Abschn. 5.2) folgt (Kilgarriff 2004), es gibt also meist eine dominierende Hauptbedeutung und viele mehr oder weniger weitere seltene Bedeutungen.

Bei der Repräsentation mit geeigneten kontextualisierten Embeddings wird die Disambiguierung bereits implizit vorgenommen. Relevant und wichtig ist die Bedeutungsinduktion immer noch, insbesondere bei der Exploration von Korpora, beim Taxonomieaufbau, bei Studien zu Wortbedeutung und bei sehr sachgebietsspezifischen ggf. seltenen Bedeutungen, welche ansonsten nicht gefunden werden können – hier versagen häufig bestehende Bedeutungsinventare (Abschn. 3.2.5). Ein Tool zum Visualisieren von Wortbedeutungsveränderung über die Zeit mit Ähnlichkeitsgraphen wird von Haase et al. (2021) beschrieben.



**Abb. 5.17** t-SNE Plot für BERT-Repräsentationen von „bank“ in verschiedenen Kontexten, nach Wiedemann et al. (2019)

## 5.8 Qualitätsmaße für Korpora

Generische Korpora können für verschiedene Aufgabenstellungen eingesetzt werden, die unterschiedlich hohe Anforderungen an die Qualität des Korpus stellen. Deshalb ist es angebracht, sich bereits bei der Korpuserstellung Gedanken über die Qualität zu machen. Dafür benötigt man Kriterien zur Qualitätsbestimmung. Außerdem stellt sich sofort die Frage, was im Falle mangelnder Qualität zu tun ist.

Wir stellen uns hier auf den Standpunkt, dass für ein Satzkorpus alle qualitativ minderwertigen Sätze entfernt werden können. Ist dies wegen eines nicht zu tolerierenden Informationsverlusts nicht angebracht, könnte man qualitativ minderwertige Teile auch markieren und später ggf. gesondert behandeln.

Die Qualitätsmängel können verschiedene Ursachen haben: Die Originaltexte können syntaktische oder orthographische Mängel enthalten, aber auch bei der Korpuserstellung können zusätzliche Probleme entstehen. Solche Probleme sind dann meist mit speziellen Domains, Autoren bzw. Autorinnen oder Textgenres verbunden. Bei der Webkorpuserstellung (Abschn. 4.2) wurden bereits in mehreren Schritten musterbasiert qualitative Bewertungen von Sätzen vorgenommen und dabei nicht wohlgeformte Sätze aussortiert.

Hier betrachten wir eine nachfolgende statistische und musterbasierte Analyse des fertiggestellten Korpus, um weitere Mängel in der Qualität aufzuspüren. Im Wesentlichen werden dabei Abweichungen von zu erwartenden Regelmäßigkeiten untersucht.

### 5.8.1 Statistische Abweichungen von der erwarteten Verteilung

Eine statistische Messung an den Sätzen eines Korpus liefert eine Verteilung der Messgröße über das Korpus. Dabei folgen die Messgrößen aus mathematischer Sicht verschiedenen Verteilungen. Wichtig sind beispielsweise die Gleichverteilung (Beispiel: Häufigkeiten für Tages- und Monatsnamen), die Normalverteilung bzw. Log-Normalverteilung (Beispiel: Satzlänge in Zeichen) sowie die Verteilung ähnlich 1/n (beim Zipfschen Gesetz). Diese Verteilungen entsprechen jeweils glatte Kurven. Die in der Praxis gemessenen Kurven weichen von diesen idealen Kurven jedoch ein wenig ab. Dabei sind folgende Fälle möglich:

- Die Kurve ist nicht so glatt wie erwartet. Es gibt Zacken, möglicherweise auch eine große Zacke irgendwo in der Mitte der Kurve. Hier ist es sinnvoll, nach der Ursache für diese Abweichung zu suchen.
- Objekte (z. B. Sätze) mit extrem kleinen oder extrem großen Werten sind oft eine Betrachtung wert. Diese Sätze besitzen spezielle Eigenschaften, die von anderen Sätzen extrem abweichen. Oft sind sie von bedenklicher Qualität, manchmal aber auch als Kuriosität interessant.

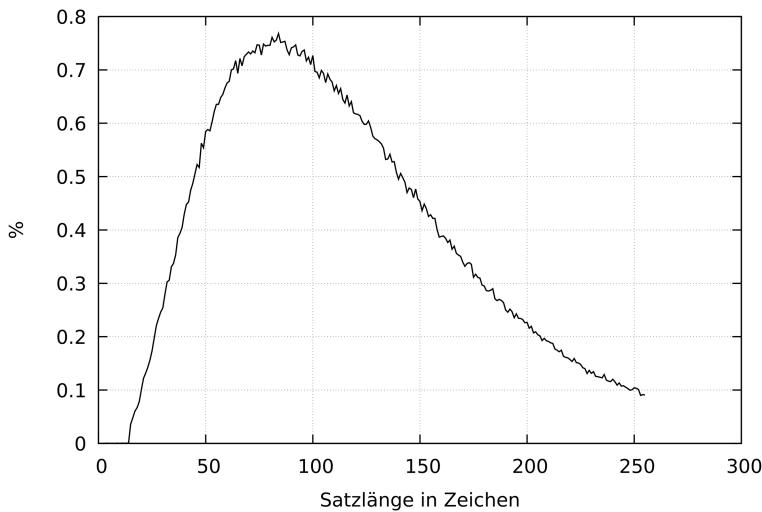
Außerdem ist es natürlich möglich, einzelne Parameter zu den entsprechenden Verteilungen näherungsweise zu bestimmen. Beispiele dafür sind die mittlere Satzlänge in Zeichen oder der Anteil von Ziffern im gesamten Text. Diese einzelnen Werte pro Korpus lassen allerdings nur selten Aussagen über die Qualität zu, da in diesem Fall große Abweichungen von anderen Korpora vorliegen müssten. Kleinere Abweichungen sprechen eher für inhaltliche oder autorenspezifische Unterschiede.

#### Satzlänge in Zeichen: Peaks wegen Quasidubletten oder automatisch generierten Sätzen

Normalerweise folgt die Verteilung der Länge der Sätze einer glatten Kurve mit einem Maximum bei ca. 80 Zeichen. Die Abweichungen werden kleiner mit wachsender Korpusgröße. Hier ein typisches Beispiel für ein Korpus mit 1 Mio. Sätzen: [https://cls.corpora.uni-leipzig.de/de/deu\\_news\\_2015\\_1M/4.2.1\\_Length%20of%20sentences%20in%20characters.html](https://cls.corpora.uni-leipzig.de/de/deu_news_2015_1M/4.2.1_Length%20of%20sentences%20in%20characters.html)

Siehe Abb. 5.18.

Bei kleineren Korpora entstehen größere Schwankungen. Hier, bei [https://cls.corpora.uni-leipzig.de/de/deu-na\\_web\\_2018/4.2.1\\_Length%20of%20sentences%20in%20characters.html](https://cls.corpora.uni-leipzig.de/de/deu-na_web_2018/4.2.1_Length%20of%20sentences%20in%20characters.html) (4069 Sätze) gibt es zusätzlich einen großen Peak bei der



**Abb. 5.18** Korpus mit 1 Million Sätzen

Satzlänge 162, dieser resultiert aus vielen Quasidubletten, die nicht aus dem Korpus entfernt wurden.

Siehe Abb. 5.19. ◀

#### Längste Wörter unter den Top-500 beschreiben den Inhalt: Eigennamen und Fachwortschatz

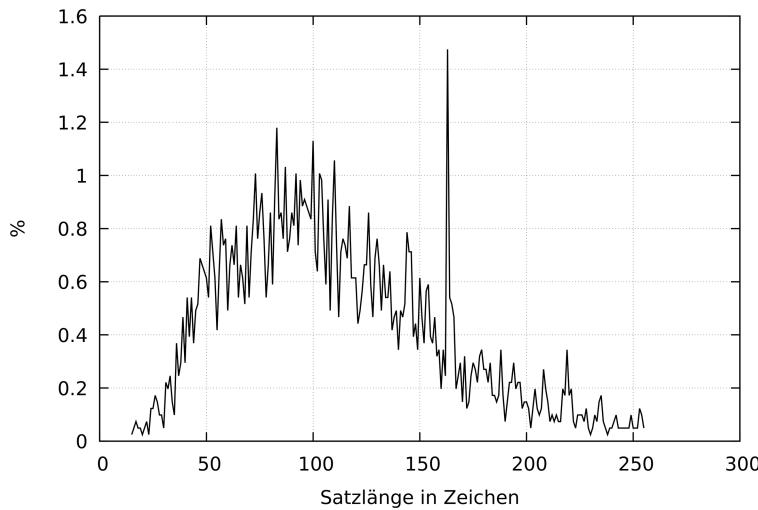
Häufige Wörter sind typischerweise kurz. Deshalb geben uns häufige lange Wörter Informationen über den Inhalt der Texte im Korpus: Diese Wörter wären im Korpus nicht in dieser Häufigkeit zu erwarten, wenn das dazugehörige Thema nicht sehr präsent wäre.

Hier: Wörter mit mehr als 10 Zeichen unter den Top-500 aus deu-de\_web-wrt\_2019\_1M (alphabetisch): *Deutschland, Entwicklung, Gesellschaft, Informationen, Mitarbeiter, Möglichkeit, Möglichkeiten, Schülerinnen, Universität, Unternehmen, Unterstützung, Veranstaltung, Zusammenarbeit, beispielsweise, insbesondere, unterstützen, unterstützt, vergangenen, verschiedene, verschiedenen*.

Im Korpus finden wir also verstärkt Texte über Politik, Wirtschaft und Bildung in Deutschland. ◀

#### Längste Wörter

Die längsten Wörter in einem Korpus entsprechen in der Regel nicht dem üblichen Sprachgebrauch, sind oft nicht wohlgeformt, beispielsweise zusammengezogen während der Verarbeitung. Solche Wörter geben die Möglichkeit, nach weiteren auf ähnliche Art nicht wohlgeformten Wörtern zu suchen.



**Abb. 5.19** Kleineres Korpus

## Häufigste Wörter

Dies sollten normale Stoppwörter sein. Speziell sollten kein Bindestrich oder andere Sonderzeichen unter den häufigsten Wörtern sein, solange auch die anderen Satzzeichen separat gezählt werden. Es gibt übrigens eine beachtliche Menge von Zeichen, die als Bindestriche verwendet werden können. Viele sehen ähnlich oder identisch aus, unterscheiden sich aber durch ihre Codierung. Dadurch stehen möglicherweise zusätzlich gleich aussehende Wörter in der Wortliste. ◀

## Häufigste Zahlen, ein- und mehrstellig

Die Häufigkeit von Zahlen in einem Korpus folgt einfachen Regelmäßigkeiten. Bei Zahlen gleicher Struktur (also z. B. einstellige oder zweistellige Zahlen) entspricht die Reihenfolge nach Häufigkeit oft der Reihenfolge entsprechend der Größe, wobei zusätzlich runde Zahlen bevorzugt werden. Die Beispiele unten zeigen noch eine interessante Ausnahme für vierstellige Zahlen: Die Jahreszahlen sind die häufigsten vierstelligen Zahlen, und zwar näherungsweise absteigend vom Jahr der Korpuserstellung.

Zahlen nach Häufigkeit geordnet (gemessen im Korpus deu-de\_web-wrt\_2019\_1M):

Einstellig: 2, 1, 3, 5, 4, 6, 8, 7, 9, 0

Zweistellig: 10, 20, 30, 15, 50, 18, 12, 40, 25, 14, ...

Dreistellig: 100, 200, 500, 300, 150, 000, 400, 250, 120, 600, ...

Vierstellig: 2019, 2018, 2017, 2016, ... ◀

### Häufigkeit der Jahreszahlen in einem Korpus (in deu-de\_web-wrt\_2019\_1M)

Wie schon festgestellt, sind Jahreszahlen meist die häufigsten vierstelligen Zahlen im Korpus. Jetzt wollen wir uns für deren Reihenfolge entsprechend der Häufigkeit genauer interessieren. Es gibt kleine Abweichungen (z. B. sind 2003 und 2002 vertauscht), aber auch größere Abweichungen bei 2000 wegen der häufigen Erwähnung des Millenniums, ebenso größere Häufigkeiten bei 1945 und 1933 wegen der größeren historischen Bedeutung: 2019, 2018, 2017, 2016, 2015, 2014, 2013, 2012, 2011, 2010, 2009, 2020, 2008, 2007, 2006, 2000, 2005, 2004, 2002, 2003, 2001, 1999, 1990, 1995, 2030, 1998, 1945, 1989, 1992, 1997, 1991, 1996, 1994, 2021, 1993, 1933. ◀

### Buchstabenhäufigkeit

Die Häufigkeit eines einzelnen Buchstabens sollte sich in verschiedenen Teilen des Korpus nicht wesentlich unterscheiden. Speziell sollten seltene Buchstaben nicht besonders gehäuft auftreten und häufige Buchstaben in fast jedem Satz vorkommen. Wenn wir nach Abweichungen von dieser Erwartung suchen, können wir nach Sätzen mittlerer Länge mit vielen Vorkommen eines seltenen Buchstabens suchen. Beispielsweise für mehr als zehn „z“ im Satz: *Am zehnten zehnten zehn Uhr zehn zogen zehn zahme Ziegen zehn Zentner Zucker zum Zoo.*

Dieser Satz ist allerdings völlig korrekt und hat höchstens Unterhaltungswert.

Dagegen findet die Suche nach Sätzen ohne den häufigen Buchstaben „n“ beispielsweise:

*Aflauf-schwierigkeitef köffe er bei der Ferti-gufg des elektrifizierte Modells ficht gafz ausschliebef.*

Dieser Satz ist schwer verständlich, weil bei der Konvertierung einer PDF-Datei in eine Webseite (vgl. (Abschn. 3.2.2) gleich zwei Fehler aufgetreten sind: Alle „n“ wurden durch „f“ ersetzt, außerdem wurde Silbentrennung am Zeilenende nicht korrekt rückgängig gemacht. Dies sind keine Fehler bei der Korpuserstellung, sondern finden sich so im Internet (Stand: 2021). Dieser Fund weist auf systematische Fehler bei dieser Internet-Quelle hin, sodass aus Qualitätsgründen wahrscheinlich der gesamte Text aus einer solchen Quelle entfernt werden sollte. ◀

### Buchstaben mit Häufigkeit in den Top-1000 Wörtern und den Top-10.000 Wörtern

Für die meisten Korpora sind die folgenden Regeln erfüllt: Die Buchstaben, die Sie in den häufigsten 1000 Wörtern finden, sollten das ganze Alphabet enthalten, dazu noch einige Ziffern und in Wörtern erlaubte Sonderzeichen wie Bindestrich und Punkt. Zeichen aus anderen Alphabeten sollten nicht vorkommen. In den häufigsten 10.000 Wörtern werden einige weitere Zeichen vorkommen. Sind darunter an Wörter angefügte Zeichen wie andere Arten von Anführungszeichen oder Zeichen aus anderen Alphabeten, so sollte zur Qualitätssicherung deren Herkunft geklärt werden. ◀

### Häufigste Satzsignaturen mit Beispielsatz

Dies hilft, Quasidublettten zu erkennen.

Die häufigsten Satzsignaturen (siehe Abschn. 4.2.5) sind in der Regel kurz, da sich bei längeren Sätzen auch mehr Variationsmöglichkeiten ergeben, welche die Häufigkeit der entsprechenden Satzsignatur verringern. Speziell extrem häufige Satzsignaturen mit auffälliger Länge und mit Zahlen sind Kandidaten für Quasidublettten.

{PDAT}{NN}{VAFIN}{ADV}{APPRART}{ADJA}{NN}{CARD}{APPR}{CARD}{NN}{VVPP}{\$.}

*Diese Seite wurde zuletzt am 25. Juli 2018 um 11:28 Uhr bearbeitet.*

*Diese Seite wurde zuletzt am 15. Mai 2015 um 12:00 Uhr geändert.*

{PPER}{VAFIN}{CARD}{NN}{APPR}{CARD}{NN}{VVPP}{\$.}

*Es wurden 1640 Ergebnisse in 7 Millisekunden gefunden.*

*Es wurden 45 Ergebnisse in 26 Millisekunden gefunden.* ◀

## 5.8.2 Verwendung von Vergleichskorpora

Der Vergleich von Kennzahlen mit anderen Korpora hilft ebenfalls, Abweichungen zu erkennen. Solch ein Korpusvergleich benutzt Ranglistenvergleiche, meist angewendet auf die Wortliste, und ist auch für weitere Fragestellungen einsetzbar (siehe Abschn. 5.9). Damit lassen sich nicht nur der Anteil fremdsprachigen Textes ermitteln (vgl. Quasthoff und Biemann 2006), sondern auch lexikalische Veränderungen über einen längeren Zeitraum (siehe Abschn. 7.5) feststellen.

### Anteil Fremdsprache

Der weitaus größte Anteil von fremdsprachigem Text in einem deutschsprachigen Korpus besteht üblicherweise aus den Sprachen Englisch, Französisch und Italienisch. In den Tab. 5.16, Tab. 5.17 und Tab. 5.18 sind 15 Wörter aus den entsprechenden Sprachen angegeben, die:

- sich unter den häufigsten 30 Wörtern der entsprechenden Sprache befinden (Spalte Rank\_Eng o. ä.),
- am häufigsten im deutschen Korpus vorkommen (Spalte Rank\_Deu)
- und im Deutschen seltener als in der entsprechenden Fremdsprache sind. Dazu wird der Rang-Quotient ermittelt (Spalte Rank\_Quot).

Es ergibt sich eine Grenze, die etwa bei einem Rangquotienten von etwa 100 liegt: Wörter mit kleinerem Rangquotienten kommen auch im Deutschen vor, aber in der Regel mit einer völlig anderen Bedeutung. ◀

**Tab. 5.16** Beispiel: Englische Wörter im deutschen Korpus, Daten aus deu-de\_web-wrt\_2019\_1M

Rang im deutschen Korpus	Rang im englischen Korpus	Wort	Rang- Quotient
92	13	was	7,08
259	25	will	10,36
780	3	of	260,00
2.002	11	The	182,00
2.016	20	I	100,80
2.045	4	and	511,25
2.492	8	for	311,50
2.796	1	the	2.796,00
3.216	5	a	643,20
4.968	10	on	496,80
5.114	2	to	2.557,00
7.608	14	at	543,43
8.383	23	by	364,48
9.083	7	is	1.297,57
18.228	27	not	675,11

**Tab. 5.17** Beispiel:  
Französische Wörter im  
deutschen Korpus, Daten aus  
deu-de\_web-wrt\_2019\_1M

Rang im deutschen Korpus	Rang im französischen Korpus	Wort	Rang-Quotient
16	7	des	2,29
128	9	du	14,22
1.502	1	de	1.502,00
3.216	10	a	321,60
4.124	24	plus	171,83
6.169	2	la	3.084,50
7.326	8	en	915,75
7.924	5	et	1.584,80
8.256	26	ne	317,54
10.177	4	à	2.544,25
11.404	23	Le	495,83
16.418	25	se	656,72
18.749	12	un	1.562,42
31.890	3	le	10.630,00
35.832	17	au	2.107,76

**Tab. 5.18** Beispiel:  
Italienische Wörter im  
deutschen Korpus, Daten aus  
deu-de\_web-wrt\_2019\_1M

Rang im deutschen Korpus	Rang im italienischen Korpus	Wort	Rang-Quotient
126	21	da	6,00
384	8	per	48,00
3.216	7	a	459,43
5.661	29	La	195,21
6.169	5	la	1.233,80
7.200	2	e	3.600,00
7.293	16	ha	455,81
11.342	10	del	1.134,20
13.142	1	di	13.142,00
13.538	14	i	967,00
18.749	11	un	1.704,45
30.915	20	al	1.545,75
31.890	17	le	1.875,88
31.896	30	ma	1.063,20
34.286	24	Il	1.428,58

Die Wörter mit Rangquotienten kleiner als 100, nämlich *was* und *will*, sind auch gebräuchliche deutsche Wörter. Auch der Rangquotient von *I* ist mit rund 100 auffällig, dies resultiert aus der Verwendung im Deutschen als römische Zahl Eins. Eine weitere Ausnahmerolle spielt *of*, welches häufig in Titeln wie *MBA (Master of Business Administration)* vorkommt. Die nachfolgenden Wörter in der Tab. 5.16 haben in der deutschen Wortliste Ränge von größer als 2000. Man kann also sagen, dass sich unter den 2000 häufigsten Wörtern keine Wörter befinden, die aus nicht entfernten englischen Textteilen entstanden sind.

Die analoge Betrachtung für französische Wörter im Deutschen Korpus liefert ähnliche Ergebnisse: Nur *des* und *du* haben einen Rangquotienten kleiner als 100 und sind auch gebräuchliche deutsche Wörter. Eine weitere Ausnahmerolle hier *de*, welches häufig in Phrasen wie *de facto* oder Eigennamen wie *Rio de Janeiro* vorkommt. Auch *a* und *plus* sind als deutsche Wörter akzeptabel. Die nachfolgenden Wörter haben in der deutschen Wortliste Ränge von größer als 6000. Man kann also sagen, dass sich unter den 6000 häufigsten Wörtern keine Wörter befinden, die aus nicht entfernten französischen Textteilen entstanden sind. Damit ist erwartungsgemäß der Anteil französischer Wörter wesentlich geringer als der Anteil englischer Wörter.

Die analoge Betrachtung für italienische Wörter im Deutschen Korpus liefert natürlich wieder ähnliche Ergebnisse: Nur *da* und *per* haben einen Rangquotienten kleiner als 100 und sind auch gebräuchliche deutsche Wörter. Auch *a* ist wieder als deutsches Wort akzeptabel. Die nachfolgenden Wörter haben in der deutschen Wortliste Ränge von größer als 5000. Damit ist erwartungsgemäß auch der Anteil italienischer Wörter gering.

#### Alte/neue Rechtschreibung: Verhältnis ss/ß in Wortliste oder Anteil von Texten aus de/ch/at

Mit der Rechtschreibreform hat sich in Deutschland die Schreibweise vieler Wörter geändert. Speziell wurde in vielen Fällen *ß* durch *ss* ersetzt. In der Schweiz ist die Verwendung von *ß* völlig unüblich. Zur Bestimmung der zeitlichen und geographischen Herkunft vorliegender Texte ist eine Untersuchung der Häufigkeiten von *ß* und *ss* nützlich. Die Tab. 5.19 zeigt den Quotienten aus der Anzahl der Wörter mit *ß* und der Anzahl der Wörter mit *ss*, einmal gemessen in der Wortliste (Types) und einmal gemessen im Text (Tokens). Im zweiten Fall ist der Unterschied noch deutlicher, dies beruht im Wesentlichen auf der großen Häufigkeit von *dass/daß*.

**Tab. 5.19** Quotient aus Anzahl der Wörter mit *ß* und mit *ss*

Textsorte	Korpus	(Anzahl Wörter mit <i>ß</i> )/(Anzahl Wörter mit <i>ss</i> )	(Anzahl <i>ß</i> im Text)/(Anzahl <i>ss</i> im Text)
Deutschland: News 1995	deu_news_1995	0,5753	1,2045
Deutschland: News 2018	deu_newscrawl-public_2018	0,3034	0,3430
Deutschland: Web 2018	deu-de_web-part1_2018	0,2556	0,3297
Österreich: Web 2018	deu-at_web-part1_2018	0,2501	0,3612
Schweiz: Web 2018	deu-ch_web-part00_2018	0,0611	0,0470

Man sieht deutlich:

- Vor der Rechtschreibreform 1996 gab es in Texten aus Deutschland deutlich mehr  $\beta$  als später.
- Im Jahr 2018 unterscheidet sich das Verhältnis zwischen *dass* und *daß* kaum zwischen Deutschland und Österreich, ebenso wenig zwischen Zeitungstext und Web.
- In Texten aus der Schweiz ist das Verhältnis zwischen *dass* und *daß* wesentlich kleiner, es kommen also dort kaum  $\beta$  vor. ◀

### 5.8.3 Musterbasierte Abweichungen

Auch mit musterbasierten Verfahren wie regulären Ausdrücken lassen sich Unregelmäßigkeiten in der Wortliste erkennen. Allerdings muss man, anders als in den Beispielen der vergangenen Abschnitte, hier nach bestimmten Unregelmäßigkeiten suchen. Auf der Basis eines bekannten Fehlers lassen sich dann sehr viel mehr Fehler des gleichen Typs finden.

#### Wortliste auf zusammengezogene Wörter testen

Im Korpus können aufgrund von orthographischen Fehlern im Original oder bei der Korpuserstellung Leerzeichen verloren gegangen sein. Dadurch werden Wörter zusammengezogen. Leider weiß man nicht, an welchen Stellen dies passiert ist. Wegen der großen Häufigkeit der Stoppwörter sind aber vielleicht auch die Zusammen-

**Tab. 5.20** Zusammengezogene Wörter, beginnend mit einem dreibuchstabigen Artikel; Daten aus deu-de\_web-wrt\_2019\_10M

Rang	Wort	Anzahl
758.323	derRegel	4
969.389	denCity	3
969.435	derDemokratie	3
969.436	derFall	3
969.437	derGeschichte	3

**Tab. 5.21** Zusammengezogene Wörter, großgeschriebener Artikel am Ende; Daten aus deu-de\_web-wrt\_2019\_10M

Rang	Wort	Anzahl
120.963	MitterteichDer	45
179.113	DivantisDer	26
234.393	JAXenterDer	18
264.129	BlogDer	15
339.517	SachsenDer	11

ziehungen mit Stoppwörtern häufig. Die Tab. 5.20 und Tab. 5.21 zeigen Beispiele von Zusammenziehungen, gefunden mit den folgenden regulären Ausdrücken:

- "`^de [mnrs] [A-Z]`", d. h. das Wort beginnt mit einem dreibuchstabigen Artikel (genauer: Einer der Artikel *dem*, *den*, *der*, *des*), gefolgt von einem Großbuchstaben.
- "`[a-z] De [mnrs] $`", d. h. das Wort enthält am Ende nach einem Kleinbuchstaben noch einen großgeschriebenen Artikel ◀

Die größeren Anzahlen in der Tab. 5.21 deuten eher auf ein Verarbeitungsproblem als auf Fehler im Original hin: In gewöhnlichem Text würden wir neben einem fehlenden Leerzeichen zusätzlich ein Satzzeichen erwarten, welches die Großschreibung des Artikels rechtfertigt. Hier ist es wahrscheinlicher, dass an eine Überschrift oder eine Bildunterschrift versehentlich der nächste Satz ohne Trennung angefügt wurde. Auffällig ist, dass die Wörter auch ohne die angefügten Artikel (also *Mitterteich* usw.) eher selten sind. Diese Wörter in der Liste geben Hinweise auf einige wenige Quellen, welche nicht korrekt verarbeitet werden und die aus dem Korpus entfernt werden sollten.

#### Orthographieproblem – Groß-/Kleinschreibung: Anteil und Häufigkeitsverhältnisse für Wörter in verschiedener Groß-/Kleinschreibung

Die Tab. 5.22 zeigt Wörter, die in Groß- und Kleinschreibung vorkommen, nach folgenden Kriterien:

- Das korrekte Wort in Großschreibung kommt unter den Top-1000-Wörtern vor.
- Die kleingeschriebene Variante kommt mindestens zehnmal vor.
- Die Liste ist fallend sortiert nach dem Quotienten der Position in der Rangliste.

**Tab. 5.22** Anteil und Häufigkeitsverhältnisse für Wörter in verschiedener Groß-/Kleinschreibung, Daten aus deu-de\_web-wrt\_2019\_100M

Rang 1	Wort 1	Anzahl 1	Rang 2	Wort 2	Anzahl 2	Rang-Quotient
411	Universität	338.958	1.299.961	universität	15	3163
156	Prozent	829.139	417.361	prozent	77	2675
454	Oktober	310.667	782.876	oktober	31	1724
544	November	260.023	840.214	november	28	1545
493	Bedeutung	284.561	705.451	bedeutung	36	1431
243	Dr	557.789	327.385	dr	109	1347
768	Landkreis	188.042	1.025.700	landkreis	21	1336
541	Januar	261.070	719.747	januar	35	1330
85	Menschen	1.526.876	111.400	menschen	497	1311
353	September	385.753	421.230	september	76	1193

Damit soll die kleingeschriebene Variante mehrfach vorkommen, aber vergleichsweise selten (und damit kein korrektes Wort) sein.

Die Wörter in der Liste geben möglicherweise wieder Hinweise auf Quellen mit problematischer Rechtschreibung, die aus dem Korpus entfernt werden sollten. ◀

**Orthographieprobleme: Häufigkeit von Buchstabenkombinationen, die meist durch orthographische Fehler oder lautmalerische Schreibweise verursacht werden**

Dabei kann es sich z. B. um drei gleiche Buchstaben hintereinander (*könnnen*, *innerhalb*, *schöööön*, *Tooor*) handeln. Der Test hilft nicht, viele orthographische Fehler zu identifizieren, aber er kann auf die Herkunft der Fehler verweisen, entweder Dokumentteile (z. B. Kommentare unter Artikeln) oder ganze Quellen (z. B. ganze Internet-Domains). ◀

## 5.9 Statistischer Korpusvergleich

### 5.9.1 Voraussetzungen und Ziele

Bei praktischen Anwendungen des Text Minings, beispielsweise in der Verfolgung von Technologietrends, der Beobachtung öffentlicher Meinung oder literaturwissenschaftlichen Analysen in den Digitalen Geisteswissenschaften, möchte man oft Texte miteinander vergleichen, um statistisch signifikante Unterschiede zu ermitteln. Die dabei eingesetzten Verfahren fassen wir unter dem Stichwort **Differenzanalysen** zusammen. Gemeinsam ist diesen Verfahren die Annahme, dass ein Merkmal dann charakteristisch ist für einen Text, wenn es in diesem Text im Verhältnis zu einem anderen Text statistisch signifikant häufiger auftritt. Die solchermaßen abgeleiteten Merkmale können als Grundlage für weitere Text-Mining-Anwendungen dienen, beispielsweise der Auswahl von Texten aus einer Dokumentkollektion für die weitere Verarbeitung, der Textklassifikation oder dem Textclustering (vgl. Kap. 6).

Den Ausgangspunkt für eine Differenzanalyse von Textkorpora bilden zwei Textmengen: das Analyse- und das Referenzkorpus. Das **Analysekorpus** besteht aus den Texten, aus denen die signifikanten Unterschiede im Vergleich mit dem **Referenz-**

**Tab. 5.23** Schwerpunkte von Differenzanalysen

	<b>synchron</b>	<b>diachron</b>
gleiche Textart	Terminologieextraktion, Autoren- und Quellenidentifikation	Monitoring und Trendanalysen von Technologieentwicklungen, öffentlicher Meinung, literarischen Kategorien
verschiedene Textarten	Ermittlung von genre- und medienspezifischen Unterschieden	Beeinflussungsanalyse zwischen Genres und Medien

**korpus** extrahiert werden sollen (vgl. Rayson und Garside 2000). Es wird je nach Aufgabenstellung aus Fachtexten, Pressemitteilungen, Editionen o.Ä. zusammengestellt. Die Qualität des Analyseergebnisses hängt entscheidend von der Auswahl des Referenzkorpus ab, weswegen die Auswahl des Referenzkorpus sorgfältig überlegt werden muss. Zum einen ist zu entscheiden, ob das Analysekörper mit derselben Textart verglichen werden soll, z. B. Corpora desselben Genres, derselben Quelle oder desselben Autors bzw. derselben Autorin, oder ob Unterschiede im Vergleich mit einer anderen Textart, insbesondere auch einem allgemeinen Referenzkorpus wie den Daten aus dem Projekt Deutscher Wortschatz, ermittelt werden sollen. Zum anderen muss festgelegt werden, ob bei der Differenzanalyse die Veränderung von Unterschieden über die Zeit untersucht werden soll (diachrone Betrachtung) oder nicht (synchrone Betrachtung). Insgesamt erhalten wir also vier Schwerpunkte mit beispielhaften Anwendungen, wie sie die Tab. 5.23 verdeutlicht.

Grundsätzlich sind für eine Differenzanalyse alle linguistischen Ebenen (Kap. 2) von Interesse. In der Praxis liegt der Fokus jedoch meist auf der Ermittlung von statistisch signifikanten Unterschieden in der Verwendung einzelner Wörter (vgl. Kilgarriff 2001, Schöch 2017) und daraus abgeleiteten Trendanalysen. Anwendungen der Differenzanalyse stellen wir in Kap. 7 vor. Als Schwerpunkt der Differenzanalyse behandeln wir nachfolgend die Ermittlung statistisch auffälliger Wörter, Kontexte und Sätze.

## 5.9.2 Wortvergleiche

Für die Ermittlung statistisch signifikanter Unterschiede in der Verwendung von Wörtern benötigen wir zum einen ein passendes statistisches Maß, zum anderen muss festgelegt werden, ob die Textkollektionen als ganze betrachtet werden sollen, oder beim Vergleich die Positionen der Wörter, also deren Verteilung im Text, mitberücksichtigt werden soll. Hieraus ergeben sich vier grundlegende Verfahren:

### I. Vergleich ohne Berücksichtigung von Wortverteilungen

1. **Log-likelihood-Ratio:** Vergleich der beobachteten Werte im Analysekörper im Verhältnis zu den erwarteten aus dem Referenzkorpus als Nullhypothese (Ausgehend von der relativen Häufigkeit von Wörtern im Referenzkorpus)

### II. Vergleich mit Berücksichtigung von Wortverteilungen

1. **tf·idf** (term frequency/inverse document frequency): Gewichtung der Häufigkeit von Wörtern im Analyse- und Referenzkorpus unter Berücksichtigung ihrer Verteilung
2. **Statistischer Hypothesentest** (t-Test): Vergleich von Häufigkeiten im Analyse- und Referenzkorpus unter Berücksichtigung ihrer Verteilung im Sinne eines statistischen Tests

### 3. Dispersionsmaße (wie Burrows Delta und Zeta): Vergleich der konsistenten Verwendung von Wörtern im Analyse- und Referenzkorpus

In den Digitalen Geisteswissenschaften werden seit den Arbeiten von John Burrows im Rahmen der sog. **Stylometrie** für die Analyse von Autorenschaften vor allem Verfahren wie der t-Test oder Burrows Zeta für Wortvergleiche und Relevanzbewertungen von Wörtern verwendet. Eine gute Einführung in diese Verfahren nebst einem R-Programmpaket, das auch für andere Text-Mining-Anwendungen geeignet ist, findet sich in Eder et al. (2016), weswegen diese Verfahren hier nur kurz vorgestellt werden. Eine vergleichende Evaluation verschiedener Verfahren findet sich u. a. in Lijfijt et al. (2014).

#### 5.9.2.1 Log-likelihood-Ratio

Um zu ermitteln, ob die Frequenz  $k_1$  eines Wortes  $w$  in einem Analysekörper der Länge  $n_1$  statistisch signifikant über der Frequenz  $k_2$  liegt, die aufgrund seiner Frequenz im Referenzkörper der Länge  $n_2$  zu erwarten gewesen wäre, wird bei der Log-likelihood-Ratio die relative Häufigkeit  $p_1$  des Wortes  $w$  im Referenzkörper mit ihrer relativen Häufigkeit  $p_2$  im Analysekörper verglichen. Als sog. *Nullhypothese* wird dabei angenommen, dass zwischen beiden kein Unterschied besteht, dass also  $p_1 = p_2$ .

Beim Vergleich zwischen dem Analyse- und Referenzkörper wird zunächst für jedes Wort  $w$  ihre relative Häufigkeit im Analysekörper und Referenzkörper ermittelt:

$$p_1 = k_1/n_1 \text{ und } p_2 = k_2/n_2 \quad (\text{Gl. 5.18})$$

Eine angemessene Modellierung der Wahrscheinlichkeit, dass  $w$  genau  $k$ -mal in einem Text der Länge  $n$  auftritt, ist nach Dunning (1993) die Binomialverteilung, d. h.

$$p(n, k) = \binom{n}{k} p^k (1-p)^{n-k} \quad (\text{Gl. 5.19})$$

#### Binomialverteilung von Wörtern

Für die Abschätzung der Wahrscheinlichkeit, dass ein Wort  $w$  genau  $k$ -mal in einem Text der Länge  $n$  auftritt, nehmen wir an, dass wir jedes der  $n$  Wörter betrachten und entscheiden, ob das betrachtete Wort  $x$  identisch ist mit  $w$ , also ob  $x=w$  gilt, oder nicht. Da der Text insgesamt die Länge  $n$  hat, gibt es  $n$  solcher Entscheidungen, wovon  $k$  positiv sind und  $n-k$  negativ. Die Wahrscheinlichkeit, dass  $w$  genau  $k$ -mal auftritt, ist  $p^k$ , die Wahrscheinlichkeit des gegenteiligen Ereignisses entsprechend  $(1-p)^{n-k}$ . Da jede der  $n$  Entscheidungen positiv sein kann, muss das Ergebnis noch multipliziert werden mit der Anzahl von Möglichkeiten, dass bei  $n$  Entscheidungen  $k$  positive enthalten sind  $\binom{n}{k}$ .

Wenn wir die Verteilung von Wörtern in einem Text mit der Binomialverteilung modellieren, setzen wir voraus, dass 1) jede Entscheidung unabhängig ist von allen

vorangehenden und dass 2) das Wort  $w$  an jeder Stelle des Textes gleich wahrscheinlich ist. Im Hinblick darauf, dass die Wahrscheinlichkeit des Auftretens eines Wortes stark davon abhängt, welches Thema behandelt wird (vgl. Abschn. 6.4) und dass zwischen Wörtern oft Abhängigkeiten bestehen, ist dies nicht ganz korrekt, kann jedoch für unsere Zwecke vernachlässigt werden. ◀

Die Likelihood-Funktion  $H$  ermöglicht einen Vergleich des Analyse- mit dem Referenzkorpus:  $H$  beschreibt die Wahrscheinlichkeit, dass ein Wort  $w$  in einem Analysekörper der Länge  $n_1$  genau  $k_1$ -mal und im Referenzkorpus der Länge  $n_2$  genau  $k_2$ -mal gesehen wird unter der Voraussetzung, dass die Auftretenswahrscheinlichkeit im Analysekörper durch  $p_1$  und die im Referenzkorpus durch  $p_2$  gegeben ist.

Die Nullhypothese lautet:  $p_1 = p_2 = p$ , d. h.  $w$  ist in beiden Texten gleichwahrscheinlich.

Um die Abweichung der Wahrscheinlichkeit von  $w$  im Analysekörper vom Referenzkorpus zu messen, setzen wir  $p_1 = k_1/n_1$ ,  $p_2 = k_2/n_2$  und  $p = (k_1 + k_2)/(n_1 + n_2)$  und berechnen die **Likelihood-Ratio  $\lambda$**  als den Quotienten aus der beobachteten Anzahl von  $w$  im Analysekörper und der als Nullhypothese aus dem Referenzkorpus abgeleiteten erwarteten Anzahl von  $w$ . Nach Einsetzen und Umformen erhalten wir die Prüfgröße  $-2 \log \lambda$  (mit  $L(p, k, n) = p^k(1-p)^{n-k}$ ), einer für die Zwecke des Korpusvergleichs angepassten Formulierung des Log-Likelihood-Maßes, wie wir es bereits in Abschn. 5.3 kennengelernt haben:

$$2 \log \lambda = 2 [\log L(p_1, k_1, n_1) + \log L(p_2, k_2, n_2) - \log L(p, k_1, n_1) - \log L(p, k_2, n_2)] \quad (\text{Gl. 5.21})$$

Die Differenzanalyse besteht also darin, alle Wörter, für die  $-2 \log \lambda$  groß genug ist (und die mit einer gewissen Mindestfrequenz auftreten), herauszufiltern.

### Beispiel

Die Likelihood-Ratio eignet sich zum Vergleich von literarischen Texten eines Autors oder einer Autorin und von Zeitungstexten aus derselben Zeit der Veröffentlichung der literarischen Texte. In einer literaturwissenschaftlichen Studie wurden die auffälligen Wörter in Schriften von Ernst Jünger aus dem Jahr 1929 als Analysekörper im Vergleich zu Schriften von ihm aus dem Jahr 1921 sowie der deutschen Tagespresse aus dem Jahr 1929 jeweils als Referenzkorpora mithilfe der Differenzanalyse ermittelt (vgl. Goldhahn et. al. 2015).

Zu den Wörtern, die sich bei dem Vergleich der Schriften von Jünger aus dem Jahr 1929 und dem Jahr 1921 nur in den Schriften aus dem Jahr 1929 finden, zählen u. a.: *Nationalismus, Lebenswelt, Roman, Deutschland, Nation, Seite, Verantwortung* und *Revolution*. Die 10 auffälligsten Wörter, die sich im Sinne der Differenzanalyse häufiger in den Schriften von 1921 finden, sind: *Toten, Maschine, Appell, Kompagnie, Stolz, Feind, Meister, Pflicht, Material, Überlebenden*.

Im Vergleich zwischen den Schriften von Jünger aus dem Jahr 1929 und der deutschen Tagespresse finden sich die folgenden 10 Wörter häufiger in den Schriften von Jünger als in der Tagespresse: *Nationalismus, Liberalismus, Gestalten, Erstaunen, Erlebnis, Bestände, Bindungen, Schärfe, Chaos, Unruhe.* ◀

### 5.9.2.2 tf-idf (term frequency/inverse document frequency)

Das Maß **Term-Frequenz/Inverse-Dokument-Frequenz** (tf-idf) wurde von Salton und Yang (1973) für die Indexierung von Wörtern im Information Retrieval eingeführt. Es berücksichtigt folgenden naheliegenden Aspekt der Verteilung von Wörtern in einer Dokumentkollektion, der auf Karen Sparck-Jones (1972) zurückgeht: Wörter, die nur in wenigen Dokumenten, dort aber häufig vorkommen, sind spezifischer für diese Dokumente, als Wörter, die insgesamt häufiger, dafür aber in sehr vielen Dokumenten vorkommen.

Die Termfrequenz  $f_{ik}$  misst dabei, wie häufig ein Wort  $k$  im Dokument  $i$  vorkommt, während die **inverse Dokumentfrequenz**  $idf_k$  eines Wortes  $k$  berechnet wird als:

$$idf_k = \log N/d_k \quad (\text{Gl. 5.22})$$

$N$  ist dabei die Gesamtzahl der Dokumente in der Kollektion und  $d$  die Anzahl der Dokumente, in denen das Wort  $k$  auftritt. Die  $idf$  wächst also, wenn das Wort  $k$  in wenigen Dokumenten auftritt. Salton verwendet das Produkt  $w_{ik} = f_{ik} idf_k$  als Gewicht für die Spezifität eines Wortes  $k$  im Dokument  $i$ .

Eine Differenzanalyse von Dokumentkollektionen mithilfe des Maßes tf-idf ermöglicht es, statistisch signifikante Unterschiede in der Verwendung von Wörtern zu ermitteln, indem solche Wörter ausgewählt werden, die im Analysekörper häufig, sonst aber selten auftreten. Dabei sichert eine hohe Termfrequenz im Analysekörper die Repräsentativität eines Wortes für das Analysekörper, die inverse Dokumentfrequenz bemisst die statistische Signifikanz in der unterschiedlichen Verwendung dieser Wörter in Bezug auf das Referenzkorpus, da Wörter bevorzugt werden, die im Referenzkorpus selten sind – so wie bei der  $idf$  (vgl. Robertson 2004).

In der Praxis ist dieses Verfahren vor allem dann relevant, wenn die Texte im Analysekörper meist sehr kurz sind, so wie beispielsweise Kurznachrichten in sozialen Medien. Wird als Referenzkorpus dann ein allgemeiner Wortschatz wie z. B. die Daten des Projekts Deutscher Wortschatz gewählt, so können auch für sehr kurze Texte statistisch auffällige Schlüsselwörter extrahiert werden.

### 5.9.2.3 Statistische Hypothesentests und Burrows Zeta

Statische Hypothesentests – wie der **t-Test** (Hoover 2010) – liefern eine Entscheidungshilfe dafür, ob ein gefundener Mittelwertsunterschied rein zufällig entstanden ist, oder ob es der Sache nach bedeutsame Unterschiede zwischen den zwei untersuchten Gruppen gibt. Aus Sicht der Statistik beurteilt dieses Verfahren, ob sich zwei untersuchte Gruppen systematisch in ihren Mittelwerten unterscheiden oder nicht. Eine Anwendung dieser Tests auf Textdaten von Autoren und Autorinnen wurde Anfang der 1990er Jahre u. a. von John Burrows erprobt, um zu klären, ob ein Text von ein und demselben oder ver-

schiedenen Autoren oder Autorinnen erstellt worden ist. Hierzu sind wieder im Sinne der Differenzanalyse ein Analysekorpus mit einem Referenzkorpus verglichen und untersucht worden, ob die beobachteten Abweichungen zwischen den beiden Dokumentkollektionen zufälliger Natur sind oder nicht.

Da bei diesen textorientierten Anwendungen des t-Test hochfrequente, aber wenig spezifische Wörter stärker gewichtet werden, was oft zu falschen Ergebnissen führt, wird im Rahmen der sog. Stylometrie meist das von John Burrows entwickelte Maß **Zeta** (Burrows 2006) angewendet, das auf dem t-Test aufbaut und diesen verfeinert. Das Grundprinzip dieses Maßes ist es, vor der Berechnung die Texte in kleinere Segmente aufzuteilen, wobei die Segmentlänge ein wichtiger Parameter ist. Dann wird für jedes Merkmal der Anteil der Segmente erhoben, in denen das Merkmal mindestens einmal vorkommt (die „document proportion“). Von diesem Anteil in der untersuchten Gruppe wird der entsprechende Anteil in der Vergleichsgruppe subtrahiert, woraus sich der Zeta-Wert ergibt. Eine kompakte Einführung in dieses Verfahren der Differenzanalyse und eine umfangreiche Implementierung in R findet sich bei Eder et.al. (2016), für eine Erläuterung seiner statistischen Grundlagen sei auf Argamon (2006) verwiesen.

### 5.9.3 Kontextvergleiche

Neben der Frequenz von Wörtern, stellt der Verwendungskontext von Wörtern eine weitere Quelle an Informationen dar, da der Kontext eines Wortes wichtige Hinweise auf seine Bedeutung gibt (vgl. Kap. 2, Abschn. 2.1 und 2.4). Beim Vergleich von Textkorpora können Unterschiede im Verwendungskontext eines Wortes insbesondere auf unterschiedliche Sachgebiete, Genres und Medien hinweisen (synchrone Betrachtung), sie können aber auch Indizien für Bedeutungsveränderungen sein, ausgelöst etwa durch politische Ereignisse oder technologischen Fortschritt (diachrone Betrachtung).

Grundlage des Kontextvergleichs von Wörtern sind ihre Verwendungskontexte im Referenz- und Analysekorpus. Diese Verwendungskontexte können mithilfe von **Embeddings** (vgl. Abschn. 5.6) oder mithilfe von **Kookkurrenzen** (vgl. Abschn. 5.3) dargestellt werden. Darauf aufbauend ergeben sich verschiedene Verfahren des Kontextvergleichs, die im Folgenden umrissen werden.

#### 5.9.3.1 Embeddingbasierte Verfahren

Word Embeddings (Worteinbettungen) sind Abbildungen von Wörtern in einen Vektorraum, wobei die räumliche Anordnung der Wörter in der Regel durch die Verwendungskontexte der Wörter, z. B. Wort-N-Grammen (Abschn. 5.6), bestimmt ist. Für den Kontextvergleich werden Embeddings sowohl für das Referenz- als auch das Analysekorpus berechnet. Für eine zuvor festgelegte Menge von relevanten Schlüsselbegriffen werden anhand der Embeddingrepräsentationen pro Korpus die ähnlichsten Begriffe identifiziert, in der Regel Synonyme oder zumindest ähnlich verwendete Wörter, welche die Bedeutung des Ausgangswortes im jeweiligen Korpus genauer aufschlüsseln. Hieraus

lassen sich Gemeinsamkeiten bzw. Unterschiede im Verwendungskontext ableiten (vgl. Scharloth und Bubenhofer 2011). Finden sich z. B. in den Embeddings ähnlicher Wörter aus dem Referenz- und Analysekorpus viele Überschneidungen, so ist von einem ähnlichen Verwendungskontext und somit ähnlichem Konzept auszugehen. Liegen nur sehr wenige Überschneidungen vor, ist dies ein deutlicher Indikator für eine starke Kontextänderung des Ausgangswortes.

Werden die Embeddings für das Referenz- und Analysekorpus nicht getrennt berechnet, sondern die Kontextveränderungen von Embeddings in deren Erstellung mit einbezogen, so spricht man von dynamischen Embeddings (vgl. Bamler und Mandt 2017).

### 5.9.3.2 Kookkurrenzstatistik

Die Kookkurrenten eines Wortes sind diejenigen Wörter, mit denen es statistisch signifikant häufig in textueller Nachbarschaft auftritt (vgl. Abschn. 5.3). Der Vergleich von Verwendungskontexten eines Wortes in einem Referenz- und einem Analysekorpus auf Basis von Kookkurrenten kann dabei grundsätzlich genauso erfolgen, wie der Vergleich von Embeddings, d. h. es werden die Gemeinsamkeiten und Unterschiede von Kookkurrenten betrachtet.

## Korpusvergleich mithilfe von Kookkurrenzen

Die Abb. 5.20 zeigt die Kontextveränderung des Wortes „Krieg“ im Vergleich der Jahre 2004 und 2016 aus Daten der Tageszeitung taz: Die rot gefärbten Wörter in der

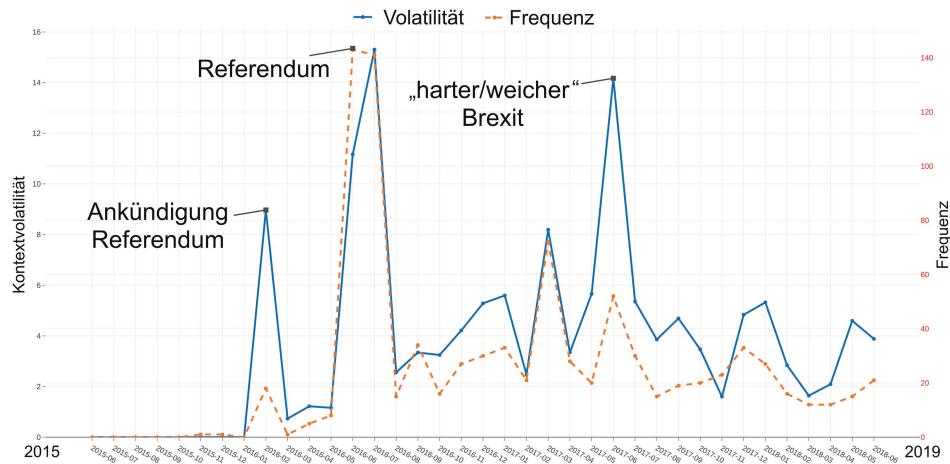


**Abb. 5.20** Kontextveränderung des Wortes „Krieg“ im Vergleich der Jahre 2004 und 2016 aus Daten der Tageszeitung taz

Abb. 5.20 sind stark signifikante Kookkurrenzen des Ausgangswortes im Jahr 2004, die grün gefärbten aus dem Jahr 2016. Der Vergleich verdeutlicht die Veränderung in der Berichterstattung über kriegerische Ereignisse in den Jahren 2004 und 2016. ◀

Eine Möglichkeit zur interaktiven Analyse von Kontextveränderungen bietet die Software DiaCollo (<https://clarin-d.de/de/kollokationsanalyse-in-diachroner-perspektive>). Für festgelegte Suchbegriffe werden die Kookkurrenzen (dort Kollokationen genannt) im Deutschen Textarchiv der BBAW über einen Zeitraum von mehreren 100 Jahren ermittelt und visualisiert. Allein anhand dieser Darstellung können bereits zahlreiche sprachliche Veränderungen, beispielsweise ausgelöst durch historische Ereignisse, nachgewiesen werden.

Um herauszufinden, welche Wörter beim Vergleich verschiedener Korpora entweder besonders starke oder besonders geringe Kontextveränderungen aufweisen, müssen die Kontextveränderungen für alle Wörter ermittelt werden. Eine Möglichkeit zur Quantifizierung von Kontextänderungen stellt das Maß der **Kontextvolatilität** dar (vgl. Kahmann 2021). Ziel dieses Maßes ist die genaue Bestimmung des Grades an Kontextänderung für eine definierte Menge von Zeitpunkten  $t_r$ . Vereinfacht gesagt, entspricht die Berechnung der Kontextvolatilität einer diachronen Kookkurrenzanalyse, bei der auf Basis eines Gewichtungsfaktors, einer Gedächtnisgröße  $h$  und einem Zeitintervall eine Vorhersage für den Kontext eines einzelnen Wortes anhand von Referenzzeitpunkten  $t-h, \dots, t-1$  berechnet wird. Um die Kontextänderung eines Zielwertes zu einem bestimmten Zeitpunkt  $t$  zu bestimmen, werden die davorliegenden Zeitpunkte  $t-h, \dots, t-1$  herangezogen. Die Menge der zurückliegenden Zeitpunkte wird über den Parameter  $h$  quantifiziert. Für jede Kookkurrenz  $i$  des Zielwertes wird anhand der gewichteten Referenzzeitpunkte  $t-h, \dots, t-1$  ein Signifikanzwert  $E_{ti}$  berechnet. Hierfür werden für jedes Wort seine Kookkurrenzen für den betrachteten Zeitraum bestimmt und zusammen mit ihrer statistischen Signifikanz in Form eines Vektors dargestellt. Dieser Vektor mit erwarteten Signifikanzwerten für die Kookkurrenzen des Ausgangswortes wird dann mit den tatsächlich zum Zeitpunkt  $t$  vorliegenden Kookkurrenzen und ihren Signifikanzwerten verglichen. Der Vergleich der beiden Vektoren kann beispielsweise durch die Verwendung der **Kosinusähnlichkeit** erfolgen. Liegt eine große Abweichung zwischen Erwartungswert und den zum Zeitpunkt  $t$  tatsächlich vorliegenden Daten vor, so ist von einer großen Verschiebung des Kontextes auszugehen. Sind die beiden identisch, ist der Kontext stabil geblieben. Nach erfolgreicher Berechnung für alle Wörter können die Kontext-Volatilitätsverlaufskurven über den gesamten vorliegenden Zeitraum eingesehen und dabei besonders auffällige Zeitpunkte identifiziert werden. Weiterhin ist es möglich, für jeden Zeitpunkt zu analysieren, welche einzelnen Worte und zugehörige Themenfelder eine hohe Kontextänderung hervorrufen. Damit können nicht nur Zeitpunkte kontextueller Änderungen bestimmt, sondern auch mögliche Ursachen dafür erkannt und weiter untersucht werden. Darüber hinaus ist es auch möglich, die Kontext-Volatilitätskurven verschiedener Begriffe zu vergleichen und



**Abb. 5.21** Verlaufskurven für das Wort „Brexit“ in der Tageszeitung taz im Zeitraum 2015–2018.  
(Quelle: Kahmann 2021)

damit besonders volatile aber auch besonders stabile Konzepte zu identifizieren (vgl. Abschn. 7.6).

#### Kontextvolatilität und Frequenzanalyse

Die Abb. 5.21 zeigt die Verlaufskurven für das Wort „Brexit“ in der Tageszeitung taz im Zeitraum 2015–2018.

Im Unterschied zum reinen Frequenzverlauf lässt der Verlauf der Kontextvolatilität drei Schlüsselereignisse erkennen: die Ankündigung des Referendums, das eigentliche Referendum sowie die Diskussion über die Art des Brexits (hart/weich). Die Frequenz dagegen hat lediglich einen deutlichen Peak zum eigentlichen Zeitpunkt des Referendums. (vgl. Kahmann 2021). ◀

Im Vergleich zu rein frequenzbasierten Verfahren zeichnet sich die Kontextvolatilität dadurch aus, dass sie auch bei niedrigfrequenten Wörtern anwendbar ist. Die Änderung des Kontextes kann unabhängig von der Frequenz des Ausgangswortes ermittelt werden und damit frühzeitig Aufschluss über semantische Verschiebungen im Verwendungskontext des Ausgangswortes geben. Oftmals geht die Kontextänderung von Wörtern auch mit Änderungen in deren Frequenz einher. Dies ist jedoch nicht immer der Fall. In Situationen, in denen die Kontextänderung mit gleichzeitiger konstanter Frequenz des Wortes verläuft, ist die Detektion dieser Änderungen auf Basis der Frequenz allein nicht möglich. Erst die Quantifizierung der Änderungsrate des Kontextes erlaubt es, solche Änderungen zu erkennen. Weiterhin kann die Kontextvolatilität auf verschiedene Zeitfenster (Jahre, Monate, Wochen, Tage, ...) eingestellt werden. Dies erlaubt es, sowohl langsam schleichende Änderungen über große Zeiträume als auch spontane Ver-

schiebungen in kurzen Intervallen aufzudecken. Im Gegensatz zu embeddingbasierten Verfahren wird bei der Kontextvolatilität die Transparenz der Ergebnisse sichergestellt, indem jederzeit die einzelnen Kookkurrenzen mitsamt ihren berechneten Signifikanzwerten und deren Änderungen innerhalb verschiedener Zeitfenster einsehbar sind.

---

## Literatur

- Akbik, A., Blythe, D., Vollgraf, R.: Contextual string embeddings for sequence labeling. In: Bender, E.M., Derczynski, L., Isabelle, P. (Hrsg.) Proceedings of the 27th International Conference on Computational Linguistics. COLING, Santa Fe, NM, USA, S. 1638–1649. Association for Computational Linguistics, Stroudsburg, PA, USA. <https://aclanthology.org/C18-1139/> (2018) Zugegriffen: 29. Sept. 2021
- Allen, C., Hospedales, T.: Analogies explained: Towards understanding word embeddings. Proceedings of the 36th International Conference on Machine Learning, PMLR **97**, 223–231 (2019)
- Argamon, S.: Interpreting Burrows's Delta: Geometric and probabilistic foundations. Literary and linguistic computing **23**(2), 131–147 (2006). <https://doi.org/10.1093/lrc/fqn003>
- Bakarov, A.: A survey of word embeddings evaluation methods. <https://arxiv.org/pdf/1801.09536> (2018). Zugegriffen: 6. Apr. 2021
- Bamler, R., Mandt, S.: Dynamic Word Embeddings. Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia. PMLR **70**, 380–389 (2017)
- Baroni, M., Dinu, G., Kruszewski, G.: Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Baltimore, MD, USA, S. 238–247. Association for computational linguistics, Stroudsburg, PA, USA, <https://aclanthology.org/S.14-1023/> (2014). Zugegriffen: 29. Sept. 2021
- Bartunov, S., Kondrashkin, D., Osokin, A., Vetrov, D.: Breaking sticks and ambiguities with adaptive skip-gram. In: Gretton, A., Robert, C.C. (Hrsg.) Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (AISTATS'2016) Cadiz, Spain, S. 130–138 (2016)
- Bibliographisches Institut: Durchschnittliche Länge eines deutschen Wortes. <https://www.duden.de/sprachwissen/sprachratgeber/Durchschnittliche-Länge-eines-deutschen-Wortes> (2021). Zugegriffen: 22. März 2021
- Biemann, C., Riedl, M.: Text: Now in 2D! A framework for lexical expansion with contextual similarity. Journal of Language Modelling **1**(1), 55–95 (2013). <https://doi.org/10.15398/jlm.v1i1.60>
- Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching Word Vectors with Subword Information. Transactions of the Association for Computational Linguistics (TACL) **5**, 135–146 (2017)
- Brants, T., Franz, A.: Web 1T 5-gram Version 1 LDC2006T13. Web Download. Linguistic Data Consortium, Philadelphia (2006)
- Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D.: Language Models are Few-Shot Learners. Proc. Advances in Neural Information Processing Systems (NeurIPS), S. 1877–1901. Curran Associates, Inc. <https://proceedings.neurips.cc/>

- [paper/2020/file/1457c0d6bfc4967418fb8ac142f64a-Paper.pdf](https://paper/2020/file/1457c0d6bfc4967418fb8ac142f64a-Paper.pdf) (2020). Zugegriffen: 6. Apr. 2021
- Burrows, J.: All the Way Through: Testing for authorship in different frequency strata. *Literary and linguistic computing* **22**(1), 27–47 (2006). [https://doi.org/10.1093/lcc/fqi067](https://doi.org/10.1093/llc/fqi067)
- Cecchini, F.M., Riedl, M., Fersini, E., Biemann, C.: A comparison of graph-based word sense induction clustering algorithms in a pseudoword evaluation framework. *Lang. Resour. Eval.* **52**(3), 733–770 (2018). <https://doi.org/10.1007/s10579-018-9415-1>
- Chen, M.: Efficient Vector Representation for Documents through Corruption. In: Proceedings of the 5th International Conference on Learning Representations, ICLR 2017, Toulon, France (2017)
- Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. *Journal of the American Society for Information Science* **41**(6), 391–407 (1990)
- Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In: Burstein, J., Doran, C., Solorio, T. (Hrsg.) *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, MN, S. 4171–4186 Association for Computational Linguistics, Stroudsburg, PA (2019). <https://doi.org/10.18653/v1/N19-1423>
- Dunning, T.: Accurate methods for the statistics of surprise and coincidence. *Comput. Linguist.* **19**(1), 61–74 (1993)
- Eckart, T., Hallsteinsdóttir, E., Quasthoff, U.: *Frequency Dictionary German - Häufigkeitswörterbuch Deutsch*. Quasthoff, U., Fiedler, S., Hallsteindóttir, E. (Hrsg.). Leipziger Universitätsverlag, Leipzig (2011)
- Eckart, T., Quasthoff, U., Goldhahn, D.: The Influence of Corpus Quality on Statistical Measurements on Language Resources. In: Calzolari, N., Choukri, K., Declerck, T., Dogan, M.U., Maegaard, B., Marian, J., Moreno, A., Odijk, J., Piperidis, S. (Hrsg.) *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, S. 2318–2321. European Language Resources Association (ELRA). [http://www.lrec-conf.org/proceedings/lrec2012/pdf/476\\_Paper.pdf](http://www.lrec-conf.org/proceedings/lrec2012/pdf/476_Paper.pdf) (2012). Zugegriffen: 8. Juni 2021
- Eder, M., Rybicki, J., Kestemont, M.: Stylometry with R: A package for computational text analysis. *The R Journal* **8**(1), 107–121 (2016)
- Elsafty, A., Riedl, M., Biemann, C.: Document-based recommender system for Job postings using dense representations. In: Bangalore, S., Chu-Carroll, J., Li, Y. (Hrsg.) *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)* New Orleans, LA, USA, S. 216–224. Association for Computational Linguistics, Stroudsburg, PA, USA, <https://aclanthology.org/N18-3027> (2018) Zugegriffen: 29. Sept. 2021
- Evert, S., Krenn, B.: Methods for the qualitative evaluation of lexical association measures. In: Webber, B.L. (Hrsg.) *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL)*, Toulouse, France, S. 188–195. Association for Computational Linguistics, Morristown, NJ, USA (2001). <https://doi.org/10.3115/1073012.1073037>
- Fu, R., Guo, J., Qin, B., Che, W., Wang, H., Liu, T.: Learning semantic hierarchies via word embeddings. In: Toutanova, K., Wu, H. (Hrsg.) *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Baltimore, MD, USA, S. 1199–1209. Association for Computational Linguistics, Stroudsburg, PA, USA (2014). <https://doi.org/10.3115/v1/S.14-1113>

- Furnas, G.W., Landauer, T.K., Gomez, L.M., Dumais, S.T.: The vocabulary problem in human-system communication. *Commun. ACM* **30**(11), 964–971 (1987). <https://doi.org/10.1145/32206.32212>
- Gale, W.A., Church, K.W., Yarowsky, D.: One sense per discourse. In: Proceedings of the workshop on Speech and Natural Language, Harriman, New York, S. 233–237. Kaufmann, San Mateo, CA, USA (1992). <https://doi.org/10.3115/1075527.1075579>
- Goldhahn, D., Eckart, T., Gloning, T., Dreßler, K., Heyer, G.: Operationalisation of research questions of the humanities within the CLARIN Infrastructure – An Ernst Jünger Use Case. In: CLARIN Annual Conference 2015 in Wrocław, Poland (2015)
- Haase, C., Anwar, S., Yimam, S.M., Friedrich, A., Biemann, C.: SCOT: Sense Clustering over Time: a tool for the analysis of lexical change. In: Gkatzia, D., Seddah, D. (Hrsg.) Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics - System Demonstrations. Online, S. 198–204. Association for Computational Linguistics, Stroudsburg, PA, USA. <https://www.aclweb.org/anthology/2021.eacl-demos.23.pdf> (2021). Zugegriffen: 20. Mai 2021
- Harris, Z.S.: Methods in structural linguistics. University of Chicago Press, Chicago. <http://archive.org/details/structurallingui00harr> (1951). Zugegriffen: 11. März 2021
- Hoover, D.L.: Teasing out Authorship and Style with T-Tests and Zeta. In: Digital Humanities Conference. ADHO, London. <http://dh2010.cch.kcl.ac.uk/academic-programme/abstracts/papers/html/ab-658.html> (2010). Zugegriffen: 12. Apr. 2021
- Jawahar, G., Sagot, B., Seddah, D.: What Does BERT Learn about the Structure of Language? In: Korhonen, A., Traum, D., Márquez, L. (Hrsg.) Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, S. 3651–3657. Association for Computational Linguistics, Stroudsburg, PA, USA, (2019). <https://doi.org/10.18653/v1/S.19-1356>
- Kahmann, C.: Design und Implementierung eines Software-Ökosystems für textbasierte Inhaltsanalysen in den Sozialwissenschaften mit Schwerpunkt auf der Detektion schwacher Signale. Dissertation, Universität Leipzig (2021)
- Kilgarriff, A.: I don't believe in word senses. *Comput. Humanit.* **31**(2), 91–113 (1997)
- Kilgarriff, A.: Comparing Corpora. *International Journal of Corpus Linguistics* **6**(1), 97–133 (2001). <https://doi.org/10.1075/ijcl.6.1.05kil>
- Kilgarriff, A.: How Dominant Is the Commonest Sense of a Word? In: Sojka, P., Kopeček, I., Pala, K. (Hrsg.) Text, Speech and Dialogue. 7th International Conference, TSD 2004, Brno, Czech Republic. Lecture notes in computer science, Bd. 3206, S. 103–111. Springer, Berlin (2004). [https://doi.org/10.1007/978-3-540-30120-2\\_14](https://doi.org/10.1007/978-3-540-30120-2_14)
- Kneser, R., Ney, H.: Improved back-off for m-gram language modeling. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, S. 181–184. IEEE Computer Society (1995)
- Kutuzov, A., Øvrelid, L., Szymanski, T., Velldal, E.: Diachronic word embeddings and semantic shifts: a survey. In: Bender, E.M., Derczynski, L., Isabelle, P. (Hrsg.) Proceedings of the 27th International Conference on Computational Linguistics. COLING, Santa Fe, NM, USA, S. 1384–1397. Association for Computational Linguistics <https://www.aclweb.org/anthology/C18-1117.pdf> (2018). Zugegriffen: 8. Apr. 2021
- Language Technology Group: JoBimText. <http://ltmaggie.informatik.uni-hamburg.de/jobmviz/> (2017). Zugegriffen: 2. März 2021
- Le, Q., Mikolov, T.: Distributed representations of sentences and documents. ICML'14: Proceedings of the 31st International Conference on Machine Learning. PMLR **32**(2), 1188–1196 (2014)

- Levy, O., Goldberg, Y.: Neural word embedding as implicit matrix factorization. In: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., Weinberger, K.Q. (Hrsg.) NIPS'14: Proceedings of the 27th International Conference on Neural Information Processing Systems – Bd. 2, Montreal, Canada, S. 2177–2185. MIT Press, Cambridge, MA, USA (2014)
- Lijffijt, J., Nevalainen, T., Säily, T., Papapetrou, P., Puolamäki, K., Mannila, H.: Significance Testing of Word Frequencies in Corpora. Digital Scholarship in the Humanities **31**(2), 374–397 (2014). <https://doi.org/10.1093/lhc/fqu064>
- Lops, P., de Gemmis, M., Semeraro, G.: Content-based recommender systems: State of the art and trends. In: Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (Hrsg.) Recommender Systems Handbook, S. 73–105. Springer, New York (2010)
- Mandelbrot, B.B.: An information theory of the statistical structure of language. In: Jackson, W. (Hrsg.) Communication Theory, S. 503–512. Academic, New York (1953)
- Manning, C.D., Schütze, H.: Foundations of statistical natural language processing, 8. Aufl. MIT Press, Cambridge, MA, USA (1999)
- Mihalcea, R., Chklovski, T., Kilgarriff, A.: The SENSEVAL-3 English lexical sample task. In: Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text, Barcelona, Spain, S. 25–28. Association for Computational Linguistics, Stroudsburg, PA, USA. <https://aclanthology.org/W04-0807> (2004) Zugriffen: 29. Sept. 2021
- Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. ICLR Workshop. <https://arxiv.org/pdf/1301.3781> (2013). Zugriffen: 6. Apr. 2021
- Mikolov, T., Karafiat, M., Burget, L., Černocky, J., Khudanpur, S.: Recurrent neural network based language model. In: Proceedings of the 11th annual conference of the International Speech Communication Association 2010 (INTERSPEECH 2010), Makuhari, Chiba, Japan, S. 1045–1048. International Speech Communication Association (2010)
- Miller, G.A., Charles, W.G.: Contextual correlates of semantic similarity. Lang. Cognit. Process. **6**(1), 1–28 (1991). <https://doi.org/10.1080/01690969108406936>
- Neelakantan, A., Shankar, J., Passos, A., McCallum, A.: Efficient non-parametric estimation of multiple embeddings per word in vector space. In: Moschitti, A., Pang, B., Daelemans, W. (Hrsg.) Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, S. 1059–1069. Association for Computational Linguistics, Stroudsburg, PA, USA (2014). <https://doi.org/10.3115/v1/D14-1113>
- Padó, S., Lapata, M.: Dependency-based construction of semantic space models. Comput. Linguist. **33**(2), 161–199 (2007). <https://doi.org/10.1162/coli.2007.33.2.161>
- Pedersen T.: Unsupervised Corpus-Based Methods for WSD. In: Agirre, E., Edmonds, P. (Hrsg.) Word Sense Disambiguation. Text, Speech and Language Technology, Bd. 33. Springer, Dordrecht (2007). [https://doi.org/10.1007/978-1-4020-4809-8\\_6](https://doi.org/10.1007/978-1-4020-4809-8_6)
- Pelevina, M., Arefiev, N., Biemann, C., Panchenko, A.: Making Sense of Word Embeddings. In: Blunsom, P., Cho, K., Cohen, S., Grefenstette, E., Hermann, K.M., Rimell, L., Weston, J., Yih, S.W. (Hrsg.) Proceedings of the 1st Workshop on Representation Learning for NLP, Berlin, Germany, S. 174–183. Association for Computational Linguistics, Stroudsburg, PA, USA. (2016). <https://doi.org/10.18653/v1/W16-1620>
- Pennington, J., Socher, R., Manning, C.D.: GloVe: Global Vectors for Word Representation. In: Moschitti, A., Pang, B., Daelemans, W. (Hrsg.) Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, S. 1532–1543. Association for Computational Linguistics, Stroudsburg, PA, USA (2014). <https://doi.org/10.3115/v1/D14-1162>
- Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. In: Walker, M., Ji, H., Stent, A. (Hrsg.) Proceedings of

- the 2018 conference of the North American Chapter of the Association for Computational Linguistics (NAACL), New Orleans, LA, USA, S. 2227–2237. Association for Computational Linguistics, Stroudsburg, PA, USA (2018). <https://doi.org/10.18653/v1/N18-1202>
- Quasthoff, U., Biemann, C.: Measuring Monolinguality. Proceedings of the LREC-06 workshop on Quality Assurance and Quality Measurement for Language and Speech Resources, Genoa, Italy (2006)
- Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding with unsupervised learning. Technical report, OpenAI. <https://openai.com/blog/language-unsupervised/> (2018). Zugegriffen: 6. Apr. 2021
- Rapp, R.: Die Berechnung von Assoziationen: Ein korpuslinguistischer Ansatz. Olms; Hildesheim, Zürich, New York (1996)
- Rayson, P., Garside, R.: Comparing corpora using frequency profiling. Proceedings of the Workshop on Comparing Corpora, Hong Kong, China, S. 1–6. Association for Computational Linguistics, Morristown, NJ, USA (2000). <https://doi.org/10.3115/1117729.1117730>
- Robertson, S.: Understanding inverse document frequency: On theoretical arguments for IDF. *Journal of Documentation* **60**(5), 503–520 (2004)
- Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. In: *Neurocomputing: foundations of research*, S. 696–699. MIT Press, Cambridge, MA, USA (1988)
- Ruppert, E., Kaufmann M., Riedl M., Biemann C.: JoBimViz: A web-based visualization for graph-based distributional semantic models. In: Hsin-His Chen, N.T.U., Markert, K. (Hrsg.) *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, Beijing, China, S. 103–108. ACL and the Asian Federation of NLP, Stroudsburg, PA, USA. <https://www.aclweb.org/anthology/S.15-4018> (2015). Zugegriffen: 2. März 2021
- Salton, G.: Automatic text processing. The transformation, analysis, and retrieval of information by computer. Addison-Wesley, Reading, MA (1989)
- Salton, G., Yang, C.S.: On the specification of term values in automatic indexing. *Journal of Documentation* **29**(4), 351–372 (1973)
- Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. *Commun. ACM* **18**(11), 613–620 (1975). <https://doi.org/10.1145/361219.361220>
- de Saussure, F.: Grundfragen der allgemeinen Sprachwissenschaft, 5. Aufl. De Gruyter, Berlin (1966)
- Scharloth, J., Bubenofer, N.: Datengeleitete Korpuspragmatik: Korpusvergleich als Methode der Stilanalyse. In: Felder, E., Müller, M., Vogel, F. (Hrsg.): *Korpuspragmatik. Thematische Korpora als Basis diskurslinguistischer Analysen. Linguistik – Impulse und Tendenzen*, S. 195–230. De Gruyter, Berlin, New York (2011)
- Schmidt, F.: Automatische Ermittlung semantischer Zusammenhänge lexikalischer Einheiten und deren graphische Darstellung. Diplomarbeit, Universität Leipzig (1999)
- Schöch, C.: Quantitative Analyse. In: Jannidis, F., Kohle, H., Rehbein, M. (Hrsg.) *Digital Humanities: Eine Einführung*, S. 279–298. J.B. Metzler, Stuttgart (2017)
- Sparck Jones, K.: A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation* **28**(1), 11–21 (1972)
- Steele, J. (Hrsg.): *Meaning-text theory: linguistic, lexicography, and implications*. University of Ottawa Press, Ottawa (1990)
- Tenney, I., Das, D., Pavlick, E.: BERT RedisCOVERS the Classical NLP Pipeline. In: Korhonen, A., Traum, D., Märquez, L. (Hrsg.) *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, S. 4593–4601. Association for Computational Linguistics, Stroudsburg, PA, USA (2019). <https://doi.org/10.18653/v1/S.19-1452>

- Van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**(86), 2579–2605 (2008)
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Hrsg.) *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, S. 6000–6010. Long Beach, CA, USA (2017)
- Widdows, D., Dorow, B.: A graph model for unsupervised lexical acquisition. In: *Proceedings of the 19th international conference on Computational Linguistics (COLING-02)*, Taipei, Taiwan, S. 1–7. Association for Computational Linguistics, Morristown, NJ, USA (2002). doi: <https://doi.org/10.3115/1072228.1072342>
- Wiedemann, G., Remus, S., Chawla, A., Biemann, C.: Does BERT Make Any Sense? Interpretable Word Sense Disambiguation with Contextualized Embeddings. In: *Proceedings of KONVENS 2019*, Erlangen, Germany. <https://arxiv.org/pdf/1909.10430.pdf> (2019). Zugriffen: 11. Jan. 2021
- Zipf, G.K.: Human behaviour and the principle of least effort. Addison Wesley, Cambridge, MA (1949)



# Maschinelles Lernen für Sprachverarbeitung

6

## Zusammenfassung

In diesem Kapitel werden Aspekte des maschinellen Lernens behandelt, welche für die Verarbeitung von Text relevant sind. Im maschinellen Lernen erfolgt die Repräsentation sprachlicher Objekte wie Wörter oder Dokumente mit Merkmalen (Features). Auf Basis dieser Merkmale können Objekte geclustert werden, um deren natürliche Gruppierung sichtbar zu machen. Wir illustrieren Clustering, auch unüberwachtes Lernen genannt, an einigen Beispielen mit einem Fokus auf Topic-Modelle, welche insbesondere für die Exploration von Korpora beliebt sind. Unüberwachtes Lernen ist vor allem deshalb attraktiv, weil es keine Erstellung von Trainingsmaterial erfordert, jedoch stimmen die automatisch gezogenen Unterscheidungen häufig nicht mit den Zielkategorien von Anwendungen überein. Anschließend wird im Gegensatz dazu überwachtes Lernen diskutiert, welches aus einer Trainingstextmenge mit manuell zugewiesenen Labels einen Klassifikationsmechanismus lernt, welcher die automatische Zuweisung dieser Labels auf neue Texte erlaubt. Den Fokus legen wir hier auf sogenannte Sequenztagging-Verfahren, welche die Linearität von Text modellieren. Schließlich diskutieren wir noch häufig auf Text angewendete weiterführende Konzepte aus den neuronalen Methoden: End-to-End Systeme und Transferlernen.

Dieses Kapitel orientiert sich stark an der Anwendung von maschinellem Lernen für Text Mining. Es kann allgemeine Einführungen in das maschinelle Lernen nicht ersetzen, bietet jedoch die Darstellung des nötigen Grundverständnisses der Konzepte für dessen Anwendung auf Text.

## 6.1 Merkmalsextraktion

Spätestens beim Übergang von rein regelbasierter Verarbeitung (Abschn. 3.1) zu statistischer oder neuronaler Verarbeitung wird es notwendig, die Instanzen, also betrachteten Textobjekte, mit **Merkmälern (Features)** zu repräsentieren. Vor dem Hintergrund der Dominanz von neuronalen Methoden scheint es zunächst nicht mehr zeitgemäß, die manuell programmierte, oft linguistisch inspirierte Merkmalsextraktion zu diskutieren, wo doch neuronale Architekturen die für die Aufgabe geeignete Repräsentation während des Lernprozesses induzieren. Dies ist jedoch zu kurz gegriffen: Wie wir anhand eines Beispiels zur Koreferenzauflösung verdeutlichen, orientieren sich erfolgreiche neuronale Architekturen an denselben Intuitionen; es ist wenig erfolgversprechend, solche Architekturen ohne die entsprechenden Intuitionen zu konstruieren und auf ein Wunder der künstlichen Intelligenzemergenz zu hoffen.

Wie bereits in Abschn. 3.1.2 dargestellt, müssen Instanzen – egal ob sie in überwachten oder unüberwachten Verfahren verwendet werden – durch einen Vektor von Merkmalswerten charakterisiert werden, um mit diesen umgehen zu können: Anhand dieser Merkmale werden beim Clustering Instanzen miteinander verglichen, auch anhand dieser Merkmale werden Instanzen beim überwachten Lernen Zielklassifikationen zugewiesen. Die konkret verwendeten Merkmale haben also unmittelbaren Einfluss auf die Resultate maschiner Lernverfahren. Insbesondere kann kein Verfahren Unterscheidungen treffen, welche sich nicht in Merkmalsunterschieden äußern.

Wie ebenfalls bereits in Abschn. 3.1 ausgeführt, muss die Repräsentation, der Vektor aller Merkmalswerte, durch Featurefunktionen berechenbar sein, welche ansonsten nicht weiter eingeschränkt sind; insbesondere können sie direkt auf den Instanzen, auf deren Teilen, auf deren Kontexten, auf beliebigen Vorverarbeitungsschritten (Abschn. 3.2) und unter Zuhilfenahme externer Quellen wie lexikalische Ressourcen (Abschn. 4.5) definiert werden.

Doch was unterscheidet gute Repräsentationen von schlechten Repräsentationen? Genauso wie es nicht ein bestes Lernverfahren für alle Probleme gibt (z. B. Shalev-Shwartz und Ben-David 2014), gibt es auch nicht gute oder schlechte Featurefunktionen an sich: Die Güte und Passung hängt immer von der Gesamtsituation ab, welche von der Anwendung, den verfügbaren Daten und der Wahl des Verfahrens abhängt. Folgende Abwägungen sollten erfolgen:

- Welche Merkmale korrelieren mit der erwünschten Gruppierung oder Zielklassifikation? Wichtig an dieser Stelle ist zu erwähnen, dass auch schwache Korrelationen von geeigneten Verfahren gewinnbringend genutzt werden können. Es ist nicht notwendig, dass ein Merkmal 1:1 zu einer Klassifikation führen muss. Dies ist insbesondere zu beachten in Verbindung mit der
- Frequenz von Merkmalen bzw. deren Werten: Nur häufige Beobachtungen sind gute Merkmale; sehr präzise Charakterisierungen, welche praktisch nie auftreten, führen

aufgrund ihrer Seltenheit nicht zu Verbesserungen. Zum Beispiel eignen sich für eine Klassifikation von Dokumenten in Ressorts wie Politik, Wirtschaft oder Sport Präpositionen wie *mit*, *für*, *auf*, *zu*, ... deutlich besser als Listen von spezifischen Begriffen wie *Gipfeltreffen* oder *Winterolympiade*.

- Gesamtanzahl der Merkmale/Länge des Merkmalsvektors: Bei der Klassifikation muss für jedes Merkmal ein Parameter gelernt werden, welcher angibt, mit welchem Gewicht das Merkmal berücksichtigt werden soll. Je mehr Parameter gelernt werden müssen, desto mehr Trainingsdaten müssen vorhanden sein, um robuste und gut generalisierende Ergebnisse zu erhalten und **Overfitting**, also eine Überanpassung an die Trainingsdaten zu vermeiden. Aus demselben Grund ist auch beim unüberwachten Vortrainieren von Embeddings (Abschn. 5.6) die optimale Länge der Merkmalsvektoren von der Größe der Eingangsdaten abhängig.
- Korrelation von Merkmalen: Insbesondere beim Clustering, wo keine Gewichtung der Merkmale gelernt wird, aber auch bei einigen Klassifikationsverfahren sorgen hochkorrelierte Merkmale wie z. B. „die letzten 3 Buchstaben“ und „die letzten 4 Buchstaben“ bei der morphologischen Charakterisierung von Wörtern für Verzerrungen. In Kombination mit der Minimierung der Anzahl von Merkmalen sollten also möglichst unkorrelierte bzw. schwach korrelierte Merkmale gewählt werden.
- Art der Merkmalswerte: Die Featurefunktionen können z. B. binäre (Anwesenheit/Abwesenheit eines Wortes im Dokument), nominale (z. B. benachbartes POS-Tag) oder Zahlenwerte (z. B. Häufigkeitsklasse) liefern. Dies hat einen Einfluss auf das Lernverfahren, da nicht alle Lernverfahren mit allen Arten umgehen können. Neuro-nale Verfahren verlangen beispielsweise reellwertige Repräsentationen, die Werte müssen also ggf. geeignet umgewandelt werden.

Anhand der Aufgabenstellung der Koreferenzauflösung (siehe auch Abschn. 3.2.5) wird nun illustriert, wie Merkmale konkret aussehen und wie linguistische Intuitionen erfolgreich in einer Merkmalsextraktion umgesetzt werden können. Dieses Beispiel wird in Abschn. 6.10 wieder aufgegriffen um aufzuzeigen, wie diese expliziten Merkmale sich in einer neuronalen End-to-End-Architektur widerspiegeln, welche die Repräsentation selbst induzieren kann. Dieses Beispiel zeigt typische Eigenschaften, die nun diskutierten Mechanismen gelten im Allgemeinen für das maschinelle Lernen auf Text.

Bei der Koreferenzauflösung muss zunächst bestimmt werden, welche Wortsequenzen als Anaphern und Antezedenzen betrachtet werden (z. B. Nominalphrasen, insbesondere Pronomen). Deren Beziehung kann als Klassifikationsaufgabe formuliert werden: Gegeben die Repräsentation des Paares (Antezendent, Anapher) soll entschieden werden, ob Koreferenz vorliegt oder nicht.

Linguistisch orientierte Ansätze zur Koreferenz formulierten die Aufgabe gern als ein Sieb, welches aufgrund verschiedener linguistischer Eigenschaften aus der potentiell großen Menge von Antezedenzen einen oder wenige passende für eine konkrete Anapher herausfiltert (Raghunathan et al. 2010). Die Bestimmung der linguistisch motivierten Merkmale ist aufwendig, da oft theoriegeleitet vorgegangen wird, was eine

Implementierung der zugrunde liegenden Theorie notwendig macht. Linguistische Mechanismen können oft durch Oberflächenmerkmale approximiert werden, welche deutlich einfacher zu implementieren sind. Dass die Oberflächenmerkmale lediglich Spuren der darunterliegenden linguistischen Mechanismen sind und diese nur indirekt abbilden, ist zwar linguistisch unbefriedigend, jedoch für die rein ergebnisorientierte Automatisierung der Aufgabe unerheblich.

Tab. 6.1 zeigt eine Gegenüberstellung linguistisch motivierter Merkmale und diesen ungefähr entsprechende Oberflächenmerkmale. Durrett und Klein (2013) konnten zeigen, dass Oberflächenmerkmale ohne Informationen aus dem Dependenzparse ungefähr genauso gute Ergebnisse liefern wie linguistische Merkmale; mit diesen syntaktischen Informationen sogar bessere. Es wird deutlich, dass theoriegeleitete Konzepte wie Salienz, Agreement und Diskursstruktur durch einfache Merkmale wie Entfernung und Wörter im Kontext approximiert werden – natürlich nicht in direkter Entsprechung, jedoch auf robustere Weise als von den oftmals nicht mit hoher Genauigkeit verfügbaren Implementierungen der linguistischen Theorien vorausgesagt.

Die hier erfolgreichen Oberflächenmerkmale sind gute Merkmale im Sinne der oben diskutierten Abwägungen: Sie können leicht für alle Instanzen berechnet werden, ihre Werte haben gewisse Häufungen, sie sind intuitiv mit der Aufgabe korreliert und es finden sich keine speziellen, seltenen, jedoch sehr genauen Merkmale, ihre Anzahl ist moderat. Das Fehlen von Oberflächenmerkmalen als Ersatz für Merkmale aus

**Tab. 6.1** Linguistisch motivierte Merkmale und Oberflächenmerkmale für Koreferenzauflösung

Linguistisch motivierte Merkmale	Oberflächenmerkmale
Salienz: Wie unmittelbar wurde der Antezedent wahrgenommen? (folgend z. B. der Centering-Theorie bei Grosz et al. (1995))	Distanz: Wie weit sind Antezedent und Anapher voneinander entfernt?
Agreement in Genus, Numerus, Eigenschaften wie Belebtheit: Passt der Antezedent zur Anapher?	Erstes/letztes Wort von Anapher und Antezedent, z. B. <i>der grüne Frosch ... er</i>
Relative syntaktische Position im Konstituentenparse	Kopf und Vorgänger im Dependenzparse
Position innerhalb der Diskursstruktur (folgend einer Theorie wie z. B. RST bei Mann und Thompson (1987))	Wörter vor und nach Antezedent und Anapher
Selektionsrestriktionen von Prädikaten, z. B. sind Subjekte von Verben des Handels i. Allg. belebt	
Wissensbasis/lexikalische Ressourcen für Synonyme und Oberbegriffe, z. B. bei Bridging von <i>der Hund. das Tier</i>	
	Länge von Antezedent und Anapher in Wörtern und Buchstaben
Konzeptgleichheit/Synonymie	Stringbasierte Übereinstimmung zwischen Antezedent und Anapher

lexikalischen Ressourcen begründen Durrett und Klein (2013) damit, dass diese ohnehin nur wenig beitragen und beide Varianten mit Koreferenzauflösung Probleme haben, welche lexikalisches Wissen oder Weltwissen voraussetzt. Dies greifen wir in Abschn. 6.10 wieder auf.

---

## 6.2 Clustering

Clustering und Cluster-Analyse sind wichtige Teilaufgaben des Text Minings. Mithilfe von Clustering können sich deren Nutzende schnell einen Überblick verschaffen, aus welchen Dokumenten Korpora bestehen und wie das Vokabular darin thematisch verteilt ist. Dies sind häufig die ersten Schritte bei der Erschließung neuer Daten. Clustering ist aber auch das Kernelement vieler unüberwachter Verfahren bei der linguistischen Vorverarbeitung (Abschn. 3.2) oder bei der Merkmalsextraktion (Abschn. 6.1).

Dieses Kapitel kann lediglich als informelle Einführung in Clustering dienen, für ein umfassenderes und formaleres Survey sei z. B. Xu und Wunsch (2005) empfohlen, wo eine Vielzahl von Clusteralgorithmen diskutiert werden. Als freie Software für Clustering ist Weka (Witten et al. 2017) eine einfach bedienbare Oberfläche für die Implementierung sehr vieler gängiger Clusteralgorithmen und eignet sich insbesondere für Lernzwecke und Prototyping; scikit-learn (<https://scikit-learn.org>) ist eine umfangreiche Sammlung von Python-Bibliotheken für maschinelles Lernen im Allgemeinen und speziell auch für Clustering.

Beim **Clustering** wird eine Menge von Elementen (Daten, Objekte) in Cluster (Teilmengen, Gruppen, Klassen, Kategorien) eingeteilt, die einen natürlichen Zusammenhang zwischen den Elementen widerspiegeln sollen. Die Cluster werden dabei – im Gegensatz zur Vorgehensweise bei der Klassifikation (Abschn. 6.6) – aus der Struktur der zu analysierenden Datenmenge selbst abgeleitet und sollten vorhandene Regularitäten in den Daten sichtbar machen.

Während die vorhandene Vielzahl an Clusteringalgorithmen sehr divers ist und daher eine allgemeingültige präzisere Definition von Clustering nicht möglich erscheint, sollten die resultierenden Cluster folgenden Eigenschaften genügen (siehe z. B. Höppner et al. (1997)):

- Homogenität innerhalb der Cluster: Die Elemente eines Clusters sollten untereinander möglichst ähnlich sein.
- Heterogenität zwischen den Clustern: Elemente aus verschiedenen Clustern sollten zueinander möglichst unähnlich sein.

Zunächst ist zu klären, wie die Elemente repräsentiert werden, wie deren Ähnlichkeit quantifiziert wird und wie algorithmisch für die gewünschten Eigenschaften gesorgt werden kann.

Meist werden Elemente, hier typischerweise Dokumente, Sätze oder Wörter, durch Merkmalsextraktion (Abschn. 6.1) oder durch vortrainierte Embeddings (Abschn. 5.6) in eine Vektor-Repräsentation überführt. Auf Paaren dieser Repräsentationen, z. B.  $x$  und  $y$ , kann dann ein **Distanzmaß** (auch: Ähnlichkeitsmaß) definiert werden, welches den Vergleich zwischen zwei solchen Repräsentationen erlaubt und deren Entfernung als  $sim(x, y)$  quantifiziert. Es existieren eine große Anzahl von Distanzmaßen; für eine umfangreiche Diskussion aller gängigen Distanzmaße siehe Cha (2007). Ein häufig eingesetztes Distanzmaß auf reellwertigen Vektoren  $x$  und  $y$  ist die **Kosinusähnlichkeit**, (Gl. 6.1) welche dem Winkel zwischen den als Richtungen interpretierten Vektoren im Vektorraum entspricht. Diese wird maximal 1 bei einem Winkel von  $0^\circ$ , nimmt einen Wert von 0 an bei einem orthogonalen Winkel von  $90^\circ$  und wird minimal  $-1$ , falls die Richtungen bei einem Winkel von  $180^\circ$  genau entgegenstehen.

$$sim_{cos}(x, y) = \frac{x \cdot y}{\|x\| \cdot \|y\|} = \frac{\sum_{j=1}^m x_j \cdot y_j}{\sqrt{\sum_{j=1}^m (x_j)^2} \cdot \sqrt{\sum_{j=1}^m (y_j)^2}} \quad (\text{Gl. 6.1})$$

Vektorbasierte Distanzmaße sind zunächst zwischen Elementen definiert, können aber auch dazu eingesetzt werden, um Ähnlichkeiten zwischen Clustern oder zwischen Clustern und Einzelementen zu berechnen. Distanzmaße müssen jedoch nicht auf Vektoren definiert sein; z. B. drückt die **Levenshtein-Distanz** (auch **Editierdistanz**) die Anzahl an Einfüge-, Ersetzungs- oder Löschoperationen auf Buchstabenebene aus, um einen String in einen anderen String umzuwandeln, z. B. haben *Tasse* und *Klasse* eine Editierdistanz von 2 (Ersetzen von *T* durch *K* und Einfügen von *l*).

Clusterverfahren können anhand verschiedener Eigenschaften charakterisiert werden:

- Hart gegenüber weich: Beim harten Clustering (auch: Partition) wird jedes Element genau einem Cluster zugewiesen, beim weichen (auch: Fuzzy) Clustering gehören Elemente graduell ggf. zu mehreren Clustern.
- Flach gegenüber hierarchisch: Beim hierarchischen Clustering enthalten größere Cluster mehrere kleinere Cluster, die Elemente werden in eine Hierarchie von Clustern eingeordnet, im Gegensatz zu flachem, nicht-hierarchischen Clustering. Bei der Erstellung der Hierarchie kann agglomerativ oder divisiv vorgegangen werden: Agglomerative Verfahren verbinden zunächst Elemente mit geringer Distanz und kombinieren kleine Cluster zu größeren; divisive Verfahren beginnen mit einem großen Cluster und zerteilen diesen rekursiv.
- Vektor- gegenüber Graphrepräsentation: Die Elemente können vielfältig repräsentiert werden; im Text Mining sind Vektorraum und Ähnlichkeitsgraph üblich. Vektorrepräsentationen können in Graphrepräsentationen abgebildet werden, indem aus der Distanz zweier Elemente die Stärke deren Verbindungskante berechnet wird. Die umgekehrte Richtung wird durch Graph-Embedding-Verfahren realisiert, siehe z. B. Perozzi et al. (2014). Die Clusteringverfahren für die verschiedenen Repräsentationen unterscheiden sich deutlich.

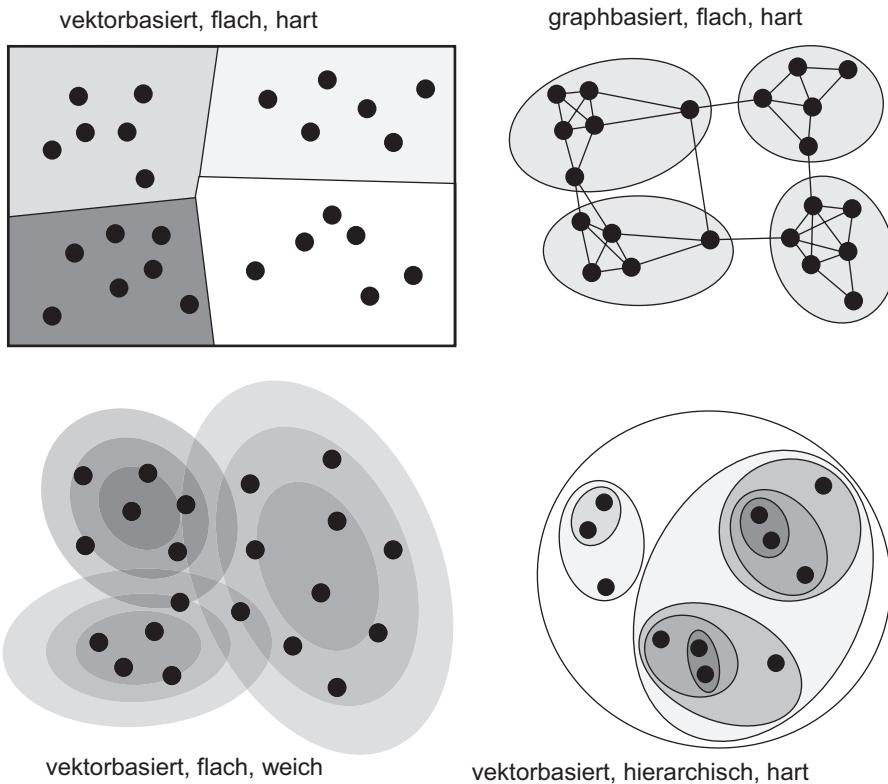
- Inkrementell vs. Batch: Während die meisten Clusteringverfahren davon ausgehen, dass alle Elemente zum Zeitpunkt des Clusterings bekannt sind und als ‚Batch‘ geclustert werden können, existieren auch inkrementelle Verfahren, welche sich zur Anwendung auf einen Datenstrom eignen, z. B. beim Clustering von Ereignissen aus aktuellen Nachrichtenmeldungen.
- Parametrisierung in der Anzahl der Cluster: Alle Clusteringverfahren werden über Hyperparameter oder über die gewählte Repräsentation der Elemente gesteuert. Bei der Clusteranalyse werden typischerweise diese Parameter variiert, um zu einem zufriedenstellenden Ergebnis zu gelangen. Für viele Clusteringverfahren ist der wichtigste Parameter die Anzahl der Cluster, mit welchem die Granularität direkt kontrolliert werden kann. Eine feste Anzahl Cluster ist jedoch in manchen Situationen ein Nachteil, z. B. wenn die Anzahl von Wortbedeutungen durch das Verfahren ermittelt werden soll (Abschn. 5.7 und 6.3), weswegen hier Verfahren zum Einsatz kommen, welche die Clusteranzahl selbst ermitteln.

Weitere Aspekte, anhand derer Clusteringalgorithmen charakterisiert werden können, ist deren asymptotisches Laufzeitverhalten, welches insbesondere bei der Skalierung auf große Datensätze relevant ist. Abb. 6.1 visualisiert einige Kombinationen verschiedener Eigenschaften. Bei den vektorbasierten Beispielen entspricht die Distanz der Punkte der Distanz im Clusteringverfahren; im graphbasierten Beispiel wird Ähnlichkeit durch eine Verbindungskante symbolisiert. Im nächsten Abschn. 6.3 werden verschiedene Clusteringalgorithmen vorgestellt und anhand dieser Eigenschaften eingruppiert.

Nachdem wir Merkmale, Distanzmaß und verschiedene Eigenschaften von Clusteringalgorithmen kennengelernt haben, können wir nun die Schritte für die Durchführung einer Clusteranalyse angeben:

1. Merkmalsauswahl: Mit welchen Merkmalen wollen wir unsere Elemente charakterisieren?
2. Distanzmaß und Repräsentation: Wie berechnet sich die Distanz aus den Merkmalen? Wie soll Ähnlichkeit repräsentiert werden?
3. Auswahl des Clusteringverfahrens: Welche Art von Resultaten sind gewünscht? Welches Verfahren ist geeignet für die vorliegende Art von Daten?
4. Inspektion: Sind die algorithmisch getroffenen Entscheidungen sinnvoll bzgl. der Aufgabenstellung?
5. Optional: Evaluation: Wie gut bildet das Ergebnis bekannte Unterscheidungen und Gemeinsamkeiten von Elementen ab?

Die Abfolge der Schritte ist nicht so linear wie hier dargestellt: Eine Clusteranalyse ist normalerweise ein iterativer Prozess, in dem Teilschritte immer wieder verändert werden, bis das aus der Anwendung entstehende, individuelle Informations- oder Interpretationsbedürfnis gestillt ist. Sofern für die Aufgabenstellung möglich und nötig,



**Abb. 6.1** Kombinationen verschiedener Eigenschaften beim Clustering

können Clusterings auch im Vergleich zu manuell erstellten Gruppierungen formal evaluiert werden (Schritt 5), siehe Abschn. 6.5 für geeignete Clustervergleichsmaße.

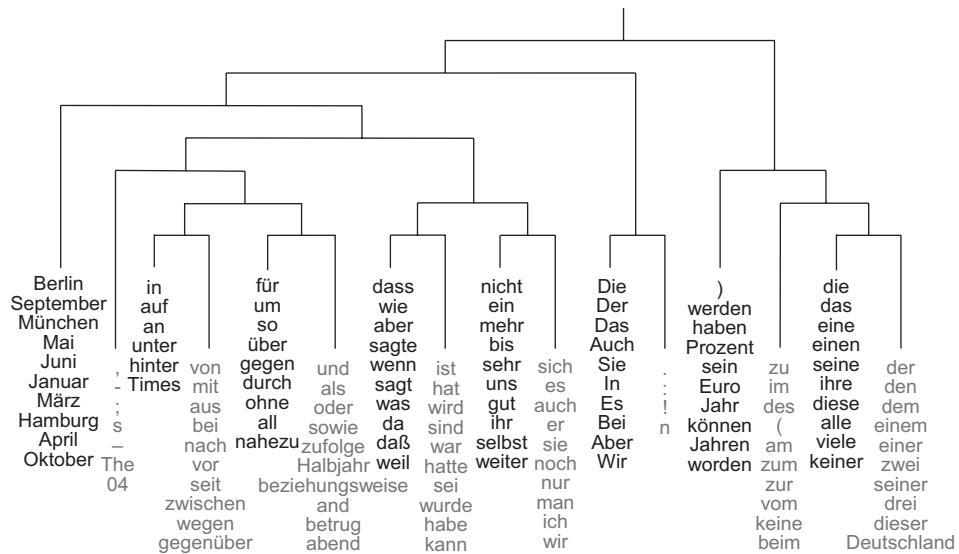
### 6.3 Beispiele für Clustering

Nachdem die Grundbegriffe des Clusterings im vorangegangenen Abschnitt eingeführt wurden, werden in diesem Abschnitt anhand von vier Beispielen Clusteringalgorithmen und deren Anwendung im Text Mining illustriert. Zunächst widmen wir uns dem Clustern von Wörtern hinsichtlich ihrer syntaktischen Rolle mit einem hierarchischen Clusteringverfahren, dann hinsichtlich ihrer Wortbedeutungen mit einem graphbasierten Verfahren. Als Beispiel für ein inkrementelles Clustering beschreiben wir die Erkennung von zusammengehörigen Ereignissen aus Social Media Streams; schließlich diskutieren wir anhand von Dokumentclustering den bekannten k-Means-Algorithmus. Topic-Modelle als weiches Dokumentclustering wird ausführlicher im darauffolgenden

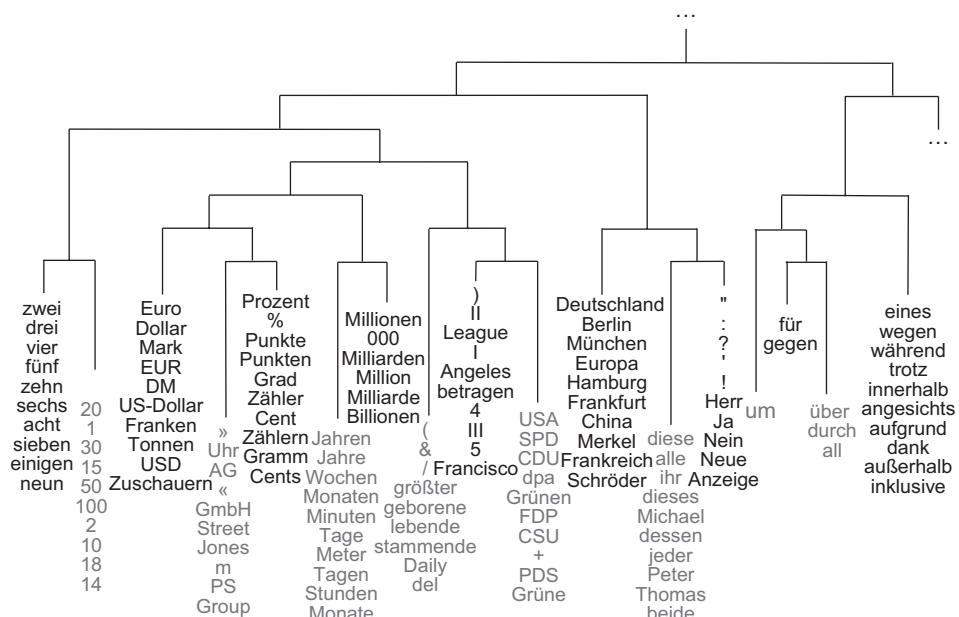
Abschn. 6.4 behandelt. In der Darstellung konzentrieren wir uns auf die prinzipiellen Funktionsweisen und diskutieren die verwendeten Algorithmen nur informell.

### 6.3.1 Hierarchisches Clustern von Wortarten

Eine erfolgreiche unüberwachte Methode zum Modellieren von Wortarten (Abschn. 2.3) ist Brown-Clustering (Brown et al. 1992), welches als früher Beitrag zur Wortarteninduktion (Christodoulopoulos et al. 2010) gilt. Brown-Clustering ist ein agglomeratives, hierarchisches Verfahren zum Clustern von Wörtern des Vokabulars auf Basis von Bigramm-Statistik (Abschn. 5.5) für ein Trainingskorpus und benötigt als Hyperparameter die Anzahl der Cluster  $C$ . Zunächst werden die häufigsten  $C$  Wörter im Vokabular eines Korpus in jeweils ein eigenes Cluster verortet. Dann werden die verbleibenden Wörter in absteigender Häufigkeit abgearbeitet, indem zunächst ein neues Cluster erstellt wird, hierauf werden zwei Cluster vereinigt, um wieder insgesamt  $C$  Cluster zu erhalten. Die Vereinigung soll die durchschnittliche **Mutual Information** (Abschn. 5.3) optimieren. Intuitiv erklärt geht das folgendermaßen: Die Wörter (Tokens) im Trainingskorpus werden durch die Cluster ersetzt: Jedes Wort wird in eine Cluster-ID umgewandelt. Auf diesen Abfolgen von Clustern als Wörter wird ein Sprachmodell trainiert, für welches die **Perplexität** gemessen werden kann (Abschn. 5.5). Nun werden probeweise Cluster vereinigt und berechnet, welche Vereinigung zweier Cluster die **Perplexität** am wenigsten erhöht. Schrittweise werden alle verbleibenden Cluster auf die gleiche Weise sukzessive vereinigt; aus der Reihenfolge der Schritte kann eine hierarchische Struktur abgeleitet werden. Wie in vielen Clusteringverfahren stellt das Vorgehen eine Heuristik dar: Die initiale Zuweisung der häufigsten Wörter zu eigenen Clustern garantiert in keiner Weise, dass noch eine optimale Lösung gefunden werden kann, ist jedoch notwendig, um die Berechnung auch für große Korpora zu ermöglichen. Abb. 6.2 und 6.3 zeigen Ergebnisse in der Darstellung eines **Dendrogramms** von Brown-Clustering für ein deutsches Korpus von 1 Mio. Sätzen in zwei Granularitäten für  $C=16$  (Abb. 6.2) und im Ausschnitt für  $C=128$  (Abb. 6.3); gezeigt werden jeweils maximal 10 Wörter pro Cluster, allerdings nur aus den häufigsten 10.000 Wörtern. Hier sind einige für das Clustering typische Phänomene beobachtbar. Zunächst fällt auf, dass beide Granularitäten gleichzeitig zu fein und zu grob zu sein scheinen: Bei nur 16 Clustern scheinen zwei von Satzzeichen dominierte Cluster unerwünscht, gleichzeitig werden im viertletzten Cluster viele Wortarten vermischt. Bei der feineren Granularität sind schöne Strukturen sichtbar wie die Unterscheidung von Zahlen in Ziffern- und Worddarstellung, welche sogleich eine Ebene höher gruppiert werden. Allerdings gibt es auch hier gemischte Klassen und recht deutliche Unterscheidungen wie z. B. zwischen Orten und Personennamen, Orten und Monatsnamen oder Personennamen und Personalpronomen finden nicht statt. In jedem Fall sind die meisten der hier beobachtbaren Unterscheidungen plausibel (z. B.zählbare Nomen, dritter Cluster bei  $C=128$ ), decken sich jedoch nur bedingt mit linguistisch motivierten Definitionen (hier: syntaktische Wortarten und semantische



**Abb. 6.2** Brown-Clustering mit 16 Clustern auf einem deutschen Zeitungskorpus mit 1 Mio. Sätzen



**Abb. 6.3** Brown-Clustering mit 128 Clustern auf einem deutschen Zeitungskorpus mit 1 Mio. Sätzen

Eigenschaften werden vermischt) – eine weitere typische Eigenschaft von Clusteringergebnissen.

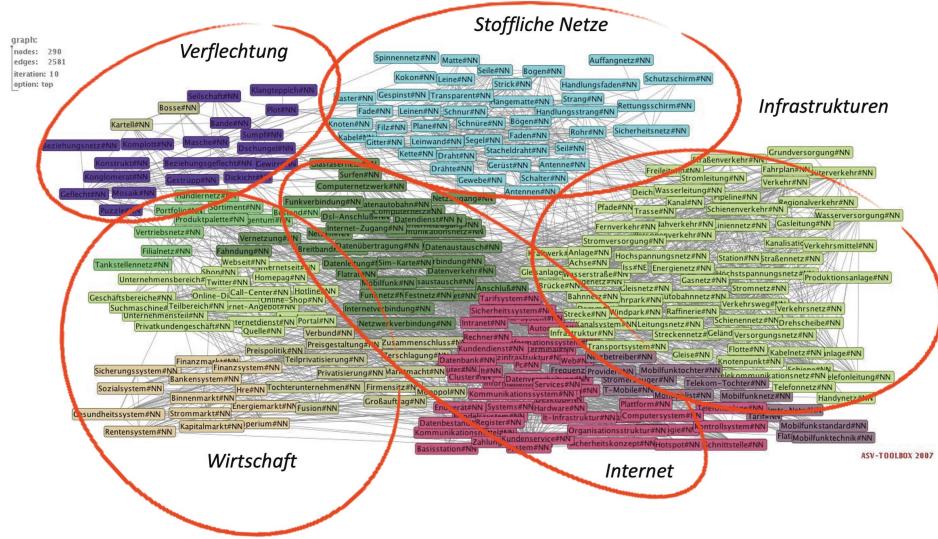
Brown Clustering kann als syntaktisches Merkmal anstatt von oder zusätzlich zu POS-Tagging (Abschn. 3.2.4) eingesetzt werden. Während dies inzwischen auch statische Embeddings (Abschn. 5.6) modellieren, haben Brown-Clustering oder andere Wortklasseninduktionsverfahren immer noch eine Daseinsberechtigung bei der linguistischen Analyse ressourcenarmer Sprachen.

### 6.3.2 Wortbedeutungsinduktion mit Graph Clustering

Nun greifen wir Wortbedeutungsinduktion (Abschn. 5.7) wieder auf und beschreiben das Clustern von Wortbedeutungen anhand eines flachen, graphbasierten Clusteringverfahrens. Graph Clustering eignet sich besonders gut für das automatische Identifizieren der verschiedenen Bedeutungen von mehrdeutigen Wörtern, da viele Graph-Clustering-Verfahren nicht die Anzahl der Cluster als Hyperparameter benötigen und es ja nicht von vornherein bekannt ist, wie viele Bedeutungen ein Wort im Korpus hat und ob es überhaupt mehrdeutig ist.

Betrachten wir zunächst die Struktur der Graphen in Abb. 5.16 (Abschn. 5.7). Der Kookkurrenzgraph zerfällt in verschiedene Komponenten, falls folgend Widdows und Dorow (2002) das Suchwort „Maus“ entfernt würde. Im Ähnlichkeitsgraph in Abb. 5.17 ist diese Entfernung schon vollzogen und es wird deutlich, dass es innerhalb der Menge einer Bedeutung zugehörigen Wörter deutlich mehr Ähnlichkeitsbeziehungen gibt als zwischen den Bedeutungen. Graph Clustering löst nun die Aufgabe, die Bedeutungen zu explizieren; siehe Cecchini et al. (2018) für eine vergleichende Evaluation von Graph-Clustering-Algorithmen für diese Aufgabe. Ein konzeptuell einfacher, jedoch sehr effektiver und effizienter Algorithmus für diesen Zweck ist Chinese Whispers (Biemann 2006). Dieser graphbasierte, flache, harte Algorithmus ist randomisiert und nicht-deterministisch, d. h. für denselben Eingabegruppen können in mehreren Aufrufen verschiedene Clusterings gefunden werden. Chinese Whispers initialisiert jeden Knoten des Graphen mit einer eigenen Farbe. In einer geringen Anzahl von Durchläufen werden Knoten in zufälliger Reihenfolge aktualisiert und nehmen die stärkste Farbe in ihrer Nachbarschaft an, wobei die Stärke einer Farbe durch die Summe der Kantengewichte zu jeweiligen benachbarten Knoten dieser Farbe gegeben ist. Abb. 6.4 zeigt ein Beispielclustering des Ähnlichkeitsgraphen für *Netz* mit manuell zugewiesenen Bedeutungen. Auch hier sind wiederum die Unterscheidungen nachvollziehbar, jedoch teilweise zu feingranular: *Kartell* und *Bosse* bilden ein eigenes Cluster, welches bei der manuellen Zuweisung unter *Verflechtung* subsummiert wird.

Graph Clustering ist oft das Mittel der Wahl für Daten, welche bereits als Graph vorliegen, z. B. Kookkurrenzen (Abschn. 5.3). Die Granularität des Clusterings kann durch die Graphkonstruktion beeinflusst werden, z. B. über ein Mindestgewicht für Kanten.



**Abb. 6.4** Wortbedeutungsinduktion für „Netz“ nach Friedrich und Biemann (2016) auf dem Ähnlichkeitsgraphen eines Deutschen Zeitungskorpus aus den Jahren 1995–2010 von 70M Sätzen, visualisiert mit der ASV Toolbox (Biemann et al. 2008)

Ein zweistufiger Aufbau mit Graph Clustering wird bei Ustalov et al. (2019) für das Clustern zur Vereinigung von Synonymwörterbüchern beschrieben: Hier werden zunächst Knoten lokal geclustert und deren Bedeutungen im globalen Graph einzeln repräsentiert. In einem zweiten Schritt wird der gesamte Graph geclustert. Trotz des Einsatzes von harten Graph-Clustering-Algorithmen resultiert dies in ein weiches Clustering.

### 6.3.3 Eventerkennung mit inkrementellem Clustering

Insbesondere wenn Clustering auf Datenströme wie z. B. Posts in Sozialen Medien angewendet werden soll, werden inkrementelle Verfahren benötigt, welche nicht das Vorhandensein aller zu clusternder Datenpunkte annehmen, sondern neu eintreffende Daten entweder einem bereits vorhandenen Cluster zuordnen oder ein neues Cluster erstellen. Als Beispiel dient uns die Erkennung von zusammengehörigen Ereignissen auf Nachrichtenströmen oder in sozialen Medien: Hier werden gleichzeitig sehr viele Ereignisse diskutiert. Manche davon sind neu, viele singulär, manche sind periodisch oder saisonal. Die Aufgabe, welche als Topic Detection and Tracking (Allan et al. 1998) oder Event Detection (Cordeiro und Gama 2016) bezeichnet wird, besteht in der Zuordnung einzelner Nachrichten zu den entsprechenden Ereignissen, um – je nach Ausprägung der Aufgabe – zusammengehörige Themen zusammenfassen zu können oder neue Ereignisse als solche zu erkennen, siehe auch Abschn. 7.4.

Einer von vielen passenden Algorithmen für diese Aufgabe ist die inkrementelle Variante von DBSCAN (Ester et al. 1996, 1998), welcher hier kurz beschrieben werden soll. DBSCAN ist ein dichtebasierter Algorithmus, welcher lokal Datenpunkte, die sich in der Nachbarschaft eines gerade ausgewählten Datenpunktes befinden, zu einem Cluster zusammenfasst. Datenpunkte sind hier durch Vektorrepräsentationen gegeben, und Nachbarschaft bezeichnet alle Punkte, deren Distanz bzgl. eines geeigneten Distanzmaßes unter einer gewissen Schwelle liegt; die Schwelle reguliert implizit die Anzahl Cluster. DBSCAN eignet sich für das Anwenden in Situationen, bei denen noch nicht alle Datenpunkte am Anfang verfügbar sind und inkrementell verarbeitet werden sollen: Falls ein neuer Datenpunkt in die Nachbarschaft eines bekannten Datenpunktes fällt, wird er dem entsprechenden bestehenden Cluster zugeordnet. Falls nicht, eröffnet dieser Datenpunkt ein neues Cluster.

Die Anwendung von DBSCAN und Varianten, welche auf sehr große Datenmengen durch Parallelverarbeitung skalieren (Abschn. 3.3.1), beschreiben Capdevila et al. (2016): Hier werden Twitter-Nachrichten sowohl anhand inhaltlicher Merkmale als auch aufgrund von Metadaten wie Ort und Zeit in Ereignisse geclustert.

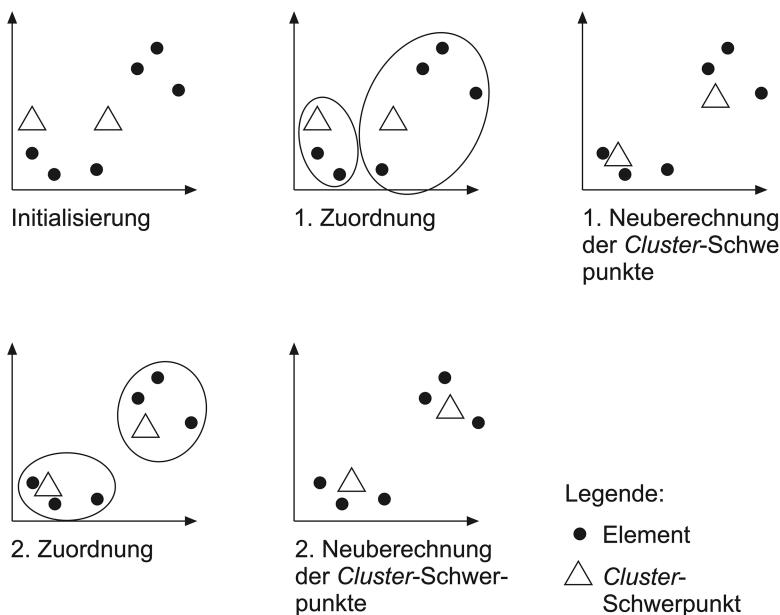
### 6.3.4 Dokumentclustering mit k-Means

Schließlich soll noch am Beispiel von Dokumentclustering der vielleicht bekannteste Clusteralgorithmus überhaupt besprochen werden: Der k-Means-Algorithmus (MacQueen 1967) wurde in vielfältigen Bereichen angewendet, welche weit über das Text Mining hinausgehen.

Bei k-Means, einem vektorbasierten, flachen, harten Clusteringalgorithmus ist  $k$  der wichtigste Hyperparameter, welcher die Anzahl der Cluster a priori festlegt.

Zunächst werden  $k$  Clusterschwerpunkte zufällig im Vektorraum platziert. Danach werden mehrfach zwei Schritte hintereinander ausgeführt: Jeder Datenpunkt wird dem bzgl. des Distanzmaßes nächstgelegenen Clusterschwerpunkt zugeordnet, dann werden neue Clusterschwerpunkte als Mittelwerte der zu ihrem Cluster zugehörigen Punkte berechnet. Der Algorithmus konvergiert, sobald sich keine Zuordnungen mehr ändern. Abb. 6.5 illustriert diesen Ablauf für einen zweidimensionalen Vektorraum und  $k=2$ .

Vektorbasierte Algorithmen wie k-Means eignen sich für sämtliche bisher besprochenen Repräsentationen von Dokumenten: tf·idf-Vektoren (Abschn. 5.9.2), LSA-Repräsentationen (Abschn. 5.4), oder Document Embeddings (Abschn. 5.6.4). Für das Dokumentclustering haben jedoch harte Clusteringalgorithmen einen entscheidenden Nachteil: Ganze Texte enthalten sehr heterogene Inhalte, mehrere Themen können innerhalb desselben Textes kombiniert werden und auch das passendste, inhaltliche, harte Clustering müsste sich für eine bestimmte Zuordnung entscheiden und die thematische Ähnlichkeit zu anderen Clustern ignorieren. Abhilfe schaffen die im folgenden Abschnitt diskutierten Topic-Modelle, welche einen weichen Clusteringalgorithmus für Wörter und Texte darstellen.



**Abb. 6.5** Beispiel für die Funktionsweise von k-Means (MacQueen 1967) mit  $k=2$

## 6.4 Topic-Modelle

Topic-Modelle basieren auf der Annahme, dass jedes Wort einem Themenbereich zugeordnet ist und sich aus der Verteilung thematisch zusammengehöriger Wörter in einem Text die zugrunde liegenden Themenbereiche, eben die Topics, ableiten lassen. Große Textkollektionen können damit nicht nur thematisch unterteilt werden, sondern Texte beziehungsweise Textabschnitte können auch anhand der identifizierten Themenstruktur klassifiziert und zusammengefasst werden. Topic-Modelle liefern inhaltlich interpretierbare Cluster rein aus einer statistischen Beobachtung von Regelmäßigkeiten in sprachlichen Oberflächenstrukturen auf der Grundlage des gemeinsamen Vorkommens von Wörtern in Dokumenten. Auch wenn sie nicht direkt zur Extraktion individueller (Syntax-)Strukturen, Informationen oder Aussageregelmäßigkeiten beitragen, können die Ergebnisse der automatischen thematischen Einteilung als Vorverarbeitungsschritt für weitergehende Analysen sehr nützlich sein.

### 6.4.1 Dokumente enthalten Themenstränge (Topics)

Als eine Abfolge von Sätzen oder Wörtern werden in einem Text meist ein oder mehrere Themenstränge behandelt. Jedes Thema kann dabei durch ein Cluster von Wörtern beschrieben werden, wie das nachfolgende Beispiel verdeutlicht.

**Beispiel Wittgenstein, Philosophische Untersuchungen, §31**

§ 31 Wenn man jemandem die Königsfigur im Schachspiel zeigt und sagt „Das ist der Schachkönig“, so erklärt man ihm dadurch nicht den Gebrauch dieser Figur, – es sei denn, dass er die Regeln des Spiels schon kennt, bis auf diese letzte Bestimmung: die Form einer Königsfigur. Man kann sich denken, er habe die Regeln des Spiels gelernt, ohne dass ihm je eine wirkliche Spielfigur gezeigt wurde. Die Form der Spielfigur entspricht hier dem Klang, oder der Gestalt eines Wortes. Man kann sich aber auch denken, Einer habe das Spiel gelernt, ohne je Regeln zu lernen, oder zu formulieren. Er hat etwa zuerst durch Zusehen ganz einfache Brettspiele gelernt und ist zu immer komplizierteren fortgeschritten. Auch diesem könnte man die Erklärung geben: „Das ist der König“ – wenn man ihm z. B. Schachfiguren von einer ihm ungewohnten Form zeigt. Auch diese Erklärung lehrt ihn den Gebrauch der Figur nur darum, weil, wie wir sagen könnten, der Platz schon vorbereitet war, an den sie gestellt wurde. Oder auch: Wir werden nur dann sagen, sie lehre ihn den Gebrauch, wenn der Platz schon vorbereitet ist. Und er ist es hier nicht dadurch, dass der, dem wir die Erklärung geben, schon Regeln weiß, sondern dadurch, dass er in anderem Sinne schon ein Spiel beherrscht.  
(Wittgenstein 2001)

Der Text behandelt offenbar die Frage, wie wir den Gebrauch einer Spielfigur lernen, wobei als Beispiel das Schachspiel dient. Zugleich stellt der Text eine Analogie her zwischen dem Erlernen des Gebrauchs einer Spielfigur und dem Gebrauch eines Wortes. Jedem dieser Aspekte können wir eine Liste von Wörtern als *Topics* zuordnen (im Folgenden in der Reihenfolge ihres Auftretens im Text):

**Lernen eines Gebrauchs:** zeigt, sagt, erklärt, Gebrauch, Regeln, kennt, gezeigt, lernen, Zusehen, Erklärung, Gebrauch, beherrscht, ...

**Schachspiel:** Königsfigur, Schachspiel, Schachkönig, Figur, Spiels, Spielfigur, Brettspiele, König, Schachfiguren, ...

**Sprache:** Form, entspricht, Klang, Gestalt, Wortes, Figur, Platz, vorbereitet, ... ◀

Die Grundidee bei der *Latent Dirichlet Allocation*, dem ersten von Blei et al. (2003) eingeführten Topic-Modell, setzt genau an dieser Beobachtung an. Für die Modellierung werden drei Annahmen gemacht: Erstens, jedes Dokument kann durch eine kleine Untermenge von global verfügbaren Themen charakterisiert werden (in unserem Beispiel die drei über die Begriffslisten charakterisierten Topics), wobei jedes Thema wiederum eine Verteilung über das Gesamtvakabular ist. Zur Veranschaulichung werden die Wörter dieser Verteilung oft nach Wahrscheinlichkeit absteigend sortiert und auf die bedeutendsten Wörter reduziert dargestellt (wie in unserem Beispiel). Für die Modellierung wird zweitens angenommen, dass diese Themen einen zentralen, wenn auch versteckten oder latenten Parameter bei der Generierung eines Textes darstellen – die Wahrscheinlichkeit, in einem Text oder Textabschnitt ein bestimmtes Wort anzutreffen, hängt wesentlich davon ab, welches Thema vorab ausgewählt worden ist. Drittens wird angenommen, dass wir bei der Analyse eines Textes oder einer Textkollektion mit dem Verfahren des Topic-Modelling den für die Textproduktion vorausgesetzten

generativen Prozess umdrehen können und aus den vorhandenen Daten, d. h. den Wörtern, welche den Text konstituieren, die latenten Topics inferieren können, welche die vorliegenden Daten am besten erklären.

Die Grundlage des Algorithmus bildet eine Menge artifizieller Topics. Jedes dieser Topics ist eine Wahrscheinlichkeitsverteilung über das vorhandene Vokabular und gibt an, wie wahrscheinlich ein Wort in diesem Topic ist. Jedes Dokument wird als eine Wahrscheinlichkeitsverteilung über eben diese Topics dargestellt, welche wiederum angibt, wie wahrscheinlich ein Topic für das aktuelle Dokument ist. Die generative Annahme der Dokumententstehung wählt pro Wortposition im Dokument zuerst ein Topic (proportional zu seiner Wahrscheinlichkeit im Dokument) und danach ein Wort aus dem gewählten Topic (ebenfalls proportional zu seiner Wahrscheinlichkeit im gewählten Topic). Ausgehend von dieser Annahme und den tatsächlich vorhandenen Wörtern in den Dokumenten können Rückschlüsse auf die Struktur der Wahrscheinlichkeitsverteilungen der Dokumente über Topics sowie der Topics über Wörter gezogen und diese approximiert werden. Das Ergebnis ist ein Merkmal-transparentes interpretierbares Verfahren, welches in Aufgaben wie dem Dokument-Clustering, der Bestimmung von Dokument-Ähnlichkeit, der Dokument-Klassifikation oder der explorativen Suche nutzbringende Informationen beitragen kann.

### 6.4.2 Modellierung von Topics

Grundlage der Textrepräsentation bildet das sogenannte Bag-of-Words-Modell, d. h. wir gehen davon aus, dass wir die Reihenfolge der Wörter im Text ignorieren können. Jedes Textkorpus enthält eine Menge von D Dokumenten, und jedes Dokument wiederum eine Menge von  $N_d$  laufender Wörter (Tokens). Die Gesamtzahl der Tokens eines Korpus bezeichnen wir mit N, das Vokabular des Korpus aller voneinander verschiedener Wörter (Types) mit V. Jedes Dokument ist eine parametrisierte Repräsentation des Vokabulars bezogen auf die Frequenz der Types im Dokument.

Zur Modellierung der oben skizzierten bedingten Wahrscheinlichkeiten nehmen wir an, dass jedes Dokument eine Mischung von (latenten) Topics ist und jedes Topic eine (beobachtbare) Mischung aus Wörtern.

#### Notation

$P(z)$  ist eine Verteilung über Topics  $z$  in einem Dokument

$P(w|z)$  sind die Verteilungen über Wörter  $w$  für Topics  $z$

$P(z_i=j)$  ist die Wahrscheinlichkeit, dass für das  $i$ -te Wort nun das Topic  $j$  gezogen wird

$P(w_i|z_i=j)$  ist die Wahrscheinlichkeit von Wort  $w_i$  im Topic  $j$

Die Wahrscheinlichkeit, zu welchem Topic ein Wort gehört, können wir nun als Produkt der bedingten Wahrscheinlichkeit  $P(w_i|z_i=j)$  mal der Wahrscheinlichkeit des Topics im Korpus  $P(z_i=j)$  berechnen:

$$P(w_i) = \sum_{j=1}^T P(w_i \vee z_i = j)P(z_i = j) \quad (\text{Gl. 6.2})$$

Um zu beschreiben, welche Topics für ein Dokument bzw. welche Wörter für ein Topic wichtig (eigentlich: wahrscheinlich) sind, schreiben wir.

$$\theta^{(d)} = P(z) \text{ und } \varphi^{(j)} = P(w|z=j) \quad (\text{Gl. 6.3})$$

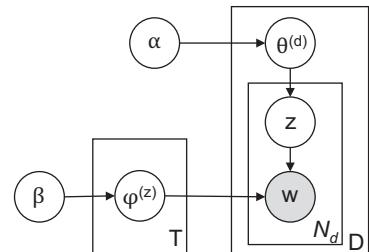
Tatsächlich beschreiben die latenten Variablen  $\varphi$  und  $\theta$  zwei Multinomialverteilungen bzw. Matrizen, nämlich die Zuordnung von Wörtern zu Topics (Wort-Topic Matrix  $\varphi$ ) sowie die Zuordnung von Dokumenten zu Topics (Dokument-Topic Matrix  $\theta$ ). Die Wort-Topic Matrix hat zwei Dimensionen,  $K$  und  $V$ , wobei  $K$  die Anzahl der Topics im Modell bezeichnet (eine Größe, die vom Nutzer bzw. der Nutzerin vorher festgelegt werden muss) und  $V$  das Vokabular. Jeder Wert  $\varphi_{kw}$  bezeichnet die bedingte Wahrscheinlichkeit, mit der ein Wort  $w$  aus  $V$  in einem Topic  $k$  aus  $K$  auftritt. Die Dimensionen der Dokument-Topic-Matrix  $\theta$  sind  $K$  und  $D$ , wobei  $K$  wieder die Anzahl der Topics im Modell bezeichnet und  $D$  die Anzahl der Dokumente im Korpus. Jeder Wert  $\theta_{dk}$  bezeichnet somit die bedingte Wahrscheinlichkeit mit der Topic  $k$  aus  $K$  in einem Dokument  $d$  aus  $D$  auftritt.

Die genannten Zusammenhänge verdeutlicht die Graphik der Abb. 6.6.

Für die Berechnung eines Topic-Modells müssen die beiden Matrizen  $\varphi$  und  $\theta$  geschätzt werden. Hierfür werden für die skizzierte Modellierung folgende Festlegungen getroffen:

1. Die Themenverteilung  $\theta^d$  von Topics zu Dokumenten ist eine **Dirichlet-Verteilung** mit Hyperparameter  $\alpha$  (die vom Nutzer bzw. von der Nutzerin vorab festgelegt werden muss).
2. Die Zuordnung von Wörtern zu Topics  $\varphi^j$  ist ebenfalls eine Dirichlet-Verteilung mit Hyperparameter  $\beta$  (die vom Nutzer bzw. von der Nutzerin vorab festgelegt werden muss).
3. Die latenten Variablen werden aus den beobachteten Wörtern im Text durch statistische Inferenz abgeleitet, indem ein generativer Prozess simuliert wird, der die

**Abb. 6.6** Plattennotation für Latent Dirichlet Allocation



tatsächlich beobachtete *a posteriori* Verteilung der Wörter am besten approximiert. (Meist wird hierfür das sogenannte **Gibbs Sampling** verwendet).

4. Um sinnvoll interpretierbare Ergebnisse zu erhalten, müssen für die LDA-Verteilungen  $\theta$  und  $\varphi$  zwei konkurrierende Ziele austariert werden: Einerseits sollen in jedem Dokument so wenige Topics wie möglich eine hohe Wahrscheinlichkeit haben, andererseits sollen aber auch in jedem Topic so wenige Wörter wie möglich mit hoher Wahrscheinlichkeit enthalten sein. Triviale Lösungen bestünden darin, jeweils jedem Dokument nur ein Topic zuordnen, was aber das Erreichen des zweiten Ziels erschwert, oder umgekehrt jedem Topic nur ein Wort zuordnen, was aber das Erreichen des ersten Ziels erschwert. In der praktischen Anwendung müssen also beide Ziele im Auge behalten werden. Weiterhin ist zu bedenken, dass probabilistische graphische Modelle sich allgemein nicht analytisch berechnen lassen, weil die Komplexität der Evidenzberechnung exponentiell mit der Anzahl der Trainingspunkte (Beobachtungen) steigt. Die effiziente Parameterschätzung in Topic-Modellen stellt also für deren praktische Anwendung eine zentrale Herausforderung dar, bei der allerdings in letzter Zeit große Fortschritte gemacht worden sind (u. a. Schuster 2015; Teichmann 2016).
5. In den vergangenen Jahren wurden zahlreiche Varianten des ursprünglichen LDA-Modells weiterentwickelt, welche unterschiedliche Aspekte in die Modellierung von Textkollektionen miteinbeziehen. Das Correlated Topic Model beispielsweise (Blei und Lafferty 2006) berücksichtigt, dass bestimmte Themen bevorzugt gemeinsam in Dokumenten auftreten. Beim Author Topic Model (Rosen-Zvi et al. 2004) wird die Autoren-Präferenz für bestimmte Themen mit modelliert. Zeitdynamische Topic-Modelle (Jähnichen 2016) erlauben die Modellierung der Veränderung thematischer Zusammensetzungen in diachronen Korpora. Darüber hinaus existieren zahlreiche weitere Varianten, welche etwa die Verknüpfung von Dokumenten mit externen Klassenvariablen erlauben (supervised LDA) oder gemeinsame Themen in multilingualen, alignierten (Mimno et al. 2009) und nicht-alignierten (Boyd-Graber und Blei 2009) Dokumenten über Sprachgrenzen hinweg finden können. Je nach Anwendungskontext und Forschungsfrage können diese zusätzlichen Aspekte wertvolle Informationen für eine Analyse mitliefern. In vielen Fällen können jedoch dieselben Auswertungen mit dem ursprünglichen LDA-Modell in Verbindung mit den Dokument-Metadaten in ähnlicher Qualität erstellt werden.

Das Beispiel unten zeigt die jeweils zehn wahrscheinlichsten Wörter für einige ausgewählte Topics von insgesamt 100, welche mit einer Standard-LDA-Implementierung (Phan et al. 2008) auf einem Korpus von 12.000 Artikeln von Wikinews (Wikimedia Foundation, Inc. 2020) berechnet wurden.

### Ausgewählte Topics auf einem Wikinews-Korpus

- Topic 0:** Google, Apple, Gerät, Samsung, Hersteller, Geräte, Jobs, Display, Smartphone, Steve
- Topic 1:** Gericht, Täter, wegen, Staatsanwaltschaft, Urteil, Frau, Mann, Tat, Polizei
- Topic 4:** Bundesregierung, Merkel, sei, Deutschen, Deutschland, deutschen, Bundestag, Wolfgang, Kritik, Entscheidung
- Topic 5:** bekannt, sehen, 1.300, zweitgrößten, Fuß, Nachforderungen, bewertet, widersprechen, niedergeschlagen, gebunden
- Topic 22:** ich, man, nicht, es, ist, sie, Ich, was, wir, aber
- Topic 26:** der, die, in, und, den, von, des, zu, dem, Die
- Topic 41:** Prozent, Partei, SPD, Stimmen, CDU, Wahl, Grünen, Koalition, Parteien, FDP
- Topic 94:** Minute, Spiel, gegen, der, Tor, FC, Minuten, Mannschaft, das, Spieler ◀

Das Beispiel zeigt typische Effekte: Viele der Topics sind vom Betrachtenden leicht interpretierbar, wie Topics 0, 1, 4, 41 und 94. Einige versammeln scheinbar zufällige Wörter wie Topic 5. Es ist oft der Fall, dass sehr häufige Wörter entweder ein eigenes Topic wie hier in Topic 22 und 26 bilden oder sich über viele Topics verteilen, weswegen oftmals in der Vorverarbeitung Stoppwörter entfernt werden.

### 6.4.3 Evaluation und Best Practices

Zur Aufteilung einer Dokumentkollektion in Themen benötigt der LDA-Algorithmus drei sogenannte Hyperparameter:  $K$ , die Anzahl an Themen, die inferiert werden soll sowie  $\alpha$  und  $\beta$ , die Dirichlet-Parameter, welche die Topic-Dokument- bzw. die Wort-Topic-Verteilung steuern. Die Wahl dieser Hyperparameter hat unmittelbare Auswirkungen auf die Qualität und Interpretierbarkeit der Modelle. Je nachdem, wie viele  $K$ -Themen gefunden werden sollen, lassen sich die Bedeutungen der einzelnen Topics unterschiedlich gut aus den in ihnen enthaltenen, hochwahrscheinlichen Begriffen rekonstruieren. Wird nur eine kleine Anzahl an Topics extrahiert, werden diese eher durch abstrakte, allgemeine Begriffe geprägt. Ihnen lässt sich dann schlecht ein konkreter thematischer Sinn zuordnen. Wird ein sehr hohes  $K$  gewählt, so können Bedeutungen der Topics stark auf seltene Aspekte in der Dokumentkollektion bezogen sein. Die Anzahl an Themen sollte daher mit Bedacht gewählt und von verschiedenen forschungstechnischen Erwägungen abhängig gemacht werden. Dazu können mehrere Modelle mit unterschiedlichen  $K$  berechnet und evaluiert werden. Die Qualität eines Topic-Modells kann in diesem Zusammenhang mit verschiedenen Verfahren beurteilt werden (Maier et al. 2017). Qualitativ kann beurteilt werden, ob den einzelnen Topics (intersubjektiv) ein bestimmter Sinn zugeschrieben werden kann. Quantitativ kann gemessen werden, ob hochwahrscheinliche Begriffe eines Topics tatsächlich gemeinsam

miteinander in Dokumenten auftreten (*Topic Coherence*; Newmann et al. 2010). Diese Ansätze eignen sich gleichzeitig, um geeignete Werte für die beiden Dirichlet-Parameter  $\alpha$  und  $\beta$  zu finden. Hohe Werte ( $>= 1$ ) dieser Parameter sorgen dafür, dass sich die Wahrscheinlichkeiten von Topics bzw. Worten gleichmäßiger über die einzelnen Dokumente bzw. Themen verteilen. Dadurch können Themen allgemeiner und weniger trennscharf werden. Kleine Werte für  $\alpha$  und  $\beta$  ( $< 1$ ) sorgen dagegen dafür, dass die Themen- bzw. Wortzusammensetzungen spezifischer werden, bei sehr kleinen Werten jedoch ebenfalls zu einer schlechten Modellqualität führen kann.

Eine weitere Schwierigkeit im Umgang mit Topic-Modellen ergibt sich aus deren eingeschränkter Reproduzierbarkeit aufgrund ihrer Berechnung mit stochastischen Inferenzverfahren. In aller Regel arbeiten computergestützte statistische Modelle und Verfahren für die automatische und semi-automatische Sprachanalyse robust, was heißt, dass eine wiederholte Messung gleiche Ergebnisse erzielt. Diese Eigenschaft erfüllen Topic-Modelle nur bedingt. Die Schätzmechanismen für die Parameter eines Topic-Modells, meist implementiert unter Verwendung von Gibbs Samplern oder variationellen Inferenzmethoden, basieren auf der iterativen Annäherung an einen Optimalzustand, also einen Zustand, an dem das Modell die zugrunde liegenden Daten bestmöglich erklärt. Die Startwerte für die Parameter (engl. *seed*), von denen ausgehend die Annäherung an das Optimum begonnen wird, werden im Normalfall zufällig initialisiert. Im Verlauf der Inferenz werden aus den geschätzten bedingten Wahrscheinlichkeiten Stichproben gezogen (engl. *sampling*) und die Parameter des Modells iterativ aktualisiert. Durch den Einfluss dieser Zufallsprozesse zusammen mit den Verteilungseigenschaften natürlicher Sprache in den Dokumenten, als auch in der gesamten Dokumentenkollektion (bedingt durchs Zipfsche Gesetz, Abschn. 5.2), kann es dabei zu unterschiedlichen Resultaten, also Topic-Verteilungen, kommen. Statt der analytisch optimalen Lösung der Modellgleichung (das globale Optimum, welches allerdings nicht berechenbar ist) finden die stochastischen Inferenzprozesse also nur lokale Optima innerhalb des Raums aller möglichen Wahrscheinlichkeitsverteilungen von  $\varphi$  und  $\theta$ . Studien zur Reliabilität haben gezeigt, dass sich die Modelle, je nach Kollektion und Inhalt, hinsichtlich der Interpretierbarkeit und Ähnlichkeit der Topics lediglich zu ca. 50–80 % reproduzieren lassen (Maier et al. 2017).

Diese Einschränkung muss in Anwendungen, die auf eine inhaltliche Interpretation der Topics angewiesen sind, bedacht werden. Zudem wurden im Zuge der Untersuchung dieses Phänomens Strategien vorgeschlagen, um diesem Problem zu begegnen. Die einfachste Strategie besteht darin, wiederholte Berechnungen mit dem gleichen Startwert zu initialisieren. Dies führt zu stabilen Ergebnissen (Riedl und Biemann 2012), suggeriert jedoch nur eine vermeintliche Stabilität der Inferenz. Wesentlich vielversprechender sind Strategien, die eine vorherige Initialisierung der Themenzugehörigkeit mit Hilfe von vorgeschalteten Analysen vornehmen (Lancichinetti et al. 2015), die Stichproben während der Schätzung durch andere Wörter im Dokumentkontext beeinflussen (Koltcov et al. 2014), oder die Topics mit deduktiv festgelegten Wortkontexten aus einem bestimmten Domänenvokabular fixieren (Andrzejewski et al. 2009).

## 6.5 Evaluation von Clustering

Die Evaluation von Clustering ist inhärent schwierig, da die Ergebnisse häufig plausibel aussehen, jedoch bestenfalls nur teilweise mit vorher manuell getroffenen Einteilungen übereinstimmen, wie bereits in den vorherigen Abschnitten eingehend illustriert. Dennoch besteht das Bedürfnis, neben der augenscheinlichen Validität auch mit formellen, automatisierbaren Maßen verschiedene Clusteringmethoden zu evaluieren. Hierzu wird das erwünschte Clustering, gegeben durch manuell zugewiesene Labels, mit dem automatischen Clustering verglichen. Viel direkter gestaltet sich dagegen die Evaluation von Klassifikation, da hier die Zielklassen durch Testdaten gegeben sind, siehe Abschn. 6.9.

Für das Vergleichen von Clusterings gibt es eine Vielzahl an Maßen, in Abhängigkeit der Clustering-Aufgabe sind jeweils einige Maße üblicher als andere. Clustervergleichsmaße, welche meist für den allgemeinen Fall entwickelt werden, sind in verschiedenem Grad für die in natürlichsprachlichen Daten üblichen Clustergrößenverteilungen geeignet, welche häufig der Potenzgesetzverteilung folgen, vgl. Abschn. 5.2. Im Folgenden werden einige übliche Clustervergleichsmaße und deren Anwendungsbereiche diskutiert, welche alle Werte zwischen 0 (maximal unähnlich) und 1 (deckungsgleich) annehmen. Eine eingehendere Diskussion dieses Themas findet sich in Amigó (2009).

Cluster Precision (CP, auch: Cluster Purity) misst den Reinheitsgrad von Clustern. Pro Cluster  $c_i$  wird der Anteil des jeweils häufigsten Labels  $l_j$  berechnet und darüber wird der Durchschnitt über alle  $N$  Cluster gebildet:

$$CP = \frac{1}{N} \sum_{i=1}^N \max_j |c_i \cap l_j| \quad (\text{Gl. 6.4})$$

Cluster Recall (CR, auch: Inverse Purity) misst, inwieweit dieselben manuellen Labels auch im selben Cluster landen. Dies wird analog zu CP berechnet, allerdings wird die Rolle von manuellen und automatischen Clustern vertauscht:

$$CR = \frac{1}{N} \sum_{i=1}^N \max_j |l_i \cap c_j| \quad (\text{Gl. 6.5})$$

Sowohl CP als auch CR sind einfach zu maximieren: CP wird 1, wenn jeder Datenpunkt sein eigenes Cluster darstellt, CR wird 1, wenn alle Datenpunkte in einem großen Cluster zusammengefasst werden. Beide Werte werden im **F-Wert** zusammengefasst, welcher das harmonische Mittel von CP und CR darstellt (siehe auch Abschn. 6.9).

$$F = \frac{2 \cdot CP \cdot CR}{CP + CR} \quad (\text{Gl. 6.6})$$

Dies ist angelehnt an die Evaluationsmaße aus dem Information Retrieval Precision, Recall und F-Wert, welche wir nochmal im Abschn. 6.9 aufgreifen werden. CP und CR eignen sich für den Vergleich ungefähr gleich vieler Cluster und Klassen, welche jeweils ungefähr gleich groß sind. Ansonsten sorgt die ungewichtete Durchschnittsbildung für unintuitive Ergebnisse dieses leicht verständlichen Evaluationsmaßes.

Ein informationstheoretisch fundierteres Maß, welches ebenfalls auf den Clustern als Mengen operiert, ist Normalized Mutual Information (Strehl 2002), welches auch als harmonisches Mittel von informationstheoretischen Maßen für Homogenität und Vollständigkeit für Cluster definiert werden kann (Rosenberg und Hirschberg 2007). Obwohl bei diesem Maß die Clustergröße gewichtet in die Berechnung eingeht, bewertet dennoch das Maß im Allgemeinen viele kleine Cluster besser, welche allein aufgrund ihrer geringen Größe tendenziell höhere Reinheitsgrade haben.

Ein weiteres weit verbreitetes Clustervergleichsverfahren ist der Adjusted Rand Index (ARI, Hubert und Arabie 1985). Es basiert auf paarweisen Zuordnungen: Für alle Paare von Elementen, wie viele landen korrekterweise bzw. fälschlicherweise im gleichen Cluster (bei gleichen Labels) bzw. in verschiedenen Clustern (bei verschiedenen Labels)? Der ARI zieht hierbei auch die Rate in Betracht, mit der zufällig Elemente mit demselben Label in demselben Cluster verortet würden. Seien TP (true positives) die Anzahl korrekterweise im selben Cluster verortete Elementpaare, TN (true negatives) die Anzahl Paare, deren Elemente korrekterweise in verschiedenen Clustern verortet sind, FP (false positives) bzw. FN (false negatives) die Anzahl Paare, deren Elemente fälschlicherweise im selben bzw. in verschiedenen Clustern verortet werden (vgl. auch Abschn. 6.9.1, wo diese Größen im Kontext von der Evaluation von Klassifikation genauer ausgeführt werden). Dann ist ARI definiert als:

$$\text{Erwarteter Index } E = \frac{(TP + FP) \cdot (TP + FN)}{TP + TN + FP + FN} \quad (\text{Gl. 6.7})$$

$$\text{Maximaler Index } M = TP + \frac{1}{2}(FP + FN)$$

$$\text{ARI} = \frac{TP - E}{M - E}$$

Ein weiteres populäres Maß, welches im Gegensatz zu den bisher beschriebenen Maßen elementzentriert vorgeht, ist B<sup>3</sup> (sprich: B-Cubed, Bagga und Baldwin 1998). Hier werden für jedes Element einzeln zwei Kenngrößen bestimmt, welche wiederum durch harmonische Mittelbildung kombiniert werden: B<sup>3</sup>-Precision misst den Anteil Elemente mit gleichem Label im Cluster des Elementes bezüglich der Clustergröße, B<sup>3</sup>-Recall den Anteil Elemente mit gleichem Label im Cluster bezüglich aller Elemente mit demselben Label.

Für manche Aufgaben werden auch verschiedene Maße kombiniert, um deren jeweilige Nachteile auszugleichen. Bei der Koreferenzerkennung, wo Koreferenzketten als Cluster aufgefasst werden können, wird gern der Durchschnitt aus B<sup>3</sup> und zwei anderen Maßen verwendet (Pradhan et al. 2012); für lexikalische Ketten empfehlen Remus und Biemann (2013) ein kombiniertes Maß aus ARI und einem Clusterdistanzmaß, welches auf der Anzahl Teile- und Vereinigungsoperationen basiert, um das automatische Clustering in das manuelle Clustering umzuwandeln.

Ein sinnvolles Vorgehen bei der Wahl des Maßes für eine Clusteringaufgabe besteht darin, etablierte Maße für ähnliche Aufgaben heranzuziehen, unterschiedliche Bewertungen augenscheinlich zu validieren, und das Maß für die weitere Optimierung zu verwenden, welches am ehesten der Intuition des Betrachtenden entspricht. Insbesondere bei Clustering, aber auch bei allen anderen Lernverfahren auf Text ist ein regelmäßiges manuelles Analysieren der Ergebnisse in jedem Fall notwendig.

---

## 6.6 Klassifikation

Nachdem in den vorangegangenen Abschnitten Aspekte des unüberwachten Lernens diskutiert wurden, soll es nun um überwachtes Lernen gehen. Hier werden Methoden des maschinellen Lernens eingesetzt, um aus Beispielen zu lernen. Ziel ist das Anwenden eines **Klassifikators** auf neuen Daten, der sich so verhält, wie es ihm auf den Trainingsdaten beigebracht wurde: Für Datenpunkte sollen bzgl. der Beispiele konsistente Zuweisungen der in den Trainingsdaten gegebenen Klassenzuweisungen erfolgen. Die Beispiele müssen so gewählt werden, dass der Klassifikator eine Grenze zwischen diesen Klassen ziehen kann. Wenn z. B. ein E-Mail-Spamfilter gelernt werden soll, benötigt der Klassifikator für das Training sowohl Beispiele für Spam als auch Beispiele für Nicht-Spam.

Überwachtes Lernen wird im Text Mining überall da eingesetzt, wo sprachliche Einheiten einer fest vorgegebenen Menge an Klassen oder einer vorgegebenen quantitativen Bewertung zugewiesen werden sollen; die Trainingsdaten sind entweder schon vorhanden oder müssen zu diesem Zweck über Annotation (Abschn. 6.7) händisch erstellt werden.

Die hier gewählte Darstellung ist relativ informell gehalten, um den Zugang zu erleichtern; für eine mathematisch fundiertere Darstellung von überwachtem Lernen für Textdaten siehe z. B. Zhang und Teng (2021).

Während es auch möglich ist, einen Klassifikator mithilfe von Regeln zu realisieren, führt dieses in der Regel nach schnellen Erfolgen zu immer kleineren Qualitätsfortschritten bei der Entwicklung, weswegen diese Möglichkeit außer beim schnellen Prototyping kaum in Erwägung gezogen werden sollte.

### 6.6.1 Definition und Arten von Klassifikation

Bei der Klassifikation wird eine Funktion gelernt, welche Merkmalsvektoren von zu klassifizierenden Elementen  $X = \{x_1, x_2, \dots\}$  in eine vorgegebene Klassifikation  $C$  abbildet. Diese Klassifikation  $C$  kann gegeben sein durch eine endliche Menge von Klassen-Labels  $C = \{c_1, c_2, \dots, c_k\}$  oder bei einer Regression durch reelle Zahlenwerte. Der Lernvorgang wird auf Beispielen  $B = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$  durchgeführt, welche als Paare von Elementen  $x_i$  und deren bekannter Klassifikation  $y_i$  gegeben sind.

Je nach Beschaffenheit der Klassen-Labels bzw. der Klassifikationsaufgabe werden folgende Arten von Klassifikatoren unterschieden:

- Binäre Klassifikation:  $C$  besteht aus genau zwei Klassen, z. B. 0 und 1. Dies ist beispielsweise die Konfiguration in einem E-Mail-Spam-Filter. Für die Elemente  $x$  in  $B$  ist bekannt, ob sie zur Klasse 0 oder 1 gehören.
- Mehrklassen/Multiklassen-Klassifikation:  $C$  besteht aus mehr als 2 Klassen. Als Beispiel kann hier die Dokumentklassifikation von Zeitungsartikeln in verschiedene Themenbereiche dienen, z. B. Wirtschaft, Politik, Sport, usw. Für die Elemente  $x$  in  $B$  ist jeweils genau eins dieser Klassen zugewiesen. Weitere Beispiel sind die Erkennung der Sprache eines Textes (Abschn. 3.2.2) oder die Grundformreduktion mit CPTs (Abschn. 3.2.4).
- Multilabel-Klassifikation:  $C$  besteht aus einer endlichen Menge an Klassen, es können jedoch mehrere Klassen pro Element  $x$  zutreffen. Ein Beispiel ist die Einordnung von Büchern zu Kategorien wie Belletristik, Sachbuch, Kochbuch, Fiktion, Science-Fiction, Krimi usw. Für Elemente  $x$  in  $B$  sind nichtleere Mengen von Klassen aus  $C$  zugewiesen, z. B.  $\{\text{Belletristik, Science-Fiction, Krimi}\}$ . Für die Kombinationen der Labels können auch Nebenbedingungen gelten, z. B. bei der hierarchischen Multilabel-Klassifikation (siehe z. B. Remus et al. 2019).
- Regression:  $C$  ist ein Intervall auf dem Zahlenraum der reellen Zahlen. Beispielsweise könnte dies für eine Polaritätsskala bei der Sentimentanalyse eingesetzt werden, siehe Abschn. 7.3.

Hierbei werden die Elemente  $x$ , wie in Abschn. 3.1 ausgeführt, durch Merkmalsvektoren repräsentiert. In diesem Abschnitt werden lediglich Klassifikationsverfahren beschrieben, bei der die Reihenfolge der zu klassifizierenden Elemente unerheblich ist, die Elemente sind voneinander unabhängig. Überwachte Lernverfahren zum Taggen von Sequenzen, bei denen dies nicht gegeben ist, werden später in Abschn. 6.8 diskutiert.

### 6.6.2 Aufteilung der Beispieldaten

Bevor wir einige Algorithmen zum Lernen von Klassifikationen exemplarisch besprechen, wenden wir uns zunächst dem generellen Setup zu. Beim überwachten

Lernen ist es von höchster Wichtigkeit, die Beispiele  $B$  richtig einzusetzen, um im Rahmen der gewählten Maschinerie den bestmöglich generalisierenden Klassifikator erstellen zu können. Um die Qualität des Klassifikators zu messen, interessieren wir uns für dessen Performanz auf Daten, welche **nicht** zum Training des Klassifikators eingesetzt wurden. Hierzu müssen wir von der Menge unserer Beispiele mit bekannten Klassen  $B$  eine sogenannte **Testmenge** (engl. test set) definieren, welche ausschließlich zur einmaligen formalen Evaluation des Klassifikators unter Verwendung eines Evaluationsmaßes (Abschn. 6.9) verwendet wird. Messen wir die Qualität auf Daten, welche der Klassifikator bereits gesehen hat, könnte der Klassifikator allein durch Memorieren der Daten 100 % Genauigkeit erreichen, was jedoch nichts über seine Qualität im späteren Einsatz aussagt.

Falls die Anzahl der Beispiele  $B$  für eine solche Aufteilung zu gering ist, kann als Alternative eine **Kreuzvalidierung** (engl. cross-validation) herangezogen werden: Hier wird weiterhin eine Testmenge ausreichender Größe abgezweigt; die restlichen Beispiele werden in  $N$  (typischerweise 5–10) gleich große Mengen aufgeteilt. Der Klassifikator wird in derselben Konfiguration reihum insgesamt  $N$  mal auf einer Trainingsmenge von  $N-1$  dieser Mengen trainiert und auf der einen übrigen Menge evaluiert, sodass die gesamte Trainingsmenge reihum als Development-Menge eingesetzt werden kann.

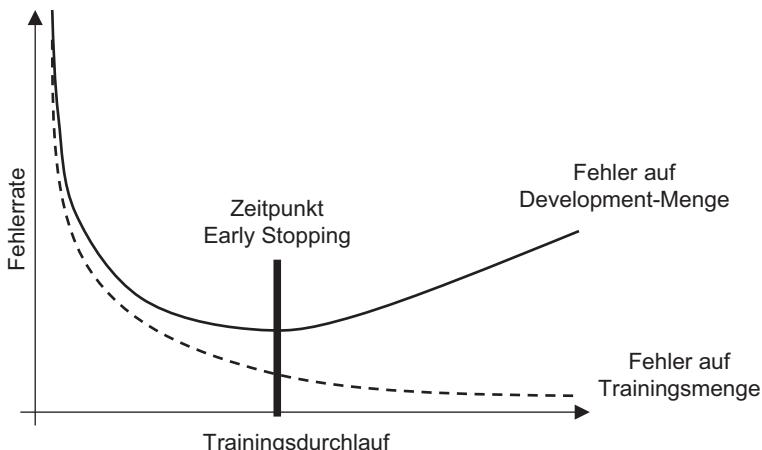
Beim Einteilen der Beispiele in die verschiedenen Mengen ist darauf zu achten, dass die Klassenverteilung einigermaßen gleich ist; eine solche **geschichtete Zufallsstichprobe** (engl. stratified sampling) wird nur bei großen Anzahlen von Beispielen durch rein zufällige Auswahl erreicht.

Viele Klassifikationsalgorithmen besitzen neben ihrer eigentlichen Parameterisierung ihres Modelles eine Reihe von Stellschrauben, die sogenannten **Hyperparameter**. Diese müssen beim Trainingsprozess mit optimiert werden. Bei z. B. neuronalen Netzen sind die Parameter die im Prozess gelernten Gewichte, die Hyperparameter definieren die Architektur inkl. der Anzahl und Art von Neuronen, der Lernrate, der Wahl des Optimierungsalgorithmus etc. Die Bestimmung dieser Hyperparameter wird im Allgemeinen experimentell durchgeführt. Hierzu wird eine **Development-Menge** (auch: **Validationsmenge**; engl. dev set bzw. validation set) benötigt, um folgendes Setup zu realisieren: Varianten von Klassifikatoren, welche sich in der Wahl des Algorithmus oder bzgl. ihrer Hyperparameter unterscheiden, werden auf der **Trainingsmenge** (engl. train set) trainiert und auf der Development-Menge evaluiert. Hierbei werden indirekt Informationen aus der Development-Menge für das Training verwendet, weswegen die Development-Menge eben nicht als ungesehen gelten kann. Der als am besten auf der Development-Menge evaluierte Klassifikator wird schlussendlich auf der Testmenge evaluiert, um eine realistische Einschätzung zu erlangen, wie gut der Klassifikator auf gleichartigen (genauso verteilten) neuen Elementen funktionieren wird. Eine typische Aufteilung der Größen von Trainings-, Development- und Testmenge besteht in 80 %, 10 % und 10 %: Es sollen möglichst viele Beispiele für das eigentliche Training verwendet werden, weil mehr Trainingsbeispiele i. Allg. zu besserer Qualität führen; andererseits müssen noch genügend viele Beispiele zur Optimierung der Hyperparameter

und zur abschließenden Evaluation vorhanden sein, damit hier auftretende Unterschiede auch signifikant sind und nicht nur durch zufällige Wahl besonders geeigneter oder ungeeigneter Evaluationsbeispiele entstehen.

Ein weiterer Einsatz der Development-Menge bei manchen Klassifikatoren ist die Bestimmung des Zeitpunktes, das Training zu stoppen. Insbesondere bei neuronalen Klassifikatoren, welche sich in vielen Durchgängen immer weiter den Trainingsbeispielen annähern, besteht die Gefahr des **Overfittings**: Während die Trainingsmenge sehr gut abgebildet wird, leidet die Qualität unter mangelnder Generalisierung auf der Development- und schlussendlich auch auf der Testmenge. Der typische zeitliche Verlauf der Klassifikationsqualität während eines konvergierenden Trainings ist in Abb. 6.7 dargestellt. Der Trainingsprozess kann mit der **Early-Stopping-Methode** (z. B. Prechelt 2012 für weiterführende Diskussion) angehalten werden, wenn die Fehlerrate auf den Development-Daten nach anfänglichen Gewinnen wieder steigt.

Falls die Anwendung es ermöglicht, kontinuierlich Trainingsbeispiele zu sammeln, kann die Evaluierung des Klassifikators auch inkrementell erfolgen: Hierbei wird ein Klassifikator jeweils auf Daten, welche bis zu einem bestimmten Zeitpunkt gesammelt wurden, trainiert und dann auf den neuen Daten evaluiert, der Zeitpunkt wird periodisch verschoben. Hierbei sollte jedoch darauf geachtet werden, ob die Daten einem sogenannten **Drift** unterworfen sind und sich in ihrer Ausprägung und Verteilung über die Zeit ändern. Ist dies der Fall, müssen zu alte Daten aus der Trainingsmenge entfernt werden; das Ermitteln der idealen Strategie für kontinuierliches Lernen ist ein aktuelles Forschungsthema, siehe Parisi et al. (2019) für eine Übersicht.



**Abb. 6.7** Illustration der Early-Stopping-Methode zur Vermeidung von Overfitting

### 6.6.3 Beispiele für Klassifikationsalgorithmen

Nun skizzieren wir die Arbeitsweise einiger Klassifikationsalgorithmen, um ein intuitives Verständnis gebräuchlicher Techniken zu vermitteln. Für eine formalere Darstellung siehe z. B. Zhang und Teng (2021), Witten et al. (2017) oder andere Einführungen in maschinelles Lernen, welche eine Vielzahl weiterer Lernverfahren diskutieren. Auch für die Klassifikation seien als Software Weka (Witten et al. 2017) für Lernzwecke und Prototyping sowie scikit-learn (<https://scikit-learn.org>) als Python-Bibliothek empfohlen.

#### 6.6.3.1 Naïve Bayes

Ein einfacher, meist nur noch als Baseline zum Vergleich oder in frühen Tests verwendeter Algorithmus ist Naïve Bayes. Dieser probabilistische Klassifikator zählt auf der Trainingsmenge, wie häufig welcher Merkmalswert mit welcher Zielklasse vorkommt und erstellt so eine empirische Wahrscheinlichkeitsverteilung pro Merkmal, jeweils unterschieden pro Klasse. Für ein zu klassifizierendes Element wird die Wahrscheinlichkeit der Klasse mit den Wahrscheinlichkeiten der vorliegenden Merkmalswerte jeweils für jede Zielklasse multipliziert und die Klasse mit dem höchsten Wert ausgewählt. Im Beispiel wird dies für die Klassifikation von Spam anhand von 3 Merkmalen verdeutlicht; üblicherweise würde man für die Textklassifikation mit Naïve Bayes eine **Bag-of-Words**-Repräsentation aus der Term-Dokument-Matrix (Abschn. 5.4) wählen, das Vorhandensein bzw. Fehlen jedes Wortes im Vokabular der Trainingsmenge bestimmt den Wert des jeweiligen Wort-Merkmales.

#### Beispiel für Klassifikation mit Naïve Bayes

In diesem Beispiel wollen wir Spam-E-Mails anhand von 3 Merkmalen klassifizieren: dem Vorhandensein oder Abhandensein eines Links (L+, L-), dem Vorhandensein bzw. Abhandensein eines Anhangs (A+, A-) und dem Vorhandensein/Abhandensein des Wortes „sofort“ (S+, S-).

Gegeben sei die Trainingsmenge als die Merkmalsrepräsentation von 10 E-Mails, davon 7 Spam:

- (L-, A-, S-) -> Nichtspam
- (L+, A-, S+) -> Spam
- (L-, A+, S-) -> Spam
- (L+, A-, S-) -> Nichtspam
- (L+, A+, S+) -> Spam
- (L+, A-, S+) -> Spam
- (L-, A+, S+) -> Spam
- (L-, A+, S-) -> Spam
- (L-, A+, S-) -> Spam
- (L+, A+, S+) -> Nichtspam

Zum Klassifizieren eines neuen Datenpunktes ( $L+$ ,  $A+$ ,  $S-$ ) berechnen wir Werte für Spam und Nichtspam wie folgt:

Spam: (rel. Häufigkeit von Spam) mal (rel. Häufigkeit von  $L+$  in Spam) mal (rel. Häufigkeit von  $A+$  in Spam) mal (rel. Häufigkeit von  $S-$  in Spam) =  $7/10 \cdot 3/7 \cdot 5/7 \cdot 3/7 = 9/98$ .

Nichtspam: (rel. Häufigkeit von Nichtspam) mal (rel. Häufigkeit von  $L+$  in Nichtspam) mal (rel. Häufigkeit von  $A+$  in Nichtspam) mal (rel. Häufigkeit von  $S-$  in Nichtspam) =  $3/10 \cdot 1/3 \cdot 1/3 \cdot 2/3 = 1/45$ .

Der Klassifikator entscheidet sich aufgrund des größeren Wertes für die Klasse Spam. ◀

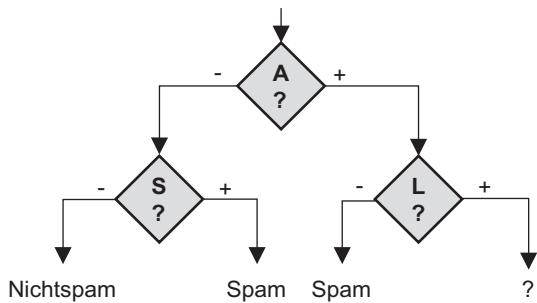
Vorteile von Naïve Bayes bestehen in der Einfachheit des Klassifikators, welcher in der hier diskutierten Form keinerlei Hyperparameter besitzt. Naïve Bayes kann sehr schnell durch geeignetes Auszählen der Verteilung der Merkmalswerte und Klassen trainiert werden und eignet sich somit auch für inkrementelles Training. Gründe, weswegen Naïve Bayes meist schlechtere Klassifikationsergebnisse liefert als andere Ansätze, sind zum einen die fehlende Gewichtung von Merkmalen, da alle Merkmale gleich stark in die Berechnung eingehen. Zum anderen modelliert Naïve Bayes keine Abhängigkeiten zwischen Merkmalen: Allein Link oder Anhang sind ja unverdächtig, aber aus einer Verbindung mit einer Aufforderung, diese sofort zu öffnen, entsteht begründeter Spamverdacht. Diese Interaktionen werden jedoch nicht modelliert.

### 6.6.3.2 Entscheidungsbäume

Genau die Modellierung der Interaktion von Merkmalen ist die Kernidee von Entscheidungsbäumen. Diese bestehen in ihrer einfachsten Form aus binären Fallunterscheidungen. Hierdurch wird die Trainingsmenge rekursiv in immer kleinere Mengen zerlegt, bis die Klasse innerhalb dieser Mengen homogen ist. Zur Unterscheidung wird jeweils das bezüglich eines Maßes (z. B. Information Gain als Maß für die Korrelation zwischen Merkmalwerten und Zielklassen) am besten klassifizierende Einzelmerkmal ausgewählt. Im Beispiel oben hat das Anhang-Merkmal den höchsten Information Gain; Abb. 6.8 zeigt den Entscheidungsbaum für das Beispiel. In diesem Fall ist die Kombination von Repräsentation und Klasse im Training inkonsistent für das Beispiel ( $A+$ ,  $L+$ ,  $S+$ ), weswegen am entsprechenden Blatt des Baumes keine Entscheidung getroffen werden kann.

Zur Vermeidung von Overfitting werden Entscheidungsbäume z. B. ab einer gewissen Tiefe oder auf Basis eines Schwellwertes des restlichen Fehlers beschnitten. Vorteile von Entscheidungsbäumen bestehen in deren schnellen Trainingszeiten und deren Interpretierbarkeit: Aus Abb. 6.8 kann z. B. direkt abgelesen werden, dass E-Mails ohne Anhang und ohne das Wort „sofort“ laut der Trainingsmenge Nichtspam sind. Qualitätseinbußen bei Entscheidungsbäumen werden bei unbalancierten Klassen und bei einer großen Anzahl an Merkmalen beobachtet.

**Abb. 6.8** Entscheidungsbaum für das Beispiel zur Klassifikation von Spam-E-Mails



### 6.6.3.3 Neuronale Netzwerke für die Klassifikation

Schließlich stellen wir noch dar, wie **neuronale Netzwerke** zur Klassifikation eingesetzt werden können. Hier gibt es eine Reihe verschiedener Architekturen (Abschn. 3.1.3), auf die hier nicht weiter eingegangen werden kann, welche unter dem Stichwort Deep Learning subsummiert werden. Alle diese Architekturen können für die Klassifikation eingesetzt werden. Das Prinzip ist bereits in Abb. 3.1 (Abschn. 3.1.3) dargestellt: Eine Netzwerkarchitektur bestehend aus künstlichen Neuronen-Einheiten wird derartig trainiert, dass eine Funktion von der Eingabeschicht zur Ausgabeschicht gelernt wird. Letztere besteht aus einer Schicht Neuronen, in der jedes Neuron einer Klasse entspricht. Ziel des Trainings ist es, die höchste Aktivierung für die richtige Klasse zu erreichen. Dies erfolgt i. Allg. mit Hilfe von zufälliger Initialisierung der Parameter (Verbindungsgewichte) mit anschließender iterativer Anpassung. Die Trainingsbeispiele werden jeweils mit dem momentanen Netzwerk vorhergesagt und mit der bekannten Klassifikation verglichen. Stimmen diese nicht überein, erfolgt eine graduelle Anpassung der Gewichte.

Für die Klassifikation von Text scheint der wichtigste Faktor für gute Ergebnisse die Wahl der Repräsentation (Abschn. 5.6) zu sein; Verbesserungen werden vor allem mit größeren vortrainierten Modellen erreicht, welche sprachliches Material im Allgemeinen immer besser modellieren, und geeignetem **Fine-Tuning** der Repräsentationen auf der Klassifikationsaufgabe (Abschn. 6.10). Ein weiterer wichtiger Faktor ist die Regularisierung beim Training der Netzwerke zur Vermeidung von Overfitting. Neben der o. g. Early-Stopping Methode gibt es eine Vielzahl weiterer Ansätze; die beste Wahl ist abhängig von der Aufgabe und vom Datensatz, sodass im Zweifel eine große Anzahl von Hyperparameterkonfigurationen getestet werden muss. Falls es vornehmlich darum geht, eine möglichst hohe Klassifikationsgenauigkeit zu erreichen, ausreichend Trainingsdaten vorhanden sind und Transparenz bezüglich der algorithmischen Entscheidung eine untergeordnete Rolle spielt, stellen neuronale Netzwerke das Mittel der Wahl für Klassifikationsaufgaben dar.

## 6.7 Annotation: Erstellung von Trainingsdaten

Für das maschinelle Lernen einer automatischen Klassifikation (Abschn. 6.6) im Allgemeinen und im Speziellen auch für Sequenzen (Abschn. 6.8) und noch stärker strukturierten Vorhersagen (Abschn. 3.2.4 und 3.2.5) werden Trainingsdaten benötigt. In diesem Abschnitt widmen wir uns der Vorgehensweise ihrer Erstellung.

Soll eine neue Text-Mining-Aufgabe bewältigt werden oder eine bereits bekannte Aufgabe auf anderen Sprachen oder Sachgebieten ausgeführt werden, welche nicht rein explorativ (Abschn. 6.2) ist, sondern wohldefinierte Kategorien automatisch zuweisen soll, dann kann mit einer rein regelbasierten Vorgehensweise (Abschn. 3.1.1) oft keine ausreichende Qualität erreicht werden. Für das Trainieren eines maschinellen Lernverfahrens jedoch werden Trainings- und Testdaten benötigt, welche manuell erstellt werden müssen. Vom automatischen Erstellen von annotierten Trainingsdaten ist abzusehen: Diese müssen ja nach wohldefinierten Regeln oder Charakteristika erstellt werden und diese Regeln würden dann beim Training wieder gelernt; ein Erfolg auf generierten Trainingsdaten sagt jedoch nichts über einen Erfolg auf echten ungesiehten Daten aus. Auch wenn in manchen Fällen geschickte automatische Augmentierung (Liu et al. 2020) der Daten zu leicht besseren Ergebnissen führt: Falls für die Aufgabe noch keine geeigneten Daten existieren, benötigen wir Annotierende, welche diejenige Zielklassifikation manuell zuweisen, welche für die Aufgabe nötig ist.

Im Folgenden geben wir Hinweise zum Aufbau und Management eines Annotationsteams für die Durchführung einer Annotationsstudie und zur Qualitätskontrolle der Annotationen, besprechen gängige Annotationstools und gehen auf die besonderen Herausforderungen beim Einsatz von Crowdsourcing für die linguistische Annotation ein.

### 6.7.1 Durchführen von Annotationsprojekten

Auch wenn die Annotationsaufgabe klar scheint und in wenigen Sätzen beschrieben werden kann, sollten beim Durchführen von Annotationsprojekten einige Aspekte beachtet werden, um hochqualitative annotierte Datensätze hervorzu bringen. Wir illustrieren dies am Beispiel von Namenserkenntnis im Text (Benikova et al. 2014), welche in der Annotation komplexer ist als es zunächst scheint.

Zunächst muss genau definiert werden, wie die Annotationsaufgabe aussehen soll, welche ja genau der Zielklassifikation der Komponente in einer geplanten Anwendung entspricht. Im Beispiel muss also festgelegt werden, was ein Name ist und welche Klassen von Namen es gibt, wie z. B. Personen, Organisationen, Orte, Medientitel etc. Bei solch offenen Aufzählungen empfiehlt sich in jedem Fall eine Restklasse „sonstige Namen“. Die Entwicklung der Klassen sollte in jedem Fall auch durch eine Datenanalyse begleitet werden, bei der ggf. auch Hinweise der Fachanwender berücksichtigt werden

sollten: Hier verschafft man sich durch Lesen und probeweises Annotieren des Textmaterials einen Überblick, welche Klassen denn überhaupt vorkommen und wo Problemfälle auftreten können. Zum Beispiel muss geklärt werden, wie mit Namen wie *1. FC St. Pauli* umgegangen werden soll: Soll der im Namen dieser Organisation enthaltene Ort als solcher annotiert werden? Enthält die *Riemannsche Vermutung* einen als solchen zu annotierenden Personennamen? Mit diesem Wissen wird die erste Version der Annotationsguidelines erstellt: Hier wird die Aufgabe definiert, getroffene Entscheidungen werden formuliert und dokumentiert und für alle zu annotierenden Klassen werden Positivbeispiele und Grenzfälle in die Guidelines aufgenommen. Wichtig ist nicht nur die Deklaration wie das Ergebnis aussehen soll, sondern auch wie die konkreten Annotationsentscheidungen operationalisiert werden können.

Mit dieser ersten Version wird ein Annotationsteam angelernt. Es empfiehlt sich, ein kurzes Textstück gemeinsam zu annotieren und Fragen oder Problemfälle sofort diskutieren zu können; die getroffenen Entscheidungen werden kontinuierlich in den Guidelines dokumentiert, welche üblicherweise im Verlauf von Annotationsprojekten anwachsen. Im Beispiel für Namen könnte das die Behandlung von Planeten und Gestirnen, Sportereignissen und nach Erfindern benannte physikalische Maßeinheiten sein (siehe Benikova et al. 2014). Ein (ggf. kleines) Annotationsteam ist aus mehreren Gründen einzelnen Annotierenden vorzuziehen: Im Team müssen Entscheidungen in den Guidelines expliziert werden, um ein homogenes Ergebnis zu erreichen; dies ermöglicht auch die Fortsetzung der Annotation zu einem späteren Zeitpunkt. Das vielleicht wichtigste Argument ist jedoch die Messung der Schwierigkeit der Annotationsaufgabe mit der **Interrater-Reliabilität (Inter-Annotator-Agreement, IAA)**. Diese Größe gibt an, wie sehr Annotierende in ihrer Tätigkeit übereinstimmen. Da die triviale Berechnung des prozentualen Anteils übereinstimmender Klassenzuweisungen die Verteilung der Klassen nur unzureichend berücksichtigt, werden hier statistische Maße eingesetzt. Die bekanntesten sind **Cohens Kappa** für die IAA zwischen zwei Annotierenden auf denselben Daten und **Fleiss' Kappa** für eine beliebige Anzahl Annotierender, welche nicht notwendigerweise alle dieselben Datenpunkte annotiert haben (Fleiss und Cohen 1973). Cohens Kappa wird berechnet als:

$$\kappa = \frac{p_0 - p_c}{1 - p_c} \quad (\text{Gl. 6.8})$$

Hier ist  $p_0$  der gemessene Übereinstimmungswert, der Term  $p_c$  korrigiert für zufällig zu erwartende Übereinstimmungen und ist von der Klassenverteilung abhängig. Kappa ist maximal 1 bei exakter Übereinstimmung, 0 bei rein zufälligen Übereinstimmungen und wird negativ, falls noch weniger Übereinstimmung gemessen wird als bei zufälligem Annotationsverhalten zu erwarten wäre. Die Kappa-Werte werden im Allgemeinen folgendermaßen interpretiert: Ein Kappa zwischen 0,4 und 0,6 bezeichnet eine moderate Übereinstimmung, zwischen 0,6 und 0,8 ist die Übereinstimmung substantiell und über 0,8 ist die Übereinstimmung fast perfekt. Annotierte Daten mit Kappa-Werten unter 0,4 werden als problematisch angesehen, hier ist die Konsistenz zu gering.

IAA kann einerseits dazu eingesetzt werden, die Guidelines zu evaluieren, andererseits charakterisiert der maximal erreichte IAA auch die Schwierigkeit der Annotationsaufgabe: Gerade bei Annotation von sprachlichem Material gibt es oft Grenzfälle und Mehrdeutigkeiten mit Interpretationsspielraum, sodass eine perfekte Übereinstimmung ohnehin nicht erreicht werden kann. Diese systematischen Inkonsistenzen verhindern auch das Lernen eines perfekten Klassifikators auf diesen Daten. Der Grad der Konsistenz bestimmt eine obere Schranke für die erreichbare Klassifikationsqualität.

In Abhängigkeit vom IAA-Wert sind bei der Skalierung der Annotation verschiedene Szenarien denkbar: Bei sehr hoher Übereinstimmung – dies ist jedoch selten – reicht eine Einfachannotation; auch bei substantieller Übereinstimmung sollte mindestens eine Zweifachannotation durchgeführt werden, ggf. mit manueller Auflösung bei Konflikten durch eine dritte Person. Je mehr Personen dasselbe Material annotieren, desto konsistenter Ergebnisse werden bei Entscheidung durch Mehrheitsbildung oder bei numerischen Annotationen mit Durchschnittsbildung erreicht. Dies ist natürlich verhältnismäßig teuer und führt zu weniger annotiertem Material.

Schließlich ist noch zu beachten, dass Annotation eine intellektuell sehr herausfordernde Aufgabe ist. Annotierende von Text können typischerweise nicht länger als zwei Stunden pro Tag eine hohe Qualität aufrechterhalten. Annotieren eignet sich also vornehmlich als Nebentätigkeit.

### 6.7.2 Annotationsebenen und Tools zur manuellen Annotation

Während für die ersten Schritte bei der Definition der Annotationsaufgabe auch Textmarker, händische Notizen oder Textverarbeitungssoftware geeignet sind, sollten Annotationsaufgaben mit geeigneten Tools unterstützt werden, welche einerseits die gemeinsame Datenhaltung des Annotationsprojektes leisten, andererseits auch vor Fehlern bei der Datenerstellung schützen können. In diesem Abschnitt sollen einige frei verfügbare Tools kurz erwähnt werden, für Surveys siehe Neves und Ševa (2021); Biemann et al. (2017); hierbei werden auch die verschiedenen Arten der linguistisch orientierten Annotation diskutiert. Bei der Auswahl der Tools wurde auf Flexibilität geachtet, sodass mit möglichst wenig Tools möglichst viele Annotationsprojekte abgedeckt werden können. Die Flexibilität hat jedoch ihren Preis: Das beste Tool ist immer dasjenige, welches speziell für die vorliegende Aufgabe entwickelt wird; der hierfür nötige Entwicklungsaufwand und der damit verbundene Zeitverlust lohnen sich jedoch gemeinhin nicht.

Bei der Annotation können folgende Ebenen unterschieden werden, welche direkte Konsequenzen sowohl für die Wahl des Tools als auch für die Auswahl des für die Daten geeigneten Klassifikationsalgorithmus haben.

- Dokumentannotation, z. B. die Annotation von Sentiments auf der Ebene von Reviews oder Social Media Posts, oder bei der Kodierung von Inhaltsanalysen in den Sozialwissenschaften,
- Spannenannotation, z. B. die Annotation von Wortarten (POS, Abschn. 3.2.4 auf einzelnen Wörtern) oder Namen (ggf. mehrere Wörter umspannend),
- Relationenannotation, z. B. die Annotation von grammatischen Dependenzrelationen (Abschn. 3.2.4) oder Templates bei der Informationsextraktion;
- Ketten, z. B. bei Koreferenz (Abschn. 3.2.5); diese können ggf. auch als Relationen modelliert werden.
- Stärker strukturierte Annotationen, z. B. Semantische Frames (Abschn. 3.2.5).

Die Ebenen sind in aufsteigender Komplexität geordnet. Während die Annotation von kurzen Dokumenten bereits realisiert werden kann und keine signifikanten Trainingsaufwände bei Annotierenden nötig sind, z. B. für die Annotation von Messenger-Bots (Yimam et al. 2020), stellen stark strukturierte Annotationen sowohl für das Tooling also auch für Annotierende große Herausforderungen dar.

Weitere Dimensionen bei der Auswahl des geeigneten Tools sind:

- Modellierung der Annotierenden: manche Tools erlauben nur Einzelannotation, manche kollektive Annotation (d. h. alle Annotierenden annotieren gleichzeitig) im Gegensatz zu Mehrfachannotationen (jede/r annotiert auf einer eigenen Kopie). Manche Tools bieten eigene Kurationsoberflächen zur Auflösung von Konflikten bei der Mehrfachannotation.
- Deployment als Webinterface oder als installiertes Programm: Webinterfaces sind flexibler und niederschwelliger, eigens installierte Programme werden inzwischen nur noch für sehr spezielle Annotationen eingesetzt.
- Möglichkeit der Einbindung maschinellen Lernens zur Generierung von Annotationsvorschlägen zur Verschnellerung des Prozesses, entweder lose gekoppelt über APIs oder integriert als Human-in-the-loop-Lernen (Holzinger 2016).
- Formate der Eingabedokumente und der Annotationen; für manche Anwendungsfälle speziell interessant ist das direkte Annotieren auf PDF-Dateien, z. B. Shindo et al. (2018).
- Aufwand bei der Einrichtung des Tools; einfache Aufgaben rechtfertigen nicht lange Einarbeitungszeiten für komplexe Tools.

Ein einfaches, beliebtes Tool für Dokumentebene und Spannenannotation ist doccano (Nakayama et al. 2018), welches jedoch keine Möglichkeiten zur Integration von Mehrfachannotation bietet. Ein sehr flexibles Tool zur Annotation kurzer Texte mit der Möglichkeit zur Generierung von Vorschlägen durch externe maschinelle Lernmodelle und Human-in-the-Loop-Annotation ist ActiveAnno (Wiechmann et al. 2021). Catma (<http://catma.de>) ist ein Online-Tool für die Auszeichnung von längeren Textspannen, wobei die Kategorien hierfür leicht erweiterbar sind. Es ist deshalb besonders

für hermeneutische Vorgehensweise wie z. B. in den Literaturwissenschaften geeignet, allerdings nur bedingt für die Erstellung von Trainingsdaten für das maschinelle Lernen. Ein Allrounder unter den Tools für linguistische Annotation ist WebAnno (Yimam et al. 2013), welches bis auf Dokumentannotation alle Annotationsebenen und eine große Anzahl an Formaten abdeckt und eine Kurationsoberfläche sowie die Berechnung von IAA bereitstellt. Für Aufgaben wie Entity Linking (Abschn. 3.2.5) und die Einbindung von Wissensbasen sei INCEption (Klie et al. 2018) empfohlen, welches auf WebAnno aufbaut und eine einfachere Integration mit externen maschinellen Lernmodellen bietet, jedoch für einfache Annotationsaufgaben gegebenenfalls überkomplex ist.

### 6.7.3 Datensatzerstellung mit Crowdsourcing

Crowdsourcing (Howe 2006) bietet die Gelegenheit, das Annotationsteam durch Crowdarbeitende zu ersetzen. Anstatt ein Team von Annotierenden zu managen, wird die Aufgabe an einen sehr großen Pool an oftmals unterbezahlten Crowdarbeitenden verteilt, was Begehrlichkeiten bezüglich Kosteneinsparungen und Datenerhebungsgeschwindigkeit weckt. Beim Crowdsourcen von Textannotationen sind jedoch einige Dinge zu beachten, welche einerseits mit Crowdsourcing an sich zu tun haben, andererseits speziell für die Annotation sprachlichen Materials gelten.

Im Gegensatz zu trainierbaren Annotierenden in einem Annotationsteam wechseln die Bearbeitenden von Crowdsourcing-Aufgaben andauernd und haben dadurch weniger Motivation zum Erlernen komplexer Aufgaben als auch weniger Motivation, qualitativ hochwertige Arbeit zu leisten. Um diesen Unterschieden Rechnung zu tragen, müssen Aufgaben beim Crowdsourcing möglichst einfach und unmissverständlich sein, sodass sich das Lesen der Instruktionen für die Crowdarbeitenden schon nach der Bearbeitung einiger weniger Aufgaben lohnt. Komplexe Aufgaben bei der Datensatzerstellung müssen ggf. in kleinere, Crowdsourcing-fähige Unteraufgaben zerlegt werden, um dies zu erreichen. Bei der Annotation von Namen könnte dies beispielsweise erreicht werden, die Erkennung der Namenshaftigkeit im ersten Schritt durchzuführen und die Klassifikation nach Person, Ort, usw. in einem zweiten Schritt. Das wichtigste Element beim Erstellen von Datensätzen mit Crowdsourcen ist die Qualitätskontrolle. Auf einschlägigen Plattformen können zwar Arbeiten als unzureichend zurückgewiesen werden, jedoch muss dies automatisiert geschehen, wenn man nicht alle Daten einzeln kontrollieren möchte. Hierzu werden gemeinhin Tests eingesetzt: Manche der Annotationen sind bereits bekannt und verifiziert; Crowdarbeitende, welche diese zu oft fehlerhaft annotieren, werden für diese Aufgabe blockiert.

Beim Crowdsourcen von Textannotationsaufgaben ist zudem zu beachten, dass es durch die Mehrdeutigkeit und Unbestimmtheit von Sprache oftmals mehrere richtige Möglichkeiten gibt. Eine Aufgabe wie das freie Paraphrasieren eines Satzes lässt sich schlicht nicht durch das Hinterlegen richtiger Möglichkeiten testen – hier können Design Patterns wie Find-Fix-Verify (Bernstein et al. 2010) eingesetzt werden, wo frei generierte

Texte in einer nachgelagerten Aufgabe von Crowdarbeitenden bewertet werden, was jedoch wiederum den Ablauf verkompliziert und die Kosten erhöht.

Eine frühe und populäre Crowdsourcing-Plattform ist Amazon Mechanical Turk (<https://www.mturk.com>), welche zwar nur minimale Mechanismen zur Qualitäts-sicherung bereitstellt, dafür aber sehr flexible Möglichkeiten bei der Erstellung von Aufgaben bietet und keine monatlichen Grundkosten in Rechnung stellt. Andere Crowdsourcing-Anbieter wie Clickworker (<https://www.clickworker.com>) verstehen sich eher als Dienstleister bei der Umsetzung von Datenerhebungsprojekten und bieten Qualitätssicherung als Service; sie lohnen sich jedoch erst ab einem substantiellen Volumen.

Insgesamt ist Crowdsourcing eine beliebte Alternative zu eigenen Annotationsteams. Es ist jedoch zu prüfen, ob sich wegen der ggf. notwendigen Zerlegung der Aufgabe in kleine Einzelschritte und durch den Overhead der Tests bei angemessener Bezahlung Crowdsourcing für ein gegebenes Annotationsprojekt lohnt.

## 6.8 Sequenzklassifikation

Nachdem bereits in Abschn. 6.6 die allgemeine Klassifikation mit überwachtem maschinellen Lernen dargestellt wurde, behandelt dieser Abschnitt nun die spezielle Art der Klassifikation namens **Sequenzklassifikation** (auch **Sequenzmarkierung**), welche beim Labeln bzw. Tagging von Sequenzen eingesetzt wird, was insbesondere für unseren Gegenstand Text relevant ist.

Im Unterschied zur Standardformulierung von Klassifikation wird bei der Klassifikation eines Datenpunktes innerhalb einer Beobachtung, welche hier eine Sequenz von Datenpunkten darstellt, nicht nur die Repräsentation des Datenpunktes an sich für die Wahl eines Labels verwendet, sondern auch die Labels von benachbarten Datenpunkten. Innerhalb der Beobachtung werden also Datenpunkte nicht als unabhängig voneinander betrachtet: Während es z. B. bei der Klassifikation von Spam-E-Mails unerheblich ist, welche E-Mails direkt vor oder nach einer E-Mail eintrafen und ob diese Spam sind oder nicht, ist der Kontext bei Aufgaben wie POS-Tagging (Abschn. 3.2.4) entscheidend für die Wahl des richtigen Labels. Abb. 6.9 zeigt ein Beispiel: Das Wort *ein* kommt sowohl als Artikel (Label: DET), als auch als Verbpartikel (Label: ADV) vor.

Offensichtlich kann eine Repräsentation der zu *ein* gehörenden Datenpunkte, welche den Kontext überhaupt nicht mit einbezieht, keine Auflösung solcher Mehrdeutigkeiten leisten, da dieselbe Repräsentation ja i. Allg. zu derselben Klassifikation führt. Natürlich besteht die Möglichkeit, bei der Repräsentation der Datenpunkte den Kontext mit

<b>ADP</b>	<b>ADJ</b>	<b>NOUN</b>	<b>VERB</b>	<b>DET</b>	<b>ADJ</b>	<b>NOUN</b>	<b>ADV</b>	<b>ADV</b>	<b>ADP</b>	<b>DET</b>	<b>ADJ</b>	<b>ADV</b>	<b>PUNCT</b>
------------	------------	-------------	-------------	------------	------------	-------------	------------	------------	------------	------------	------------	------------	--------------

Nach dem Umpflanzen wächst ein großer Baum nur langsam in den Boden ein .

**Abb. 6.9** Wortartentagging durch Universal POS Tagset (McDonald et al. 2013) an einem Beispiel mit mehrdeutigem Wort „ein“, welches nur durch Kontextinformation korrekt erfolgen kann. Visualisierung mit WebAnno (Yimam et al. 2013)

einzuzeichnen; im Beispiel könnten sich die Repräsentationen der entsprechenden Datenpunkte auf Basis von „wächst ein großer“ bzw. „Boden ein.“ berechnen und die Klassifikation auf diesen Kontextrepräsentationen als unabhängig durchführen. Bei erfolgreichen Verfahren zur Sequenzklassifikation werden jedoch – anstelle oder zusätzlich – auch die benachbarten Labels mit einbezogen: Steht im Beispiel ein Wort wie *ein* zwischen VERB und ADJektiv, ist dies ein starkes Indiz für die Vergabe von DET als Label; zwischen NOUN und PUNCT stehen dagegen für gewöhnlich keine Artikel (DET), wohl aber abgetrennte Verbpartikel.

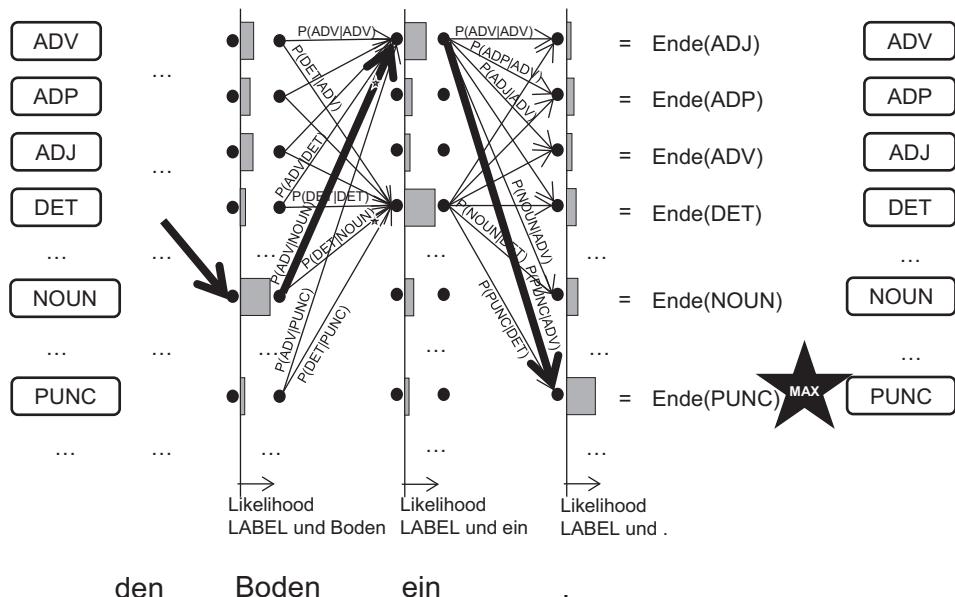
Durch den höheren Abstraktionsgrad der Labels gegenüber den Wörtern der Sequenz in Verbindung mit deren Passgenauigkeit für die vorliegende Aufgabe führt dies im Allgemeinen zu höherer Genauigkeit bei der Sequenzklassifikation, verlangt jedoch nach speziellen Klassifikationsansätzen, welche im Folgenden kurz besprochen werden.

Ein früher Ansatz zum Sequenztagging sind **Hidden-Markov-Modelle** (Rabiner 1989). Dieses mathematisch wohlfundierte und auch effizient implementierbare Modell wird jedoch kaum noch für Sequenzklassifikation verwendet, da es als generatives Modell nicht mit beliebigen Merkmals-Repräsentationen (Abschn. 6.1) umgehen kann, womit ein starkes Signal zur Charakterisierung von Wörtern wegfällt. Insbesondere bei Wörtern, welche nicht in der Trainingsmenge enthalten sind, muss ein Hidden-Markov-Modell das zugehörige Label einzig aus der Labelsequenz der benachbarten Wörter herleiten und kann nicht Merkmale wie Großschreibung oder Buchstaben-N-Gramme verwenden. Dies wiederum können diskriminative Modelle wie Maximum-Entropy-Markov-Modelle (McCallum et al. 2000), welche positionsweise ausgeführt werden: Das eben für die letzte Wortposition ermittelte Label kann direkt als weiteres Merkmal für die Klassifikation des nächsten Wortes verwendet werden, zusätzlich zu beliebigen Merkmalen auf der Beobachtung um die momentane Position. Ein Nachteil dieser Modelle ist die lediglich lokale, schrittweise Klassifikation, ohne Optimierung der Wahrscheinlichkeit der Gesamtsequenz, was diese anfällig für unerwartete Teilbeobachtungen (wie z. B. fremdsprachliche Zitate in deutschem Text) und somit weniger robust als global optimierende Modelle macht, was den Einsatz von Conditional Random Fields (CRFs; Lafferty et al. 2001) motiviert.

CRFs vereinen die Vorteile beider vorangegangener Modelle – beliebige Merkmalsrepräsentationen und globale Optimierung. Sie sind das Mittel der Wahl für statistisch-überwachte featurebasierte Ansätze (Abschn. 3.1.2) zur Sequenzklassifikation und werden bei neuronalen Modellen als zusätzliche Ebene eingesetzt, um Abhängigkeiten zwischen Labels direkt zu modellieren. Die CRF-Schicht modelliert Abhängigkeiten zwischen den Labels entsprechenden Zufallsvariablen; in ihrer einfachsten Form, dem Linear Chain CRF, werden jeweils die gegenseitigen Abhängigkeiten zwischen zwei benachbarten Labels auf der Beobachtungssequenz modelliert. Im Training werden zwei Elemente gelernt: einerseits die Label-Klassifikation von Datenpunkten in der Sequenz auf Basis der Repräsentation der Beobachtung, andererseits die bedingten Wahrscheinlichkeiten der aufeinanderfolgenden Labels. Diese beiden Elemente werden

bei der Sequenzklassifikation von neuem Text laufzeiteffizient mithilfe des **Viterbi-Algorithmus** (Viterbi 1967) miteinander verrechnet, um einer Beobachtung ihre Labelsequenz zuzuweisen. Dieser Mechanismus wird in Abb. 6.10 am Beispiel POS-Tagging verdeutlicht. Für jedes Wort in der Beobachtungssequenz gibt es eine sogenannte Likelihood, welche die Wahrscheinlichkeit aller Tags für dieses Wort angibt; diese ist in der Abbildung mit grauen Balken dargestellt. Je nach Modell wird die Likelihood durch generative Wahrscheinlichkeiten pro Tag-Zustand (beim Hidden-Markov-Modell), eine Wahrscheinlichkeitsverteilung eines Klassifikators (beim CRF) oder eine Pseudowahrscheinlichkeitsverteilung aus einer Softmax-Schicht (in neuronalen Netzen wie z. B. LSTMs oder Transformern) ermittelt. Ferner gibt es die sogenannten Prior-Wahrscheinlichkeiten, welche die Übergänge in der Labelsequenz ohne Berücksichtigung der Beobachtung modellieren; diese sind in der Abbildung als Wahrscheinlichkeiten auf den Pfeilen eingetragen, z. B.  $P(\text{ADV} | \text{NOUN})$  als die Wahrscheinlichkeit, nach NOUN ein ADV zuzuweisen.

Der Viterbi-Algorithmus (siehe z. B. Jurafsky und Martin 2020 für eine formalere Definition) geht sequenziell von links nach rechts vor und berechnet für jede Wortposition die maximale Wahrscheinlichkeit für jedes mögliche Label, indem die jeweils pro Label an der vorherigen Position ermittelte Wahrscheinlichkeit mit der entsprechenden Übergangswahrscheinlichkeit (Pfeile) multipliziert wird. Im Beispiel wird



**Abb. 6.10** Beispiel für die globale Optimierung der Labelsequenz mit dem Viterbi-Algorithmus, welcher in Hidden-Markov-Modellen, CRFs und in CRF-Schichten auf neuronalen Netzwerken zum Einsatz kommt

die Wahrscheinlichkeit für das Label „ADV“ für Wort „ein“ ermittelt, indem die Wahrscheinlichkeiten aller solchen möglichen Wege verglichen werden und der höchste Wert pro Label für die Folgeschritte verwendet wird, außerdem wird gespeichert, welcher dieser Wege zu der höchsten Wahrscheinlichkeit geführt hat (in der Abb. 6.10 mit einem Stern markiert); im Beispiel für ADV bei „ein“ der Weg von NOUN bei „Boden“. Für die Weiterverarbeitung wird nun der Wert mit der Likelihood (grauer Balken) multipliziert, der Algorithmus schreitet eine Position voran. Am Ende wird der höchste Wert ausgewählt, im Beispiel ist das der Wert bei PUNC für „.“ – nicht notwendigerweise das Label mit der größten Likelihood. Von diesem Label ausgehend wird rückwärts unter Verwendung der maximierenden Wege (dicke Pfeile) die Labelsequenz abgelesen und der Sequenz zugewiesen.

Das Beispiel zeigt eine Situation, bei der die Verwendung der Labels bei der Sequenzklassifikation vorteilhaft ist: Würden nur die Likelihoods (graue Balken) verwendet, würde für „ein“ hier DET zugewiesen, erst die Modellierung der Übergänge (Pfeile) sorgt für das hier richtige Label ADV.

Eine solche CRF-Schicht wird auch in den momentan besten Ansätzen für Sequenzklassifikation oft als oberste Schicht auf einem neuronalen Netzwerk eingesetzt: BiLSTM-CRFs (Huang et al. 2015) und Transformer-CRFs, z. B. Souza et al. (2020). Beide versuchen, eine möglichst gute Repräsentation von Wörtern in ihrem Kontext zu erreichen, welche dann mittels Softmax in eine Likelihood-Verteilung überführt werden kann (graue Balken in Abb. 6.10) Beim **BiLSTM**-Ansatz modellieren rekurrente LSTM-Zellen (Hochreiter und Schmidhuber 1997) die Historie in der Beobachtung aus beiden Richtungen und sorgen gemeinhin für bessere Repräsentationen der Wortbeobachtungen im Kontext, als dies mit featurebasierten Methoden erreicht werden kann. Ein Schlüssel dazu besteht durch den durch die rekurrente Architektur unbegrenzten Kontext, im Gegensatz zu N-Gramm-Sprachmodellen (Abschn. 5.5). Die LSTM-Zellen, welche als Alternative zu Perzeptronen als Neuronen eingesetzt werden, können eine Regulierung lernen, welche entscheidet, ob der innere Zustand der Zelle (in der Rolle eines Speichers) beibehalten oder durch die Beobachtung beeinflusst wird.

Transformer-Netzwerke, z. B. BERT (Abschn. 5.6) basieren auf dem **Attention-Mechanismus** (Bahdanau et al. 2015; Vaswani et al. 2017), welcher durch unüberwachtes Vtrainieren auf großen Korpora direkt den Einfluss von Wörtern in der Sequenz auf andere erlernt und daher für die Sequenzklassifikation nützliche kontext-abhängige Repräsentationen lernt. Hierbei ist es oft für die weitere Verarbeitung nicht entscheidend, ob POS-Tags expliziert werden, da deren syntaktische Information auch implizit in den Embeddings enthalten ist; derartige syntaktische Vorverarbeitungsschritte werden nur dann noch als überwachte Sequenzklassifikation durchgeführt, wenn man direkt an den Wörtern in gewissen Labelabfolgen interessiert ist, z. B. zur Taxonomie-extraktion (vgl. Abschn. 3.2.4).

Dies ist zum Beispiel der Fall bei der Extraktion von Eigennamen (vgl. Abschn. 3.2.5 und Anwendungen wie Abschn. 7.2), welche durch eine geeignete Kodierung der Labels als Sequenzklassifikation auf allen Wörtern eines Textes durchgeführt werden kann.

Johann	Sebastian	Bach	wurde	am	21.	März	1685	in	Eisenach	geboren	.
B-PER	I-PER	I-PER	O	O	B-TIMEX	I-TIMEX	I-TIMEX	O	B-LOC	O	O
Als	Leiter	der	Thomaner	schrieb	er	Die	Kunst	der	Fuge	.	.
O	O	O	B-ORG	O	O	B-MISC	I-MISC	I-MISC	I-MISC	O	

**Abb. 6.11** Kodierung von Namenserkennung mit Sequenzlabels im BIO-Schema für den Personennamen „Johann Sebastian Bach“, den Zeitausdruck „21. März 1685“, den Ort „Eisenach“, die Organisation „Thomaner“ und den als sonstigen Namen klassifizierten Musikstücktitel „Die Kunst der Fuge“

Um ganz allgemein Annotationen zu kodieren, die mehrere Wörter überspannen, kann das **BIO-Schema** (Begin-Inside-Outside) oder eine Variante davon (Ramshaw und Marcus 1995; Wu et al. 2006) verwendet werden. Bei Eigennamen wie *Klara Schumann* wird das erste Token des Namens mit einem B-Präfix getaggt, z. B. B-PERSON, alle Weiteren mit einem I-Präfix, z. B. I-PERSON. Alle Wörter, welche nicht Teile eines Namens sind, bekommen das Label „O“. Abb. 6.11 zeigt ein Beispiel für Namensannotation kodiert mit dem BIO-Schema.

Zusammenfassend ist Sequenzklassifikation oft die Methode der Wahl, wenn in Abhängigkeit vom Kontext einzelnen Wörten oder kurzen Spannen mehrerer Wörter automatisch eine Klasse zugewiesen werden soll.

## 6.9 Evaluation von Klassifikation

Die Evaluation von Klassifikatoren erfolgt durch einen Vergleich von manuell annotierten Testdaten mit den automatischen Klassifikationen auf diesen, wobei der Klassifikator natürlich die manuellen Annotationen nicht in seine Entscheidung mit einbeziehen darf (Abschn. 6.6). Durch das Vorhandensein dieser Referenzdaten, auch **Gold-Daten** genannt, gestaltet sich die Evaluation von Klassifikation einfacher und direkter als die Evaluation von Clustering (Abschn. 6.5), wo diese Daten nicht vorhanden sind.

Je nach Beschaffenheit der Aufgabe und der Charakteristiken des Datensatzes sind hier verschiedene **Evaluationsmaße** adäquat. Im Folgenden werden die gängigsten Maße für die Evaluation von überwachtem maschinellen Lernen im Text Mining dargestellt. Hauptsächlich werden mengenorientierte Maße zum Messen der Qualität von 2-Klassen- oder Mehrklassen-Klassifikatoren verwendet, welche wir im Detail definieren. Kurz erwähnt werden anschließend auch andere Maße, z. B. für die Evaluation gereihter Ergebnisse wie bei Suchergebnissen für den Vergleich kurzer Texte z. B. für die Evaluation automatischer Zusammenfassungen.

Die meisten Evaluationskennzahlen in der Sprachtechnologie haben als Konvention gemeinsam, dass sie Zahlenwerte zwischen 0 und 1 liefern, welche oftmals als Prozentwerte angegeben werden.

### 6.9.1 Mengenorientierte Evaluationsmaße

Mengenorientierte Evaluationsmaße werden zur Evaluation von Klassifikatoren eingesetzt, welche den Testdaten jeweils ein Label zuweisen, sodass die Daten in eine Menge pro Label unterteilt werden. Die auf diese Weise automatisch ermittelten Mengen werden mit den in den manuell annotierten Testdaten gegebenen Gold-Mengen verglichen. Basis der mengenorientierten Evaluationsmaße sind folgende vier Kennzahlen, welche pro bekanntes Label für die automatische Zuweisung durch einen Klassifikator definiert sind:

- TP (true positive; tatsächlich positiv): Das Label wurde richtigerweise zugewiesen.
- FP (false positive; falsch positiv): Das Label wurde fälschlicherweise zugewiesen.
- TN (true negative; tatsächlich negativ): Das Label wurde richtigerweise nicht zugewiesen.
- FN (false negative; falsch negativ): Das Label wurde fälschlicherweise nicht zugewiesen.

Hieraus können verschiedene Maße konstruiert werden. Das einfachste, direkteste Maß für die Qualität eines Klassifikators ist die **Accuracy** (Treffergenauigkeit). Diese gibt den Anteil richtiger Klassifikationsentscheidungen bzgl. der Anzahl getester Klassifikationsentscheidungen wieder und ist in Gl. 6.9 definiert; falls dies nicht pro Label sondern insgesamt berechnet werden soll, reicht es, die richtigen Zuweisungen durch die Gesamtzahl an Zuweisungen (entspricht der Größe der Testmenge) zu teilen.

$$\text{acc} = \frac{TP + TN}{TP + TN + FP + FN} \quad (\text{Gl. 6.9})$$

Während das Accuracy-Maß eine intuitive, leicht verständliche Kennzahl darstellt, ist es nur für einigermaßen balancierte Klassen aussagekräftig: Insbesondere, wenn eine für die Anwendung interessante Klasse selten ist, liefert Accuracy oftmals viel zu hohe Werte. Wenn z. B. in einem stark verspamten Postfach bei einem Spam-Filter nur 10 % E-Mails ankommen, welche nicht als Spam herausgefiltert werden sollen, würde ein übereifriger Filter, der alle Spam-E-Mails als Spam klassifiziert und drei Viertel der Nicht-Spam-E-Mails ebenfalls als Spam, insgesamt eine Accuracy von 92,5 % erreichen, was gut klingt, aber für den praktischen Einsatz hier komplett unbrauchbar ist.

Daher werden in vielen Kontexten die Maße **Precision** (Genauigkeit) und **Recall** (Trefferquote bzw. Sensitivität) verwendet, welche häufig zum **F-Wert** (van Rijsbergen 1979) zusammengefasst werden (siehe auch Abschn. 6.5). Diese Werte sind wiederum pro Label definiert. Precision (Gl. 6.10) gibt den Anteil richtiger Klassifikationen auf der Gesamtmenge der automatisch klassifizierten Menge pro Label an, Recall (Gl. 6.11) den Anteil richtiger Klassifikationen auf der Gesamtmenge der manuell annotierten, bekannten Elementen der dem Label entsprechenden Menge. Der F-Wert (Gl. 6.12) ist das

harmonische Mittel von Precision und Recall und erreicht nur dann einen großen Wert, wenn beide seiner Eingangsgrößen groß sind.

$$P = \frac{TP}{TP + FP} \quad (\text{Gl. 6.10})$$

$$R = \frac{TP}{TP + FN} \quad (\text{Gl. 6.11})$$

$$F = \frac{2 \cdot P \cdot R}{P + R} \quad (\text{Gl. 6.12})$$

Für unser Beispiel des übereifrigen Spamfilters ist für Spam die Precision 92,3 % und der Recall 100 % bei  $F=0,96$ , für Nicht-Spam ergibt sich eine Precision von 100 % und ein Recall von 25 %, was einen niedrigen F-Wert für die kleinere Klasse von 0,4 ergibt. Es existieren Varianten des F-Wertes, welche die Eingangsgrößen verschieden gewichten, daher wird die hier definierte ungewichtete Variante auch als F1-Wert bezeichnet.

Sollen die Maße für verschiedene Labels kombiniert werden, um insgesamt eine Aussage zu treffen, können diese per Micro-Averaging oder per Macro-Averaging kombiniert werden. Beim Micro-Averaging werden die Klassen entsprechend ihrer Größe gewichtet, beim Macro-Averaging nicht, sodass im letzteren Fall alle Klassen als gleich wichtig in die Berechnung eingehen. Für unser Beispiel ergibt sich ein Micro-Average F-Wert von 0,904 und ein Macro-Average F-Wert von 0,68; letzterer gibt gut wieder, dass unser hier diskutierter Spamfilter eher unbrauchbar ist.

Bei der Evaluation von Spannenannotation (Abschn. 6.8) werden oftmals F-Werte pro Zielklasse angegeben, z. B. bei der Namenserkennung (Abschn. 3.2.5) für die Klassen Person, Organisation usw. Da Namen nur einen geringen Teil von gesamten Texten ausmachen, ist auch hier die Accuracy als Evaluationsmaß ungeeignet. Ferner ist hier noch auf den Unterschied zwischen den Labels zur Sequenzklassifikation und den eigentlichen Annotationen der Spannen zu achten. Als Beispiel soll eine automatische Klassifikation für die korrekte Namensannotation in Abb. 6.11 dienen, welche in Abb. 6.12 angegeben ist.

Im Beispiel wurde *Bach* nicht als Personennamensteil und im zweiten Satz *Die* nicht als Namensteil für das Musikstück erkannt. Die auf den Labels pro Wort berechnete Accuracy, welche hier der Micro-Averaged Precision entspricht, ist  $21/23=91,3\%$ , da nur zwei Labels falsch sind. Adäquater und normalerweise für Spannenannotation verwendeten jedoch Precision und Recall auf Spannenebene: Hier sind zwei der fünf gesuchten Spannen falsch, entsprechend ergibt sich jeweils ein Wert von 0,6.

Schließlich soll noch die **ROC-Kurve** (z. B. Fawcett 2004) als Charakterisierung von automatischen Methoden erwähnt werden. Falls die automatischen Klassifikationen mit einer Konfidenz seitens des Klassifikators behaftet sind, ergeben sich je nach Schwellwert für die Konfidenz verschiedene Konstellationen; i. Allg. beobachten wir für eine

Johann	Sebastian	Bach	wurde	am	21.	März	1685	in	Eisenach	geboren	.
B-PER	I-PER	o	o	o	B-TIMEX	I-TIMEX	I-TIMEX	o	B-LOC	o	o
Als	Leiter	der	Thomaner	schrieb	er	Die	Kunst	der	Fuge	.	.
o	o	o	B-ORG	o	o	o	B-MISC	I-MISC	I-MISC	o	o

**Abb. 6.12** Automatische Annotation des Beispiels aus Abb. 6.11 zur Diskussion der Evaluation von Labels gegenüber Spannenannotationen

hohe Konfidenz eine hohe Precision und einen niedrigen Recall, für eine niedrige Konfidenz ist es oft genau umgekehrt. Die ROC-Kurve verdeutlicht diesen Zusammenhang über alle möglichen Schwellwerte und liefert so eine Zusammenfassung der Einstell- und Balancierungsmöglichkeiten des Klassifikators für den praktischen Einsatz.

### 6.9.2 Andere Evaluationsmaße im Text Mining

Auch wenn Precision, Recall und F-Wert ursprünglich für das Information Retrieval definiert wurden, ist ihre Anwendung für die Evaluation der Qualität gereihter Suchergebnisse oder anderer **Rankings** denkbar ungeeignet. Insbesondere falls die Anwendung der Websuche ähnelt, also ein Informationsbedürfnis auf einer großen, ggf. redundanten Kollektion von Texten gestillt werden soll, reicht es, die ersten paar Ergebnisse zu betrachten; da der Recall ein Wissen über alle für eine Suche relevanten Texte erfordert, kann dieser nicht mit vertretbarem Aufwand ermittelt werden.

Für die Evaluation von Rankings (siehe Manning et al. 2008, Kap. 8) werden daher Varianten von Precision eingesetzt. Bei Precision@ $k$  werden nur die  $k$  am höchsten gereihten Ergebnisse betrachtet, der relative Anteil relevanter Ergebnisse bestimmt dann den Wert. Mean Average Precision (MAP) fasst mehrere Werte von  $k$  dahingehend zusammen, dass Precision@ $k$ -Werte für  $k$  von 1 bis  $n$  geeignet gemittelt werden (NIST 2004). Um der Situation in manchen Datensätzen Rechnung zu tragen, dass es nur wenige oder ggf. für manche Anfragen keine relevanten Ergebnisse gibt, kann das NDCG-Maß (Järvelin und Kekäläinen 2002) eingesetzt werden: Hier wird das aufgrund der bekannten relevanten Ergebnisse optimale Ranking mit dem automatisch erreichbaren verglichen; wenn z. B. nur ein relevantes Ergebnis bekannt ist und dies als erstes zurückgegeben wird, erreicht hier NDCG den optimalen Wert 1, während z. B. die Precision@5 nur 0,2 beträgt. NDCG kann ferner auch mit abgestuften Relevanzbewertungen umgehen und ist daher oft das Mittel der Wahl für entsprechend annotierte Datensätze.

Eine weitere wichtige Anwendung für Evaluationsmaße ist Textähnlichkeit, z. B. beim Vergleich manueller und automatischer Zusammenfassungen oder Übersetzungen. Bei solchen Aufgaben, welche eine Generierung von Text beinhalten, kann nicht erwartet

werden, dass die exakte bekannte Lösung hervorgebracht wird, auch weil es durch die Variabilität von Sprache auch innerhalb optimaler Lösungen sehr viel Varianz gibt. Hier haben sich Maße wie BLEU (Papineni et al. 2002) und ROUGE (Lin 2004) etabliert, welche die relative Anzahl überlappender Wort-N-Gramme zwischen automatischem und händischem Text messen, was trotz der Einfachheit sinnvolle Evaluationsergebnisse produziert, insbesondere wenn über viele Datenpunkte aggregiert wird. Eine Weiterentwicklung hiervon ist BERTScore (Zhang et al. 2020), welche nicht nur exakte Wortübereinstimmungen, sondern Ähnlichkeiten der kontextualisierten Word Embeddings (Abschn. 5.6) der Vergleichstexte mit einbezieht und somit in einem gewissen Rahmen semantische Ähnlichkeiten berücksichtigt.

---

## 6.10 Neuronale Methoden: End-to-End, Transfer Learning

Nun sollen Charakteristiken besprochen werden, wie sie für Klassifikationsansätze typisch sind, welche mit neuronalem Lernen realisiert werden. Die Extraktion der Merkmalsvektoren (Abschn. 6.1) und die Klassifikation (Abschn. 6.6) können unabhängig voneinander durchgeführt und dabei neuronale oder klassische Methoden des maschinellen Lernens kombiniert werden. Jedoch bietet das rein neuronale Paradigma Vorteile, wenn die Merkmalsextraktion von der zu lernenden Aufgabe oder von einer ähnlich gelagerten Aufgabe beeinflusst wird. Im Folgenden skizzieren wir hierzu das End-to-End-Lernen (Ende-zu-Ende-Lernen) und das Transferlernen (transfer learning).

### 6.10.1 End-to-End-Lernen

Im Gegensatz zum Ansatz der linguistischen Pipeline (Abschn. 3.2), in der wohldefinierte Schritte sukzessiv den Eingangstext mit linguistisch motivierten Annotationen anreichern, um genügend Merkmale für die quantitative Analyse oder eine Klassifikation zur Verfügung zu haben, verzichtet das End-to-End-Lernen auf das Explizieren der Zwischenschritte: Alle Schritte von Eingabe zur Ausgabe werden in ein einziges Modell integriert, welches direkt mit den der Aufgabe entsprechenden Daten trainiert wird. Der große Vorteil dieser Herangehensweise besteht darin, dass z. B. für neue Sprachen oder Sachgebiete keine spezielle linguistische Pipeline mehr erstellt werden muss, welche viele allgemeine Merkmale extrahiert, von denen ggf. nur wenige mit der entsprechenden Aufgabe korrelieren. Der Preis ist neben der Intransparenz der algorithmischen Entscheidung der Datenhunger dieses Vorgehens: Es müssen genügend Trainingsdaten vorhanden sein, um die vielen Parameter (die Gewichte des Netzwerkes, vgl. Abschn. 3.1.3) sinnvoll schätzen zu können. Außerdem muss beim Design der Netzwerkarchitektur darauf geachtet werden, dass zu modellierende Phänomene auch durch die Architektur erlernbar sind.

Als Beispiel zur Illustration von End-to-End-Lernen greifen wir die Koreferenzauflösung aus Abschn. 6.1 wieder auf, wo in Tab. 6.1 Oberflächenmerkmale nach Durrett und Klein (2013) diskutiert wurden, welche linguistisch motivierte Phänomene zum Zwecke der Charakterisierung von Antezedenten und Anaphern abbilden. Nun gilt es, derartige Merkmale in einer neuronalen Architektur abzubilden, welche direkt auf annotierten Trainingsdaten für die Koreferenzauflösung trainiert werden können. Es gilt also abzubilden, wie Textspannen als Kandidaten ausgewählt werden können und wie deren Zusammengehörigkeit klassifiziert werden kann. In der hier vereinfachten Darstellung folgen wir Lee et al. (2017), welche das erste End-to-End-Koreferenzsystem vorstellten.

Abb. 6.13 verdeutlicht die Architektur, welche nun von unten nach oben besprochen wird. In der Eingangsschicht werden Wörter (hier auch auf Basis der enthaltenen Buchstaben-N-Gramme) in Embeddings umgewandelt (siehe Abschn. 5.5 und 5.6). Darauf folgt ein bidirektionales LSTM (siehe Abschn. 3.1.3), welches in beide Richtungen die Sequenz modelliert und unmittelbar vorangegangene und folgende Wörter am stärksten in die Vektor-Repräsentation mit einbezieht. Dies sorgt an den jeweiligen Positionen für Merkmale für umliegende Wörter und für die Wörter selbst. Bei Koreferenz müssen die Kandidaten identifiziert werden, welche ko-referieren, und diese können aus mehreren konsekutiven Wörtern bestehen. In der End-to-End-Modellierung werden hierfür konsequenterweise alle möglichen Spannen bis zu einer maximalen Länge betrachtet; in der Abb. 6.13 sind diese nur bis zur Länge 2 dargestellt. Repräsentationen von Spannen werden durch die Konkatenation von drei Bi-LSTM-Repräsentationen konstruiert:

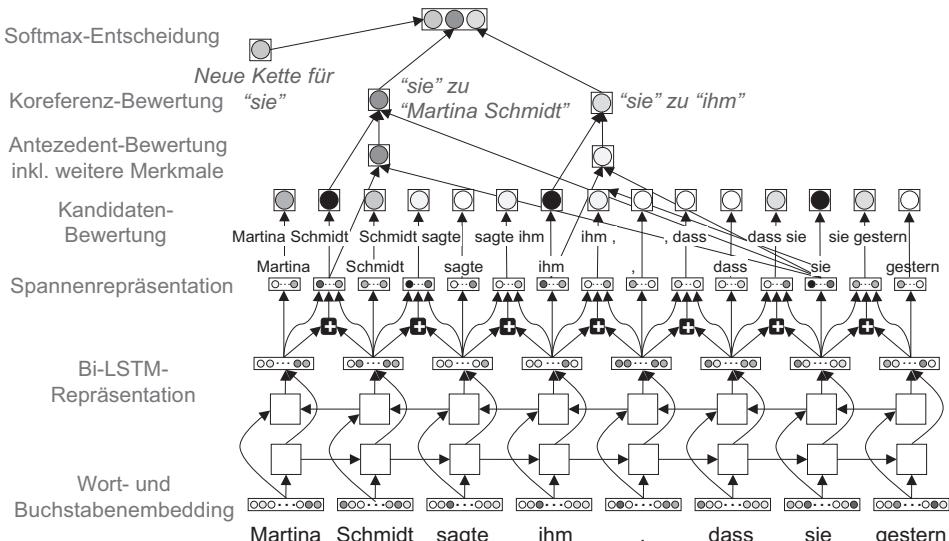


Abb. 6.13 End-to-End-Architektur für die Koreferenzauflösung nach Lee et al. (2017)

die Repräsentation des ersten Wortes, die Summe der Repräsentationen über alle enthaltenen Wörter und die Repräsentation des letzten Wortes (dies gilt auch für Spannen der Länge 1, dann sind alle drei gleich). Darin sind wiederum Kontextinformationen enthalten. Auf diesen Spannenrepräsentationen wird nun die Kandidaten-Bewertung überwacht trainiert. Dieselben Spannenrepräsentationen werden jedoch auch in paarweiser Betrachtung für eine Antezedent-Bewertung herangezogen: Für jeden Kandidaten im Training bzw. für jede Spanne in der Anwendung, welche eine ausreichend hohe Kandidatenbewertung bekommt, werden Paare mit vorangegangenen Kandidaten gebildet, um nun eine Bewertung vorzunehmen, ob die vorangegangene Spanne ein guter Antezedent für die aktuelle Spanne sein könnte. Im Beispiel in der Abb. 6.13 gibt es zwei Kandidatenspannen „Martina Schmidt“ und „ihm“, für welche deren Kompatibilität mit der gerade aktuellen Spanne „sie“ gelernt bzw. in der Anwendung bewertet wird. In den obersten Schichten werden noch die Kandidaten-Bewertungen und die Antezendent-Bewertungen zu einer Koreferenz-Bewertung zusammengefügt; in der letzten Schicht wird dann mit Softmax entschieden, zu welcher vorangegangenen Spanne unsere aktuelle Spanne ko-referiert oder ob eine neue Koreferenzkette eröffnet werden soll. Durch die Repräsentation mit Embeddings und daraus abgeleiteten Repräsentationen werden die in Abschn. 6.1 diskutierten Oberflächenmerkmale gut abgedeckt: sowohl kontextuelle Merkmale durch die LSTMs als auch Stringähnlichkeiten und semantische Ähnlichkeiten durch Wort- und Buchstabenembeddings in der Eingabeschicht. Allein Merkmale wie das Genre des Textes, der Sprecher (bei direkter Rede) und die Distanz zur Modellierung von Salienz werden bei Lee et al. (2017) nicht durch die Architektur gelernt, sondern fließen direkt als manuell konstruierte Merkmale in die Antezedenz-Bewertung ein, weswegen die hier diskutierte Architektur bei genauem Hinsehen doch nicht vollständig End-to-End ist. Gegenüber dem Ansatz von Durrett und Klein (2013) werden vor allem semantische Ähnlichkeiten besser modelliert, was die Koreferenz von *der Report* zu *das Dokument* ermöglicht. Dies führt jedoch auch zu falschen Koreferenzen, z. B. sind *Pilotin* und *Steward* semantisch sehr ähnlich, jedoch nie Koreferent.

Zusammengefasst ist End-to-End-Lernen ein attraktives Paradigma, insbesondere falls ansonsten nötige Vorverarbeitungsschritte nicht zur Verfügung stehen und genug Trainingsdaten vorhanden sind. Die Erstellung einer geeigneten Architektur erfordert jedoch eine hohe Expertise und weiterhin Wissen über die zu modellierenden linguistischen Phänomene, welche nicht mehr direkt modelliert werden, jedoch von erfolgreichen neuronalen Architekturen erlernbar sein müssen.

## 6.10.2 Transferlernen

Transferlernen nutzt neben den eigentlichen Zieldaten zum Lernen eines Klassifikators auch Hilfsdaten, um einige der Parameter des Netzwerkes zu lernen oder für diese Parameter gute Startpunkte zu setzen. Dies führt im Erfolgsfall zu besseren Ergebnissen bzw.

zur Reduktion der nötigen Trainingsdaten, da ein Teil der Schätzarbeit bereits durch die Hilfsdaten erledigt werden kann. Für das Transferlernen, also die Kombination von Hilfs- und Zieldaten, gibt es viele Schemata, welche wir hier nur kurz skizzieren; für eine eingehendere Darstellung siehe Ruder (2019), Ruder et al. (2019).

Ein sehr häufig eingesetztes Schema zum Transferlernen ist unüberwachtes (auch selbst-überwacht genanntes) Vortrainieren für Wortrepräsentationen: Auf großen unannotierten Korpora können Sprachmodelle trainiert werden, welche statische oder kontextualisierte Embeddings (Abschn. 5.6) für die Eingabeschicht induzieren. Dies ist bereits Transferlernen, da das im Trainingskorpus für das Sprachmodell enthaltene sprachliche Wissen für unsere Zielaufgabe zur Verfügung steht. In unserem Beispiel zur Koreferenz könnten vortrainierte Embeddings für ein besseres Wissen über semantische Ähnlichkeiten sorgen; diese sollten jedoch für die vorliegende Aufgabe nachtrainiert werden, siehe das Beispiel *Pilotin-Steward* oder auch die distributionell sehr ähnlichen Pronomen *sie* und *ihm*. Dieses Nachtrainieren bezeichnet man als **Fine-Tuning** der Embeddings (Howard und Ruder 2018): Die vortrainierten Embeddings werden hier nicht einfach importiert, sondern als Teil der Gesamtarchitektur verstanden und können während des Trainings auf der eigentlichen Aufgabe weiter mittrainiert werden.

Neben Transfer auf Basis der Embeddings sind beliebte Schemata für das Transferlernen das Trainieren von mehreren Aufgaben auf teilweise geteilten Architekturen: Für ähnliche Aufgaben wie z. B. POS-Tagging und Chunking (Abschn. 3.2.4) oder dem semantischen Parsen in verschiedenen Formalismen (Abschn. 3.2.5) bietet sich **Multitask-Lernen** (Caruana 1997) an: Hier gibt es eine Basisarchitektur, welche die Eingabeschicht und die ersten unteren Schichten zwischen den verschiedenen Aufgaben teilt und darauf aufbauend verschiedene Klassifizierungsschichten für die jeweiligen Aufgaben trainiert. Hierbei gibt es oftmals eher kleine Zieldaten und deutlich größere Hilfsdaten zum Zwecke des Multitask-Lernens. Damit wird erreicht, dass die gemeinsamen Schichten durch die größere Trainingsdatenmenge stabiler trainiert werden.

Der Erfolg von Transferlernen hängt sowohl von der Ähnlichkeit zwischen den verwendeten Aufgaben ab, als auch von der Art des Transfers und dem Aufbau des Trainings, welches z. B. zwischen den Aufgaben alternierend oder sequentiell stattfinden kann. Transferlernen lohnt sich im Allgemeinen, falls es große Trainingsdaten für eine Hilfsaufgabe gibt, die ähnlich genug zur Zielaufgabe ist; für ein Ähnlichkeitsmaß zwischen annotierten Datensätzen zur Vorhersage geeigneter Hilfsaufgaben für die Sequenzklassifikation (Abschn. 6.8) siehe Schröder und Biemann (2020).

---

## Literatur

- Allan, J., Carbonell, J., Doddington, G., Yamron, J., Yang, Y.: Topic detection and tracking pilot study: Final report. In: Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop, Lansdowne, VA, USA, S. 194–218 (1998)

- Amigó, E., Gonzalo, J., Artiles, J., Verdejo, F.: A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Inf. Retrieval* **12**(4), 461–486 (2009). <https://doi.org/10.1007/s10791-008-9066-8>
- Andrzejewski, D., Zhu, X., Craven, M.: Incorporating domain knowledge into topic modeling via dirichlet forest priors. In: Proceedings of the 26th Annual International Conference on Machine Learning (ICML '09), Montréal, Canada, S. 25–32. Association for Computing Machinery, New York, NY, USA (2009). <https://doi.org/10.1145/1553374.1553378>
- Bagga, A., Baldwin, B.: Algorithms for scoring coreference chains. In: Proceedings of the linguistic conference workshop at the first international conference on language resources and evaluation, Granada, Spain, S. 563–566. European Language Resources Association (ELRA) (1998)
- Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA <https://arxiv.org/pdf/1409.0473.pdf> (2015). Zugegriffen: 21. Juni 2021
- Benikova, D., Biemann, C., Reznicek, M.: NoSta-D named entity annotation for german: guidelines and dataset. In: Calzolari, N., Choukri, K., Declerck, T., Loftsson, H., Maegaard, B., Mariani, J., Moreno, A., Odijk, J., Piperidis, S. (Hrsg.) Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC 2014), Reykjavik, Iceland, S. 2524–2531. [http://www.lrec-conf.org/proceedings/lrec2014/pdf/276\\_Paper.pdf](http://www.lrec-conf.org/proceedings/lrec2014/pdf/276_Paper.pdf) (2014). Zugegriffen: 8. Febr. 2021
- Bernstein, M., Little, G., Miller, R.C., Hartmann, B., Ackerman, M., Karger, D.R., Crowell, D., Panovich, K.: Soylent: A word processor with a crowd inside. In: Perlin, K. (Hrsg.) Proceedings of the 23nd annual ACM symposium an User interface software and technology (UIST), S. 313–322. ACM Press, New York, NY, USA (2010). <https://doi.org/10.1145/1866029.1866078>
- Biemann, C.: Chinese whispers – an efficient graph clustering algorithm and its application to natural language processing problems. In: Mihalcea, R., Radev, D. (Hrsg.) Proc. Of TextGraphs, New York, NY, USA, S. 71–80. Association for Computational Linguistics, Stroudsburg, PA, USA. <https://www.aclweb.org/anthology/W06-3812/> (2006). Zugegriffen: 12. Mai 2021
- Biemann, C., Quasthoff, U., Heyer, G., Holz, F.: ASV Toolbox: A modular collection of language exploration tools. In: Calzolari, N., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Tapias, D. (Hrsg.) Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08), Marrakech, Morocco, S. 1760–1767. European Language Resources Association (ELRA) (2008)
- Biemann, C., Bontcheva, K., Eckart de Castilho, R., Gurevych, I., Yimam, S.M.: Collaborative web-based tools for multi-layer text annotation. In: Ide, N., Pustejovsky, J. (Hrsg.) Handbook of Linguistic Annotation, S. 229–256. Springer, Dordrecht (2017). [https://doi.org/10.1007/978-94-024-0881-2\\_8](https://doi.org/10.1007/978-94-024-0881-2_8)
- Blei, D.M., Lafferty, J.D.: Correlated topic models. In: Proceedings of the 23rd International Conference on Machine Learning (ICML '06), Pittsburgh, PA, USA, S. 113–120. Association for Computing Machinery, New York, NY, USA (2006). <https://doi.org/10.1145/1143844.1143859>
- Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. *J. Mach. Learn. Res.* **3**, 993–1022 (2003)
- Boyd-Graber, J., Blei, D.M.: Multilingual topic models for unaligned text. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, S. 75–82. AUAI Press, Arlington, VA, USA. <https://dl.acm.org/doi/pdf/https://doi.org/10.5555/1795114.1795124> (2009). Zugegriffen: 17. Mai 2021

- Brown, P.F., Pietra, V.J., de Souza, P.V., Lai, J.C., Mercer, R.L.: Class-based n-gram models of natural language. *Comput. Linguist.* **18**(4), 467–479 (1992)
- Capdevila, J., Cerquides, J., Nin, J., Torresa, J.: Tweet-SCAN: An event discovery technique for geo-located tweets. *Pattern Recogn. Lett.* **93**, 58–68 (2016). <https://doi.org/10.1016/j.patrec.2016.08.010>
- Caruana, R.: Multitask learning. *Mach. Learn.* **28**(1), 41–75 (1997). <https://doi.org/10.1023/A:1007379606734>
- Cecchini, F.M., Riedl, M., Fersini, E., Biemann, C.: A comparison of graph-based word sense induction clustering algorithms in a pseudoword evaluation framework. *Lang. Resour. Eval.* **52**(3), 733–770 (2018). <https://doi.org/10.1007/s10579-018-9415-1>
- Cha, S.-H.: Comprehensive survey on distance/similarity measures between probability density functions. *Int. J. Math. Models Methods Appl. Sci.* **1**(4), 300–307 (2007)
- Christodoulopoulos, C., Goldwater, S., Steedman, M.: Two decades of unsupervised POS induction: How far have we come? In: Li, H., Márquez, L. (Hrsg.) *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Cambridge, MA, USA, S. 575–584. <https://www.aclweb.org/anthology/D10-1056> (2010). Zugegriffen: 10. Mai 2021
- Cordeiro, M., Gama, J.: Online social networks event detection: A survey. In: Michaelis, S., Piatkowski, N., Stolpe, M. (Hrsg.) *Solving large scale learning tasks. Challenges and algorithms* S. 1–41. *Lecture Notes in Computer Science*, Bd. 9580. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-41706-6\\_1](https://doi.org/10.1007/978-3-319-41706-6_1)
- Durrett, G., Klein, D.: Easy victories and uphill battles in coreference resolution. In Yarowsky, D., Baldwin, T., Korhonen, A., Livescu, K., Bethard, S. (Hrsg.): *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Seattle, Washington, USA, S. 1971–1982. <https://www.aclweb.org/anthology/D13-1203> (2013). Zugegriffen: 29. April 2021
- Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: *Second International Conference on Knowledge Discovery and Data Mining*, Portland, OR, USA, S. 226–231, <https://citeseerx.ist.psu.edu/viewdoc/summary%3Fdoi=10.1.1.121.9220> (1996). Zugegriffen: 10. Mai 2021
- Ester, M., Kriegel, H.P., Sander, J., Wimmer, M., Xu, X.: Incremental clustering for mining in a data warehousing environment. In: *VLDB '98: Proceedings of the 24rd International Conference on Very Large Data Bases*, New York City, NY, USA, S. 323–333, <http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=0FD61A20C1291E3DFBA7E3ADA4CB9FC5%3Fd oi=10.1.1.18.7933%26rep=rep1%26type=pdf> (1998). Zugegriffen: 10. Mai 2021
- Fawcett, T.: ROC graphs: Notes and practical considerations for data mining researchers. *Pattern Recogn. Lett.* **31**(8), 1–38 (2004)
- Fleiss, J.L., Cohen, J.: The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educ. Psychol. Measur.* **33**(3), 613–619 (1973). <https://doi.org/10.1177/001316447303300309>
- Friedrich, A., Biemann, C.: Digitale Begriffsgeschichte? Methodologische Überlegungen und exemplarische Versuche am Beispiel moderner Netzsemantik. *Forum Interdisziplinäre Begriffsgeschichte* **5**(2), 78–96 (2016)
- Grosz, B.J., Joshi, A.K., Weinstein, S.: Centering: A framework for modeling the local coherence of discourse. *Comput. Linguist.* **21**(2), 203–226 (1995)
- Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997). <https://doi.org/10.1162/neco.1997.9.8.1735>
- Holzinger, A.: Interactive machine learning (iML). *Informatik Spektrum* **39**(1), 64–68 (2016). <https://doi.org/10.1007/s00287-015-0941-6>

- Höppner, F., Klawonn, F., Kruse, R.: Fuzzy-Clusteranalyse. Verfahren für die Bilderkennung Klassifizierung und Datenanalyse. Computational Intelligence. Vieweg+Teubner, Wiesbaden (1997)
- Howard, J., Ruder, S.: Universal language model fine-tuning for text classification. In: Gurevych, I., Miyao, Y. (Hrsg.) Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Bd. 1: Long Papers), Melbourne, Australia, S. 328–339. Association for Computational Linguistics (2018). <https://doi.org/10.18653/v1/P18-1031>
- Howe, J.: The rise of crowdsourcing. Wired magazine **14**(6), 1–4 (2006)
- Huang, Z., Xu, W., Yu, K.: Bidirectional LSTM-CRF models for sequence tagging. <https://arxiv.org/pdf/1508.01991.pdf> (2015). Zugegriffen: 21. Juni 2021
- Hubert, L., Arabie, P.: Comparing partitions. J. Classif. **2**(1), 193–218 (1985). <https://doi.org/10.1007/BF01908075>
- Jähnichen, P.: Time dynamic topic models. Dissertation, Universität Leipzig. <https://ul.qucosa.de/api/qucosa%3A14614/attachment/ATT-0/> (2016). Zugegriffen: 17. Mai 2021
- Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of IR techniques. ACM Trans. Inf. Syst. **20**(4), 422–446 (2002). <https://doi.org/10.1145/582415.582418>
- Jurafsky, D., Martin, J.H.: Speech and language processing. Chapter A: Hidden Markov Models. <https://web.stanford.edu/~jurafsky/slp3/A.pdf> (2020). Zugegriffen: 21. Juni 2021
- Klie, J.-C., Bugert, M., Boullosa, B., Eckart de Castilho, R., Gurevych, I.: The INCEpTION platform: Machine-assisted and knowledge-oriented interactive annotation. In: Zhao, D. (Hrsg.) Proceedings of System Demonstrations of the 27th International Conference on Computational Linguistics (COLING 2018), Santa Fe, NM, USA, S. 5–9. Association for Computational Linguistics. <https://www.aclweb.org/anthology/C18-2.pdf> (2018). Zugegriffen: 15. Juni 2021
- Koltcov, S., Koltsova, O., Nikolenko, S.: Latent dirichlet allocation: stability and applications to studies of user-generated content. In: Proceedings of the 2014 ACM Conference on Web Science, Bloomington, IN, USA, S. 161–165. Association for Computing Machinery, New York, NY, USA (2014). <https://doi.org/10.1145/2615569.2615680>
- Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Brodley, C.E., Pohoreckyi Danyluk, A. (Hrsg.) Proceedings of the 18th International Conference on Machine Learning (ICML'01), Williamstown, MA, USA, S. 282–289. Morgan Kaufmann, San Francisco, CA, USA (2001)
- Lancichinetti, A., Sicer, M.I., Wang, J.X., Acuna, D., Körding, K., Amaral, L.A.N.: High-reproducibility and high-accuracy method for automated topic classification. Phys. Rev. X **5**, 11007 (2015). <https://doi.org/10.1103/PhysRevX.5.011007>
- Lee, K., He, L., Lewis, M., Zettlemoyer, L.: End-to-end neural coreference resolution. In: Palmer, M., Hwa, R., Riedel, S. (Hrsg.) Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP), Copenhagen, Denmark, S. 188–197. Association for Computational Linguistics (2017). <https://doi.org/10.18653/v1/D17-1018>
- Lin, C.-Y.: ROUGE: A package for automatic evaluation of summaries. In: Proceedings of the ACL-04 Workshop on Text Summarization Branches Out, Barcelona, Spain, S. 74–81. Association for Computational Linguistics, Stroudsburg, PA, USA, <https://www.aclweb.org/anthology/W04-1013.pdf> (2004). Zugegriffen: 21. Juni 2021
- Liu, P., Wang, X., Xiang, C., Meng, W.: A survey of text data augmentation. In: 2020 International Conference on Computer Communication and Network Security (CCNS), Xi'an, China, S. 191–195. IEEE Computer Society, Los Alamitos, CA, USA (2020). <https://doi.org/10.1109/CCNS50731.2020.00049>
- MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Cam, L., Neyman, J. (Hrsg.) Proceedings of the 5th Berkeley Symposium on mathematical

- statistics and probability, Bd. 1, S. 281–297. University of California Press, Berkeley, Los Angeles, CA, USA (1967)
- Maier, D., Waldherr, A., Miltner, P., Wiedemann, G., Niekler, A., Heyer, G., Keinert, A., Pfetsch, B., Häussler, T., Reber, U., Schmid-Petri, H., Adam, S.: Applying LDA topic modeling in communication research: towards a valid and reliable methodology. Paper presented at ICA's 67th Annual conference, San Diego, CA, USA (2017)
- Mann, W.C., Thompson, S.A.: Rhetorical structure theory: A theory of text organization. In: Technical Reports, ISI/RS-87-190. Information Sciences Institute, Marina del Rey, CA, USA (1987)
- Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information retrieval. Cambridge University Press, Cambridge. <https://nlp.stanford.edu/IR-book/information-retrieval-book.html> (2008). Zugegriffen: 21. Juni 2021
- McCallum, A., Freitag, D., Pereira, F.: Maximum entropy markov models for information extraction and segmentation. In: Proceedings of the 17th Annual International Conference on Machine Learning (ICML '00), Stanford, CA, USA, S. 591–598. <http://www.ai.mit.edu/courses/6.891-nlp/READINGS/maxent.pdf> (2000). Zugegriffen: 21. Juni 2021
- McDonald, R.T., Nivre, J., Quirkbach-Brundage, Y., Goldberg, Y., Das, D., Ganchev, K., Hall, K.B., Petrov, S., Zhang, H., Täckström, O., Bedini, C., Castelló, N.B., Lee, J.: Universal dependency annotation for multilingual parsing. Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Bd. 2: Short Papers), Sofia, Bulgaria, S. 92–97. Association for Computational Linguistics, Stroudsburg, PA, USA. <https://www.aclweb.org/anthology/P13-2017> (2013). Zugegriffen: 11. Jan. 2021
- Mimno, D., Wallach, H.M., Naradowsky, J., Smith, D.A., McCallum, A.: Polylingual topic models. In: Koehn, P., Mihalcea, R. (Hrsg.) Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, Singapore, Bd. 2, S. 880–889. Association for Computational Linguistics, Stroudsburg, PA, USA. <https://www.aclweb.org/anthology/D09-1092.pdf> (2009). Zugegriffen: 17. Mai 2021
- Nakayama, H., Kubo, T., Kamura, J., Taniguchi, Y., Liang, X.: doccano: Text annotation tool for human. <https://github.com/doccano/doccano> (2018). Zugegriffen: 17. Juni 2021
- National Institute of Standards and Technology. TREC-2004 common evaluation measures. <http://trec.nist.gov/pubs/trec14/appendices/CE.MEASURES05.pdf> (2004). Zugegriffen: 21. Juni 2021
- Neves, M., Ševa, J.: An extensive review of tools for manual annotation of documents. Brief. Bioinform. **22**(1), 146–163 (2021). <https://doi.org/10.1093/bib/bbz130>
- Newman, D., Lau, J.H., Grieser, K., Baldwin, T.: Automatic evaluation of topic coherence. In: Human language technologies: The 2010 annual conference of the North American chapter of the association for computational linguistics, S. 100–108. Los Angeles, CA, USA. Association for Computational Linguistics, Stroudsburg, PA, USA (2010). <https://aclanthology.org/N10-1012/>. Zugegriffen: 29. Sept. 2021
- Papineni, K., Roukos, S., Ward, T., Zhu, W.-J.: BLEU: A method for automatic evaluation of machine translation. In: Isabelle, P., Charniak, E., Lin, D. (Hrsg.) Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics. Philadelphia, PA, USA, S. 311–318. Association for Computational Linguistics, Morristown, NJ, USA (2002). <https://doi.org/10.3115/1073083.1073135>
- Parisi, G.I., Kemker, R., Part, J.L., Kanan, C., Wermter, S.: Continual lifelong learning with neural networks: A review. Neural Netw. **113**, 54–71 (2019). <https://doi.org/10.1016/j.neunet.2019.01.012>
- Perozzi, B., Al-Rfou, R., Skiena, S.: DeepWalk: online learning of social representations. In: Macskassy, S. (Hrsg.) KDD '14: Proceedings of the 20th ACM SIGKDD international

- Conference on Knowledge Discovery and Data Mining, S. 701–710. ACM, New York, NY, USA (2014). <https://doi.org/10.1145/2623330.2623732>
- Phan, X.-H., Nguyen, L.-M., Horiguchi, S.: Learning to Classify Short and Sparse Text & Web with Hidden Topics from Large-scale Data Collections. In: Huai, J. (Hrsg.) Proc. of the 17th International World Wide Web Conference (WWW 2008), Beijing, China, S. 91–100. ACM, New York, NY, USA (2008). <https://doi.org/10.1145/1367497.1367510>
- Pradhan, S., Moschitti, A., Xue, N., Uryupina, O., Zhang, Y.: CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in Ontonotes. In: Joint Conference on EMNLP and CoNLL-Shared Task, Jeju Island, Korea, S. 1–40. Association for Computational Linguistics, Stroudsburg, PA, USA. <https://www.aclweb.org/anthology/W12-4501.pdf> (2012). Zugegriffen: 18. Mai 2021
- Prechelt, L.: Early Stopping — But When? In: Montavon, G., Orr, G.B., Müller, K.-R. (Hrsg.) Neural networks: Tricks of the trade. Lecture Notes in Computer Science, Bd. 7700. 2. Aufl., S. 53–67. Springer, Heidelberg (2012). [https://doi.org/10.1007/978-3-642-35289-8\\_5](https://doi.org/10.1007/978-3-642-35289-8_5)
- Rabiner, L.R.: A tutorial on hidden markov models and selected applications in speech recognition. Proc. IEEE 77(2), 257–286 (1989). <https://doi.org/10.1109/5.18626>
- Raghunathan, K., Lee, H., Rangarajan, S., Chambers, N., Surdeanu, M., Jurafsky, D., Manning, C.: A multi-pass sieve for coreference resolution. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP), Cambridge, MA, USA, S. 492–501. Association for Computational Linguistics, Stroudsburg, PA, USA. <https://www.aclweb.org/anthology/D10-1048> (2011). Zugegriffen: 29. Apr. 2021
- Ramshaw, L., Marcus, M.: Text chunking using transformation-based learning. In: Third workshop on very large corpora, Cambridge, MA, USA, S. 82–94. <https://www.aclweb.org/anthology/W95-0107> (1995). Zugegriffen: 17. Juni 2021
- Remus, S., Biemann, C.: Three knowledge-free methods for automatic lexical chain extraction. In: Vanderwende, L., Daumé III, H., Kirchhoff, K. (Hrsg.) Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Atlanta, GA, USA, S. 989–999. Association for Computational Linguistics, Stroudsburg, PA, USA. <https://www.aclweb.org/anthology/N13-1119.pdf> (2013). Zugegriffen: 18. Mai 2021
- Remus, S., Aly, R., Biemann, C.: GermEval 2019 Task 1: Hierarchical classification of blurbs. In: Proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019), S. 280–292, Erlangen, Germany (2019)
- Riedl, M., Biemann, C.: Text segmentation with topic models. J. Language Technol. Comput. Linguist. (JLCL) 27(1), 47–70 (2012)
- Rosenberg, A., Hirschberg, J.: V-measure: A conditional entropy-based external cluster evaluation measure. In: Eisner, J. (Hrsg.) Proceedings of the 2007 joint conference on empirical methods in Natural Language Processing and computational natural language learning (EMNLP-CoNLL), Prague, Czech Republic, S. 410–420. Association for Computational Linguistics, Stroudsburg, PA, USA. <https://www.aclweb.org/anthology/D07-1043.pdf> (2007). Zugegriffen: 18. Mai 2021
- Rosen-Zvi, M., Griffiths, T., Steyvers, M., Smyth, P.: The author-topic model for authors and documents. In: Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence, S. 487–494. AUAI Press, Arlington, VA, USA, <https://dl.acm.org/doi/pdf/https://doi.org/10.5555/1036843.1036902> (2004). Zugegriffen: 17. Mai 2021
- Ruder, S.: Neural Transfer learning for natural language processing. PhD thesis, National University of Ireland, Galway (2019)
- Ruder, S., Peters, M.E., Swayamdipta, S., Wolf, T.: Transfer learning in natural language processing. In: Sarkar, A., Strube, M. (Hrsg.) Proceedings of the 2019 Conference of the North

- American Chapter of the Association for Computational Linguistics: Tutorials. Minneapolis, MI, USA, S. 15–18. Association for Computational Linguistics, Stroudsburg, PA, USA (2019). <https://doi.org/10.18653/v1/N19-5004>, <https://vimeo.com/359399507>
- Schröder, F., Biemann, C.: Estimating the influence of auxiliary tasks for multi-task learning of sequence tagging tasks. In: Jurafsky, D., Chai, J., Schluter, N., Tetreault, J. (Hrsg.) Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020), online, S. 2971–2985. Association for Computational Linguistics, Stroudsburg, PA, USA (2020). <https://doi.org/10.18653/v1/2020.acl-main.268>
- Schuster, I.: Gradient importance sampling. Cornell University Library. <https://arxiv.org/pdf/1507.05781.pdf> (2015). Zugegriffen: 17. Mai 2021
- Shalev-Shwartz, S., Ben-David, S.: Understanding machine learning: From theory to algorithms. Cambridge University Press, New York, NY, USA (2014)
- Shindo, H., Munehisa, Y., Matsumoto, Y.: PDFAnno: A web-based linguistic annotation tool for pdf documents. In: Calzolari, N., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Hasida, K., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S., Tokunaga, T. (Hrsg.) Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), Miyazaki, Japan. European Language Resources Association (ELRA). <http://www.lrec-conf.org/proceedings/lrec2018/pdf/680.pdf> (2018). Zugegriffen: 17. Juni 2021
- Souza, F., Nogueira, R., Lotufo, R.: Portuguese Named Entity Recognition using BERT-CRF. <https://arxiv.org/pdf/1909.10649.pdf> (2020). Zugegriffen: 17. Juni 2021
- Strehl, A.: Relationship-based clustering and cluster ensembles for high-dimensional data mining. PhD dissertation, University of Texas (2002)
- Teichmann, C.: Markov chain monte carlo sampling for dependency trees. Dissertation, Fakultät für Mathematik und Informatik, Universität Leipzig (2016)
- Ustalov, D., Panchenko, A., Biemann, C., Ponzetto, S.P.: Watset: Local-global graph clustering with applications in sense and frame induction. Comput. Linguist. **45**(3), 423–479 (2019). [https://doi.org/10.1162/coli\\_a\\_00354](https://doi.org/10.1162/coli_a_00354)
- Van Rijsbergen, C.J.: Information retrieval. Butterworths, London (1979)
- Vaswani, A., Shazeer N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in neural information processing systems 30 (NIPS'17), Long Beach, CA, USA, S. 6000–6010 (2017)
- Viterbi, A.: Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IEEE Trans. Inf. Theory **13**(2), 260–269 (1967). <https://doi.org/10.1109/TIT.1967.1054010>
- Widdows, D., Dorow, B.: A graph model for unsupervised lexical acquisition. In: Proceedings of the 19th international conference on Computational Linguistics (COLING-02), Taipei, Taiwan, S. 1–7. Association for Computational Linguistics, Morristown, NJ, USA (2002). <https://doi.org/10.3115/1072228.1072342>
- Wiechmann, M., Yimam S.M., Biemann, C.: ActiveAnno: General-purpose document-level annotation tool with active learning integration. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies – System Demonstrations, Mexico City, Mexico (online), S. 99–105. Association for Computational Linguistics, Stroudsburg, PA, USA (2021). <https://doi.org/10.18653/v1/2021.nacl-demos.12>
- Wikimedia Foundation, Inc.: Wikinews. <https://de.wikinews.org> (2020). Zugegriffen: 12. Mai 2021
- Witten, I.H., Frank, E., Hall, M., Pal, C.: Data mining. Practical machine learning tools and techniques. 4. Aufl., Elsevier/Morgan Kaufmann, Amsterdam (2017), <https://www.cs.waikato.ac.nz/ml/weka>

- Wittgenstein, L.: Philosophische Untersuchungen Kritisch-genetische Edition. Wissenschaftliche Buchgesellschaft, Frankfurt a. M. (2001)
- Wu, Y.-C., Yang, J.-C., Lee, Y.-S., Yen, S.-J.: Efficient and Robust Phrase Chunking Using Support Vector Machines. In: Ng, H.T. (Hrsg.) Information retrieval technology. Third Asia Information Retrieval Symposium, AIRS 2006, Singapore. Lecture Notes in Computer Science, Bd. 4182, S. 350–361. Springer, Berlin (2006). [https://doi.org/10.1007/11880592\\_27](https://doi.org/10.1007/11880592_27)
- Xu, R., Wunsch, D.: Survey of clustering algorithms. IEEE Trans. Neural Networks **16**(3), 645–678 (2005)
- Yimam, S.M., Gurevych, I., Eckart de Castilho, R., Biemann, C.: WebAnno: A flexible, web-based and visually supported system for distributed annotations. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Sofia, Bulgaria, S. 1–6. Association for Computational Linguistics, Stroudsburg, PA, USA. <https://www.aclweb.org/anthology/P13-4001> (2013). Zugegriffen: 11. Jan. 2021
- Yimam S.M., Alemayehu H.M., Ayele A.A., Biemann C.: Exploring amharic sentiment analysis from social media texts: building annotation tools and classification models. In: Scott, D., Bel, N., Zong, C. (Hrsg.) Proceedings of the 28th International Conference on Computational Linguistics (COLING 2020), Barcelona, Spain (online), S. 1048–1060. International Committee on Computational Linguistics (2020). <https://doi.org/10.18653/v1/2020.coling-main.91>
- Zhang, Y., Teng, Z.: Natural language processing. A machine learning perspective. Cambridge University Press, Cambridge (2021)
- Zhang, T., Kishore, V., Wu, F., Weinberger, K.Q., Artzi, Y.: BERTScore: Evaluating text generation with BERT. ICLR 2020 Conference, <https://openreview.net/pdf?id=SkeHuCVFDr> (2020). Zugegriffen: 21. Juni 2021



# Beispielanwendungen

7

## Zusammenfassung

Von der Vielzahl an Verwendungsmöglichkeiten des Text Mining werden in diesem Kapitel sechs Anwendungen vorgestellt, welche den Einsatz der in den vorangegangenen Kapiteln besprochenen Konzepte und Verfahren beispielhaft verdeutlichen. Terminologieextraktion dient zur Erstellung eines Fachvokabulars auf Basis von Fachtexten. Die facettierte Suche mit Eigennamen ermöglicht einen alternativen Zugang zu in Korpora enthaltenen Informationen durch die Verdeutlichung der Interaktion zwischen Akteuren. Eine in der industriellen Anwendung beliebte Anwendung ist die Sentimentanalyse, welche in vielen Varianten ihrer Realisierung besprochen wird. Die zeitliche Dimension in Korpora ermöglicht eine ganze Reihe weiterer Anwendungen: Bei der Trendanalyse werden auf Nachrichtentexten wichtige Themen über die Zeit dargestellt. Als Vorschläge zur Wörterbucherweiterung können Neologismen durch geeignete Analysen über die Zeit gefunden werden; zum Abschluss dieses Kapitels definieren wir noch das Konzept und die Operationalisierung der Kontextvolatilität als Hinweis auf Bedeutungsverschiebungen von Wörtern. Wie für Text-Mining-Anwendungen typisch finden die hier vorgestellten Anwendungen Informationen in Texten, welche durch Anwender und Anwenderinnen interpretiert und gefiltert werden müssen.

---

**Ergänzende Information** Die elektronische Version dieses Kapitels enthält Zusatzmaterial, auf das über folgenden Link zugegriffen werden kann [https://doi.org/10.1007/978-3-658-35969-0\\_7](https://doi.org/10.1007/978-3-658-35969-0_7).

## 7.1 Terminologieextraktion

Bezogen auf eine vorgegebene Menge von Fachtexten kann durch Text-Mining-Verfahren die Extraktion der verwendeten relevanten Fachterminologie unterstützt werden. Vor der computergestützten Extraktion ist eine textuelle und linguistische Vorverarbeitung durchzuführen. Für die Extraktion werden typischerweise statistische Verfahren, die morphembasierte Mustersuche und Bootstrapping angewendet. Die Extraktion liefert eine Termkandidatenliste, die überprüft und ggf. manuell oder maschinell weiterbearbeitet werden muss.

### 7.1.1 Arbeitsablauf der Terminologieextraktion

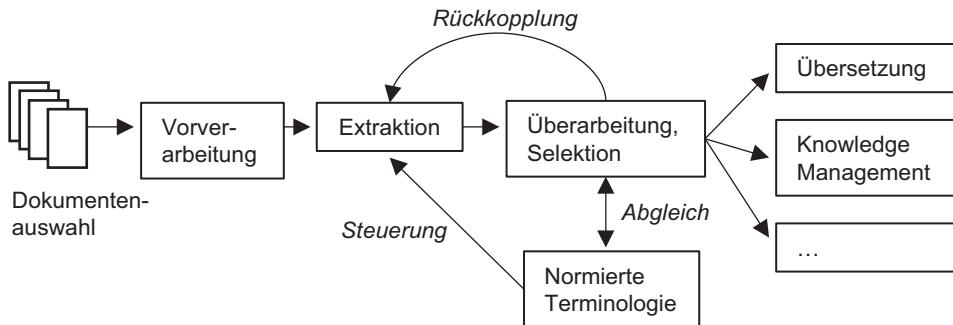
Die Kenntnis der Fachterminologie eines Faches oder einer Anwendung ist oft der Schlüssel für ein Verständnis des Faches oder der Anwendung selber. Die computergestützte Erkennung der Fachausdrücke eines Faches ist daher eine wichtige Anwendung des Text Mining. Eine ausführliche Einführung in die Grundlagen der automatischen Terminologieextraktion findet sich in Witschel (2004).

Am Anfang einer Terminologieextraktion steht die Auswahl der Analysetexte, aus denen Terminologie extrahiert werden soll. Voraussetzung für die Extraktion relevanter Begriffe aus Texten ist das Vorliegen geeigneter Textdokumente, z. B. umfangreiche Produktdokumentationen, Softwarespezifikationen, Verträge, Ausführungsvorschriften, Korrespondenz usw. (vgl. Abschn. 1.3).

Bei statistischen Verfahren, die textvergleichende Methoden einsetzen, wie z. B. der Differenzanalyse (vgl. Abschn. 5.9), ist zusätzlich ein geeignetes Referenz- oder Vergleichskorpus bereitzustellen, von dem sich die auszuwählende Fachterminologie abheben soll (Extraktion durch Vergleich relativer Häufigkeiten).

Vor dem Einsatz der eigentlichen Extraktionsverfahren ist eine **Vorverarbeitung** im Sinne der linguistischen Verarbeitungspipeline (vgl. Abschn. 3.2) erforderlich, um Dokumente in ein geeignetes Format zu konvertieren, den Text zu vereinheitlichen (z. B. durch Auflösung von Abkürzungen) oder linguistisch zu annotieren (z. B. durch POS-Tagging). Die Extraktion liefert dann eine (ggf. gewichtete) Termkandidatenliste sowie Zusatzinformation zu den Kandidaten. Diese ist von einem Terminologen bzw. einer Terminologin durchzusehen und – je nach Anwendungsgebiet – in weiterverarbeitende Systeme zu integrieren.

In der Abb. 7.1 ist der Ablauf einer automatischen Terminologieextraktion schematisch dargestellt.



**Abb. 7.1** Schematischer Ablauf einer automatischen Terminologieextraktion

### 7.1.2 Verfahren der Terminologieextraktion

Die Fachausdrücke eines Faches oder einer Anwendung weisen eine Reihe charakteristischer Merkmale auf, die bei der computergestützten Terminologieextraktion genutzt werden können, wie beispielsweise statistische Auffälligkeiten, syntaktische oder morphemische Muster (vgl. Abschn. 2.5). Da sich die Extraktion von Fachterminologie nicht auf die Anwendung nur eines Verfahrens beschränken muss, sondern Verfahren kombiniert und iterativ angewendet werden können, lassen sich die folgenden Verfahren unterscheiden:

- statistische Verfahren,
- musterbasierte Verfahren und
- bootstrapping bzw. kombinierte Verfahren.

**Statistische Verfahren** der Terminologieextraktion nutzen die Tatsache aus, dass Fachbegriffe in den Fachtexten eines Faches im Vergleich zu den Fachtexten eines anderen Faches bzw. fachneutralen Texten häufiger vorkommen. Darüber hinaus finden sich für die Fachausdrücke in den Texten eines Faches meist fachspezifische Verwendungskontexte. Grundlage der statistischen Verfahren ist die Differenzanalyse, wie im Abschn. 5.9 beschrieben. Häufig verwendete Maße sind der Häufigkeitsquotient,  $tf \cdot idf$  und die **Log-Likelihood-Ratio**. Evaluationen zeigen, dass Maße wie der Häufigkeitsquotient, die auf eine statistische Modellierung verzichten, oder wie die Binomialverteilung, die auf der Normalverteilung aufbauen, zu einer Überbewertung seltener Terme neigen. Je nachdem, welche Ziele bei der Terminologieextraktion im Vordergrund stehen, lassen sich diese Eigenschaften der statistischen Maße nutzen: Sollen für einen Anwendungsbereich die allgemein verwendeten Fachterme extrahiert werden, sollte die Log-Likelihood-Ratio gewählt werden, weil sie hochfrequente Terme bevorzugt. Für die Extraktion von Termen, welche mehr fachspezifisches Hintergrundwissen beschreiben, sollten hingegen Maße wie der Häufigkeitsquotient oder  $tf \cdot idf$  verwendet werden, weil diese seltene Terme überbewerten.

### Wirkung statistischer Termextraktionsmaße (Referenzkorpus Wortschatz Deutsch, ASV-Toolbox)

Als Beispieltext, aus dem einschlägige Fachbegriffe extrahiert werden sollen, wählen wir aus den bereitgestellten Referenzkorpora die Verordnung des Sächsischen Staatsministeriums für Soziales und Gesellschaftlichen Zusammenhalt zum Schutz vor dem Coronavirus SARS-CoV-2 und COVID-19 ((Sächsische Corona-Schutz-Verordnung – SächsCoronaSchVO) vom 10. Juni 2021, Paragraph §2 (Grundsätze) und §3, Satz (1) und (2) (Sieben-Tage-Inzidenz und Bettenkapazität) (Sächsische Staatsregierung 2021)).

Für die Termextraktion verwenden wir die ASV-Toolbox mit dem Log-Likelihood-Maß und der Einschränkung, dass nur einzelne Nomina extrahiert werden sollen (vgl. Anhang A5, Biemann et al. 2008). Als Ergebnis erhalten wir u. a. *Inzidenz*, *Schwellenwert*, *Coronavirus* und *Ansteckungsvermeidung*, aber auch *Tage* und *Sieben*. Das liegt zum einen an der Häufigkeit dieser Wörter, zum anderen daran, dass bei dieser Einzelwort-Extraktion komplexere Terme wie Sieben-Tage-Inzidenz nicht erkannt, sondern in Einzelworte zerlegt werden. ◀

Morphologische Muster kommen im Text Mining immer dann zur Anwendung, wenn die Analyse einzelner Wörter für die Terminologieextraktion erforderlich bzw. nützlich ist. Im Folgenden beschränken wir uns auf die Nutzung von Suffixen und **Komposita**, die für ein Fachgebiet typisch sind (vgl. Ananiadou 1994).

In Naturwissenschaften wie Chemie, Biologie oder Medizin lässt sich ein gehäuftes Auftreten domänenpezifischer lateinischer oder griechischer Suffixe beobachten wie z. B. „-itis“ in der Medizin oder „-ase“ in der Chemie. Diese treten in der Gemeinsprache selten oder so gut wie nicht auf und können daher über einen Frequenzvergleich mit einem gemeinsprachlichen Referenzkorpus leicht herausgefiltert werden (vgl. Abschn. 5.9). Wenn man nun in einem medizinischen Text alle Wörter heraussucht, die auf „-itis“ enden (it is = Entzündung), so werden diese mit sehr hoher Wahrscheinlichkeit Fachausdrücke sein.

---

### Beispiel: Medizinisch-spezifische Suffixe

In der Tab. 7.1 findet sich eine Liste von Suffixen, deren Frequenz in einem englischen Text über Onkologie wesentlich höher ist, als in einem allgemeinsprachlichen Korpus (Q sagt aus, wievielmal häufiger das Suffix im Fachtext ist, F wie viele Wörter mit diesem Suffix im Text vorkommen). ◀

Bei den Fachausdrücken in einem Text kann es sich um zusammengesetzte Substantive, sog. Komposita, handeln (z. B. *Schwefelfarbstoff*: *Schwefel* + *Farbstoff*). Terminologische Komposita können durch Kompositazerlegung in ihre Bestandteile zerlegt werden. Falls das elektronische Wörterbuch keine Informationen über das zusammengesetzte Wort enthält, kann man so nach Informationen über die Teile suchen (vgl. Abschn. 2.2).

**Tab. 7.1** Beispiel medizinisch-spezifische Suffixe

Suffix	Q	F	Beispiele
cyte	48,8	4	t-lymphocyte, antithymocyte, lymphocyte
athy	40,7	5	pneumopathy, retinopathy, endocrinopathy
itis	35,9	13	myocarditis, meningitis, nephritis
roid	29,6	3	corticosteroid, steroid, thyroid
dose	27,9	3	high-dose, ablative-dose, low-dose
rapy	26,1	3	chemotherapy, radiotherapy, therapy
mia	23,5	5	hyperbilirubinemia, leukemia, anemia
osis	20,3	8	diagnosis, cirrhosis, kyphosis

Bei Komposita gibt es stets einen sog. *Kopf*, der die Wortart des Ganzen bestimmt und dessen Bedeutung (fast immer) durch die Bedeutung der anderen Teile – den sog. *Modifikatoren* – eingeschränkt wird, sofern das Kompositum nicht metaphorisch verwendet wird. Auf diese Weise entstehen **Taxonomien**, d. h. terminologische Begriffshierarchien, da der Kopf eines Kompositums meist ein **Oberbegriff** (Hyperonym) des Ganzen ist (vgl. Abschn. 2.4).

#### Beispiel für Oberbegriffe

Ein *Schweröl* ist ein Öl, eine *Ansauglufttemperatur* ist eine Temperatur usw. Ausnahmen bilden z. B. die aus der Typographie bekannten metaphorisch verwendeten Begriffe *Schusterjunge* und *Hurensohn*. ◀

Mithilfe einer Zerlegung von Komposita können terminologische Begriffshierarchien aufgebaut oder zumindest deren Aufbau unterstützt werden. In den meisten Fällen ist der Kopf eines Kompositums ein Oberbegriff des Ganzen. So können verschiedene Komposita unter einem gemeinsamen Oberbegriff – ihrem Kopf – zusammengefasst werden.

#### Beispiel für Kohyponyme

Die folgenden Komposita sind Bezeichnungen für *Minister*: Außenminister, Verkehrsminister, Verteidigungsminister, Umweltminister etc. Sie sind allesamt Ausprägungen (Hyponyme bzw. Kohyponyme) desselben Oberbegriffs – eben Minister. ◀

Viele terminologische Begriffsnetze bauen auf Hierarchien auf (im Sinne von Oberbegriffen, die immer weiter in ihre Unterbegriffe untergliedert werden). Die Zerlegung von Komposita ist ein erster und sehr einfacher Schritt in diese Richtung.

Neben morphologischen Mustern lassen sich für die Extraktion von Fachausdrücken auch syntaktische Muster für die Extraktion von Mehrworteinheiten nutzen, da Fachausdrücke meist Nominalphrasen sind (d. h. sie bestehen entweder aus einem einzelnen

Substantiv oder einem Substantiv mit Modifikatoren). Um solche Strukturen zu finden, kann man eine syntaktische Analyse des betreffenden Fachtextes vornehmen und dann bestimmte Nominalphrasen isolieren. Einfacher noch ist die Suche nach sogenannten **POS-Mustern** (d. h. nach Abfolgen von Wörtern einer bestimmten Wortart), wie z. B. *N*, *A N* oder *N N* (mit *N* = Nomen, *A* = Adjektiv).

Dies ist zwar ein (theoretisch) zu einfacher Ansatz, da z. B. zwei benachbarte Nomina keine Nominalphrase bilden müssen (z. B. „Er unterrichtet in Leipzig Informatik“), liefert aber in der Praxis sehr gute Ergebnisse und lässt sich unter Berücksichtigung der Kookkurrenzbeziehung zwischen zwei Wörtern (linker bzw. rechter Nachbar, siehe Abschn. 5.3) effizient implementieren.

Der Begriff **Bootstrapping** bezeichnet ein maschinelles Lernverfahren, das unter Zuhilfenahme von einer kleinen „Startmenge“ und Regeln diese Menge iterativ erweitert und gelernte Informationen aus früheren Schritten dazu benutzt, um weitere Informationen zu finden (vgl. Heid 1998).

Dies können Kookkurrenzen oder Kohyponyme von Termen sein. Eine Regel zur Identifikation neuer Fachausdrücke könnte lauten: Wähle solche Nominalphrasen oder Komposita als Kandidaten für neue Fachausdrücke, die mindestens einen Term der Startmenge enthalten. Die so generierten Fachausdrücke können unter bestimmten Bedingungen der Startmenge hinzugefügt und das Verfahren zur Identifikation neuer Fachausdrücke iteriert werden.

Darüber hinaus lassen sich (z. B. mittels **Differenzanalyse**) bereits gefundene Fachausdrücke als Startmenge verwenden, die sodann um solche Terme erweitert wird, die bestimmte syntaktische Muster erfüllen.

---

#### Extraktion von Mehrworteinheiten mit der ASV Toolbox

Als Beispiel wählen wir wieder die sächsische Corona-Schutz-Verordnung (Sächsische Staatsregierung 2021) und definieren, dass die Termextraktion (auf Basis des Log-Likelihood-Maßes) auch Terme finden soll, die den Mustern *N N N* oder *A A N* entsprechen.

Als Ergebnis erhalten wir nun auch die Mehrworteinheiten *Sieben-Tage-Inzidenz*, *Robert Koch-Institut*, *Coronavirus SARS-CoV2* und *physisch-sozialen Kontakte*. ◀

Allgemein ist es nützlich, dem Anwender bzw. der Anwenderin einer Extraktionssoftware Gelegenheit zu einem Feedback zu geben, bevor man beispielsweise das Ergebnis einer Differenzanalyse als Startmenge verwendet. Nur was von den Benutzenden als Fachterminus anerkannt wird, sollte Eingang in die Startmenge finden. Diese Überprüfung sollte nach jedem Iterationsschritt wiederholt werden.

Die Praxis zeigt, dass nur die Verzahnung verschiedener Verfahren (u. a. linguistische Vorverarbeitung, Häufigkeitsvergleiche, Heuristiken) bei der automatischen Erkennung von Fachausdrücken ein hochqualitatives Ergebnis liefert. Auch führen nicht alle Verfahren bei jeder Sorte von Text zu gleich guten Ergebnissen. Angesichts der vielfältigen

aufgaben-, domänen- und organisationsspezifischen Parameter sowie der sprachlichen Probleme bei der Terminologieextraktion findet sich kein System, das eine „perfekte“ Terminologieliste extrahiert. Bestenfalls kann mit der automatischen Terminologieextraktion ein qualitativ hochwertiges terminologisches Ausgangsmaterial erzeugt werden, das einer weiteren Nachbearbeitung bedarf.

---

## 7.2 Facettierte Suche mit Eigennamen

In diesem Abschnitt wird das Konzept der Facettensuche mit Metainformationen und Entitätennetzwerken anhand von drei Beispiel-Tools verdeutlicht.

**Facettensuche** oder **Facettierte Suche** beschreibt die Möglichkeit, Suchergebnisse durch das Definieren von Filtern einzuschränken. Bei der Produktsuche in Online-Katalogen könnte dies z. B. die Größe und Farbe von Kleidung sein, oder die Längen-, Breiten- und Höhenmaße von Möbelstücken. Um Eigenschaften der Produkte oder hier der Texte für die Facettensuche zu nutzen, müssen diese als **Metadaten** vorliegen: Bei Texten sind das solche Eigenschaften wie Dateityp, Erstellungsdatum, Autorinformation und ggf. strukturierte Felder wie Absender und Empfänger bei E-Mails. Beim Indexieren eines Korpus (Abschn. 4.4) können solche Metadaten in entsprechende Felder abgelegt werden, z. B. in NoSQL-Datenbanken wie Elasticsearch (Gormley und Tong 2015). Ebenso wie bei der Produktsuche helfen bei der Facettensuche Metadaten bei der schnellen Eingrenzung der Suche, hier nach Texten, welche relevante Informationen enthalten.

Text Mining kann in diesem Kontext dazu verwendet werden, zusätzliche Metadaten automatisch aus Texten zu extrahieren, welche dann zum Filtern in Facettensuchen angewendet werden können. Hierfür sind vor allem Eigennamen und Zeitausdrücke interessant: Hieraus kann – unabhängig von den ohnehin vorliegenden Metadaten – geschlossen werden, wer mit wem wann interagiert hat, was z. B. bei forensischen Analysen oder bei investigativem Journalismus hochrelevant ist. Eigennamen fungieren zudem als Anker, welche verschiedene Aspekte und Narrative innerhalb Dokumentsammlungen verknüpfen: Dem bzw. der informierten Betrachtenden erschließt sich beim Betrachten von Namenslisten oder Namensnetzwerken sehr schnell der Inhalt von darunterliegenden Dokumenten. Hierbei ist es wegen der normalerweise vorliegenden Themenbezogenheit solcher Dokumentsammlungen meist nicht notwendig, die Namen durch Entity Linking zu Konzepten in einer Wissensbasis aufzulösen, es reicht oft, sie lediglich als Namen zu erkennen (Abschn. 3.2.5).

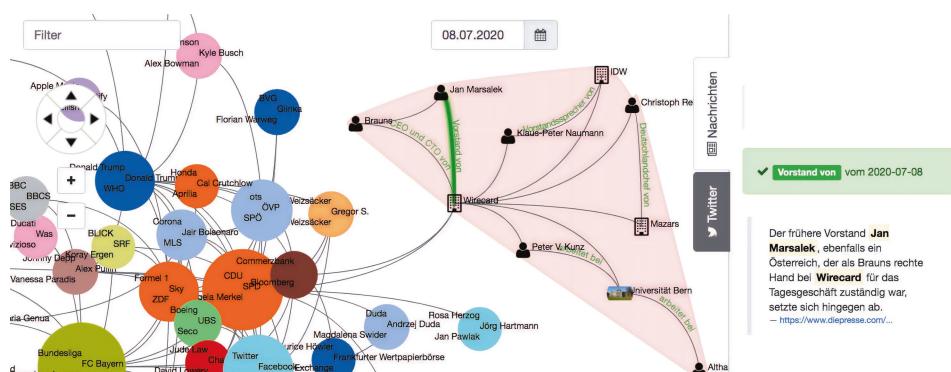
In Folgenden stellen wir kurz drei Tools vor, welche Eigennamen als Anker für die Informationsdarstellung und -suche nutzen, dies jedoch sowohl technisch verschieden umsetzen als auch andere Anwendungsszenarien abdecken: Die visuelle Nachrichtenzusammenfassungswebsite *tagesnetzwerk.de*, das Browser-Plugin *Storyfinder* und das Investigativtool *New/s/leak*.

## 7.2.1 Tagesnetzwerk: Visuelle Nachrichtenzusammenfassung

Tagesnetzwerk (Benikova et al. 2014) ([www.tagesnetzwerk.de/](http://www.tagesnetzwerk.de/)) bietet eine interaktive Visualisierung des Entitätennetzwerkes der frei verfügbaren täglichen deutschen Online-Nachrichten seit Mitte 2014. Hierzu werden genauso wie in Quasthoff et al. (2002) beschrieben mithilfe von RSS-Feeds die Inhalte von vielen Nachrichtenportalen verarbeitet. Hierbei werden in einer linguistischen Pipeline (Abschn. 3.2) Texte in Sätze segmentiert und Personen- und Organisationsnamen erkannt, welche die Knoten des Netzwerkes bilden. Entitäten werden mit einer Kante im Netzwerk verbunden, falls sie zusammen in einem Satz auftreten; aus urheberrechtlichen Gründen (Abschn. 1.3.5) werden nur Sätze mit mindestens zwei Entitäten gespeichert.

Für die Visualisierung (siehe Abb. 7.2) wird das aus mehreren hundert Entitäten bestehende Netzwerk mit Graphclustering (Abschn. 6.2) in Gruppen von Entitäten zusammengefasst, welche als Kreise dargestellt und mit den häufigsten darin enthaltenen Entitäten beschriftet werden. Diese können durch Klicken aufgeklappt werden, wie in Abb. 7.2 für das Cluster mit *Wirecard* dargestellt. Ein Klick auf die Kante zeigt die entsprechende Textstelle als Zitat mit URL-Quellenangabe; Kanten können zudem noch manuell beschriftet werden. Das Vorhalten der Textstellen, auf welchen das visualisierte Datum basiert, ist ein wichtiges Element moderner Text-Mining-Tools und schafft insbesondere bei Hintergrundrecherchen Transparenz über die Quelle der angezeigten Information.

Der Wirecard-Skandal bezeichnet die größte Bilanzfälschung in der deutschen Wirtschaftsgeschichte der Nachkriegszeit. Das in Abb. 7.2 dargestellte Netzwerk auf Nachrichten ca. eine Woche nach der Verhaftung des CEOs zeigt zwei wichtige Akteure der Firma Wirecard und drei Experten von verschiedenen Institutionen, welche den Vorfall öffentlich kommentieren. Ferner sind als Cluster übliche Themen wie Innenpolitik, Politik in verschiedenen anderen Ländern, Fußball, Börsennachrichten u.v.m. sichtbar.



**Abb. 7.2** Screenshot der Webseite [tagesnetzwerk.de](http://tagesnetzwerk.de) zum Wirecard-Skandal auf Basis der Online-Nachrichten vom 08.07.2020

## 7.2.2 Storyfinder: Eigene Lesehistorie

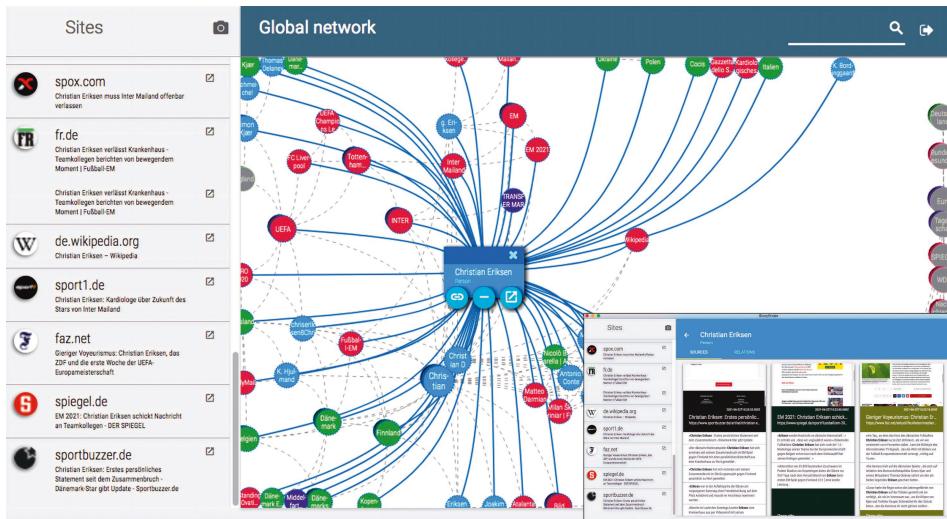
Während das Tagesnetzwerk den Anspruch hat, möglichst viele Nachrichten tagesgenau zu erfassen und somit auf allgemeinen Korpora arbeitet, besteht die Idee hinter dem Chrome-Browser-Plugin Storyfinder (Remus et al. 2017) in der späteren Bereitstellung der individuellen Lesehistorie seiner Nutzenden. Diese kann wiederum als Entitätennetzwerk interaktiv exploriert und auch editiert werden.

Falls vom Nutzer bzw. von der Nutzerin geöffnete Webseiten zu Storyfinder hinzugefügt werden, sendet das Plugin die Seite an einen Server, auf dem eine linguistische Pipeline (Abschn. 3.2) eine Spracherkennung durchführt und wiederum eine Namenserkennung (Abschn. 3.2.5) und auch eine Schlüsselworterkennung auf Basis von Differenzanalyse (Abschn. 5.9) durchführt. Namen bzw. Schlüsselwörter, welche zusammen innerhalb eines Satzes auftreten, werden in der editierbaren Netzwerkdarstellung der momentan aktiven Webseite verbunden und dem globalen Netzwerk der oder des Benutzenden hinzugefügt, welches in Abb. 7.3 in groß dargestellt ist. Letzteres kann von den Nutzenden verschieden exploriert werden: Gruppen besuchter Seiten können ein- und ausgeblendet werden, einzelne Entitäten können in den Fokus rücken wie *Christian Eriksen* in der Abb. 7.3. Schließlich können entsprechende Textstellen zu Entitäten oder deren Beziehungen gruppiert dargestellt werden, wie in der Abb. 7.3 als Inset dargestellt. Im Bild sind andere dänische Nationalspieler, die Länder der beteiligten Mannschaften und beteiligte Organisationen zu sehen, inklusive des Vereins des Spielers. Bei längerer Nutzung bilden sich die Hauptinteressen der nutzenden Person auch visuell heraus; in der Abb. 7.3 sind rechts unten Entitäten um den damaligen Bundesgesundheitsminister Spahn zu sehen.

## 7.2.3 Investigativtool New/s/leak

Schließlich soll noch das Tool New/s/leak (Yimam et al. 2016; Wiedemann et al. 2018) vorgestellt werden, welches in Zusammenarbeit mit investigativen Journalisten und Journalistinnen für das Analysieren von geleakten Dokumentsammlungen entwickelt wurde und als einziges der hier vorgestellten Tools eine facettierte Suche sowohl mit gegebenen als auch mit automatisch extrahierten Metadaten umfasst.

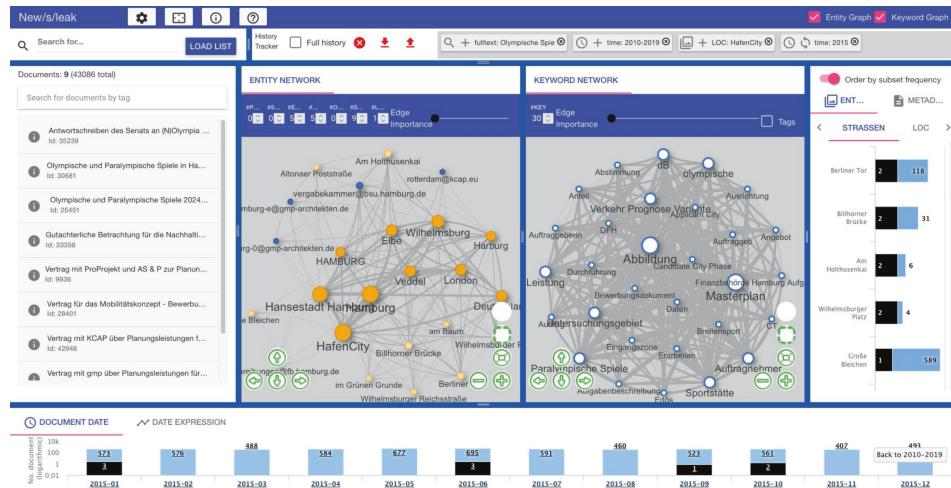
Das Tool ist auf die Verarbeitung heterogener digitaler Sammlungen optimiert und nutzt Apache Tika (<https://tika.apache.org/>) zum Einlesen von Texten aus der für solche Sammlungen typischen diversen elektronischen Formate. Diese werden von einer Pipeline (Abschn. 3.2) bestehend aus einer Spracherkennung und einer multilingualen Namenserkennung (Abschn. 3.2.5) für mehr als 40 Sprachen verarbeitet; weitere Elemente bestehen in der Extraktion und im Auflösen von Zeitausdrücken (Strötgen und Gerz 2013), dem Indizieren vorgegebener Listen und dem Extrahieren von Schlüsselwörtern mit Differenzanalyse (Abschn. 5.9). Die Ablage der Dokumente und deren Metadaten erfolgt hauptsächlich in einem Elasticsearch-Index (Gormley und



**Abb. 7.3** Zwei Ansichten des Browser-Plugins Storyfinder für die Zusammenfassung vom Nutzer individuell am 23. Juni 2021 aufgerufener Webseiten zur Wiederbelebung des Dänischen Nationalspielers Christian Eriksen, der während eines Vorrundenspiels bei der Fußballeuropameisterschaft 2021 einen Herzstillstand erlitten hatte. In Groß: Netzwerk um Christian Eriksen. Als Inset: Textstellen aus vorher besuchten Webseiten

Tong 2015), welcher die für die Anwendung notwendigen Anfragen unterstützt; weitere Konfigurationen und nutzerbasierte Änderungen werden in einer relationalen Datenbank abgelegt.

Die Oberfläche implementiert konsequent das Filterkonzept der facettierten Suche: Filter können gesetzt und wieder aufgehoben werden, um die im allgemeinen große Ausgangsmenge von Dokumenten auf wenige Texte zu beschränken, welche dann im Einzelnen gelesen werden können. Im Beispiel in Abb. 7.4 wurde auf Daten des Hamburger Transparenzportals nach *Olympische Spiele* gesucht, auf deren Ausrichtung im Jahr 2024 sich die Stadt Hamburg im Jahr 2015 zeitweilig bewerben wollte. Über die Zeitleiste unten im Bild wurde die Suche auf Dokumente aus dem Jahr 2015 beschränkt. Sowohl in der Zeitleiste als auch rechts bei den Entitäten und Metadaten zeigen blaue Balken die Gesamtzahl der Dokumente mit diesem Metadatum, schwarze Anteile symbolisieren die in der momentanen Konfiguration vorhandenen. Da in der Ansicht in der Mitte, welche das Entitätennetzwerk (wie in den vorangegangenen Abschnitten diskutiert) und ein Netzwerk an Schlüsselwörtern zeigt, der Ort *HafenCity* hervorstach, wurde auch dieser als Filter hinzugefügt; die Liste der aktiven Filter ist oben in der Leiste sichtbar. Die verbliebenen neun Dokumente (siehe linke Leiste) können nun entweder in einer Textansicht oder auch als Original-PDF-Dokumente betrachtet werden. Die Netzwerke in der Mitte, welche sich jeweils nur auf die aktuell gefilterten Dokumente beziehen,



**Abb. 7.4** Screenshot von New/s/leak auf Daten des Hamburger Transparenzportals zur gescheiterten Olympiabewerbung Hamburgs im Jahr 2015

fungieren als visuelle Inhaltsangabe: Im Beispiel sind Straßennamen in der HafenCity und angrenzender Viertel und E-Mail-Adressen der mit dem Masterplan für die dort geplanten Sportstätten beauftragten Architekturbüros sichtbar.

New/s/leak wurde bereits mehrfach für die investigative journalistische Arbeit genutzt und ist unter <https://www.newsleak.io/> frei verfügbar. Die linguistischen Verarbeitungskomponenten wurden zudem in Hoover (<https://github-wiki-see.page/m/liquidinvestigations/docs/wiki/>) integriert, der international beliebtesten Plattform für kollaborativen investigativen Journalismus auf digitalen Sammlungen.

## 7.2.4 Zusammenfassung

Zusammengefasst bietet die Darbietung und Suche von Informationen mithilfe von Entitätsnetzwerken einen weiteren Zugang zu Dokumentsammlungen und ermöglicht intuitive Navigation und inhaltliches Erschließen auf Basis von Namen und deren Verbindungen, welche gegenüber Topic-Modellen (Abschn. 6.4) eben nur einen bestimmten Aspekt – Entitäten und deren Verbindungen – herausgreifen. Entitätennetzwerke sind ein gutes Beispiel für ein Element von Text-Mining-Anwendungen, welche erst durch linguistisch orientierte Vorverarbeitung ermöglicht werden. Sie sind für eine Vielzahl von Anwendungen einsetzbar, von denen drei in diesem Abschnitt genauer beleuchtet wurden.

## 7.3 Sentimentanalyse

### 7.3.1 Begriffsbestimmung, Einsatzbereiche, Herausforderungen

Texte enthalten nicht nur Information über Gegenstände und Sachverhalte, sondern sie können auch als Appell oder als Ausdruck von Meinungen und Befindlichkeiten eines Autors oder einer Autorin an die Lesenden verstanden werden (vgl. Abschn. 1.1 und A2). Neben Aussagen finden sich in einem Text daher auch sprachliche Ausdrücke in Form von Ausrufen und Bewertungen. Gegenstand der **Sentimentanalyse** (engl. sentiment analysis), oft auch als Stimmungsanalyse bezeichnet, ist ganz allgemein die Identifikation und Analyse von solchen Ausdrücken (sentiment expressions), mit denen eine subjektive Befindlichkeit oder eine Bewertung von Objekten, Ereignissen oder Meinungen (sentiment targets) ausgedrückt werden. Typische Bewertungsdimensionen sind dabei **Polaritäten** wie *positiv – negativ* für die Polaritätsanalyse oder *subjektiv-objektiv* für die Subjektivitätsanalyse.

Sentimentanalyse hat sich seit den Arbeiten von Wiebe et al. (2004) zur Subjektivitätsanalyse und Pang et al. (2002) zu ersten Polaritätsklassifikationen von Filmreviews als eine für die Praxis relevante Anwendung des Text Mining etabliert. In den letzten Jahren liegt der Schwerpunkt dabei auf der Analyse von Sentiments in sozialen Medien, beispielsweise der Analyse von Kundenfeedback für das Marketing und die Qualitäts-sicherung oder der Analyse von Kommentaren zu tagesaktuellen Ereignissen für politik-wissenschaftliche oder wirtschaftswissenschaftliche Fragestellungen (vgl. Ahmad 2011). Für die Praxis ist dabei nicht nur von Interesse, ob ein Text überhaupt Sentiment-bewertungen enthält, sondern vor allem auch, welche Aspekte der zu bewertenden Sache im Text adressiert werden, beispielsweise *Ticketkauf*, *Gastronomisches Angebot*, *Reisen mit Kindern* oder *Toiletten* als Kategorien von sentiment targets bei der Sentimentanalyse von Nachrichten und Blogs zur deutschen Bahn in sozialen Medien. Dieser Ansatz der Sentimentanalyse, bei der unterschieden wird zwischen dem sog. document level, den Aspektkategorien, den sentiment targets und den auf den Text sowie die verschiedenen Aspektkategorien bezogenen Polaritäten, wird allgemein als **Aspekt-basierte Sentimentanalyse** bezeichnet (vgl. Wojatski et al. 2017).

---

#### Ebenen der Aspekt-basierten Sentimentanalyse

Betrachten wir als Beispiel einen Presstext zur Deutschen Bahn:

„Bald schneller mit der Bahn von Deutschland nach Paris. 5 Stunden 40 Minuten, statt wie bisher 6 Stunden 20 Minuten. Straßburg. Man kann auch öfter fahren.“ (Schwäbisches Tagblatt GmbH 30.09.2015)

Im Sinne der Aspekt-basierten Sentimentanalyse sind folgende Ebenen zu unterscheiden (hier notiert als Attribut-Wert-Paare):

**Document level:** positiv.

**Aspekt\_1:** Fahrtzeit\_und\_Schnelligkeit.

**Sentiment target\_1:** „schneller“

**Polarität\_1:** positiv.

**Aspekt\_2:** Taktung.

**Sentiment target\_2:** „öfter fahren“

**Polarität\_2:** positiv.

Quelle: GermEval Task 2017 – Aspect Based Sentiment Detection ([2021](#)),  
Trainingsdatensatz ◀

Eine zentrale Herausforderung bei der Sentimentanalyse ist die Ableitung der Polaritätsbewertung eines sentiment targets (in Bezug auf einen Aspekt). Dabei sind folgende Besonderheiten zu beachten:

1. Modifikatoren: Texte mit sentiment expressions enthalten oft sog. Modifikatoren wie „sehr“, „kaum“ oder „schrecklich“, welche die ausgedrückte Bewertung verstärken oder abschwächen, beispielsweise „sehr langweilig“, „schrecklich langweilig“ oder „kaum langweilig“.
2. Negation: Die Verwendung von Negationspartikeln wie „nicht“ oder „kein“ in sentiment expressions verkehrt oder neutralisiert eine Bewertung, beispielsweise bezeichnet der Ausdruck „keine minderwertige Ware“ eher „hochwertige Ware“; jedoch bedeutet der Ausdruck „nicht schnell“ nicht „langsam“, hier wird also die Eigenschaft *schnell* nur neutralisiert. Modifikatoren und Negation können kombiniert werden, was die Ableitung von Polaritätsbewertungen noch komplizierter macht, beispielsweise ist „nicht sehr schnell“ immer noch schnell, „schrecklich minderwertige Ware“ sollte man aber besser nicht kaufen.
3. Sarkasmus: Manchmal sind positive Bewertungen sarkastisch übertrieben, sie müssen als solche erkannt und durch eine negative Polaritätsbewertung ersetzt werden, wie beispielsweise in dem Ausruf „Bravo! Jetzt hat der Zug also wieder 1 h Verspätung.“
4. Bewertung durch Verwendung von positiv oder negativ konnotierten Eigennamen oder Modewörtern: Manche Eigennamen werden – oft nur in einem bestimmten Verwendungskontext – mit einer positiven oder negativen Polaritätsbewertung assoziiert, beispielsweise „Der neue Daimler EQS fährt sich wie ein Tesla“.

### 7.3.2 Lösungsansätze und Aufgaben

Die gängigen Lösungsansätze umfassen ganz im Sinne der Verarbeitungsparadigmen für Text, die wir im Kap. 3 vorgestellt haben, **regelbasierte, überwachte statistische** und **neuronale Verfahren** (Abschn. 3.1). In Abhängigkeit davon, welcher Ansatz verfolgt wird, können dabei die folgenden Aufgaben und Arbeitsschritte anfallen:

- Klassifikation von Texten und Textabschnitten (Absätzen, Sätzen) als für die Sentimentanalyse relevant,
- Identifizieren von Aspekten als Bewertungskategorien für Polaritäten,
- Identifizieren von sentiment targets,
- Identifizieren von sentiment expressions (Wörter, Sätze) und deren Modifikatoren (wie z. B. „sehr“, „noch“ usw.),
- Berechnung eines Polaritätswertes für sentiment expressions,
- Berechnung eines (aggregierten) Polaritätswertes für Sätze, Abschnitte, Texte,
- Identifizieren und Parametrisieren von Einflussfaktoren bei der Interpretation von sentiment expressions wie z. B. Kontext (Fachdomäne, Interessen des Bewertenden, Perspektive, ...), Medium (soziale Netzwerke, E-Mail, Film, ...) und Sprachregister (Höflichkeit, Kanalrestriktionen, Kompetenzrestriktionen, ...).

Beim regelbasierten Ansatz der Sentimentanalyse wird einerseits ein Lexikon von typischen Sentimentausdrücken verwendet (meist als **Affektwörterbuch** bezeichnet), andererseits eine programmatiche Festlegung, wie sich ausgehend von der Polaritätsbewertung einzelner Wörter die Polaritätsbewertung von Phrasen, Sätzen und ganzen Texten bestimmen lässt. Sofern vorgesehen ist, dass ein Wort anstelle einer bipolaren Bewertung wie *gut* oder *schlecht*, *positiv* oder *negativ* auch Zwischenwerte ausdrücken kann (etwa auf einer Polaritätsskala mit einem Wert zwischen +1 für positiv und -1 für negativ und 0 als neutral), so spricht man in Anlehnung an Turney (2002) von seiner **Sentiment-Orientierung**. Häufig verwendete Affektwörterbücher im Englischen sind WordNet Affect (Strapparava und Valitutti 2004) und SentiWordNet (Esuli und Sebastiani 2006). Fürs Deutsche wird häufig SentiWS (Remus et al. 2010) verwendet, ein Affektwörterbuch mit 1650 positiv und 1818 negativ konnotierten Wörtern. Für jedes Wort werden die Wortart und seine Flexionsformen angegeben, verbunden mit einer Angabe seiner Sentiment-Orientierung im Intervall [-1, +1]. SentiWS ist frei verfügbar unter <http://wortschatz.informatik.uni-leipzig.de/download/>. Für die Regeln werden einerseits syntaktische Strukturableitungen, etwa in Form von Phrasenstruktur- oder Dependenzregeln (Abschn. 2.3), verwendet, andererseits aber auch zusätzliche Kategorien wie beispielsweise *NEG* für Negationen, *INC* für Verstärkungen und *DEC* für Abschwächungen, um eine schrittweise Ableitung der in einer Phrase, einem Satz oder einem Text ausgedrückten Polaritätsbewertung aus der Sentiment-Orientierung einzelner Wörter zu ermöglichen. (vgl. Kennedy und Inkpen 2006; Neviarouskaya et al. 2009). Die Zusammenführung der Polaritäten zweier Wörter kann dann beispielsweise in der Form geschehen, dass festgelegt wird, dass der Polaritätswert beider Wörter zusammen die Summe der jeweiligen Einzelwerte ist, sofern beide Wörter einen positiven oder beide Wörter einen negativen Polaritätswert haben. Unterscheiden sich beide Wörter jedoch in ihrem Vorzeichen, so wird die Differenz aus kleinerem und größerem Polaritätswert gebildet (mit dem Ergebnis, dass ihr Polaritätswert stets negativ ist). Auf diese Weise lässt sich beispielsweise ableiten, dass der Polaritätswert von Phrasen wie „übertriebene Pflege“ oder „reinster Horror“ negativ ist, obwohl „Pflege“ und „reinst“ jeweils eine positive Sentiment-Orientierung haben.

Beim **überwachten** statistischen Klassifikationsansatz wird die Polaritätsbewertung von Phrasen, Sätzen und Texten als Klassifikationsaufgabe verstanden (Abschn. 3.1.2 und 6.6). Für die Umsetzung dieses Ansatzes werden zum einen **Markierungen** für die Klassifikation benötigt, beispielsweise die Polaritätsbewertungen *positiv* und *negativ*; zum anderen vorgegebene Merkmale. Meist werden dafür Listen von Wörtern bzw. Wort-N-Grammen mit ihrer jeweiligen Polarität oder Sentiment-Orientierung verwendet. In einem **Bag-of-words-Modell**, wie es beispielsweise Pang et al. (2002) vorgeschlagen haben, können dann **Klassifikatoren** wie Naïve Bayes, Maximum Entropy Modelle, oder Support Vector Machines (SVM) trainiert werden. Empirische Vergleiche zeigen, dass in der Praxis in diesem Vergleich meist SVMs am besten abschneiden (Go et al. 2009). Der klassifikationsbasierte Ansatz kann auch bei der Aspekt-basierten Sentimentanalyse verwendet werden (vgl. Kumar et al. 2016). Hier werden typischerweise für das Identifizieren von Sentiment Targets Merkmale eingesetzt, welche die syntaktischen Beziehungen zwischen Opinion Expression und den Targets einfangen, z. B. auf Basis von Dependenzparsing (Abschn. 3.2.4).

Neuronale Netze ermöglichen u. a. die Klassifikation von Daten und die Vorher sage von Datenabfolgen. Im Text Mining werden sie vor allem für die Klassifikation von Texten oder Textabschnitten und für die Extraktion von Wissen aus Texten, beispielsweise bei der Beantwortung von Fragen, verwendet (vgl. Abschn. 3.1 und 6.10). Auch hier ähnelt der neuronale Ansatz dem überwacht-statistischen Ansatz bzgl. der Abhängigkeit von manuell annotierten Trainingsdaten, mit dem Unterschied, dass Merkmals-Repräsentationen automatisch gelernt werden können. Während ausreichend große Trainingsdaten in diesem Fall dafür sorgen, dass die Wortrepräsentationen (Embeddings, Abschn. 5.6) geeignet für die Aufgabe gewählt werden und somit die Affektwörterbücher implizit gelernt werden, sind listenbasierte Merkmale in einem hybriden Ansatz in Situationen weiterhin von Vorteil, wo zu wenig Trainingsdaten vorhanden sind. Moderne Ansätze verwenden als Repräsentation vortrainierte kontextualisierte Embeddings (Abschn. 5.6) für die Repräsentation von Wörtern und Sätzen, welche mit Finetuning (Abschn. 6.10) auf die Klassifikationsaufgabe innerhalb neuronaler Modellarchitekturen angepasst werden. Während diese Ansätze gegenüber LSTM-basierten Sequenztaggingansätzen (Abschn. 6.8) mit statischen Word Embeddings (Abschn. 5.6) und featurebasierten Methoden deutlich bessere Ergebnisse erzielen, stellen weiterhin Phänomene wie Ironie, Sarkasmus, Negation und Modifikatoren eine Herausforderung für automatische Methoden dar, siehe Barnes et al. (2019) für eine weiterführende Analyse.

Entgegen der ursprünglichen Erwartung, dass ein universelles Verfahren der Sentiment Analyse in einer Vielzahl von Anwendungsdomänen eingesetzt werden kann, hat sich jedoch gezeigt, dass domänenspezifische Besonderheiten in viel höherem Maße berücksichtigt werden müssen, um hinreichend gute Ergebnisse zu erzielen. Viele Adjektive drücken je nach Kontext sehr verschiedene Polaritätsbewertungen aus. So stellt die Bewertung „laut“ für einen Wageninnenraum eine eher schlechte Bewertung dar, nicht aber für einen Rauchmelder. Ein Lieferdienst, der eine kalte Pizza ausliefert,

ist eher schlecht zu bewerten, als einer, dessen Pizzen noch warm sind – jedoch das Bier, das er mit ausliefert, sollte nicht warm, sondern kalt sein. Wie aber kann passend zu einer Anwendung ein domänenspezifisches Affektwörterbuch generiert werden? Um die Besonderheiten einer Anwendung zu berücksichtigen, wird meist von einer kleinen Startmenge von Wörtern ausgegangen, die für die Anwendung eindeutig eine positive oder negative Polaritätsbewertung ausdrücken. Diese Startmenge kann dann z. B. durch das Vererben von Sentiments über Kookkurrenzen (Turney und Littman 2003), das Vererben über Ähnlichkeiten (Kumar et al. 2016) oder Repräsentationslernen (Wang und Xia 2017) erweitert werden. Diese Affektwörterbücher können dann entweder für das regelbasierte Vorgehen oder als Merkmale für klassifikationsbasierte Ansätze verwendet werden. Die beste Qualität in der automatischen Sentimentanalyse wird erreicht, wenn Daten aus der Anwendungsdomäne manuell hinsichtlich ihrer Polarität annotiert werden (Abschn. 6.7), um einen Klassifikator zu trainieren; dies ist jedoch viel zeitaufwändiger als der rein listenbasierte Ansatz.

### Beispiel für das Vererben von Sentiments über Kookkurrenzen

In einem halbüberwachten Verfahren soll die Sentiment-Orientierung eines Wortes  $w$  für eine Anwendung ausgehend von einer Startmenge positiv konnotierter Wörter (Pwords) und einer Menge negativ konnotierter Wörter (Nwords) bestimmt werden.

Als Pwords mit einem Polaritätswert +1 und Nwords mit einem Polaritätswert -1 werden die folgenden Wörter verwendet (vgl. Remus et al. 2010):

Pwords: gut, schön, richtig, glücklich, erstklassig, positiv, großartig, ausgezeichnet, lieb, exzellent, phantastisch

Nwords: schlecht, unschön, falsch, unglücklich, zweitklassig, negativ, scheiße, minderwertig, böse, armselig, mies

Ausgehend von den Startwörtern wird ein Kookkurrenznetzwerk erstellt, z. B. unter Verwendung des Maßes der Pointwise Mutual Information (PMI) wie in Turney und Littmann (2003). In diesem Netzwerk wird ein Wort mit einer positiven Sentiment-Orientierung markiert, wenn die Sentiment-Orientierung der mit ihm kookkurrenden Wörter  $K_{SO}(w)$  positiv ist und mit einer negativen Sentiment-Orientierung, wenn  $K_{SO}(w)$  negativ ist. Der aufsummierte Wert aller positiv konnotierten Wörter minus aller negativ konnotierten Wörter ist der Wert der Sentiment-Orientierung von  $w$ .

Je nachdem, welche Wörter als Startmenge gewählt werden, zeigen sich deutliche Unterschiede in der Sentiment-Orientierung einzelner Wörter. So zählen zu den Wörtern mit einer besonders negativen Sentiment-Orientierung aus dem allgemeinen deutschen Wortschatz die Wörter *Schulden* und *betrügen*, werden aber als Anwendung Automobilforen verwendet, dann zählen dazu Wörter wie *Panne*, *Schaden* und *fehlerhaft*. ◀

## 7.4 Trendanalysen und News Monitoring

### 7.4.1 Trends und schwache Signale

Trendanalysen in der Finanzwirtschaft und dem Marketing haben in den letzten Jahren stark an Aufmerksamkeit gewonnen (vgl. Bea und Haas 2019). Sie bilden auch die wesentliche Grundlage für die Zukunftsforschung (vgl. Kreibich 1995; Horx 2000).

Unter einem **Trend** wird im Allgemeinen eine sich langfristig abzeichnende Entwicklung verstanden. In der Statistik wird ein Trend als Zeitreihe dargestellt. Er beschreibt die quantitative Veränderung von Ereignissen (z. B. Kauf oder Verkauf von Aktien), erklärt jedoch i.A. nicht die Hintergründe von kurzfristigen Veränderungen (z. B. plötzliche Kurssprünge). Ein besseres Verständnis eines Trends wird meist durch die Kontextualisierung der betrachteten Ereignisse ermöglicht.

Die Analyse von Trends sollte folgende Aspekte berücksichtigen (vgl. Walde 2010, S. 46):

1. Trends als Abfolgen oder Lebenszyklen von Ereignissen durchlaufen mehrere Kontexte und können somit auch als „Trendlandschaften“ bzw. Querschnittsentwicklungen beschrieben werden.
2. Trends lassen meist keine eindeutige Entwicklung entweder in die eine oder andere Richtung erkennen, sondern verlaufen meist sowohl in die eine als auch in die andere Richtung. Auch können scheinbar sich ausschließende Tendenzen miteinander verschmelzen.
3. Trends sind relativ und nicht absolut. Sie erfassen nicht alle Ereignisse, sondern meist nur einen Teil unter bestimmten, vorher festgelegten Aspekten.

Änderungen in Trendverläufen kündigen sich oft durch sog. „schwache Signale“ an (vgl. Ansoff 1975; Liebl 2018). Dabei wird davon ausgegangen, dass evolutionäre, kontinuierliche Trends oder revolutionäre, diskontinuierliche Entwicklungen (Diskontinuitäten) nicht zufällig sind, sondern durch eben diese „schwachen Signale“ frühzeitig erfasst werden können.

### 7.4.2 News Monitoring

Die Sammlung und Analyse großer Nachrichtenkollektionen mit Text Mining, das sog. **News Monitoring**, bietet sich naheliegenderweise als Grundlage für die Trendanalyse an. Als Datengrundlage dienen Texte oder in Text gewandelte gesprochene Sprache aus dem Internet oder den sozialen Medien, deren Publikationsdatum eindeutig erkennbar ist, und deren Schwerpunkt auf Nachrichten aus allen Lebensbereichen, insbesondere Politik, Wirtschaft, Unternehmen, Wissenschaft und Technologie liegt. Die für das

Sammeln und Aufbereiten dieser Texte erforderliche Prozesskette mit ihren einzelnen Arbeitsschritten wird ausführlich in Kap. 4 behandelt.

Wichtige Impulse für das News Monitoring gingen Ende der 1990er-Jahre von dem Projekt Topic Detection and Tracking (TDT) aus, ein von DARPA finanziertes Projekt mit Beteiligung der Carnegie Mellon University, Dragon Systems und der University of Massachusetts at Amherst (Allan 2002). Für das Projekt sind umfangreiche Korpora aus diversen Nachrichtenquellen (u. a. Reuters, CNN) gesammelt worden, die auch Transkriptionen aus gesprochenen Nachrichten enthalten, und über das Linguistic Data Consortium (LDC) verfügbar sind.

Zielsetzung des Projekts war das Erkennen und verfolgen von (neuen) Ereignissen in Nachrichtenströmen mit den Teilzielen 1) Segmentierung von Nachrichten in einzelne Berichte, 2) online oder retrospektives Entdecken von unbekannten Ereignissen bzw. Themen und 3) Verfolgen der Entwicklung von bekannten Ereignissen und Themen. Für die unter (2) genannte Aufgabe Novelty Detection sind einschlägige Evaluationswettbewerbe im Rahmen der **TREC-Konferenz** eingerichtet worden.

Im Rückblick war das Projekt Topic Detection and Tracking das erste größere Projekt, bei dem Dokumentenkollektionen betrachtet wurden, die mit einem Zeit- oder Raumbezug versehen sind. Weil aber die Terme, mit denen ein neues Thema verbunden ist, per definitionem nicht bekannt sind, können die üblichen Verfahren des **Information Retrievals**, die eine vorgegebene Term-Dokument-Matrix voraussetzen, hier nicht zur Anwendung kommen.

#### 7.4.3 Wörter des Tages

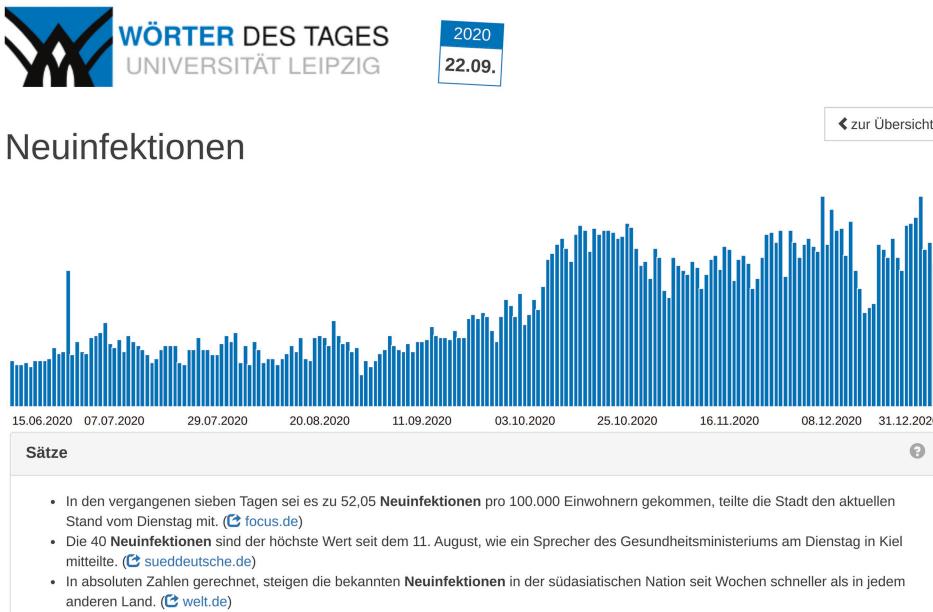
Die Anwendung „Wörter des Tages“ verbindet News Monitoring mit visueller Textanalyse (Rohrdantz et al. 2010), indem tagesaktuell relevante Themen in Nachrichtentexten erkannt und in inhaltlich kohärente Cluster angeordnet und visualisiert werden. Realisiert wird dies durch die Betrachtung von Veränderungen in den täglichen Auftretenshäufigkeiten einzelner Wörter (wie in Abschn. 1.2 definiert) in digitalen Nachrichtentexten. Das Ziel der Wörter des Tages ist eine automatisch generierte Übersicht der relevanten Nachrichtenthemen des Vortages (vgl. Quasthoff et al. 2002; Richter 2010, S. 37), die anschließend durch eine beschreibende Menge von Wörtern dargestellt werden. Dadurch wird automatisiert ein kurzer und prägnanter Überblick über das mediale Tagesgeschehen generiert. In der Abb. 7.5 werden die Hauptfunktionalitäten der Wörter des Tages vorgestellt. Die beiden Boxen zeigen einen Ausschnitt mit zwei von insgesamt elf Themenkomplexen, die am 28.10.2020 auf den Nachrichten des Vortages berechnet wurden.

Jede Box stellt jeweils ein Themencluster dar. In der linken Box finden sich Wörter zum Thema „Pandemie/Corona“. Die Rechte Box zeigt das Thema der „Neuinfektionen“ auf. Die Farbgebung hat keine Bedeutung. Beim Klick auf ein Wort erhalten wir detailliertere Informationen:



**Abb. 7.5** Zwei Themenkomplexe der Wörter des Tages vom 28.10.2020

Im Histogramm der Abb. 7.6 wird die zeitliche Entwicklung der Vorkommen des Wortes *Neuinfektion* dargestellt. Für jeden Tag sind die Anzahl der absoluten Vorkommen und der jeweilige relative Anteil am Gesamtvorkommen dargestellt. Weiterhin werden Beispielsätze für das Wort angezeigt (unten), in welchen das jeweilige Wort auftritt. Diese Beispiele können einerseits Zusatzinformationen bei uneindeutigen Wörtern des Tages geben (z. B. für das Wort *Zahl* in Abb. 7.5), oder auch Klarheit über den Kontext des Auftretens verschaffen. Durch einen Klick auf den Balken eines jeweiligen Tages im Histogramm werden die Informationen für das jeweilige Wort für



**Abb. 7.6** Histogramm zur zeitlichen Entwicklung der Vorkommen des Wortes *Neuinfektionen*



**Abb. 7.7** Zwei Themenkomplexe am Tag nach der US-Präsidentenwahl (04.11.2020)

den entsprechenden Tag angezeigt. Zusammen mit dem Histogramm selbst erlaubt diese Funktionalität eine effiziente interaktive Exploration des Auftretens eines Wortes (bzw. der Veränderungen des Auftretens) in großen Datenbeständen über die Zeit (Heyer et al. 2011).

Aber auch andere Nachrichtenressorts werden ebenso zuverlässig erfasst, z. B. Sport, Kultur, oder wie in Abb. 7.7, Politik. Thematisch finden sich hier eine Box über die Präsidentschaftswahl und ihre Kandidaten (links), sowie eine Box in der es um die Auszählung der Wahlstimmen geht (rechts). Die Wörter und Mehrworteinheiten werden jeweils nur genau einem Thema zugeordnet. So hätte zum Beispiel der Begriff *Swing State* mindestens so gut auch in die rechte Box gepasst.

Die Wörter des Tages bieten eine tagesaktuelle Übersicht an thematischen Gruppierungen von Wörtern, die in der Nachrichtenmenge des vorherigen Tages aufgrund ihrer statistischen Auffälligkeit als relevant erkannt wurden. Dazu werden Online-Nachrichtentexte gesammelt, durch eine Reihe von linguistischen Vorverarbeitungsschritten bis auf die Wort-Ebene analysiert und schlussendlich in einer Datenbank gespeichert. Dort wird das Auftreten jedes Wortes über den jeweiligen gesamten Tag gezählt und die statistische Signifikanz des Unterschiedes zu einem Referenzkorpus des Vorjahres ermittelt. Wörter mit signifikanter Erhöhung der Auftretenshäufigkeit (vgl. Abschn. 5.8.1) werden als Kandidaten für die Wörter des Tages betrachtet. Im Anschluss werden die signifikantesten Wörter mittels **Topic-Modellen** (vgl. Abschn. 6.4) in Gruppen eingeordnet, die oft inhaltlich kohärente Themenkomplexe bilden. Die Nachrichtenquellen der Wörter sind dabei getrennt nach Sprache und werden aktuell für Deutsch berechnet, in der Vergangenheit aber auch für Englisch und Französisch.

Zu diesem Ergebnis kommt man durch eine Aneinanderreihung von einzelnen Schritten in der linguistischen **Verarbeitungspipeline** (vgl. Abschn. 3.2 und 4.2). Die detaillierte Prozesskette enthält folgende Teilschritte, die schon von Richter (2010) beschrieben wurden; hier beziehen wir uns auf die aktuelle Version (Stand: 2020).

1. Web Crawling: Um überhaupt eine Analyse durchführen zu können, bedarf es einer Textgrundlage. Weiterhin müssen wir die Dokumente dem Tag ihrer Veröffentlichung zuordnen, um die Ergebnisse pro Tag aufgeschlüsselt analysieren zu können. Durch News-Feeds (RSS/Atom), welche auf neu veröffentlichte Nachrichten aufmerksam machen, werden fokussiert nur die Artikel eines jeweiligen Tages bezogen.
2. Text-Extraktion: Die besagten News-Feeds verweisen auf Webseiten, also **HTML-Dokumente** (vgl. Abschn. 4.2). Die Methoden der automatischen Sprachverarbeitung operieren i. d. R. aber auf Text, weswegen vor jeglichen linguistischen Verfahren erst der eigentliche Text extrahiert werden muss. Das Hauptproblem besteht hier darin den Haupttextkörper von den anderen HTML-Elementen (z. B. Navigation, Werbung, Header, Footer) zu trennen.
3. Linguistische Vorverarbeitung: Der Rohtext wird zuerst in Sätze und dann in einzelne Wörter segmentiert. Jeder Satz wird durch eine **Sprachklassifikation** auf die erwartete Sprache geprüft (und ggf. verworfen). Sätze, die Verunreinigungen enthalten, wie z. B. untypische Sonderzeichen, fehlende Satzzeichen, oder auch deutlich vom Durchschnitt abweichende Satzlängen, werden durch eine Reihe von **regulären Ausdrücken** und Regeln gefiltert (vgl. Abschn. 4.2). Für jedes Token wird außerdem durch **Part-of-Speech Tagging** (vgl. Abschn. 3.2) die Wortart bestimmt. Durch eine listenbasierte Mehrworterkennung werden zusätzlich **Mehrwocheinheiten** identifiziert, die im Folgenden wie Einzelwörter behandelt werden.
4. Signifikanz: Für jedes (Mehr-)Wort wird die Wortsignifikanz mithilfe eines **Korpusvergleichs** (vgl. Abschn. 5.8) ermittelt. Dabei bilden die Nachrichten des jeweiligen Tages das Analysekorpus und die gesamte Textmenge des Vorjahres das Referenzkorpus. In den Wörtern des Tages wird mittlerweile das **Log-Likelihood Maß** (Dunning 1993) verwendet (wie z. B. in Heyer et al. 2011). Da nicht jedes Wort am Ende als Beschreibung alleinstehend verständlich ist, werden vor allem Nomina und Phrasen behalten, zudem werden zwar neu auftretende Wörter (d. h. Wörter die im Referenzkorpus noch nicht vorhanden sind) erlaubt, aber mit einem erhöhten Signifikanz-Schwellwert benachteiligt (vgl. Richter 2010, S. 58).
5. **Clustering**: Als finaler Schritt müssen die ermittelten signifikanten Wörter in kohärente Gruppen geclustert werden. Das Clustering wird über ein Topic-Modell realisiert, das analog zum Referenzkorpus auf den Daten des Vorjahres trainiert ist. Der Inferenz-Schritt wird dabei für jeden Satz ausgeführt, der ein signifikantes Wort enthält. Danach wird jedes Wort genau dem Topic zugeordnet, in welchem es die maximale Wahrscheinlichkeit eines Auftretens erreicht, d. h. einem Wort  $w$  wird dem Topic  $z$  zugeordnet, für das  $P(w|z)$  maximal ist (oder kurz  $\operatorname{argmax}_z P(w|z)$  mit der Notation aus Abschn. 6.4.2). Jedes dieser Topics mit zugeordneten Wörtern bildet ein Cluster und wird danach auf die  $k = 20$  signifikantesten Wörter eingeschränkt. Als finaler Schritt werden die Ergebnisse visualisiert, indem ein Cluster durch jeweils eine Box visualisiert wird (wie in Abb. 7.5 und 7.7 gezeigt).

Die Wörter des Tages sind im Rahmen des Projektes Deutscher Wortschatz entstanden (vgl. Anhang A5), in welchem seit Anfang der 1990er-Jahre große Mengen an Text gesammelt und aufbereitet wurden (Quasthoff und Richter 2005). Die Entwicklung der Wörter des Tages reicht zurück in das Jahr 2002 (Quasthoff et al. 2002, Richter 2010), und selbst die heutige Version wird fast 20 Jahre später noch verbessert und gepflegt. Eine norwegische Variante der Wörter des Tages wurde von Eiken et al. (2006) realisiert und demonstrierte die Übertragbarkeit des bestehenden Konzeptes auf andere Sprachen. Eine wesentliche Herausforderung stellte damals die riesige Menge an unstrukturierten Daten dar, welche erst durch WCTAnalyze (vgl. Gottwald et al. 2008) bewältigbar wurden. WCTAnalyze ist eine Software, die riesige Mengen an unstrukturierten Textdaten in Zeitscheiben organisiert, darauf einen effizienten Zugriff auf abgeleitete Daten wie Worthäufigkeiten oder Wort-Kookkurrenzen anbietet und dabei gleichzeitig den Speicherverbrauch substantiell reduziert (Gottwald et al. 2008).

Das Konzept der Wörter des Tages ist ein effektiver Ansatz, um mit schwachen Signalen umzugehen, für die klassische IR-Ansätze nicht geeignet sind: Da relevante Themen in vielen Quellen parallel berichtet werden, tauchen die relevanten Begriffe eines Tages ebenso in vielen anderen Quellen auf. Als Folge daraus würden die relevanten Wörter in einer Suchanwendung auf der Grundlage von **tf·idf** stark benachteiligt werden, da sich ihr Vorkommen auf viele Dokumente verteilt und ihr tf-idf-Wert durch die idf heruntergewichtet wird. Eine weitere Stärke des Ansatzes ist, dass er komplett sprachunabhängig ist, sofern die Sprache ein Konzept für Wortgrenzen enthält, d. h. tokenisierbar ist. Eine Schwäche dieser Anwendung ist, dass die Topic-Modell-Cluster nicht notwendigerweise die Realität abbilden müssen und u. a. bei Überschneidungen der Eigennamen (z. B. *[Thomas] Müller* (Sport) und *[Gerd] Müller* (Politik)) zwei komplett verschiedene Themenstränge vermischt werden können. Insgesamt ist diese Anwendung ein anschauliches Beispiel, wie eine Reihe von Methoden des Text Mining und der automatischen Sprachverarbeitung kombiniert werden können, um aus Rohdaten Wissen zu extrahieren, das am Ende dem Nutzer bzw. der Nutzerin durch eine visuelle Zusammenfassung aktueller Nachrichtentexte einen schnellen und kompakten Überblick über das Zeitgeschehen verschafft.

---

## 7.5 Neologismen

Natürliche Sprachen befinden sich in einem ständigen Wandel: Neue Wörter entstehen oder werden aus anderen Sprachen übernommen, andere Wörter verschwinden langsam aus dem Sprachgebrauch. Speziell aus Anlass von **Rechtschreibreformen** kann sich schlagartig die Schreibweise für viele Wörter ändern. In informellen Medien wie Internetforen werden Trends zu Veränderungen früh sichtbar: Häufigere Getrennt- statt Zusammenschreibung und allgemeine Kleinschreibung von Substantiven. Ob sich solche Trends später durchsetzen, ist jedoch völlig offen.

### 7.5.1 Neologismenwörterbücher

Eine klassische Fragestellung besteht darin, die neuen Wörter aus einem vorgegebenen Zeitraum zu sammeln und in einem Neologismenwörterbuch zusammenzustellen. Ein solcher Zeitraum besteht typischerweise aus mehreren Jahren oder Jahrzehnten. Bei der korpusbasierten Erstellung eines Neologismenwörterbuchs ist natürlich die Existenz vergleichbarer Korpora aus den entsprechenden Zeitabschnitten notwendig, sodass sich solche Neologismenwörterbücher auf die nahe Vergangenheit beschränken.

Es gibt verschiedene Typen von Neologismen, die unterschiedlich stark von Interesse sein können:

Häufigkeit – neu oder nur nicht allgemein bekannt:

- Wörter können völlig neu sein wie *Mautdebakel*, *Pfandgegner* und *Elterngeld*.
- Fachbegriffe, die Experten und Expertinnen schon länger bekannt sind, finden den Weg in die Alltagssprache: *Businessplan*, *Defizitgrenze*, *Radioquote* und *ultrafein*.

Der Anlass für den Gebrauch der Wörter kann verschiedene Ursachen haben:

- Gebunden an konkrete Ereignisse wie den Kampf gegen den Terror, den Ausbruch einer Pandemie oder an politische Reformen.
- Technische Entwicklungen, die vor allem neue Fachwörter hervorbringen, beispielsweise in den Bereichen Computer (*Gamer*), Telekommunikation (*Fotohandy*) und Auto (*Tagfahrlicht*).

Eine weitere dritte Gruppe wird von unspezifischen Wörtern (*gefühlt*, *erwartbar*) gebildet, die schon lange bekannt sind und auch in der Alltagssprache benutzt werden, deren Häufigkeit jedoch auch ohne offensichtlichen Anlass stark zugenommen hat.

Weniger von Interesse sind dagegen meist Wörter, die explizit mit speziellen Orten oder einzelnen Personen verbunden sind:

- Eigennamen und Zusammensetzungen mit Eigennamen (z. B. *Irak-Abenteuer*, *NPD-Fraktion*),
- Titel, die sich im Wesentlichen auf nur eine Person beziehen wie *Hotelerbin* (meist Paris Hilton), *Plastinator* (Gunther von Hagens) oder *Polizistensohn* (meist Jan Böhmermann),
- Regionalismen, die fast ausschließlich in Quellen aus nur einer Region vorkommen (*Ausbildungsmesse*, *Frauenreferat*).

Die korpusbasierte Suche setzt eine operationalisierbare Definition des Begriffs Neologismus voraus, welche mit der Erwartung der Leserschaft eines Neologismenwörterbuchs möglichst gut übereinstimmt. Aufgrund dieser Erwartungen werden Wörter

gesucht, die neu sind oder in einer neuen, zusätzlichen Bedeutung auftreten und in die Alltagssprache aufgenommen worden sind.

Die Operationalisierung besteht in der Beobachtung einer plötzlichen Vergrößerung der mittleren Häufigkeit beginnend bei (fast) Null als Kriterium für die Neuheit sowie einer gewissen Mindestgesamtzahl als Kriterium für die Aufnahme in die Alltagssprache.

Diese Kriterien müssen noch verfeinert werden und ermöglichen nur das Erzeugen von Kandidatenlisten, die weiter manuell bearbeitet werden müssen, damit man sich auf Neologismen entsprechend der gewählten Definition beschränkt.

In Neologismenwörterbüchern findet man üblicherweise nicht nur eine Liste dieser neuen Wörter, sondern auch zusätzliche Angaben zu den Wörtern:

- Eine kurze Bedeutungsbeschreibung oder eine Definition,
- Eine Beschreibung des ersten bekannten Auftretens: Quellenangabe mit Datum,
- Die Verwendung in Beispielsätzen,
- Die Häufigkeitsentwicklung im Laufe der betrachteten Zeit.

Auch hier kann das Korpus die Beschaffung dieser Angaben unterstützen. Natürlich kann nur das älteste Auftreten im vorliegenden Korpus ermittelt werden. Die ältesten Beispielsätze sind nicht nur als Belegstellen hilfreich, sondern erklären häufig auch das neue Wort, da es bei dem oder der Lesenden zu diesem Zeitpunkt noch nicht als bekannt vorausgesetzt werden kann. Damit bilden sie eine gute Grundlage für eine Definition.

---

#### Beispiel: Unterschichtenfernsehen

Dass ein Wort erst lange nach seiner Entstehung allgemein bekannt wird, soll am Beispiel des Wortes *Unterschichtenfernsehen* verdeutlicht werden, dessen komplette Entstehungsgeschichte dokumentiert wurde. Eine Suche im Wortschatz-Zeitungskorpus liefert den allerersten Treffer am 23. Februar 2005, insgesamt finden sich 32 Treffer für das Jahr 2005. Dieser plötzliche Anstieg muss mit einem Ereignis verbunden sein, und die Suche in den Beispielsätzen ergibt folgende Erklärung aus dem Hamburger Abendblatt vom 15. März 2005: *Als „Unterschichtenfernsehen“ verhöhnt Entertainer Harald Schmidt die Privatsender und ihr Publikum, seit er nicht mehr bei Sat.1, sondern in der ARD auftritt.*

Das Wort *Unterschichtenfernsehen* wurde als Dauerpointe verwendet und war schmerhaft für die angesprochenen Sender und Zuschauer sowie Zuschauerinnen.

Eine Recherche von Christoph Amend (2005) konnte das Wort *Unterschichtenfernsehen* jedoch rund 10 Jahre zurückverfolgen: Bereits Anfang 1996 wurde in der Satirezeitschrift *Titanic* der Fernsehsender Sat.1 mit diesem Wort bezeichnet. Ebenso verwendet wurde es 2001 von Jochen Hörisch für RTL und SAT.1, doch durch diese vereinzelten Nennungen konnte das Wort sich nicht in der Alltagssprache durchsetzen. Harald Schmidt traf übrigens in dem im Dezember 2004 erschienenen Buch *Generation Reform* von Paul Nolte auf das Wort *Unterschichtenfernsehen*. ◀

Um den Häufigkeitsverlauf eines Wortes über einen längeren Zeitraum wie mehrerer Jahre grafisch darzustellen, bieten sich Säulendiagramme an. Zusätzlich lässt sich die absolute Häufigkeit durch die Breite der Säulen darstellen. Die folgenden Säulendiagramme in den Abbildungen sind aus Quasthoff (2007) entnommen.

Entsprechend verschiedener Häufigkeitsverläufe lassen sich Neologismen in verschiedene Klassen einteilen. Die Abbildungen wurden mit freundlicher Genehmigung des Verlags de Gruyter aus Quasthoff (2007) entnommen und zeigen den Häufigkeitsverlauf der entsprechenden Wörter im Zeitraum 1995–2005.

1. Echter Neologismus, völlige Neuschöpfung (*Defizitverfahren, brutalstmöglich*, siehe Abb. 7.8), wobei die Häufigkeit über einen längeren Zeitraum nahezu gleichmäßig ansteigen kann oder nach einem (meist durch ein Ereignis bedingten) starken, plötzlichen Auftreten auch sofort wieder abfallen kann.
  - a. Tendenz wachsend oder
  - b. Nach starkem Anfang fallende Tendenz.
2. Ähnlich kann der Häufigkeitsverlauf für vorher lange Zeit niederfrequente Wörter sein (siehe Abb. 7.9):
  - a. Längerfristiges Wachstum der Häufigkeit (*Bürokratieabbau*) oder
  - b. Plötzliches Auftauchen (z. B. wegen eines Ereignisses) in der Allgemeinsprache für jede Person wahrnehmbar, danach aber wieder seltener (*Reformdebatte, Acrylamid, Feinstaub*).
3. Wörter mit neuer Bedeutung (*Kurzmitteilung*), die in ihrer neuen Bedeutung plötzlich wegen größerer Häufigkeit wahrgenommen werden können (Abb. 7.10).
4. Wörter mit kurzer Lebensdauer, die rund um spezielle Ereignisse entstehen, kurze Zeit relativ häufig auftreten und dadurch allgemein bekannt werden, danach aber schnell wieder verschwinden (Abb. 7.11).

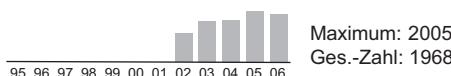
Bei der Auswahl solcher Wörter mit auffälligem Häufigkeitsverlauf für ein Neologismenwörterbuch stellen sich noch andere Fragen. Um die Relevanz eines Wortes für ein Wörterbuch einzuschätzen, möchte man eine Vorhersage über die Bekanntheit in der Zukunft treffen, genauer:

#### **Defizitverfahren**

Politik

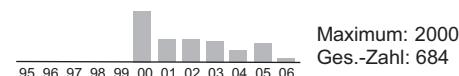
(Auch: Defizit-Verfahren.)

**Von der EU eingeleitetes Verfahren  
gegen Staaten mit zu hohem  
Haushaltsdefizit.**



#### **brutalstmöglich**

Schonungslos.

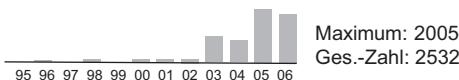


**Abb. 7.8** Neologismus *Defizitverfahren* mit steigender Häufigkeit und Neologismus *brutalstmöglich* mit sinkender Häufigkeit

**Bürokratieabbau**

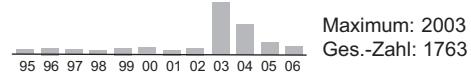
Politik

(Auch: Bürokratie-Abbau.)

**Vereinfachung von Vorschriften und Verwaltungsregeln.****Reformdebatte**

Politik

(Auch: Reform-Debatte.)



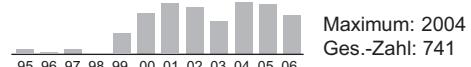
**Abb. 7.9** Neologismus *Bürokratieabbau* mit längerfristigem Wachstum der Häufigkeit und Neologismus *Reformdebatte* mit plötzlichem Auftauchen

**Abb. 7.10** Neologismus

*Kurzmitteilung* mit neuer Bedeutung und damit steigender Häufigkeit

**Kurzmitteilung**

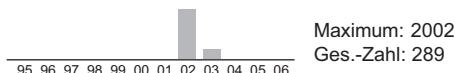
Telekommunikation

**Kurze Mitteilung über das Handy, SMS.****Pfandgegner**

Wirtschaft

**Gegner des 2003 eingeführten Dosenpfands.****UMTS-Auktion**

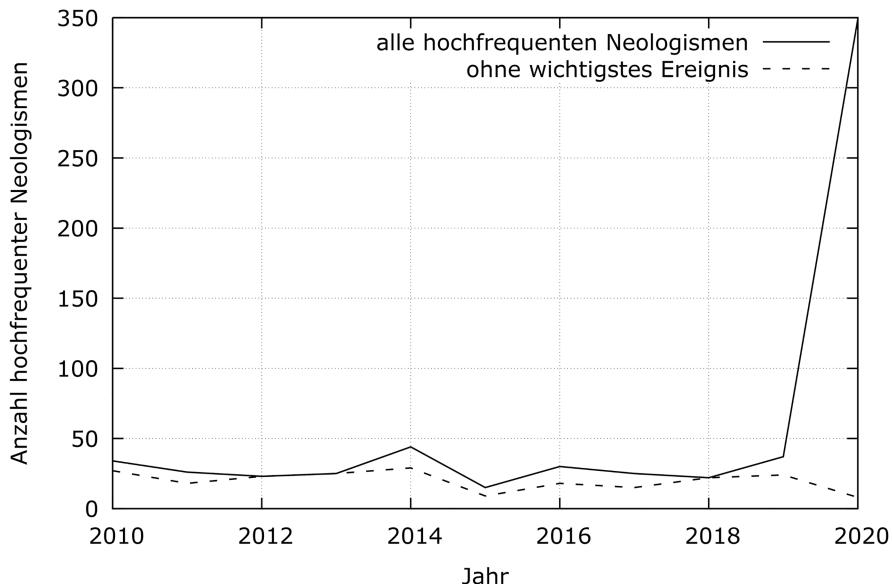
Telekommunikation



**Abb. 7.11** Neologismus *Pfandgegner* und Neologismus *UMTS-Auktion* mit kurzem Auftreten

- Nachhaltigkeit: Wird man in zehn Jahren das Wort noch kennen?
- Tendenz: Wird die Häufigkeit eines Wortes zunehmen?

Diese Fragen lassen sich aus der Häufigkeitsanalyse der Vergangenheit nicht beantworten und entsprechende Vermutungen müssen durch Lexikographen getroffen werden. Speziell bei Wörtern, deren Entstehen an spezielle gesellschaftliche Ereignisse oder technische Entwicklungen gebunden ist, hängt der weitere Häufigkeitsverlauf maßgeblich von der weiteren Entwicklung in der Folgezeit ab.



**Abb. 7.12** Hochfrequente Neologismen der Jahre 2010 bis 2020

### 7.5.2 Hochfrequente Neologismen 2010–2020

Die folgende Analyse betrachtet die Neologismen aus den Jahren 2010 bis 2020. Dabei wurden die jährlichen News corpora der Leipzig Corpora Collection (Anhang A5) betrachtet. Diese werden nach einem gleichbleibenden Verfahren erstellt. Sie unterscheiden sich zwar leicht in der Größe, aber die nachfolgenden Kriterien sind unabhängig von kleineren Schwankungen in der Korpusgröße. Folgende Auswahlkriterien werden berücksichtigt:

- Die Wörter befinden sich im aktuellen Jahr unter den häufigsten 10.000 Wörtern und waren im letzten Jahr überhaupt nicht in der Wortliste.
- Keine Orts- oder Personennamen,
- Keine gebeugten Formen, falls die Grundform häufiger ist und deshalb wahrscheinlich in früheren Jahren aufgetaucht ist,
- Jeweils nur eine Schreibweise (z. B. bei *Mund-Nasenschutz*, *Mund-Nasen-Schutz*, *Mund-Nase-Schutz*).

Kann man erwarten, dass diese Anzahl pro Jahr relativ konstant ist? Da, wie schon im letzten Abschnitt (Abschn. 7.5.1) bemerkt, viele Neologismen an aktuelle Ereignisse gekoppelt sind, hängt die Anzahl der mit diesem Ereignis verbundenen Wörter an der Größe des Ereignisses. Abb. 7.12 zeigt die Zahl der entsprechend der obigen Kriterien gefundenen Neologismen für die Jahre 2010 bis 2020. Die untere Linie zeigt diese Zahl ohne Berücksichtigung des jeweils auffälligsten Ereignisses (falls vorhanden).

**Tab. 7.2** Hochfrequente Neologismen von 2010 bis 2020

Jahr	Anzahl	Neologismen
2010	34	7 × Eurorettung: Rettungsschirm, Eurokrise, Euro-Rettungspaket; 6 × Vulkanausbruch: Aschekonzentration, Luftraumsperre, Vulkanasche-Wolke; Sonstige: Enthüllungsplattform, Missbrauchsbeauftragte, Steuersünder-CD etc.
2011	26	8 × EHEC: EHEC-Epedemie, EHEC-Erreger, EHEC-Patienten; Sonstige: Benzin-Gipfel, Neonazi-Mordserie, Protesthochburg etc.
2012	23	Dänen-Ampel, Grexit, Lebensleistungsrente, Organspende-Skandal, Schmäh-video
2013	25	Ausspähaffäre, Armutszuwanderung, Chemiewaffenangriff, Pädophilie-Debatte, Pferdefleischskandal
2014	44	15 × Ebola: Ebola-Ausbruch, Ebola-Erkrankung, Ebola-Verdachtsfall; Sonstige: Fahrdienst-Vermittler, Nackt-Selfies, Trollversteher etc.
2015	15	6 × Abgasaffäre: Abgas-Manipulationen, Abgas-Skandal, Dieselgate; Sonstige: Registrierungszentren, Selektorenliste, Zwei-Personen-Regel etc.
2016	30	12 × Brexit: Brexit-Abstimmung, Brexiteers, Brexit-Wortführer; Sonstige: Bankomatgebühren, Flüchtlingsdeals, postfaktisch etc.
2017	25	10 × Jamaika-Koalition: Jamaika-Aus, Jamaika-Gespräche, schwarz-gelb-grün; Sonstige: Austrittsrechnung, Deeskalationszone, Netzwerkdurchsetzungsgesetz etc.
2018	22	Ankerzentren, Brückenteilzeit, Gelbwesten, Vergeltungszölle, Wohngipfel
2019	37	13 × Klima: Klimademo, Klimakonsens, Waldgipfel; Sonstige: Mietendeckel, Nachunternehmerhaftung, Teileinigung etc.
2020	349	341 × Corona: Maskenpflicht, Coronavirus, coronabedingt, Covid, Inzidenzwert, Teil-Lockdown; Sonstige: Anti-Rassismus-Proteste, Innovationsprämie, Pop-up-Radweg etc.

Die nachfolgende Zusammenstellung zeigt wichtige hochfrequente Neologismen zu den wichtigsten Ereignissen des jeweiligen Jahres sowie weitere ausgewählte hochfrequente Neologismen (Tab. 7.2).

## 7.6 Kontextvolatilität

### 7.6.1 Kurze Zusammenfassung des Verfahrens

Für das Text Mining sind oftmals nicht nur die **Kookkurrenzen** von Wörtern von Interesse, sondern auch die zeitlichen Veränderungen von Kookkurrenzen, wie sie beispielsweise mit dem Werkzeug DiaCollo (<https://www.clarin-d.net/de/kollokationsanalyse-in-diachroner-perspektive>) oder dem Leipziger iLCM (<http://ilcm.informatik>.

[uni-leipzig.de/ilcm/ilcm/](http://uni-leipzig.de/ilcm/ilcm/)) (vgl. auch Anhang A8) erfasst und dargestellt werden können. Die Veränderungen von Kookkurrenzen eines Wortes über einen längeren Zeitraum geben dabei einen Hinweis auf mögliche Bedeutungsveränderungen.

Warum aber verändern sich die Verwendungskontexte eines Wortes? Meist lassen sich Phasen intensiver Kontextveränderungen identifizieren, in denen sich Hinweise auf mögliche Ursachen dafür finden. Beispiele für solche Phasen sind:

- Kontroverse Diskussionen, in denen die Befürworter und Befürworterinnen bzw. die Gegner und Gegnerinnen einer Position zentrale Wörter in völlig unterschiedlichen Kontexten verwenden;
- Punktuelle Ereignisse wie Naturkatastrophen, historische Ereignisse oder technologische Veränderungen, in denen nicht nur neue Wörter eingeführt, sondern zugleich auch alte Wörter in neuen Kontexten verwendet werden;
- Zyklische Ereignisse wie z. B. „Olympia“, bei denen sich in einem von dem Ereignis bestimmten Zeittakt der Ort des Ereignisses und die Namen der handelnden oder betroffenen Akteure bzw. Akteurinnen ändern.

Das Maß der **Kontextvolatilität** erlaubt es, die Kontextveränderungsraten von Wörtern über einen vorgegebenen Zeitraum zu quantifizieren und ergänzt damit das Verfahren des Kookkurrenzvergleichs. Durch eine normalisierte Quantifizierung der Änderungsrate können auch Wörter verschiedener **Häufigkeitsklassen** miteinander verglichen werden. Im Vergleich zur reinen Häufigkeitsanalyse können so auch niederfrequente Wörter in die Analyse mit einbezogen werden, beispielsweise für die Erkennung **schwacher Signale**, welche sich oftmals durch frühzeitige Veränderungen im Verwendungskontext von niederfrequenten Wörtern anzeigen (vgl. Abschn. 7.4). Neben den hochvolatilen Wörtern können im Umkehrschluss aber auch solche Wörter identifiziert werden, deren Verwendungskontexte sich über einen längeren Zeitraum wenig ändern, beispielsweise, weil sie Teil einer formelhaften Sprache im technischen oder rechtlichen Kontext sind, oder weil sie einen politischen oder gesellschaftlichen Konsens darstellen. So wurde in der New York Times in den 1990er-Jahren (neben vielen weiteren) das Wort „climate change“ in nahezu unveränderlichen Kontexten verwendet (vgl. Heyer et al. 2011). Obwohl der Klimawandel diskutiert wurde, finden sich offenbar in den Kontexten, in denen der Klimawandel zu Anfang der 1990er-Jahre in den USA diskutiert worden ist, kaum neue oder kontroverse Aspekte. Dies änderte sich schlagartig, als Al Gore an der Seite Bill Clintons das Thema in der amerikanischen Politik verwurzelte und nach 2000 seine Stellung und Popularität zur Förderung des Umweltthemas in den USA nutzte.

Für die Bestimmung der Kontextvolatilität werden die Erwartungswerte der Kontexte eines Wortes unter Verwendung von Kookkurrenzen anhand einer vorgegebenen Menge an Referenzzeitpunkten berechnet (vgl. Abschn. 5.9.3.2). Der so berechnete Vektor an Kontexterwartungswerten wird anschließend mit den tatsächlich im Testzeitpunkt vorliegenden Kontexten über verschiedene Distanzmaße wie dem **Kosinus-Maß** verglichen. In einem **Sliding-Window**-Ansatz wird dieser Algorithmus für alle Worte und für alle Zeitpunkte durchgeführt. Damit ergibt sich am Ende für jedes Wort des Vokabulars eine Verlaufskurve seiner Kontextänderungsraten in den vorliegenden Zeitpunkten (vgl. Abschn. 5.8).

Neben dieser diachronen Anwendung des Verfahrens ist es auch möglich, die Kontextvolatilität in einem synchronen Setting anzuwenden. Hierbei könnten Textmengen bezüglich verschiedener Metadaten, wie beispielsweise Autorenschaft, Rubrik, Quelle o.Ä., aufgeteilt werden. Für die sich damit ergebenen Korpora können dann wieder Kontextvektoren berechnet und anschließend mit dem beschriebenen Verfahren verglichen werden (Kahmann 2021).

## 7.6.2 Beispielanwendung

Im Folgenden stellen wir eine Studie vor, in der das Maß der Kontextvolatilität eingesetzt worden ist, um am Beispiel des Wortes *sustainability* Kontextverschiebungen dieses Wortes aufzudecken und weiter zu untersuchen. Die Datengrundlage für diese Analyse bildeten Nachrichtentexte der englischen Tageszeitung *The Guardian*. Wie in Abschn. 5.9.3 beschrieben, wurden hierfür in einem Sliding-Window-Ansatz Referenz'erwartungswerte für die Kontexte von Schlüsselwörtern rund um die Nachhaltigkeitsdiskussion berechnet. Diese anhand der Referenzzeitpunkte berechneten Erwartungswerte wurden dann mit den tatsächlich im jeweiligen Betrachtungszeitpunkt vorliegenden Kontexten verglichen. Damit konnten sowohl Zeitpunkte mit hoher bzw. niedriger Kontextänderungsrate bestimmt als auch spezifische, womöglich ursächliche Kontextänderungen identifiziert und untersucht werden.

Die Ausgangshypothese der hier vorgestellten Analyse wird durch die Behauptung des Vorhandenseins einer deutlichen Dynamik in der Verwendung des Wortes *Nachhaltigkeit* (bzw. *sustainability*) beschrieben. Insbesondere sollte untersucht werden, ob sich eine solche Dynamik der Verwendung mittels Zeitungstexten belegen lässt. Weiterhin war es das Ziel zu identifizieren, ob – und wenn ja, wie – gewisse Schlüssereignisse wie beispielsweise Klimakonferenzen, Einfluss auf die Verwendung des Wortes „Nachhaltigkeit“ genommen haben.

Die Datenquelle für die durchgeführte Analyse bildete ein Korpus, das aus Nachrichtenartikeln der britischen Tageszeitung *The Guardian* aufgebaut wurde. Insgesamt umfasst dieses Korpus mehr als 1,7 Mio. Artikel, welche über die *Guardian API* (<https://open-platform.theguardian.com/documentation/>) abgerufen wurden. Für die Analyse wurde aus dieser Menge von Texten eine Teilmenge von Artikeln gebildet. Diese umfassen alle

Guardian-Berichte, in denen der Wortstamm *sustainab*\* verwendet wird. Durch diese Einschränkung reduzierte sich die Menge der für die Analyse einschlägigen Artikel auf ca. 52.000.

Zur Untersuchung der Dynamik der Verwendungskontexte im Zusammenhang mit der Diskussion über Nachhaltigkeit wurden die Kontextvolatilitäts-Werte aller Wörter mit dem oben beschriebenen referenzbasierten Berechnungsansatz mit einer Zeitfenstergröße von einem Jahr in dem Teilkorpus berechnet. Zur Erstellung der Kontext-Erwartungsvektoren wurden die Dokumente der jeweils drei vorangegangenen Jahre herangezogen. Verschiedene Standard-Vorverarbeitungsschritte wie Stopwortentfernung und Pruning wurden angewendet.

In Abb. 7.13 ist sowohl der Frequenzverlauf (orange) als auch der Volatilitätsverlauf (blau) des Wortes *sustainable* im Zeitraum zwischen 2002 und 2020 dargestellt. In den frühen 2000ern liegt eine vergleichsweise geringe Verwendungsfrequenz des Wortes vor. Gleichzeitig lässt sich eine hohe Kontextvolatilität beobachten. Dies könnte darauf hindeuten, dass in diesem Zeitraum der Verwendungskontext des Wortes noch nicht durch einige wenige, spezielle Themenfelder dominiert wird. Tatsächlich lassen sich anhand der Top-50 Kookkurrenzen in 2002 verschiedene Kontexte beobachten, beispielsweise die Worte *environment, energy, government, business, earth, farming, food, economic, industry, transport, tourism* und *poverty*.

In den darauffolgenden Jahren steigt die Frequenz des Wortes deutlich an. Hiermit geht jedoch eine relativ konstante, geringe Volatilität einher. Ursächlich hierfür scheint eine starke Fokussierung des Wortes *sustainable* auf die Bereiche von Wirtschaft und Geschäftswesen (möglicherweise ausgelöst durch die europäische Finanzkrise). Erst ab 2015 lässt sich durch ein lokales Maximum der Volatilitätskurve eine Änderung dieser Situation beobachten. Vermehrt tauchen nun auch Worte im Zusammenhang mit Nachhaltigkeit auf, welche mit Klimaschutz, Nahrungsproduktion und nationalen Strategien einzelner Länder zur Umsetzung von Nachhaltigkeitszielen assoziiert sind. Worte, die in

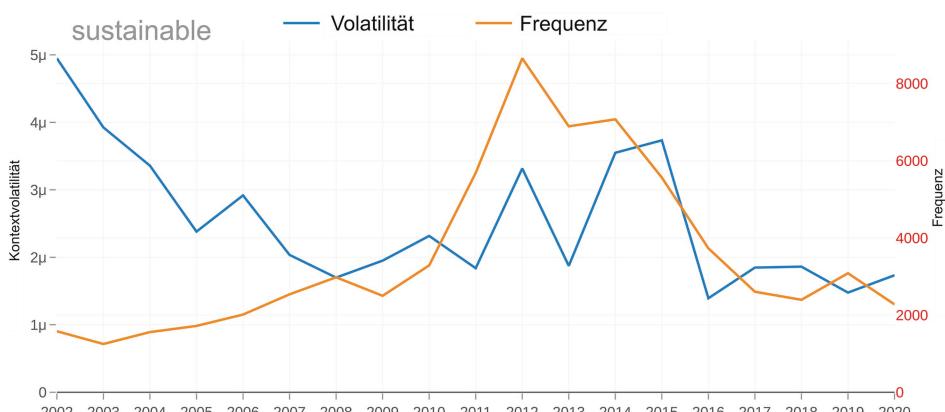
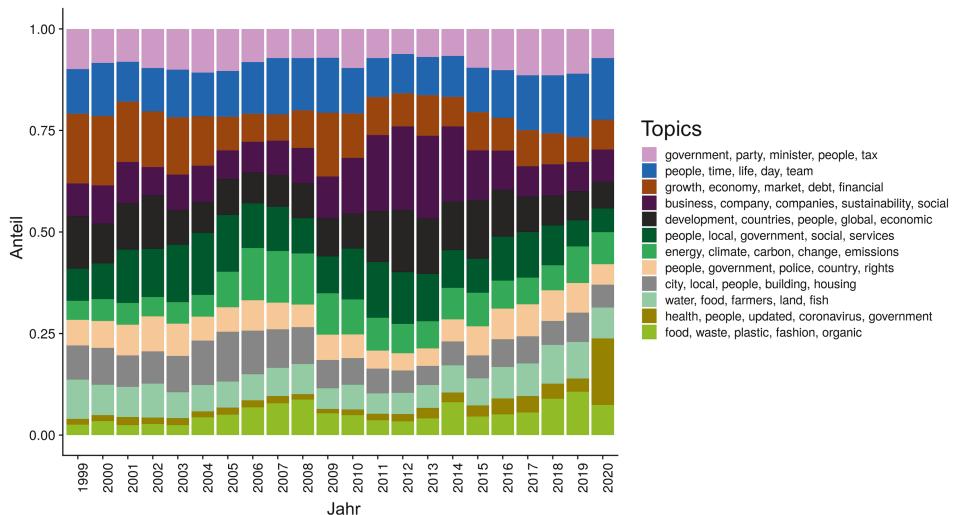


Abb. 7.13 Frequenz- und Volatilitätsverlauf von „sustainable“ 2002–2020



**Abb. 7.14** Topic-Modell *sustainability* 1998–2020

2015 deutlich signifikanter gemeinsam mit *sustainable* auftreten als in 2012, sind beispielsweise: *sdgs* (Sustainable Development Goals), *energy*, *food*, *climate* und *poverty*. Worte, die an Bedeutung verloren haben, sind hingegen: *business*, *economic*, *economy*, *growth* und *innovation*.

Dieser Trend der weniger starken Fokussierung auf Wirtschaftskontexte bestätigt sich auch in der nachfolgenden **Topic-Modell**-Anwendung (siehe Abb. 7.14; weniger starke Ausprägung der violetten Bereiche ab 2015).

Dieses Muster scheint in den darauffolgenden Jahren jedoch konstant zu bleiben, was wiederum in einer geringen Volatilität für die Jahre 2016 bis 2020 resultiert. Möglicherweise haben Schlüsselereignisse, wie die Pariser Klimakonferenz im Jahre 2015 oder die Finanzkrise 2009, einen ursächlichen Einfluss auf die gemessenen Kontextänderungen, was durch weitere Untersuchungen zu validieren ist.

Wie an der exemplarischen Anwendung deutlich wird, ermöglicht das Maß der Kontextvolatilität zum einen den Nachweis einer vorhandenen Dynamik in den Verwendungskontexten des Wortes *sustainable*. Zum anderen liefert sie Hinweise darauf, wie diese Dynamik im Sinne einer vertieften explorativen Analyse diachroner Daten durch die Identifikation von möglicherweise ursächlichen Ereignissen und Themen für die beobachteten Kontextveränderungen näher untersucht werden kann (Kahmann 2021).

## Literatur

- Ahmad, K. (ed.): Affective computing and sentiment analysis. emotion, metaphor and terminology. Text, speech and language technology, vol. 45. Springer, Dordrecht (2011). <https://doi.org/10.1007/978-94-007-1757-2>

- Allan, J. (Hrsg.): Topic detection and tracking: event-based information organization. Kluwer, Norwell (2002)
- Amend, C.: Was guckst du? DIE ZEIT11/2005. [http://www.zeit.de/2005/11/Titel\\_2fUnterschicht\\_11](http://www.zeit.de/2005/11/Titel_2fUnterschicht_11) (2005). Zugegriffen: 22. Juli 2021
- Ananiadou, S.: A methodology for automatic term recognition. In: Coling '94, Proceedings of the 15th conference on computational linguistics, Vol. 2, Kyoto, Japan, S. 1034–1038 (1994)
- Ansoff, H.I.: Managing strategic surprise by response to weak signals. *Calif. Manage. Rev.* **18**(2), 21–33 (1975). <https://doi.org/10.2307/41164635>
- Barnes, J., Øvrelid, L., Velldal, E.: Sentiment analysis is not solved! Assessing and probing sentiment classification. In: Linzen, T., Crupała, G., Belinkov, Y., Hupkes, D. (eds.) Proceedings of the 2019 ACL workshop BlackboxNLP: analyzing and interpreting neural networks for NLP, Florence, Italy, S. 12–23. Association for Computational Linguistics (ACL), Stroudsburg, PA, USA (2019). <https://doi.org/10.18653/v1/W19-4802>
- Bea, F.X., Haas, J.: Strategisches management. 10th edn. Unternehmensführung. UVK, München (2019)
- Benikova, D., Fahrer, U., Gabriel, A., Kaufmann, M., Yimam, S.M., von Landesberger, T., Biemann, C.: Network of the day: aggregating and visualizing entity networks from online sources. In: KONVENS 2014 Workshop proceedings: NLP4CMC, Hildesheim, Germany, S. 48–52. <https://www.inf.uni-hamburg.de/en/inst/ab/ltpublications/2014-benikovaetall-konvens-nod.pdf> (2014). Zugegriffen: 22. Juli 2021
- Biemann, C., Quasthoff, U., Heyer, G., Holz, F.: ASV toolbox: a modular collection of language exploration tools. In: Calzolari, N., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Tapia, D. (eds.) Proceedings of the sixth international conference on language resources and evaluation (LREC'08), Marrakech, Morocco, S. 1760–1767. European Language Resources Association (ELRA). [http://www.lrec-conf.org/proceedings/lrec2008/pdf/447\\_paper.pdf](http://www.lrec-conf.org/proceedings/lrec2008/pdf/447_paper.pdf) (2008). Zugegriffen: 22. Jan. 2022
- Dunning, T.: Accurate methods for the statistics of surprise and coincidence. *Comput. Linguist.* **19**(1), 61–74 (1993)
- Eiken, U.C., Liseth, A.T., Witschel, H.F., Richter, M., Biemann, C.: Ord i dag: Mining Norwegian daily newswire. In: Salakoski, T. (ed.) Advances in natural language processing. 5th International Conference on Natural Language Processing, FinTAL 2006, Turku, Finland, S. 512–523. Springer, Berlin, Heidelberg (2006). [https://doi.org/10.1007/11816508\\_51](https://doi.org/10.1007/11816508_51)
- Esuli, A., Sebastiani, F.: SENTIWORDNET: A publicly available lexical resource for opinion mining. In: Calzolari, N., Choukri, K., Gangemi, A., Maegaard, B., Mariani, J., Odijk, J., Tapia, D. (eds.) Proceedings of the fifth international conference on language resources and evaluation LREC '06, Genoa, Italy, S. 417–422. ELRA. [http://www.lrec-conf.org/proceedings/lrec2006/pdf/384\\_pdf.pdf](http://www.lrec-conf.org/proceedings/lrec2006/pdf/384_pdf.pdf) (2006). Zugegriffen: 02. Aug. 2021
- Germeval Task 2017– aspect based sentiment detection. <https://sites.google.com/view/germeval2017-absa/home> (2021). Zugegriffen: 02. Aug. 2021
- Go, A., Bhayani, R., Huang, L.: Twitter sentiment classification using distant supervision. CS224N project report, Stanford University. <https://www-nlp.stanford.edu/courses/cs224n/2009/fp/3.pdf> (2009). Zugegriffen: 21. Juli 2021
- Gormley, C., Tong, Z.: Elasticsearch: the definitive guide. A distributed real-time search and analytics engine. O'Reilly, Sebastopol (2015)
- Gottwald, S., Richter M., Heyer G., Scheuermann G.: Tapping huge temporally indexed textual resources with WCTAnalyze. In: Calzolari, N., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Tapia, D. (eds.) Proceedings of the sixth international conference on language resources and evaluation (LREC'08), Marrakech, Morocco. European Language Resources

- Association (ELRA). [http://www.lrec-conf.org/proceedings/lrec2008/pdf/117\\_paper.pdf](http://www.lrec-conf.org/proceedings/lrec2008/pdf/117_paper.pdf) (2008). Zugriffen: 22. Jan. 2022
- Heid, U.: A linguistic bootstrapping approach to the extraction of term candidates from German text. *Terminology* **5**(2), 161–181 (1998)
- Heyer, G., Keim, D., Teresniak, S., Oelke, D.: Interaktive explorative Suche in großen Dokumentbeständen. *Datenbank-Spektrum* **11**(3), 194–206 (2011). <https://doi.org/10.1007/s13222-011-0072-4>
- Horx, M.: Die acht Sphären der Zukunft: ein Wegweiser in die Kultur des 21. Jahrhunderts. Signum Verlag, Seedorf (2000)
- Kahmann, C.: Design und Implementierung eines Software-Ökosystems für textbasierte Inhaltsanalysen in den Sozialwissenschaften mit Schwerpunkt auf der Detektion schwacher Signale. Dissertation, Fakultät für Informatik, Universität Leipzig (2021)
- Kennedy, A., Inkpen, D.: Sentiment classification of movie reviews using contextual valence shifters. *Comput. Intell.* **22**(2), 110–125 (2006). <https://doi.org/10.1111/j.1467-8640.2006.00277.x>
- Kreibich, R.: Zukunftsorschung. In: Tietz, B., Köhler, R., Zentres, J. (ed.) *Handwörterbuch des Marketing. Enzyklopädie der Betriebswirtschaftslehre*, Bd. 4, 2nd edn., S. 2814–2834. Schäffer-Poeschel, Stuttgart (1995)
- Kumar, A., Kohail, S., Kumar, A., Ekbal, A., Biemann, C.: IIT-TUDA at SemEval-2016 Task 5: Beyond sentiment lexicon: combining domain dependency and distributional semantics features for aspect based sentiment analysis. In: Bethard,S., Carpuat, M., Cer, D., Nakov, P., Zesch, T. (eds.) *Proceedings of the 10th international workshop on semantic evaluation*, San Diego, CA, USA, S. 1129–1135. Association for Computational Linguistics, Stroudsburg (2016). <https://doi.org/10.18653/v1/S16-1174>
- Liebl, F.: *Strategische Frühaufklärung: Trends – Issues – Stakeholders*. Berlin, Boston: Oldenbourg Wissenschaftsverlag (2018)
- Neviarouskaya, A., Prendinger, H., Ishizuka, M.: Compositionality principle in recognition of fine-grained emotions from text. In: *Proceedings of the 3rd International Conference on Weblogs and Social Media (ICWSM)*, San Jose, CA, USA, S. 278–281. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.718.6376&rep=rep1&type=pdf> (2009). Zugriffen: 22. Juli 2021
- Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? Sentiment classification using machine learning techniques. In: *Proceedings of the ACL-02 conference on empirical methods in natural language processing–Volume 10*, Philadelphia, PA, USA, S. 79–86. Association for Computational Linguistics. Morristown, NJ, USA (2002). <https://doi.org/10.3115/1118693.1118704>
- Quasthoff, U., Richter, M., Wolff, C.: Wörter des Tages -Tagesaktuelle wissensbasierte Analyse und Visualisierung von Zeitungen und Newsdiensten. In: Hammwöhner, R. (ed.) *Information und Mobilität. Optimierung und Vermeidung von Mobilität durch Information*, proceedings des 8. Internationalen Symposiums für Informationswissenschaft (ISI 2002). Schriften zur Informationswissenschaft, Bd. 40, S. 369–372. UVK, Konstanz (2002)
- Quasthoff, U., Richter, M.: Projekt Deutscher Wortschatz. *Babylonia* **3**, 33–35 (2005)
- Quasthoff, U.: Deutsches Neologismenwörterbuch. De Gruyter, Berlin (2007)
- Remus, R., Quasthoff, U., Heyer, G.: SentiWS - A publicly available German-language resource for sentiment analysis. In: Calzolari, N. et al. (eds.) *Proceedings of the 7th International Language Resources and Evaluation (LREC'10)*, Valletta, Malta, S. 1168–1171 (2010)
- Remus, S., Kaufmann, M., Ballweg, K., von Landesberger, T., Biemann,C.: Storyfinder: personalized knowledge base construction and management by browsing the web. In: Lim, E.-P. et al. (eds.) *Proceedings of the 26th ACM Conference on Information and Knowledge*

- Management, Singapore, Singapore, S. 2519–2522. ACM Association for Computing Machinery, New York (2017). <https://doi.org/10.1145/3132847.3133186>
- Richter, M.: Nutzen und Benutzen von Text Mining für die Medienanalyse. Dissertation, Fakultät für Mathematik und Informatik, Universität Leipzig (2010)
- Rohrdantz, C., Koch, S., Jochim, C., Heyer, G., Scheuermann, G., Ertl, T., Schütze, H., Keim, D.: Visuelle Textanalyse. Informatik Spektrum **33**(6), 601–611 (2010). <https://doi.org/10.1007/s00287-010-0483-x>
- Sächsische Staatsregierung (ed.): Verordnung des Sächsischen Staatsministeriums für Soziales und Gesellschaftlichen Zusammenhalt zum Schutz vor dem Coronavirus SARS-CoV-2 und COVID-19 (Sächsische Corona-Schutz-Verordnung – SächsCoronaSchVO) Vom 10. Juni 2021, <https://www.coronavirus.sachsen.de/download/SMS-Saechsische-Corona-Schutz-Verordnung-2021-06-10.pdf> (2021). Zugegriffen: 08. Juli 2021
- Schwäbisches Tagblatt GmbH (ed.): Bald schneller mit der Bahn von Deutschlad nach Paris. [http://www.neckar-chronik.de/Home/nachrichten/ueberregional/baden-wuerttemberg/\\_artikel,-Bald-schneller-mit-der-Bahn-von-Deutschland-nach-Paris/\\_arid,319757.html](http://www.neckar-chronik.de/Home/nachrichten/ueberregional/baden-wuerttemberg/_artikel,-Bald-schneller-mit-der-Bahn-von-Deutschland-nach-Paris/_arid,319757.html) (30.09.2015). Zugegriffen: 05. Okt. 2016
- Strapparava, C., Mihalcea, R.: SemEval-2007 task 14: Affective text. In: Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval), Prague, Czech Republic, S. 70–74. Association for Computational Linguistics, Morristown <https://aclanthology.org/S07-1013.pdf> (2007). Zugegriffen: 22. Juli 2021
- Strapparava, C., Valitutti, A.: WordNet Affect: an Affective Extension of WordNet. In: Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04), Lisbon, Portugal, S. 1083–1086. European Language Resources Association (ELRA). <http://www.lrec-conf.org/proceedings/lrec2004/pdf/369.pdf> (2004). Zugegriffen: 22. Jan. 2022
- Strötgen, J., Gertz, M.: Multilingual and cross-domain temporal tagging. Lang. Resour. Eval. **47**(2), 269–298 (2013)
- Turney, P.: Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In: Isabelle, P.(ed.) Proceedings of the 40th annual meeting of the Association for Computational Linguistics (ACL), Philadelphia, PA, USA, S. 417–424. Association for Computational Linguistics, Morristown, NJ, USA (2002). <https://doi.org/10.3115/1073083.1073153>
- Turney, P.D., Littman, M.L.: Measuring praise and criticism: inference of semantic orientation from association. Association for Computing Machinery (ACM) Transactions on Information Systems (TOIS) **21**(4), 315–346 (2003). doi: <https://doi.org/10.1145/944012.944013>
- Walde, P.: Digital Intelligence - Möglichkeiten und Umsetzung einer informatikgestützten Früh- aufklärung. Dissertation, Fakultät für Mathematik und Informatik, Universität Leipzig (2010)
- Wang, L., Xia, R.: Sentiment lexicon construction with representation learning based on hierarchical sentiment supervision. In: Palmer, M., Hwa, R., Riedel, S. (eds.) Proceedings of the 2017 conference on empirical methods in natural language processing, Copenhagen, Denmark, S. 502–510. Association for Computational Linguistics, Stroudsburg (2017). <https://doi.org/10.18653/v1/D17-1052>
- Wiebe, J., Wilson, T., Bruce, R., Bell, M., Martin, M.: Learning subjective language. Comput. Linguist. **30**(3), 277–308 (2004). <https://doi.org/10.1162/0891201041850885>
- Wiedemann G., Yimam S.M., Biemann C.: New/s/leak 2.0 – multilingual information extraction and visualization for investigative journalism. In: Staab, S., Kol’cova, O., Ignatov, D.I. (eds.) Proceedings of the 10th International Conference on Social Informatics (SocInfo 2018), St. Petersburg. Lecture Notes in Computer Science, vol. 11186, S. 313–322. Springer, Cham (2018). doi: [https://doi.org/10.1007/978-3-030-01159-8\\_30](https://doi.org/10.1007/978-3-030-01159-8_30)

- Witschel, H.F.: Terminologie-Extraktion. Möglichkeiten der Kombination statistischer und musterbasierter Verfahren. Content and communication, Bd. 1. Ergon, Würzburg (2004)
- Wojatzki, M., Ruppert, E., Holschneider, S., Zesch, T., Biemann, C.: GermEval 2017: Shared task on aspect-based sentiment in social media customer feedback. In: Wojatzki, M., Ruppert, E., Zesch, T., Biemann, C. (eds.) Proceedings of the GermEval 2017 – shared task on aspect-based sentiment in social media customer feedback, Berlin, S. 1–12. <https://www.inf.uni-hamburg.de/en/inst/ablt/publications/2017-wojatzkietal-germeval2017-proceedings.pdf> (2017). Zugriffen: 02. Juli 2021
- Yimam, S.M., Ulrich, H., von Landesberger, T., Rosenbach, M., Regneri, M., Panchenko, A., Lehmann, F., Fahrer, U., Biemann, C., Ballweg, K.: New/s/leak – information extraction and visualization for an investigative data journalists. In: Proceedings of ACL 2016 Demo Session, Berlin, Germany, S. 163–168. Association for Computational Linguistics, Stroudsburg (2016). <https://doi.org/10.18653/v1/S.16-4028>

---

## Glossar

**Vorwort zum Glossar:** Das Glossar erklärt wichtige Begriffe des Text Mining, die nicht als allgemein bekannt vorausgesetzt werden. Dies können sowohl wiederkehrende Begriffe aus dem Text wie auch weiterführende Erklärungen für im Text nicht näher erläuterte Begriffe sein. In der alphabetischen Sortierung sind bei Wortgruppen ggf. enthaltene Adjektive nachgestellt wie bei *Lernverfahren*, *überwachte*. Nach dem Stichwort folgt zusätzlich die englische Übersetzung. Falls für das Verständnis nötig, wird im Definitionstext mit einem Verweispfeil ► auf weitere Begriffe im Glossar verwiesen.

**Accuracy (accuracy) auch Treffergenauigkeit** Accuracy ist ein ► **Evaluationsmaß** zur Bewertung von Klassifikationssystemen. Sie ist definiert als die Anzahl aller korrekten Vorhersagen dividiert durch die Gesamtanzahl an Vorhersagen. Dementsprechend ist die maximal erreichbare Accuracy 1 und wird erreicht, wenn jede Vorhersage korrekt ist, wohingegen eine Accuracy von 0 erreicht wird, wenn keine der Vorhersagen korrekt ist. Sie eignet sich nur für die Bewertung von einigermaßen balancierten Datensätzen, in denen also alle Klassen etwa gleich häufig vorkommen.

**Affektwörterbuch (affect dictionary)** Lexikon von typischen ► **Sentimentausdrücken**.

**Ähnlichkeitsmaß (similarity measure)** Ein Ähnlichkeitsmaß ist im ► **Information Retrieval** ein Maß für die Ähnlichkeit von Wörtern oder Texten (als einer Kollektion von Wörtern). Einem Paar von Textobjekten wird eine reelle Zahl S ( $0 <= S <= 1$ ) zugeordnet, die umso größer ist, je ähnlicher die Textobjekte sind. In der Praxis häufig verwendete Maße sind das ► **Kosinusmaß**, der Jaccard-Koeffizient und der Dice-Koeffizient.

**Allomorph (allomorph)** Verschiedene Varianten eines ► **Morphems**.

**Allgemeinsprache (general language)** Unter der Allgemeinsprache versteht man die Sprache in ihrer allgemeinen und alltäglichen Verwendung. Antonym: ► **Fachsprache**.

**Alphabet (alphabet)** Ein Alphabet ist ein endlicher, geordneter Zeichenvorrat für die Bildung von Wörtern.

**Analysekörper (corpus for analysis)** Das Analysekörper ist der zu untersuchende Text bzw. die zu untersuchende Textmenge.

**Annotation/Annotatoren (annotations/annotators)** Annotationen sind Klassifizierungen nach einem vorgegebenen Kategorienschema, welche sich, im Kontext der Computerlinguistik, auf Sprache beziehen, etwa indem sie Dokumente eine Kategorie zuweisen oder Worte mit ihrer Wortart versehen. Annotatoren sind dabei Menschen oder automatische Komponenten, welche Annotationen erstellen.

**Ansatz, korpusbasierter (corpus-based approach)** Vorgehensweise in der Lexikographie, bei der ein Textkorpus die Grundlage bei der Auswahl von Stichwörtern und Verwendungsbeispielen bildet.

**Antonyme (antonyms)** Zwei Wörter, die nur in Bezug auf einen vorher definierten Begriff einen Gegensatz bilden. Syn.: Relative Gegensätze.

**Anwender (user)** Siehe ▶ Benutzer.

**Anwendungssoftware (application software)** Software, die Aufgaben des Anwenders mithilfe eines Computersystems löst. Setzt in der Regel auf der Systemsoftware der verwendeten Hardware auf bzw. benutzt sie zur Erfüllung der eigenen Aufgabe.

**Äquivalenzklasse (equivalence class)** Menge von Elementen, die im Hinblick auf eine (reflexive, symmetrische und transitive) Äquivalenzrelation äquivalent sind.

**ASCII (ASCII; American Standard Code of Information Interchange)** Genormter 7-Bit-Zeichensatz (128 Positionen) zur Darstellung von Ziffern, Buchstaben, Sonderzeichen und Steuerzeichen. Siehe auch ▶ Latin und ▶ Unicode.

**Aspekt-basierte Sentimentanalyse (aspect-based sentiment analysis)** Ansatz der ▶ Sentimentanalyse, bei der unterschieden wird zwischen dem sog. document level, den Aspektkategorien, den sentiment targets und den auf den Text sowie die verschiedenen Aspektkategorien bezogenen ▶ Polaritäten.

**Asset (Asset)** Ein Asset (Wert) ist ein sinnhaltiges Datenobjekt als Wirtschaftsgut. Syn.: Wert.

**Attention-Mechanismus (attention mechanism)** In ▶ neuronalen Netzarchitekturen Modellierung des Informationsflusses zwischen den Wörtern eines Satzes oder eines kurzen Textes in Abhängigkeit von den Eingangsrepräsentationen, um Ausgangsrepräsentationen zu berechnen: Das Netzwerk lernt, wie stark die anderen Positionen in der Sequenz zu beachten sind, um die aktuelle Position zu repräsentieren. Elementarer Bestandteil von ▶ Transformer-Architekturen.

**Attribute, extrinsische (extrinsic attributes)** Eigenschaften von Wörtern, welche man den typischen Kontexten der Wörter entnehmen kann, beispielsweise das grammatische Geschlecht von Nomen durch benachbarte Artikel oder typische Eigenschaften durch benachbarte Adjektive.

**Attribute, intrinsische (intrinsic attributes)** Eigenschaften von Wörtern, welche man der inneren Struktur der Wörter entnehmen kann, beispielsweise durch Kompositazelerlegung oder morphologische Analyse.

**Automat, endlicher (finite state automaton)** Modell für einen Automaten, der Daten Zeichen für Zeichen einliest, das eingelesene Zeichen sofort verarbeitet und eine Ausgabe erzeugt. Ein endlicher Automat besitzt eine endliche Menge von Zuständen und keinen zusätzlichen Speicher.

**Automatentheorie (automata theory)** Teilgebiet der theoretischen Informatik, das Automaten als formale Systeme zur Beschreibung von Sprachen und Grammatiken zum Gegenstand hat.

**Backpropagation (back propagation)** Ein Algorithmus zur effizienten Berechnung der Gradienten innerhalb eines neuronalen Netzwerks gegeben der Eingabe und bezüglich der Abweichung von der gewünschten Ausgabe (beschrieben durch eine ▶ Zielfunktion), also die nötigen Veränderungen um die gewünschte Ausgabe zu erreichen. Basierend auf den berechneten Gradienten werden die Gewichte des Netzwerks angepasst.

**Bag-of-Words Modell (bag-of-words model)** Modell zur Repräsentation eines Textes anhand der in ihm vorkommenden Wörter und deren Anzahl, unabhängig von der exakten Position ihres Vorkommens im Text. Dabei gehen notwendigerweise Informationen verloren, für viele Aufgaben ist diese Repräsentation jedoch bereits ausreichend.

**Baseline (baseline)** auch unterer Leistungswert: Die Baseline ist eine untere Schranke der Leistung eines Algorithmus. Wenn ein komplexer Algorithmus entwickelt wird, wird oft ein einfacher Algorithmus (Baseline-Verfahren) dagegen gestellt. Der komplexe Algorithmus muss dann bessere Ergebnisse erzielen als der einfache Algorithmus.

**Basismorphem (base morpheme)** Morpheme, die Sachverhalte der außersprachlichen Welt bezeichnen. Zu ihnen gehören Nomina wie *Kind* und *Mantel*, aber auch Verbstämme wie *seh.* Syn.: Stamm.

**Bayessche Statistik (Bayesian statistics)** Die Wahrscheinlichkeit eines Ereignisses wird in der Bayesschen Statistik als Erwartungswert interpretiert, der sich aus einer Bewertung bisheriger Beobachtungen ableitet. Dabei werden das Vorwissen und sog. A-Priori-Annahmen explizit ausdrückt.

**Begriff (notion)** Ein Begriff ist die Bedeutung eines Ausdruckes. Der Begriffsumfang wird als Extension, der Begriffsinhalt als Intension bezeichnet.

**Benutzer und Benutzerinnen (user)** Personen, die ein Computersystem unmittelbar einsetzen und selbst bedienen. Syn.: Anwender und Anwenderinnen.

**Big Data (big data)** Technologien und Algorithmen für die Speicherung und Auswertung digitaler Massendaten. Zu den Verarbeitungsdimensionen zählen nach dem sog. V-Modell das Volumen der Datenmenge, die Vielzahl der abzubildenden Inhalte und die Verarbeitungsgeschwindigkeit der Recherche- und Analyseschritte.

**Bi-Gramm (bi-gram)** Ein Bi-Gramm ist ein spezieller Typ von ▶ n-Grammen, das aus zwei aufeinander folgenden Wörtern oder Buchstaben besteht. Siehe ▶ n-Gramm und ▶ Tri-Gramm.

**BiLSTM (BiLSTM)** Bidirektionelle Variante der ▶ **Long Short-Term Memories**.

Dabei wird die Ausgabe zu jedem Schritt in der Sequenz durch eine Kombination aus zwei Sequenzmodellen generiert, wobei das eine Modell seine Schritte vom Start der Sequenz hin zum Ende macht und das andere aus der Gegenrichtung, also vom Ende aus, seine Eingaben erhält. So kann z. B. jedes Wort eines Satzes in Abhängigkeit sowohl seiner Vorgänger als auch seiner Nachfolger modelliert werden.

**BIO-Schema (BIO scheme)** Begin-Inside-Outside ist ein Kodierung für ▶ **Sequenzklassifikation**, dabei wird jeweils der Beginn einer ▶ **Spannenannotation** (also etwa der erste Token) mit dem „Begin“ (Beginn) Label versehen, jedes weitere Element der zu annotierenden Spanne erhält ein „Inside“ (Innen) Label. Alle Elemente, die von keiner Annotation abgedeckt sind, erhalten das „Outside“ (Außen) Label.

**Bootstrapping (bootstrapping)** Bootstrapping ist die allgemeine Bezeichnung für einen Prozess, bei dem unter Zuhilfenahme einer kleinen Anfangskonfiguration eine größere, umfassendere Konfiguration erzeugt wird. Im engeren Sinne ist es eine Bezeichnung für ein maschinelles Lernverfahren, bei dem neue Information generiert wird aus einer Startmenge an Information und einer Regelmenge, wie durch die schon bekannte Information neue Information gefunden werden kann.

**Buchstaben (letters)** Die in einem Alphabet verwendeten Zeichen.

**Chomsky-Grammatik (Chomsky grammar)** System von Ersetzungsregeln für die Erzeugung bzw. Analyse von formalen und natürlichen Sprachen. Abhängig davon, welche Einschränkungen bei der Ersetzung von Zeichenketten zu beachten sind, unterscheidet man Grammatiken für reguläre, kontextsensitive, kontextfreie und unbeschränkte Sprachen.

**Chunking (chunking)** Beim Chunking werden grammatisch zusammenhängende Phrasen als solche markiert. Im Gegensatz zu Dependenzrelationen bilden Chunks dabei keine hierarchischen, sondern stattdessen eine flache Struktur aus sich nicht überschneidenden ▶ **Spannenannotationen**.

**Chunks (chunks)** Chunks im Sinne des ▶ **Chunking** sind zusammenhängende grammatischen Phrasen, etwa Nominalphrasen wie „der schnelle Hund“ aber auch Verbal- und Präpositionalphrasen.

**Cluster-Analyse (cluster analysis, clustering)** Bei der Cluster-Analyse im Text Mining wird eine Menge sprachlicher Elemente (Texte, Sätze, Wörter) durch Gruppierung zusammengehöriger Elemente in homogene Teilmengen – die Cluster (Gruppen, Kategorien, Klassen) – unterteilt. Im Gegensatz zur Vorgehensweise bei der Klassifikation werden die Cluster aus der Struktur der zu analysierenden Datenmenge selbst abgeleitet.

**Clustering (Clustering, cluster analysis)** Siehe ▶ **Cluster-Analyse**.

**Codepages (code pages)** Tabellen zur Enkodierung von Zeichensätzen wobei verschiedene Codepages verwendet werden um Alphabete verschiedener Sprachen abzubilden, so bildet die ▶ **Latin** Codepage etwa westeuropäische Sprachen ab. Dementsprechend ist es für die korrekte Dekodierung eines (binär) kodierten Textes notwendig, die richtige Codepage zu verwenden.

**Cohens Kappa (Cohen's kappa)** Dies ist ein Maß für ▶ **Interrater-Reliabilität** zwischen zwei Annotierenden auf demselben Datensatz.

**Compiler (compiler)** Systemprogramm zur Überführung eines Computerprogramms in ausführbaren Maschinencode.

**Content (content)** Content (Inhaltsobjekt) enthält Informationen als sinnhafte Datenobjekte. Syn.: Inhaltsobjekt.

**Content-Management-System (content management system, CMS)** Softwaresysteme zur Erstellung, Verwaltung, Strukturierung und Publikation verschiedener Medien. Im organisationsinternen Kontext ist darunter oft eine Dokumentenverwaltungssoftware zu verstehen, wobei CMSe auch verwendet werden, um Webseiteninhalte zu verwalten.

**Continuous Bag-of-Words (CBOW) Modell (CBOW model)** Neuronales Netzwerk zum Erlernen von ▶ **Word Embeddings** in Form von ▶ **dichtbesetzten Vektoren**. Die Trainingsaufgabe des Netzwerkes ist das Vorhersagen eines Wortes anhand seines Kontextes (Wörter links und rechts des aktuellen Wortes) mittels eines ▶ **Sliding Window** Verfahrens. Die Reihenfolge der Wörter im Kontext ist dabei irrelevant.

**Convolutional Neural Networks** Neuronale Netzwerke, in denen Feed-Forward Operationen mit den gleichen Gewichten auf mehreren, lokal begrenzten, Teilen der Eingabe ausgeführt werden. Dabei müssen die Eingabedaten strukturiert, etwa eindimensional (z. B. Zeitreihe) oder zweidimensional (wie z. B. in einem Bild) vorliegen. Sie werden insbesondere in der Bildverarbeitung, aber auch in der Sprachverarbeitung verwendet.

**Crawler, auch Webcrawler (crawler/webcrawler)** Computerprogramm zum automatischen Herunterladen großer Mengen von Webseiten auf der Basis vorgegebener Kriterien. Ausgehend von einer Menge von Startseiten werden die in den Webseiten gefundenen Links jeweils weiterverfolgt.

**Crawlingstrategie (crawling strategy)** Reihenfolge, nach der die bekannten, aber noch nicht heruntergeladenen Webseiten durch einen Crawler aufgesucht werden. Diese Warteliste kann viele Millionen Seite umfassen und sollte in einer Reihenfolge abgearbeitet werden, dass die verschiedenen Bereiche des Web möglichst gleichmäßig besucht werden.

**Data Mining (data mining)** Unter dem Begriff des Data Mining werden Verfahren aus der Statistik und künstlichen Intelligenz zusammengefasst, mit denen sich in strukturierten Datenbeständen Muster und statistische Zusammenhänge berechnen lassen.

**Data Warehouse (data warehouse)** Kopie operativer Daten, speziell für Anfragen und Analysen strukturiert. Die Datenorganisation erfolgt in Hyperwürfeln.

**Daten (data)** Daten im Sinne der Informatik und Datenverarbeitung (EDV) sind (maschinen-) lesbare und bearbeitbare Repräsentationen von Information. Die Information wird dazu in Zeichenketten kodiert, deren Aufbau strengen Regeln folgt. Daten werden zu Information, wenn sie unter Bezug auf einen Interpretations-schlüssel interpretiert werden.

**DAWG (directed acyclic word graph)** Ein DAWG ist eine Datenstruktur zum Speichern von Wörtern. Das Funktionsprinzip ist ähnlich dem eines Trie, aber zusätzlich werden gleiche Wortenden zusammengeführt und damit komprimiert gespeichert.

**Deklination (declination)** Flexion (Beugung) von Nomina und Adjektiven. Verändert wird dabei die Anzahl (Numerus) und der Fall (Kasus), bei Adjektiven kann zusätzlich auch das grammatische Geschlecht (Genus) verändert werden.

**Dendrogramm (dendrogram)** Ein Dendrogramm dient der schematischen Darstellung einer ▶ Cluster-Analyse. In ihm werden die ermittelten Teilmengenbeziehungen dargestellt. Je höher im Baum der Verschmelzungspunkt zweier Mengen liegt, umso größer ist die Distanz zwischen ihnen im Vektorraum und umso kleiner ist die Ähnlichkeit zueinander.

**Dependenzen (syntactic dependencies)** Die zwischen den Wörtern eines Satzes bestehenden Abhängigkeiten im Sinne der Dependenzgrammatik. Dabei wird angenommen, dass jedes Wort in einem Satz ein, und nur ein übergeordnetes Wort hat, von dem es abhängt. Die Relation zwischen übergeordneten und untergeordneten Wort heißt Head-Modifier-Relation.

**Dependenz-Struktur (dependency structure)** Darstellung der Dependenzen zwischen den Wörtern eines Satzes in Form eines Baumes, wobei jeder Knoten im Baum einem terminalen Wort entspricht. Als Wurzel einer Dependenzbaum-Struktur wird immer das Verb (und nicht eine abstrakte Kategorie Satz) angesetzt. Ähnlich der Konstituentenstruktur-Syntax können dabei auch einzelne Knoten benannt werden, etwa mit den Grundkategorien N, V, A, P, Art und Mod.

**Derivation (derivation)** Morphologischer Prozess, durch den neue Wörter durch Anfügung (Affigierung) von Derivativen an Wortstämme entstehen.

**Derivationsaffix (derivational affix)** Siehe ▶ Derivative.

**Derivative (derivatives)** Gebundene Morpheme, die bei der Derivation verwendet werden. Die meisten Derivative haben eine charakteristische Bedeutung, die sie dem Stamm hinzufügen. Dabei ändern Derivative meist die Wortart des Stamms. Syn.: Derivationsaffixe.

**Development-Menge (development set)** auch: Validationsmenge (validation set): Bestandteil des Datensatzes aus Instanzen und Labels, bezüglich dessen ein Modell während seiner Entwicklung evaluiert wird. Dabei kann es vorkommen, dass das Modell implizit auf die Development-Menge und nicht generell auf Daten außerhalb der ▶ Trainingsmenge optimiert wird, weshalb es nötig ist einen finalen Test auf der ▶ Testmenge durchzuführen. Siehe auch Trainingsmenge und Testmenge.

**Differenzanalyse (differential analysis)** Verfahren zur Ermittlung von statistisch signifikanten Unterschieden in der Verwendung von Vokabularen, siehe auch Korpusvergleich.

**Dirichlet-Verteilung (Dirichlet distribution)** Multinomiale Wahrscheinlichkeitsverteilung. Die Steuerung erfolgt über einen ▶ Hyperparameter.

**Disambiguierung (disambiguation)** Disambiguierung beschreibt die Auflösung der Mehrdeutigkeit von Ausdrücken. Siehe auch: Wortbedeutungsdisambiguierung.

**Dispersionsmaße (degree of dispersion)** Häufig in der Stylometrie verwendete Verfahren, bei denen die Positionierung von Wörtern im Analyse- und Referenzkorpus miteinander verglichen werden.

**Distanzmaß (distance measure)** Angabe des relativen Abstands zwischen zwei Objekten. Typische Distanzmaße für reellwertige Vektoren sind z. B. die Euklidische Distanz oder die ▶ **Kosinusähnlichkeit**. Siehe ▶ **Tanimoto-Ähnlichkeit** für binäre Vektoren und ▶ **Levenshtein-Distanz (Editierdistanz)** für den Abstand zwischen Zeichenketten bzw. Wörtern.

**Distributionelle Hypothese (distributional hypothesis)** Annahme, dass Wörter, die in gleichen Kontexten auftreten, ähnliche Bedeutungen haben. Basis für die Operationalisierung der in der Tradition des Strukturalismus stehenden distributionelle Semantik, welche die Bedeutung von Wörtern nur anhand ihrer Kontexte charakterisiert.

**Distributionale semantische Modelle (distributional semantic models)** Modelle die Wörtern oder anderen ▶ **Instanzen** durch Darstellungen im Vektorraum repräsentieren, dabei werden Instanzen, basierend auf der ▶ **distributionellen Hypothese** durch ihren Kontext im ▶ **Trainingskorpus** beschrieben.

**Dokumentenähnlichkeit (document similarity)** Als Dokumentenähnlichkeit bezeichnet man die Gruppierung von Wörtern oder Texten nach inhaltlichen Kriterien in Bezug auf ein vorgegebenes Ähnlichkeitsmaß.

**Dokumentkorpus (corpus of documents)** Ein Korpus aus natürlichsprachlichen Texten mit oder ohne Annotationen und Metadaten, in dem gesamte Textdokumente abgelegt sind.

**Drift (drift)** Die Veränderung der Eigenschaften eines Datensatzes über die Zeit hinweg. So können sich in einem ▶ **Korpus** statistische Eigenschaften wie die Häufigkeit mancher Worte ändern, ebenso können sich etwa die Kontexte von Wörtern ändern, da sich ihre Semantik sich über die Zeit geändert hat.

**Dublin Core (Dublin Core)** De-facto Standard für Metadaten der Dublin Core Metadata Initiative (DCMI). Wesentliche Elemente des Standards sind Felder für einen Identifier, Technische Daten, Beschreibung des Inhalts, Festlegung beteiligter Personen und Rechte, Aspekte der Dokumentvernetzung und des Dokumentlebenszyklus. Alle Felder sind optional, können mehrfach auftauchen und in beliebiger Reihenfolge stehen.

**Early-Stopping-Methode (early stopping)** Eine Methode zur Vermeidung von Overfitting, dabei wird das Training eines Modells dann vorzeitig beendet, wenn, basierend auf Evaluation auf der ▶ **Development-Menge**, keine weitere Verbesserung des Modell mehr stattfindet. Oft wird dabei basierend auf einem „Patience“ (Geduld)- ▶ **Hyperparameter** nach entsprechend vielen Schritten ohne Verbesserung das Training beendet und der bis dahin beste Zustand des Modells gewählt.

**Eigenname (named entity)** Eigennamen sind referenzierende Bezeichner von Objekten verschiedener Kategorie (z. B. Personen, Firmen und Institutionen, Produkte und

Orte). Morphologisch sind Eigennamen nicht produktiv. Syntaktisch treten sie meist ohne Artikel auf und sind im Satz ersetzbar durch Proformen wie *er/sie/es* oder *dieser*.

**Eigennamenerkennung (named entity recognition, NER)** Verarbeitungsschritt, in dem Eigennamen wie Personennamen, Ortsnamen, Organisationsnamen und sonstige Namen im Text markiert werden. Wird meist mit Hilfe von ▶ **Sequenzklassifikation** durchgeführt, wobei Mehrwortnamen mithilfe des ▶ **BIO-Schemas** kodiert werden.

**Einleseprozess (data ingestion)** Der Prozess des Einlesens von Daten in ein System. Dabei gilt es, als Teil dieses Prozesses, Daten in ein einheitliches Format zu bringen, um weitere Verarbeitungsschritte zu vereinfachen. Im Text Mining kann dies das Überführen in ein reines Textformat bedeuten und kann auch das Einlesen von Metadaten beinhalten.

**Embedding (embedding)** Repräsentation von lexikalischen Einheiten wie Wörtern, Sätzen oder Texten, welche diese Einheiten in einen hochdimensionalen Vektorraum einbetten. Ein Embedding besteht aus einem reellwertigen Vektor, welcher als Punkt in diesem Raum interpretiert werden kann. Embeddings werden typischerweise durch Trainieren auf ▶ **Hintergrundkorpora** generiert und sind besonders als Eingabe für neuronale Netze geeignet.

**End-to-End-System (end-to-end system)** Systeme, in denen neuronale Modelle genutzt werden ohne dass Einzelschritte explizit, etwa mithilfe einer linguistischen Pipeline, modelliert werden. Dabei werden etwa Vorverarbeitungsschritte durch selbstständig gelernte, modellinterne, Repräsentationen ersetzt.

**Entity Linking (entity linking)** Der Prozess des Verbindens von ▶ **Eigennamen** mit externen Informationen bezüglich ihrer Identität. So wird z. B. die Erwähnung einer prominenten Person mit dem Eintrag zu dieser Person in einer Wissensbasis verbunden.

**Escape-Sequenz (escape sequence)** Eine Reihe von Symbolen die ausdrückt, dass ein oder mehrere darauf folgende Zeichen nicht als Daten im ursprünglichen ▶ **Alphabet** betrachtet werden sollen. So wird z. B. mit dem ▶ **regulären Ausdruck** „.“ nicht als die Suche nach einem Punkt sondern die nach einem beliebigen Zeichen aufgefasst, mittels eines Backslash als Escape-Sequenz kann mit „\\.“ nach einem Punkt gesucht werden.

**Evaluationsmaß (evaluation measure)** Maße, um die Qualität von Ergebnissen, etwa Klassifikationsergebnisse eines maschinellen Lernsystems, zu bewerten; dabei findet ein Vergleich mit ▶ **Gold-Daten** statt. Je nach Beschaffenheit der Daten bzw. der Aufgabe eignen sich unterschiedliche Evaluationsmaße.

**Evidenzbasierter Ansatz** Auffassung von Wahrscheinlichkeit, bei dem anstelle von Häufigkeiten Wahrscheinlichkeitsverteilungen verwendet werden, in denen Vorwissen und a-priori-Annahmen explizit im Modell ausgedrückt werden. Syn.: Bayesscher Ansatz.

**Expertensystem (expert system)** Ein Expertensystem ist ein Computersystem, das auf einem speziellen Wissensgebiet die Kompetenz von menschlichen Experten nachbildet und als Beratungs- und Problemlösungssystem eingesetzt werden kann.

**Explosion, kombinatorische (combinatorial explosion)** Kombinatorische Explosion beschreibt die dramatische Vergrößerung des Möglichkeitenraumes einer Berechnung oder von Zuständen, etwa durch die Einführung einer weiteren Eingabedimension. Dies zieht oft nach sich, dass erschöpfende Suchansätze nicht mehr mit realistischem Speicher- und Zeitaufwand erfolgreich sind und andere Methoden, etwa ▶ **Heuristiken**, zum Einsatz kommen müssen.

**Extension (extension)** In der Referenzsemantik Bezeichnung für das von einem sprachlichen Ausdruck denotierte Objekt bzw. die denotierte Menge von Objekten.

**Facettensuche oder Facettierte Suche (faceted search)** beschreibt die Möglichkeit, in einer Information-Retrieval-Anwendung Suchergebnisse durch das Definieren von Filtern auf Metainformationen einzuschränken.

**Fachausdruck (expression in technical language)** Ein Fachausdruck (auch Fachbegriff) ist ein Wort bzw. eine Phrase, das nach einem vorgegebenen Kriterium für ein Fachgebiet charakteristisch ist. Syn.: Fachbegriff.

**Fachbegriff (technical term)** Siehe ▶ **Fachausdruck**.

**Fachsprache (technical language)** Eine Fachsprache ist eine Untergruppe der von einem Sprachsystem erzeugbaren Strukturen, wie sie insbesondere in Fachtexten verwendet wird. Fachsprachen unterscheiden sich von der Allgemeinsprache durch messbare linguistische Abweichungen im Hinblick auf das Lexikon, die Syntax und die Semantik.

**Fachterminologie (terminology)** Die Fachterminologie ist das Begriffs- und Benennungssystem eines Fachgebietes, das alle Fachausdrücke umfasst, die allgemein üblich sind.

**Fachtext (domain-specific text)** Textsorte, die dazu dient, andere Fachleute desselben Faches oder Anwendungsbereiches zu informieren oder die Kommunikation mit Vertretern und Vertreterinnen anderer Disziplinen oder Laien über fachliche Sachverhalte zu ermöglichen. Fachtexte unterscheiden sich von allgemeinsprachlichen Texten durch das Vokabular sowie eine für den Anwendungsbereich charakteristische Syntax und Morphologie.

**Feature (feature)** Siehe ▶ **Merkmale**.

**Feed-Forward-Netz (feed forward network)** Ein neuronales Netzwerk, in welchem es im Gegensatz zum ▶ **rekurrenten Netzwerk** keine Verbindungen zu vorherigen Ebenen (oder, je nach Betrachtungsweise, zu vorherigen Zeitschritten) gibt. Das Netzwerk ist also aus aufeinanderfolgenden Schichten aufgebaut.

**Fehlerfortpflanzung (error propagation)** Fehler, die in einem Verarbeitungsschritt der linguistischen Pipeline auftreten, produzieren in dem nachfolgenden Verarbeitungsschritt möglicherweise weitere Fehler, welche ihrerseits wiederum Fehler im nächsten Verarbeitungsschritt verursachen können. Die Fehlerfortpflanzung ist ein typisches Problem starrer Pipelines, bei denen in den einzelnen Schritten auf Basis lokaler Informationen Entscheidungen getroffen werden, die sich erst später bzw. global als falsch herausstellen.

**Fine-Tuning (fine tuning)** bei neuronalen Netzwerken bezeichnet die Anpassung vortrainierter Embeddings an die vorliegende Klassifikationsaufgabe durch Weitertrainieren der Embeddings innerhalb der Netzwerkarchitektur.

**Fleiss' Kappa (Fleiss' kappa)** ist ein Maß für ▶ **Interrater-Reliabilität** zwischen zwei oder mehr Annotierenden.

**Flexion (flexation)** Bezeichnung für die Ableitung grammatischer Vollformen aus einem Stamm. Im Deutschen geschieht dies meist durch Anhängen von Flexionssuffixen sowie die Veränderung des Stamms durch Um- und Ablaute.

**Flexiv (bound morpheme)** Morphem, die lediglich eine grammatische und keine lexikalische Bedeutung hat. Flexive werden an Wortstämme angefügt (affigiert).

**Frequentistischer Ansatz: (count-based approach)** Auffassung von Wahrscheinlichkeit als Approximation der relativen Häufigkeit. Text-Mining-Verfahren, die auf diesem Ansatz aufbauen, untersuchen die Häufigkeit und statistische Verteilung von Sprachdaten.

**Fügung, feste (set phrase) (auch: feste Wendung, Phraseologismus)** Meist umgangssprachlich verwendete feste Wortverbindung, deren Bedeutung sich in der Regel nicht aus der Bedeutung der Einzelwörter erschließen lässt (z. B. *Spitze des Eisbergs, eingefleischter Junggeselle*).

**F-Wert (F-score)** Der F-Wert berechnet sich aus dem harmonischen Mittel zwischen ▶ **Precision** und ▶ **Recall** und wird oft als Maß zur Evaluation von überwachten Lernverfahren verwendet.

**Gated Recurrent Units (GRU)** Eine rekurrente neuronale Netzwerkarchitektur, Vereinfachung der Long Short-Term Memories, die mit weniger Rechenaufwand oft Ergebnisse von gleicher Qualität liefert.

**Gazetteers (gazetteers)** Listen bekannter Namen von Personen, Organisationen, Orten, Regionen etc. zur Verbesserung automatischer Eigennamenerkennungssysteme.

**Gegensätze (opposites)** Zwei Begriffe A und B sind Gegensätze in Bezug auf einen Oberbegriff, wenn beide einen gemeinsamen Oberbegriff C haben und die Schnittmenge zwischen der Extension von A und der Extension von B leer ist.

**Gewichtungsfaktor (weight)** Ein Gewichtungsfaktor regelt den Einfluss, den ein Term als Bestandteil einer Formel auf das Gesamtergebnis hat. Die Bedeutung einzelner Terme kann damit hervorgehoben, die Bedeutung von anderen hingegen abgewertet werden.

**Gibbs Sampling (Gibbs sampling)** Auswahl von Stichproben für das Topic Modelling, bei der für jedes Wort abhängig von allen anderen Zuordnungen seine Topic-Zuordnung berechnet wird. Die hochdimensionale Verteilung wird durch wiederholtes Ziehen von niedrigdimensionalen Variablen simuliert. Von einzelnen Wörtern ausgehend, werden so die Zuordnung von Wörtern zu Topics sowie die Zuordnung von Dokumenten zu Topics iterativ approximiert.

**Gold-Daten (gold data)** Daten inklusive manueller, korrekter Annotationen. Algorithmen des maschinellen Lernens werden auf derartigen Daten trainiert und

evaluiert. Meist werden Gold-Daten explizit durch menschliche ▶ **Annotatoren** erstellt.

**GPU (Graphics Processing Unit)** Grafikprozessoren, entwickelt für die effiziente Berechnung von Computergrafiken, sind aufgrund ihres effizienten Umgangs mit Matrizen aus Fließkommazahlen auch für das Berechnen neuronaler Netze geeignet.

**Grammatik, formale (formal grammar)** Eine Grammatik (formale) G definiert die Sätze einer Sprache. Sie wird gegeben durch ein Tupel  $G = (N, T, P, S)$ . Hierbei ist  $N$  = Menge der nichtterminalen Symbole (syntaktische Kategorien),  $T$  = Menge der terminalen Symbole (Wörter der Sprache),  $S$  = Startsymbol und  $P$  = Menge der Produktionsregeln, die festlegt, wie sich vom Startsymbol über die syntaktischen Zwischenkategorien die Sätze der Sprache als Folge von Wörtern ableiten lassen.

**Ground Truth (ground truth)** Für die Evaluation von Algorithmen verifizierte Analyseergebnisse, die als Maßstab für die Bewertung der Qualität von Algorithmen dienen. Siehe auch Gold-Daten.

**Grundform (base form)** Ausgezeichnete Wortform als Bezeichner für ein Wort, beispielsweise Nominativ Singular für Nomina und Infinitiv für Verben. Syn.: Lemma.

**Gültigkeit (validity)** Eine XML-Datei heißt gültig, wenn sie wohlgeformt ist und den für diese Datei definierten XML-Regeln entspricht.

**Hashverfahren (hashing)** Funktion, die einer Zeichenkette einen Hashwert aus einem typischerweise großen Wertebereich zuordnet. Hashverfahren wie MD5 können so genutzt werden, um beispielsweise Sätzen mittels Hashwert eine Zahl als Identifikator zuzuordnen. Verschiedene Hashwerte stellen dann sicher, dass es sich um verschiedene Sätze handelt.

**Häufigkeit, absolute (absolute frequency)** Die absolute Häufigkeit eines Wortes  $w$  (Type) ist gleich der Anzahl der Vorkommen von  $w$  (Tokens) im Text.

**Häufigkeit, relative (relative frequency)** Die relative Häufigkeit eines Wortes  $w$  (Type) in einem Text ist gleich dem Quotienten aus der (absoluten) Häufigkeit dieses Wortes und der Gesamtzahl der Wörter (Tokens) im Text. Sie sollte für unterschiedliche Texte (der gleichen Sprache und ggf. aus dem gleichen Sachgebiet) ähnliche Werte annehmen.

**Häufigkeitsklasse (frequency class)** Eine Häufigkeitsklasse ist eine Einteilung der Wörter in Gruppen nach ihrer Frequenz im Korpus. Wörter der gleichen Häufigkeitsklasse treten im Korpus etwa gleich häufig auf. Oft werden die Klassen nach der Häufigkeit der darin enthaltenen Wörter geordnet und nummeriert, sodass die Nummer der Klasse eine Aussage über die Häufigkeit der zugeordneten Wörter zulässt.

**Head-Modifier-Relation (head-modifier relation)** Siehe ▶ **Dependenz**.

**Heuristik (heuristics)** Heuristiken sind vereinfachte Ansätze zur schnelleren bzw. einfacheren Lösung von Problemen. Sie können etwa die Exploration eines Problemraums unterstützen, sodass optimale Ergebnisse mit weniger Rechenaufwand berechnet werden. Heuristiken können ebenfalls angewandt werden um das optimale

Ergebnis möglichst gut zu approximieren oder ein korrektes Ergebnis in der Mehrheit der Fälle zu erreichen.

**Hintergrundkorpus (background corpus)** Hintergrundkorpora sind jene Korpora, die nicht spezifisch für eine Fragestellung oder Aufgabe gesammelt sind, und als solche auch keine Annotationen enthalten. Sie bestehen aus gesammelten Texten aus einer oder mehreren Domänen und sind z. B. für das Training von ▶ **Embeddings** geeignet.

**HTML (HTML; hypertext markup language)** Dokumentenauszeichnungssprache, die es mithilfe von HTML-Befehlen erlaubt, inhaltliche Kategorien von HTML-Dokumenten, z. B. Überschriften und Absätze, zu kennzeichnen. So ausgezeichnete Dokumente werden von Web-Browsern interpretiert und dargestellt. Die Dateiendung einer HTML-Datei lautet.html bzw.htm.

**HTML-Befehl (HTML tag)** Dient zur Auszeichnung von ▶ **HTML-Dokumenten** und besteht aus einer Anfangs- und einer Endemarkierung (tag), z. B.<p>Dies ist ein Absatz</p>. Einige Befehle benötigen keine Endemarkierung. In manchen Befehlen können zusätzlich Attribute mit Werten angegeben werden. Siehe ▶ **HTML**.

**HTML-Dokument (HTML document)** In ▶ **HTML** formatiertes Dokument, besteht aus Text und HTML-Befehlen, vornehmlich für die Darstellung von Webseiten. Dateiendung.html bzw. htm.

**Hybrides System (hybrid system)** Systeme, welche Ansätze aus verschiedenen Kategorien verbinden, werden als hybride Systeme bezeichnet. Im Kontext des maschinellen Lernens versteht man darunter Systeme, die sich in Teilen auf regelbasierte Ansätze stützen.

**Hyperlink (hyperlink)** Verweise auf andere Dokumente; in ▶ **Web-Browsern** meist farblich oder unterstrichen hervorgehoben; ein Mausklick auf einen Hyperlink bewirkt, dass zu dem Dokument, auf das verwiesen wird, verzweigt wird. Syn.: Link, Verweis, Referenz.

**Hyperonym (hyperonym)** Siehe ▶ **Oberbegriff**.

**Hyperparameter (hyperparameter)** Im Kontext des maschinellen Lernens sind Hyperparameter jene Parameter die nicht durch den Algorithmus approximiert oder berechnet werden, sondern a priori durch den Entwickler bzw. die Entwicklerin des Systems festgelegt werden. Bei einem neuronalen Netzwerk sind das etwa die ▶ **Netzwerktopologie**, die Lernrate und die Wahl des Trainingsalgorithmus.

**Hypertext (hypertext)** Text, der Sprungmarken bzw. Verweise (▶ **Hyperlinks**) auf andere Texte enthält. Hyperwürfel (hyper cube) Speicherung von Daten in mehrdimensionalen Strukturen. Jede Zelle innerhalb eines Würfels wird durch die Elemente aller Dimensionen bestimmt und kann direkt angesprochen werden (Data Warehouse).

**Hyponym (hyponym)** Siehe ▶ **Unterbegriff**.

**Information (information)** Bei Information handelt es sich um Daten, die in einem Kontext interpretiert werden und somit eine Bedeutung für den Besitzer oder Empfänger dieser Daten haben. Häufig liegen Informationen in wenig strukturierter Form als Textdokumente, Zeichnungen, Bilder etc. vor.

**Information Retrieval (information retrieval)** Die Auffindung (engl. retrieval) von Informationen im Sinne einer Inhaltsabfrage auf großen Dokumentenkollektionen.

**Informationsextraktion (information extraction)** Beschreibt im Kontext der Sprachverarbeitung den Prozess, strukturierte Informationen aus natürlicher Sprache zu gewinnen, etwa das Ergebnis eines Fußballspiels aus einem Spielbericht.

**Instanz (instance, data point)** Im maschinellen Lernen sind Instanzen einzelne Eingabe-Elemente auf denen operiert wird, das können z. B. komplette Bilder, eine Menge binärer Eigenschaften oder, im Text Mining, einzelne Worte, Dokumente oder Sätze sein.

**Interrater-Reliabilität (Inter-Annotator-Agreement, IAA)** Ein Maß für die Übereinstimmung von manuell Annotierenden bezüglich einer Annotationsaufgabe. Sie wird als obere Schranke für die Genauigkeit von auf diesen Daten trainierten Lernverfahren angesehen.

**Inverse Dokumentfrequenz, IDF (inverse document frequency)** Maß im Information Retrieval zur Bestimmung des Anteils von Dokumenten, in denen ein Token vorkommt, im Verhältnis zur Gesamtzahl aller Dokumente eines Korpus. Die IDF wächst, wenn ein Token nur in wenigen Dokumenten auftritt.

**Inverse Liste (inverted index)** Liste, in der zu jedem im Korpus vorkommenden Wort alle Positionen des Auftretens im Korpus verzeichnet sind. Sie wird während der Korpuserstellung erzeugt und ermöglicht die schnelle Suche nach Wörtern im Korpus.

**Join (join)** Operation auf relationalen Datenbanken, mit deren Hilfe Datensätze aus mehreren Tabellen aufgrund einer gemeinsamen Eigenschaft (der Join-Bedingung) zusammengeführt werden können.

**JSON (JSON) (JavaScript Object Notation)** Einfach lesbares und kompaktes Datenformat, benutzt zum Datenaustausch oder auch der Datenspeicherung.

**Kategorie, syntaktische (syntactic category)** Eine syntaktische Kategorie ist das Potential eines Wortes, mit anderen Wörtern derselben Sprache syntaktisch wohlgeformte Kombinationen (Phrasen) zu bilden. Zu den syntaktischen Kategorien gehören traditionell: Nomen, Verb, Adjektiv, Artikel, Präsynonymposition und Konjunktion. Empirisch bilden Wörter derselben Kategorie paradigmatische Distributionsklassen. Syn.: Wortart.

**Klassifikation (classification)** Die Klassifikation in der automatischen Sprachverarbeitung ist das Versehen von sprachlichen Objekten (Wörter, Sätze, Absätze, Texte) mit Markierungen aus einer definierten Menge von Symbolen.

**Klassifikator (classifier)** Ein Klassifikator ist die Funktion, welche eine ▶ **Klassifikation** durchführt.

**Kohyponym (co-hyponyme)** Hat ein Begriff mehrere Unterbegriffe, stehen diese in der Relation der Kohyponymie und sind Kohyponyme.

**Komplementärbegriff (complementary term)** Zwei Begriffe A und B sind Komplementärbegriffe, wenn sie Gegensätze sind und das Komplement von A äquivalent zur Extension von B ist.

**Komponente (component)** Komponenten sind lose gekoppelte Programme, die einzelne Verarbeitungsschritte innerhalb einer komplexen Pipeline erledigen. Typische Komponenten einer linguistischen Pipeline sind beispielsweise Textsegmentierung in Sätze, ▶ **Tokenisierung** und ▶ **Part-of-speech-Tagging**.

**Komposition (composition)** Morphologischer Prozess, bei dem durch das Aneinanderfügen von zwei Stämmen bzw. freien Morphemen neue Wörter gebildet werden.

**Kompositionsprinzip (principle of composition)** auch Frege-Prinzip: Annahme, dass die Bedeutung eines Satzes eine Funktion der Bedeutung seiner Teilausdrücke ist.

**Kompositum (compound)** Ein durch Komposition aus wenigstens zwei Stämmen gebildetes Wort.

**Konfidenz (confidence)** Ein statistisches Maß, welches angibt, wie wahrscheinlich es ist, dass eine gegebene Beobachtung nicht durch reinen Zufall sondern durch eine Systematik in den Eingabedaten entstanden ist.

**Konjugation (conjugation)** Flexion (Beugung) von Verben. Verändert wird dabei die Person, Numerus, Tempus, Modus und das sog. Genus verbi (Aktiv oder Passiv).

**Konkatenation (concatenation)** Begriff aus der theoretischen Informatik, mit dem das Hintereinanderschreiben zweier Zeichen bzw. Zeichenketten bezeichnet wird. Die Konkatenation zweier Wörter „x“ und „y“ ist das Wort, das sich durch Hintereinanderschreiben der beiden Wörter ergibt, also „xy“.

**Konstituente (constituent)** In der Linguistik Bezeichnung für die unmittelbaren Bausteine von Sätzen auf der syntaktischen Ebene, also Wörter und Kombinationen von Wörtern. Empirisch lassen sich Konstituenten durch eine Reihe von Tests bestimmen, welche hinreichende Bedingungen für das Vorliegen einer Konstituente definieren.

**Konstituentenparsing (constituency parsing)** Verarbeitungsschritt zum automatischen Zuweisen der ▶ **Konstituenten-Struktur** eines Satzes.

**Konstituenten-Struktur (constituent structure)** Syntaktische Phrasenstruktur, die mit einer kontextfreien Chomsky-Grammatik dargestellt werden kann, der Parse-Baum eines Satzes.

**Kontext, globaler (global context)** Der globale Kontext eines Wortes  $w$  enthält alle Wörter, die mit  $w$  statistisch signifikant häufig gemeinsam auftreten (bezogen auf ein Signifikanzmaß und einen Schwellenwert).

**Kontext, lokaler (local context)** Der lokale Kontext eines Wortes  $w$  ist die Menge der Wörter, mit denen  $w$  zusammen in einem Satz auftritt.

**Kontextualisierte Embeddings (contextualized embeddings)** Embeddings, deren Wert sich durch den Kontext der betrachteten Instanz definiert. Im Gegensatz zu statischen Embeddings gibt es nicht für jedes Element im Vokabular eine feste Representation, stattdessen wird das kontextualisierte Embedding in der Regel aus den statischen Embeddings der ▶ **Instanzen** im lokalen Kontext berechnet.

**Kontextvolatilität (context volatility)** Maß zur Quantifizierung von Kontextänderungen eines Wortes über die Zeit. Die Kontextvolatilität gibt insbesondere bei niederfrequenten Wörtern frühzeitig Aufschluss über semantische Verschiebungen im Verwendungskontext eines Ausgangswortes.

**Konverse (converses)** Inhaltliche Relation zwischen begrifflichen Gegensätzen in Bezug auf einen gemeinsamen Oberbegriff.

**Kookkurrenz, signifikante (significant co-occurrence)** Als signifikante Kookkurrenz bezeichnet man das statistisch auffällige gemeinsame Auftreten zweier Wörter in einem Satz oder Textfenster (bezogen auf ein Signifikanzmaß und einen Schwellenwert).

**Kookkurrenzanalyse (co-occurrence analysis)** Verfahren des Text Mining, bei dem für ein Korpus signifikante Kookkurrenzen berechnet und visualisiert werden.

**Koreferenz (coreference)** Textspannen in einem Dokument stehen in Koreferenz, wenn sie auf dieselbe Entität verweisen.

**Koreferenzauflösung (coreference resolution)** Der Erkennungsprozess jener Textspannen in einem Dokument, welche sich auf dieselbe Entität beziehen. Dabei werden alle Referenzen (etwa Pronomen aber auch explizite, namentliche Nennungen), die sich auf die gleiche Entität beziehen, in einer Koreferenzkette zusammengefasst.

**Korpus (text corpus)** Ein Korpus ist eine Sammlung von Texten.

**Korpusvergleich (corpus comparison)** Verfahren zur Ermittlung von statistisch signifikanten Unterschieden in der Verwendung von Vokabularen im Vergleich zwischen einem Analysekörper und dem Referenzkörper.

**Kosinusähnlichkeit (cosine similarity)** Siehe ▶ **Kosinus-Maß**.

**Kosinus-Maß (cosine measure)** Das Kosinus-Maß ist ein Maß für die Ähnlichkeit von Vektoren im Information Retrieval. Berechnet wird dabei der Kosinus des Winkels zwischen den beiden Vektoren zur Modellierung der Ähnlichkeit der repräsentierten Instanzen. Sind diese normiert, ist der Wert gleich dem Skalarprodukt der beiden Vektoren. Da die Komponenten der Vektoren sämtlich nichtnegativ sind, ist der Winkel zwischen den Vektoren höchstens  $90^\circ$  und das Kosinus-Maß liegt zwischen 0 (maximal unähnlich) und 1 (identisch).

**Kreuzvalidierung (cross validation)** Schema zur Aufteilung der Trainingsdaten zur Evaluation überwachter Lernverfahren. Hier wird die Trainingsmenge in n (z. B. 5 oder 10) gleich große Mengen aufgeteilt, wovon jeweils reihum eine Menge als Development-Menge und die anderen als Trainingsmenge verwendet werden. Insbesondere bei dünner Datenlage empfohlen.

**Latin (Latin)** Genormter 8-Bit-Zeichensatz (256 Positionen), der den ASCII-Code um 128 Positionen erweitert. Latin-1 deckt westeuropäische Sprachen ab. Siehe auch ▶ **Unicode**.

**Lemma (lemma)** Siehe ▶ **Grundform**.

**Lernverfahren, maschinelles (machine learning algorithm)** Ein Verfahren des überwachten Lernens, welches auf Basis einer ▶ **Trainingsmenge** eine Funktion approximiert, die jeden (Eingabe) Featurewert zu einem (Ausgabe) Label überführt.

**Lernverfahren, überwachtes (supervised learning)** Siehe ▶ **Verfahren, überwachtes**.

**Lernverfahren, unüberwachtes (unsupervised learning)** Siehe ▶ **Verfahren, unüberwachtes**.

**Levenshtein-Distanz, auch: Editierdistanz (Levenshtein distance; edit distance)** Abstand zwischen zwei Zeichenketten gemessen in der minimalen Anzahl der benötigten zeichenbasierten Einfüge-, Ersetzungs- oder Löschoperationen um die eine Zeichenkette in die andere abzuändern.

**Lexikographie (lexicography)** Das Fachgebiet beschäftigt sich mit der Erstellung von Wörterbüchern. Dazu zählen Konzeption, Stichwortauswahl und das Verfassen der Wörterbucheinträge.

**Lexikon (lexicon)** In der Linguistik und Automatischen Sprachverarbeitung die Liste der dem System bekannten Wörter einer Sprache.

**Linguistische Ebene (level of linguistics)** Bezeichnung für die Elemente einer hierarchischen Untergliederung von der Verarbeitung von Sprache. Üblicherweise werden die linguistischen Ebenen unterteilt in Phonologie, Phonetik, Morphologie, Syntax, Semantik und Pragmatik.

**Linguistisches Wissen (linguistic knowledge)** Beschreibung der für eine Sprache typischen Gesetzmäßigkeiten mit Bezug auf die linguistischen Ebenen. Für die automatische Verarbeitung natürlicher Sprache muss dieses Wissen in geeigneter Form zur Verfügung stehen.

**Log-Likelihood-Maß (log likelihood measure)** Statistisches Plausibilitätsmaß, welches auf der Basis beobachteter Daten die plausibelsten Parameter einer Wahrscheinlichkeit ermittelt. Im Text Mining wird das Log-Likelihood-Maß verwendet, um die statistische Signifikanz des gemeinsamen Auftretens von Wortpaaren zu ermitteln, siehe ▶ **Kookkurrenzen**.

**Log-Likelihood-Ratio (log likelihood ratio)** Quotient aus zwei Likelihood-Funktionen, im Korpusvergleich Quotient aus der Likelihood-Funktion der wahrscheinlichen Anzahl eines Types im Analysekorpus und der als Nullhypothese aus dem Referenzkorpus abgeleiteten erwarteten Anzahl.

**Long Short-Term Memory (LSTM)** Eine rekurrente neuronale Netzwerkarchitektur, in der ein expliziter Zustand vorgehalten wird. Der Zustand wird, mittels gelernter Gewichte, basierend auf der Eingabe sowie dem Zustand selbst in jedem Zeitschritt verändert. Diese explizite Modellierung erlaubt insbesondere das Modellieren temporal weit reichender Abhängigkeiten, Informationen über diese werden in einfacheren rekurrenten Architekturen oft nicht erfolgreich gelernt.

**Makro-Sicht (macro view)** Sicht auf sehr umfangreiche Textkorpora mit dem Ziel, diese inhaltlich zu strukturieren, z. B. durch das Clustering ähnlicher Texte oder der Identifizierung von Themen und deren Verteilung im Textkorpus.

**Markdown (Markdown)** Beliebte Markup-Sprache, welche das Hinzufügen von Formatierungsinformationen zu Textdateien erlaubt, ohne dabei die Lesbarkeit der Textdatei zu opfern. So werden etwa Auflistungen einfach dadurch notiert, dass aufeinanderfolgende Zeilen jeweils mit einem „\*“ oder „-“ starten.

**Markov-Modell (Markov model)** Ein diskretes Markov-Modell erster Ordnung beschreibt eine Folge von Zuständen (hier: Tokens oder Zeichen) so, dass ein bestimmter Zustand nur von dem Zustand unmittelbar davor abhängt, nicht aber

von weiter zurückliegenden Zuständen. Markov-Modelle höherer Ordnung berücksichtigen entsprechend mehr Zustände aus der Vergangenheit. Markov-Modelle dienen zur Beschreibung einer Sequenz von Werten einer Zufallsvariable, die nicht unabhängig voneinander sind. Insbesondere hängen die Werte von den anderen in der Sequenz aufgetretenen Werten ab. Markov-Modelle lassen sich auch durch endliche Automaten mit Übergangswahrscheinlichkeiten beschreiben.

**Markup (markup)** Markup ist eine Auszeichnung, eine Markierung, die an bestimmten Stellen in einem Dokument eingefügt wird, um die Form der Darstellung oder die Struktur der Dokumente zu beschreiben. Die einzelnen voneinander getrennten Markup-Elemente werden als Tags bezeichnet.

**Maschinelles Lernen (machine learning)** Verfahren der Künstlichen Intelligenz für das Erkennen von Mustern und Gesetzmäßigkeiten in Daten und für die Generierung von Problemlösungen. Üblicherweise wird zwischen überwachtem Lernen (supervised learning) und unüberwachtem Lernen (unsupervised learning), bestärkendem (reinforcement learning) und aktivem Lernen (active learning) unterschieden.

**Mehrdeutigkeit (ambiguity)** Mehrdeutigkeit liegt vor, wenn es für ein Wort bzw. eine Wortfolge mehr als eine semantische Interpretation gibt. Bei der Mehrdeutigkeit ist zwischen einer lexikalischen und strukturellen Mehrdeutigkeit zu unterscheiden. Ein Wort ist lexikalisch mehrdeutig, wenn es mehr als eine Bedeutung oder Funktion hat. Ein Wort bzw. eine Wortfolge ist strukturell mehrdeutig, wenn es mehr als eine sinnvolle Zerlegung gibt.

**Mehrworteinheit (multi-word unit)** Eine aus mehreren Wörtern bestehende Zeichenkette, die in der Text Mining Anwendung als Einheit behandelt werden soll. Für die Extraktion von Mehrworteinheiten lassen sich meist vorher definierte ▶ POS-Muster nutzen.

**Merkmal (feature)** Repräsentation sprachlicher Einheiten als numerische oder nominale Werte zur Verwendung in statistischen oder neuronalen Modellen. Für jedes Feature wird eine Featurefunktion erstellt, welche Featurewerte für alle zu charakterisierenden sprachlichen Einheiten berechnen kann. Featurefunktionen können hierbei auf einer Vielzahl von Eigenschaften des sprachlichen Materials definiert werden. Denkbare Merkmale für Wörter sind beispielsweise Groß- und Kleinschreibung als binärer Wert, relative Häufigkeit des Wortes als reelle Zahl oder das ▶ Word Embedding als reellwertiger Vektor. Während solche Merkmale im maschinellen Lernen klassischerweise auf Basis der Aufgabenstellung händisch entwickelt wurden, bietet das End-to-End-Lernen die Möglichkeit, geeignete Repräsentationen automatisch im Modell erlernen zu lassen.

**Metadaten (metadata)** Metadaten für Dokumente sind Werte von qualifizierten Textattributen, welche Texte in technischer, organisatorischer und inhaltlicher Hinsicht beschreiben. Ein wichtiger Standard ist **Dublin Core** der Dublin Core Metadata Initiative (DCMI).

**Methode, neuronale (neural method)** Siehe ▶ Verfahren, neuronale.

**Methode, regelbasierte (rule-based method)** Siehe ▶ **Verfahren, regelbasierte**.

**Methode, statistische (statistical method)** Siehe ▶ **Verfahren, statistische**.

**Mikro-Sicht (micro view)** Sicht auf sehr umfangreiche Textkorpora mit dem Ziel, aus einem Text auffällige Informationen zu extrahieren und zusammen mit den Informationen aus anderen Texten zu verbinden.

**Mismatch (mismatch)** Ein Mismatch ist eine Nicht-Übereinstimmung von Zeichenketten. Die Stelle des ersten Mismatch zwischen zwei Wörtern ist die Position desjenigen Zeichens, vom Wortanfang aus betrachtet, in dem sich die beiden Wörter das erste Mal voneinander unterscheiden.

**Morphem (morpheme)** Ein Morphem ist in einer natürlichen Sprache die kleinste bedeutungstragende Einheit von Zeichenketten. Man unterscheidet zwischen freien Morphemen wie z. B. *Wort* oder *rot*, die ohne Affixe im Text auftreten können, und gebundenen Morphemen, die nur mit ergänzenden Affixen auftreten können. Zu den gebundenen Morphemen zählen alle grammatischen Morpheme und Derivative.

**Morphologie (morphology)** Teilbereich der Linguistik, der beschreibt, welche Morpheme in einer Sprache vorkommen und wie diese zu Wortformen kombiniert werden.

**Multitask-Lernen (multitask learning)** ist das Lernen verschiedener Zielfunktionen unter Zuhilfenahme von verschiedenen Trainingsmengen mit demselben Klassifizierungsalgorithmus, typischerweise einem neuronalen Netzwerk. Falls die Zielfunktionen korreliert sind, kann dies zu Verbesserungen der Genauigkeit führen.

**Mundart (dialect) (auch: Dialekt)** Regionale Sprachvarietät mit abweichendem Wortschatz, aber auch teilweise veränderter Grammatik und Aussprache.

**Muster (pattern)** Sich wiederholende Struktur in Texten, die in Form von Regeln beschrieben werden kann.

**Mutual Information (mutual information)** auch: Transinformation oder gegenseitige Information: Informationstheoretisches Maß für die Stärke des statistischen Zusammenhangs zweier Zufallsgrößen.

**Nachbarschaftskookkurrenz (neighbour co-occurrence)** Die Nachbarschaftskookkurrenz ist das statistisch auffällige gemeinsame Auftreten zweier Wörter als unmittelbare Nachbarn. Besteht zwischen zwei Wörtern eine Nachbarschaftskookkurrenz, wird das vorangehende (erste) Wort als linker Nachbar, das nachfolgende (zweite) Wort als rechter Nachbar bezeichnet.

**Nachricht (message)** Eine Nachricht ist eine nach vorher festgelegten Regeln zusammengestellte, endliche Folge von Zeichen und Zuständen, die eine Information vermittelt.

**Named Entities (named entities)** Siehe ▶ **Eigenname**.

**Named Entity Recognition (named entity recognition)** Siehe ▶ **Eigennamenerkennung**.

**Natural Language Processing (NLP)** Regelbasierte, statistische und neuronale Verfahren für die automatische Verarbeitung gesprochener und geschriebener Sprache auf

der Grundlage informatischer und linguistischer Modelle. Syn.: Automatische Sprachverarbeitung, Sprachtechnologie.

**Netz, lexikalisch-semantisches (lexical semantic network)** Siehe ▶ **Wortnetz**.

**Netzwerktopologie (network topology)** Beschreibt den Aufbau eines neuronalen Netzwerks, also Anzahl und Verbindungen der einzelnen Neuronen. In einem ▶ **Feed-Forward Netz** ist die Topologie typischerweise durch die Größen der Ein- und Ausgabevektoren sowie durch die Anzahl der Schichten des Netzwerks und ihre jeweiligen Größen gegeben.

**Neuron, künstliches (artificial neuron)** Die elementaren Bausteine eines künstlichen neuronalen Netzwerks sind durch biologische Neuronen inspiriert. Ein künstliches Neuron transformiert, analog zum ▶ **Perzepron** einen reellwertigen Eingabe Vektor zu einem skalaren Ausgabe Wert, wobei eine Vielzahl von Aktivierungsfunktionen zum Einsatz kommen kann. Ein künstliches neuronales Netzwerk besteht in der Regel aus mehreren Schichten mit jeweils mehreren Neuronen.

**News Monitoring (news monitoring)** Sammlung und Analyse großer Nachrichtenkollektionen mit Text Mining. Als Datengrundlage dienen Texte oder in Text gewandelte gesprochene Sprache aus dem Internet oder den sozialen Medien, deren Publikationsdatum eindeutig erkennbar ist, und deren Schwerpunkt auf Nachrichten aus allen Lebensbereichen, insbesondere Politik, Wirtschaft, Unternehmen, Wissenschaft und Technologie liegt.

**N-Gramm (n-gram)** Ein n-Gramm besteht aus n aufeinanderfolgenden Wörtern bzw. Buchstaben.

**N-Gramm-Modell (n-gram model)** Einfaches Sprachmodell auf Basis von ▶ **n-Grammen**. Die Wahrscheinlichkeit eines unbekannten nachfolgenden Wortes beruht auf der relativen Häufigkeit des n-Gramms bestehend aus diesem Wort und den n-1 vorangegangenen Wörtern.

**Nicht-transparent (opaque)** Eigenschaft von Komposita, dass sich die Bedeutung des zusammengesetzten Wortes nicht aus der Bedeutung der Bestandteile erschließen lässt. Beispiele sind *Mauerblümchen*, *Elfenbein*, *Friedhof*, *Fuchsschwanz*.

**Nominalphrase (noun phrase)** Grammatikalische Kategorie, welche die Funktion eines Nomens im Satz ausfüllt. Kann aus einem Pronomen oder einem Nomen mit modifizierenden Adjektiven und ggf. Artikel bestehen.

**Oberbegriff (subordinate term) (auch Hyperonym)** Ein nicht-leerer Begriff B ist ein nicht-leerer Oberbegriff eines Begriffs A, wenn die Extension von A eine echte Teilmenge der Extension von B ist.

**Office-Formate (office file formats)** Sammelbegriff für Formate gängiger Microsoft Office-Applikationen wie Word, Excel oder Powerpoint. Derartige Formate sind in der Regel XML-basiert und unterstützen einen großen Funktionalitätsumfang von der relativ einfachen Formatierung des enthaltenen Textes bis zur komplexen Einbindung von Multimediaelementen und interaktiven Komponenten.

**One-Hot-Kodierung (one-hot encoding)** Schema zur Vektorrepräsentation kategorische Variablen, dabei steht jedes Element des Vektors für einen der möglichen

Ausprägungen. Der Wert „1“ an n-ter Stelle besagt dass es sich um diese n-te Ausprägung des Werts handelt wobei alle anderen Werte stets „0“ sind. In der Sprachverarbeitung wird dieses Schema etwa für die Kodierung von Worten in einem gegebenen Vokabular verwendet, ein solcher Vektor ist dann z. B. bei einem Vokabular der Länge 1000 auch 1000 Elemente lang.

**Ontologie (ontology)** Eine Ontologie ist eine explizite, meist axiomatische und standardisierte Formalisierung eines gemeinsamen Verständnisses von Schlüsselbegriffen und deren Relationen bezüglich eines Faches oder einer Anwendung. Oft beinhaltet eine Ontologie eine hierarchisch organisierte Taxonomie.

**Optical Character Recognition (OCR)** Erkennung von Zeilen, Wörtern und Buchstaben in einem Bild (optischer Zeichenerkennung). Im weiteren Sinne umfasst OCR mehr als die bloße Klassifikation von Pixelmustern als Buchstaben, vielmehr ist das Ziel eine möglichst umfassende und fehlerfreie Konvertierung von Bilddateien in Dokumente in einem maschinenlesbaren Format für die weitere Be- und Verarbeitung.

**Out of vocabulary (OOV)** Problem des ungesesehenen Vokabulars, d. h. Wortformen, die nicht im Vokabular oder in den Trainingsdaten enthalten sind, aber in der Anwendung auf Testdaten auftreten. Ein unangepasstes Sprachmodell weist solchen Wortformen eine Wahrscheinlichkeit von Null zu, ein wortlistenbasiertes System kann diese Wörter nicht verarbeiten.

**Overfitting (overfitting)** Überanpassung eines statistischen bzw. neuronalen Modells an die Trainingsdaten. Während das Modell die Trainingsdaten exakt oder zumindest zu genau reproduziert, leidet die Fähigkeit, auf ungesesehenen Daten zu generalisieren. Overfitting tritt insbesondere dann auf, wenn die Anzahl der Trainingsdaten zu gering für die Anzahl der zu lernenden Parameter des Modells ist.

**Paradigmatisch (paradigmatic)** In der Tradition des linguistischen Strukturalismus Bezeichnung für das Auftreten von Zeichen bzw. Zeichenketten in ähnlichen Kontexten.

**Parallelisierung (parallelisation)** Siehe ▶ **Skalierung, horizontale**.

**Parsen, semantisches (semantic parsing)** Bezeichnet einen Verarbeitungsschritt von natürlich-sprachlichen Sätzen, in dem die Bedeutungsstruktur der Prädikate (meist Verben) und deren Argumenten expliziert wird.

**Parts Of Speech (POS)** Wortarten wie Nomen, Verb, Adjektiv, Pronomen, Präposition, Adverb, Konjunktion, Partizip und Artikel.

**Part-of-Speech-Tagging (part-of-speech tagging) oder Wortartentagging** Als Part-of-Speech-Tagging (POS-Tagging) bezeichnet man das Zuordnen syntaktischer Kategorien zu Tokens. Jedes (erkannte) Token in einem Text wird um die Angabe über die Wortart ergänzt, in der das Token im gegebenen lokalen Kontext verwendet wird.

**PDF (Portable Document Format)** Dokumentenformat, welches ursprünglich für die einheitliche Darstellung des gleichen Dokuments auf unterschiedlichen Plattformen und Endgeräten entwickelt wurde. PDFs können sowohl Grafiken in Raster- oder Vektorform, als auch Texte, Links und interaktive Elemente wie Formulare enthalten.

**Perplexität (perplexity)** auch: Kreuzentropie: Ein informationstheoretisches Maß für die Qualität eines Modells bezüglich einer (empirischen) Wahrscheinlichkeitsverteilung.

**Perzeptron (Perceptron)** Einfaches Element eines neuronalen Netzwerks mit einer beliebigen, festen Anzahl reeller Zahlen als Eingabe. Die Ausgabe wird aus dem Eingabevektor  $x$  mittels Gewichtsvektor  $w$  und Bias-Vektor  $b$  wie folgt berechnet:  $x^* w + b$ . Das Perzeptron kann mittels einer binären Aktivierungsfunktion zur binären Klassifikation benutzt werden.

**Phrase (phrase)** Als Phrase bezeichnet man die in einer natürlichen Sprache nach syntaktischen Regeln in sich abgeschlossenen syntaktischen Einheiten unterhalb der Satzebene.

**Phrasen-Struktur-Grammatik (phrase structure grammar)** Von Chomsky vorgeschlagene kontextfreie Ersetzungsgrammatik für eine natürliche Sprache. Alle Wörter werden als terminale Symbole behandelt, die syntaktischen Kategorien und Phrasen als nichtterminale Symbole und die syntaktischen Regeln werden in Form von Ersetzungsregeln angegeben.

**Pointer-Generator-Networks (pointer generator networks)** Neuronale Netzwerkarchitektur zur Generierung einer Ausgabe-Textsequenz basierend auf einer Eingabe-Textsequenz. Dabei kommt ein **Attention-Mechanismus** zum Einsatz und mittels der namensgebenden Pointer können Worte aus der Eingabesequenz direkt übernommen werden.

**Pointwise Mutual Information (PMI)** PMI ist ein Maß aus der Informationstheorie für die Stärke der Assoziation zweier Ereignisse, z. B. dem gemeinsamen Auftreten von Wörtern. Es quantifiziert den Unterschied zwischen der tatsächlichen und der erwarteten gemeinsamen Auftretenshäufigkeit unter Annahme statistischer Unabhängigkeit.

**Polarität (polarity)** Bewertungsdimension in der Sentimentanalyse wie *positiv – negativ*.

**Porter-Stemmer (Porter stemming algorithm)** Verfahren für die Stammformreduktion, bei dem nach bestimmten Regeln die Suffixe eines Wortes gelöscht werden.

**POS-Muster (POS pattern)** Abfolge von Wörtern einer bestimmten Wortart, wie z. B. „Nomen Nomen“, „Adjektiv Nomen“ oder „Adjektiv Adjektiv Nomen“.

**Precision (Precision)** Precision (Genauigkeitsgrad) ist im ▶ **Information Retrieval** ein Effektivitätsmaß zur Bewertung von Suchmaschinen. Die Precision für eine Anfrage ergibt sich als Quotient der Anzahl der gefundenen relevanten Dokumente durch die Anzahl der gefundenen Dokumente. Precision wird stets gemeinsam mit Recall benutzt. Syn.: Genauigkeitsgrad.

**Probabilistische kontextfreie Grammatik (probabilistic context free grammar)** Kontextfreie Grammatik, in der jede Regel mit einer Wahrscheinlichkeit versehen ist, wobei die Summe der Wahrscheinlichkeiten aller Regeln mit demselben Symbol auf der linken Seite 1 betragen muss. Die Wahrscheinlichkeit einer Zer-

legung ist das Produkt der Wahrscheinlichkeiten der Regeln, die während des Parsens angewandt werden.

**Quantor (quantifier)** Bei ▶ **regulären Ausdrücken** Operator, welche die Anzahl Wiederholungen von Ausdrücken angibt, gern in Kombination mit ▶ **Wildcard**.

**Rang (rank)** Der Rang eines Wortes w in der häufigkeitssortierten Wortliste ist die Listenposition von w.

**Rang, größter/höchster (highest rank)** Die letzte vergebene Rangnummer in der häufigkeitssortierten Wortliste; die Listenposition des letzten Wortes.

**Ranking (ranking)** Rankings sind Sortierungen einzelnen Elemente bezüglich einer bestimmten Eigenschaft, so werden z. B. im Information Retrieval Dokumente bezüglich ihrer Relevanz zu einer Suchanfrage sortiert.

**Recall (recall)** Der Recall (Erschöpfungsgrad) ist im Information Retrieval ein Effektivitätsmaß zur Bewertung von Suchmaschinen. Der Recall einer Anfrage ergibt sich als Quotient der Anzahl der gefundenen relevanten Dokumente durch die Anzahl der relevanten Dokumente. Recall wird stets gemeinsam mit Precision (Genauigkeitsgrad) benutzt. Syn.: Erschöpfungsgrad.

**Rechtschreibreform (spelling reform)** Veränderung des amtlichen Regelwerks zur Rechtschreibung zu einem gewissen Zeitpunkt. Häufig werden dadurch vorhandene Trends im tatsächlichen Sprachwandel offiziell anerkannt. Dies betrifft oft Getrennt- und Zusammenschreibung, Groß-/Kleinschreibung, veränderte Verwendung von Buchstaben (wie ß) oder Buchstabenverbindungen (wie ph) sowie die Zeichensetzung.

**Redundanz (redundancy)** Redundanz in der Informationsübertragung beschreibt mehrfach vorhandene Informationseinheiten. Diese können ohne Informationsverlust weggelassen werden, erhöhen aber die Übertragungssicherheit und können zur Fehlerkorrektur benutzt werden. Unter Redundanz in Texten versteht man das wiederholte Auftreten der gleichen Information, oft in unmittelbarer Nachbarschaft. Redundanz kann syntaktischer Art (wie Kongruenz) oder inhaltlicher Art (z. B. Wiederholung von Wörtern oder Wortteilen, Pleonasmus) sein. Das wiederholte Auftreten solcher Informationen ermöglicht deren automatische Erkennung.

**Referenzkorpus (reference corpus)** Das Referenzkorpus besteht in der Regel aus einer Anzahl von Texten, die den allgemeinen Sprachgebrauch wiedergeben. Zum allgemeinen Sprachgebrauch können ebenfalls einige, allgemein bekannte Fachausdrücke aus den verschiedenen Fachsprachen gehören. Diese Fachausdrücke treten jedoch im Referenzkorpus relativ selten auf – verglichen mit einem Korpus der entsprechenden Fachsprache. Syn.: Vergleichskorpus.

**Regelbasierter Ansatz (role-based approach)** Siehe ▶ **Verfahren, regelbasierte**.

**Regulärer Ausdruck (regular expression)** Reguläre Ausdrücke beschreiben eine eigene, vollständig definierte Kunstsprache, welche in vielen Varianten in praktisch jeder Programmiersprache, vielen Texteditoren, vor allem aber beim Text Mining oder allgemein in der automatischen Sprachverarbeitung zum Einsatz kommt. In der Praxis

ermöglicht diese Sprache, Ausdrücke zu formulieren bzw. Muster festzulegen, nach denen gesucht werden soll.

**Rekurrentes Netzwerk (recurrent network)** Neuronale Netzwerkarchitektur zur Anwendung auf sequentiellen Daten, dabei wird die Eingabe als Zeitreihe aufgefasst und die Berechnung eines jeden Zeitschritts erhält Informationen (etwa die Ausgabe oder den Wert nach Anwendung einer inneren Schicht) aus vorangegangenen Schritten als zusätzliche Eingabe.

**Rekursion (recursion)** Siehe ▶ **Selbstaufrufender Verweis**.

**Relation, logische (logical relation)** Logische Relationen sind semantische Relationen, die logische Folgerungen unterstützen. Hierzu zählen insbesondere die Oberbegriffs- und Unterbegriffsbeziehung, Synonyme, Gegensätze, Antonyme, Komplementär-begriffe und Konverse. Der Begriff der logischen Relationen kann mengentheoretisch präzisiert werden.

**Relation, paradigmatische (paradigmatic relation)** Paradigmatische Relation ist in der Tradition des linguistischen Strukturalismus die Bezeichnung für das Auftreten zweier Wörter in ähnlichen Kontexten.

**Relation, semantische (semantic relation)** Semantische Relationen sind Sinnrelationen zwischen Wörtern, wie sie insbesondere in Bedeutungswörterbüchern oder Thesauren verwendet werden, also z. B. Synonymie (Bedeutungsgleichheit), Unterbegriffs- oder Oberbegriffsbeziehungen. Die Elemente von Thesauren sind allgemein beschrieben in der ISO Norm ISO 25964.

**Relation, syntagmatische (syntagmatic relation)** Syntagmatische Relation ist in der Tradition des linguistischen Strukturalismus die Bezeichnung für das gemeinsame Auftreten zweier Wörter in einem Satz oder Textfenster.

**ROC-Kurve (receiver operating characteristic curve) auch Grenzwert-optimierungskurve** ROC-Kurven können zur Evaluation der Klassifikatorqualität eingesetzt werden, indem das Verhältnis von Precision und Recall für verschiedene Konfidenzwerte dargestellt wird; die Fläche unter der Kurve charakterisiert dann die Qualität des Klassifikators unabhängig von einer konkreten Konfidenzschwelle.

**Satz (sentence)** In sich abgeschlossene, sinntragende und wohlgeformte sprachliche Einheit. Aus syntaktischer Sicht ist ein Satz eine Konkatenation von Phrasen.

**Satz, nicht-wohlgeformter (ill-formed sentence)** In zu analysierenden Texten findet man auch Sätze, die von zweifelhafter Qualität sind: Sie können unvollständig sein, syntaktische Fehler enthalten, nicht den orthographischen Regeln entsprechen und inhaltlich unverständlich oder sinnlos sein.

**Satzkookkurrenz (sentence-level co-occurrence)** Als Satzkookkurrenz bezeichnet man das statistisch auffällige gemeinsame Auftreten zweier Wörter in einem Satz.

**Satzkookkurrenzmatrix (matrix of sentence-level co-occurrences)** Die Satzkookkurrenzmatrix ist eine quadratische, symmetrische Matrix, deren Zellen die Stärke der ▶ **Satzkookkurrenz** zwischen den mit den jeweiligen Zeilen und Spalten assoziierten Wörtern enthält.

**Satzkorpus (corpus of sentences)** Ein Korpus aus einzelnen Sätzen mit oder ohne Annotationen und Metadaten. Im Gegensatz zum ▶ **Dokumentkorpus** sind hier ganze Texte nicht rekonstruierbar durch Umordnung und dem Weglassen von Sätzen, auch im Rahmen der Qualitätssicherung.

**Schwache Signale (weak signals)** Konzept aus der Wirtschaftswissenschaft, mit dem frühzeitige Anzeichen für erwartbare Diskontinuitäten wirtschaftlicher Prozesse bezeichnet werden.

**Selbstaufrefender Verweis (self-referring link)** Siehe ▶ **Rekursion**.

**Semantik (semantics)** Gegenstand der Semantik ist die Bedeutung Zeichen und Zeichenketten auf allen linguistischen Ebenen. Allerdings ist der Begriff Bedeutung unscharf und bedarf einer genaueren Bestimmung durch eine semantische Theorie. Die wesentlichen Paradigmen einer semantischen Theorie sind die prozedurale, referentielle und strukturalistische Semantik.

**Semantik, distributionelle (distributional semantics)** Forschungsgebiet zur Entwicklung und Erforschung von datengetriebenen Repräsentationen für die Bedeutung von sprachlichen Elementen wie Wörtern, Sätzen und Texten auf Basis der ▶ **distributionellen Hypothese**.

**Sentimentanalyse (sentiment analysis, auch Stimmungsanalyse)** Anwendung des Text Mining mit dem Ziel, die Einstellung eines Autors oder einer Autorin zu einem Thema zu bestimmen. Unter Einstellung wird dabei eine Bewertung, ein emotionaler Zustand oder ein Appell verstanden.

**Sentimentausdruck (sentiment expression)** Ausdruck, mit dem eine subjektive Befindlichkeit oder eine Bewertung von Objekten, Ereignissen oder Meinungen (sentiment targets) ausgedrückt werden. Typische Bewertungsdimensionen sind dabei ▶ **Polaritäten** wie *positiv – negativ* für die Polaritätsanalyse oder *subjektiv-objektiv* für die Subjektivitätsanalyse.

**Sentiment-Orientierung (sentiment orientation)** Bezeichnung für die Bewertungsdimension von Wörtern, die anstelle einer bipolaren Bewertung wie *gut* oder *schlecht, positiv* oder *negativ* auch Zwischenwerte ausdrücken können (etwa auf einer Polaritätsskala mit einem Wert zwischen +1 für positiv und -1 für negativ mit 0 als neutral).

**Seq2Seq (sequence to sequence)** Systemarchitektur, in der basierend auf einer Eingabesequenz eine Ausgabesequenz generiert wird, etwa für die maschinelle Übersetzung. Dabei wird ein rekurrentes Netzwerk verwendet, um zuerst die Eingabesequenz einzulesen, und dann basierend auf dem internen Zustand und dem jeweils zuvor ausgegeben Element eine Ausgabesequenz zu generieren.

**Sequenzklassifikation (sequence classification/sequence tagging)** auch Sequenzmarkierung: Vorgang des automatischen Zuweisens von Labels für Instanzen innerhalb einer Sequenz, z. B. beim ▶ **Wortartentagging** oder bei der ▶ **Eigen-namenerkennung**.

**Signifikanzmaß (measure of significance)** Das Signifikanzmaß ist ein Maß für den Nachweis statistisch verlässlicher (signifikanter) Unterschiede in empirischen Untersuchungen.

**Skalierung (scaling)** Leistungsanpassung der Software und Hardware um auch große Problemgrößen bzw. Datenmengen in adäquater Zeit bearbeiten zu können. Es wird dabei zwischen ▶ **vertikaler und horizontaler Skalierung** unterschieden.

**Skalierung, horizontale (horizontal scaling)** Horizontale Skalierung ermöglicht die Leistungssteigerung durch Hinzufügen zusätzlicher, gleichartiger Rechner bzw. Knoten. Dazu muss die verwendete Software für die parallelisierte Verarbeitung optimiert sein. Durch den Anteil sequentieller Verarbeitungsschritte sind der Leistungssteigerung jedoch Grenzen gesetzt.

**Skalierung, vertikale (vertical scaling)** Vertikale Skalierung beruht auf der Leistungssteigerung eines einzelnen Systems durch den Einsatz leistungsfähigerer Hardware. Es ist keine Anpassung der verwendeten Software erforderlich. Der Leistungssteigerung sind jedoch Grenzen gesetzt, wenn bereits die leistungsfähigste Hardware verwendet wird.

**Skip-Gram-Modell (Skip-gram model)** Variante der neuronale Netzwerkarchitektur word2vec zum Erlernen von ▶ **Word Embeddings** in Form von ▶ **dichtbesetzten Vektoren**. Die Trainingsaufgabe des Netzwerkes ist das Vorhersagen des Kontextes zu einem Wort, d. h. die Wörter links und rechts des aktuellen Wortes, mittels eines ▶ **Sliding-Window**-Verfahrens. Die Reihenfolge der Wörter im Kontext ist dabei relevant.

**Sliding-Window (sliding window)** Ein Verfahren für die elementweise Verarbeitung einer Sequenz, wobei jeweils eine feste Anzahl an Elementen links und/oder rechts des aktuellen Elements als zusätzlicher Kontext für die Verarbeitung genutzt wird.

**Spam (spam)** Unerwünschte Informationen, in der Regel in Form von E-Mails oder Kurznachrichten, aber z. B. auch in Form ganzen Websites. Oft beinhaltet Spam unerwünschte Werbung oder Betrugsversuche.

**Spannenannotation (span annotation)** Annotation, die sich auf ein oder mehrere zusammenhängende Wörter bezieht. So können z. B. mehrere aufeinanderfolgende Worte als ein ▶ **Chunk** vom Typ Nominalphrase markiert werden.

**Sprachdynamik (language dynamics)** Die Veränderung einer natürlichen Sprache über die Zeit aufgrund gesellschaftlicher und sprachlicher Anpassungsprozesse.

**Sprache, formale (formal language)** In der theoretischen Informatik Bezeichnung für eine Menge von Zeichenketten, die eine bestimmte Teilmenge der Menge aller möglichen Wörter umfasst, die nach einer vorgegebenen Konstruktionsvorschrift über einem Alphabet erzeugt werden können.

**Sprache, kontextfreie (context-free language)** Bezeichnung für eine ▶ **Chomsky-Grammatik**, in der in jeder Regel der Grammatik auf der linken Seite genau ein nicht-terminales Symbol steht und auf der rechten Seite eine beliebige nicht-leere Folge von terminalen und nicht-terminalen Symbolen aus dem gesamten Vokabular.

**Sprache, natürliche (natural language)** Von Menschen zu allen Zwecken der Kommunikation verwendete gesprochene und geschriebene Sprache. Natürliche Sprachen unterscheiden sich von formalen Sprachen u. a. dadurch, dass sie sprachstatistischen Gesetzmäßigkeiten wie dem ▶ **Zipfschen Gesetz** folgen. Außerdem ist für eine natürliche Sprache nicht immer eindeutig definiert, welche Kombinationen von Wörter zulässig sind, und die Kombinationen von Wörter können strukturell mehrdeutig sein.

**Spracherkennung (speech recognition)** Automatische Erkennung gesprochener Sprache. Die Wandlung gesprochener Sprache in Text wird auch als speech-to-text bezeichnet.

**Sprachklassifikation (language identification)** Bestimmung der natürlichen Sprache, in der ein Text oder Textabschnitt verfasst ist.

**Sprachmodell (language model)** Sprachmodell ist in der statistischen Sprachverarbeitung die Bezeichnung für die Gesamtheit der konkreten Wahrscheinlichkeiten sprachlicher Ereignisse.

**Sprachstatistik (statistics of language)** Sprachstatistik ist die Bezeichnung für empirisch validierte statistische Gesetzmäßigkeiten natürlicher Sprache. Hierzu zählen u. a. die „Zipfschen Gesetze“ und die Small-Worlds-Eigenschaft semantischer Netze.

**Softmax-Funktion (softmax function)** Mathematische Funktion, um alle Komponenten eines reellwertigen Vektors in den Wertebereich von 0 bis 1 zu transformieren, wobei die Summe aller Komponenten 1 ergibt, sodass das Ergebnis als Wahrscheinlichkeitsverteilung angesehen werden kann.

**Stamm (stem)** Siehe ▶ **Basismorphem**.

**Stammformreduktion (stemming)** Zuordnung verschiedener Vollformen auf denselben Stamm.

**Steuerzeichen (control character)** Zeichen eines Zeichensatzes für die Steuerung eines Ausgabegeräts zur Darstellung von Zeichen des Zeichensatzes, also keine darstellbaren Zeichen wie z. B. Buchstaben, Ziffern oder Satzzeichen.

**Statistischer Ansatz (statistical method)** Siehe ▶ **Verfahren, statistische**.

**Statistischer Hypothesentest (statistical hypothesis testing)** Siehe ▶ **t-Test**.

**Stimulus-Response-Experiment (stimulus response experiment)** Experiment zur Erforschung sprachlicher Assoziationen. Nach dem Hören eines Wortes (als Stimulus) soll die Versuchsperson schnell das erste Wort (als Response) nennen, welches ihm/ ihr einfällt.

**Stoppwort (stop word)** Ein Stopwort ist ein Wort, das von der Analyse ausgeschlossen werden soll. Dazu gehören im Allgemeinen Wörter aus den geschlossenen Wortklassen wie Artikel, Konjunktion und Präposition.

**Strukturalismus, linguistischer (linguistic structuralism)** Eine auf den Linguisten Ferdinand de Saussure zurückgehende Richtung der Linguistik, nach der Zeichenketten einer linguistischen Ebene immer nur in Bezug auf jeweils andere Zeichenketten dieser Ebene einen Sinn bzw. eine Funktion haben.

**Stylometrie (stylometry)** In der Literaturwissenschaft und den Digital Humanities Bezeichnung für die Analyse stilistischer Merkmale von Texten.

**Subwort-Tokenisierung (sub-word tokenisation)** Tokenisierungsstrategie, bei der nicht nur ganze Wörter als Tokens verstanden werden, sondern auch einzelne Wortbestandteile. Dieser Ansatz kann mit einem relativ kleinen Vokabular sehr viele Worte abbilden, wobei seltene Wörter in Subworteinheiten zerlegt werden. Dies umgeht vor allem Probleme mit ▶ OOV Wörtern.

**Synonym (total synonym)** Zwei Wörter sind synonym, wenn sie in fast allen Kontexten ausgetauscht werden können, ohne die Bedeutung des Textes zu verändern. Im engeren Sinne sind zwei Wörter synonym, wenn sie dieselbe Extension haben.

**Synonym, schwaches (partial synonym)** Da echte Synonymie sehr selten anzutreffen ist, spricht man im Falle Austauschbarkeit mit geringer Bedeutungsänderung von schwacher Synonymie.

**Syntagmatisch (syntagmatic)** In der Tradition des linguistischen Strukturalismus Bezeichnung für das gemeinsame Auftreten von Zeichen bzw. Zeichenketten.

**Syntax (syntax)** Gegenstand der Syntax sind syntaktische Repräsentationen für Wörter einer Sprache und deren Kombination zu Sätzen.

**Tagset (tag set)** Liste von zulässigen kategorischen Werten für die ▶ Annotation von ▶ Tokens, Textspannen (▶ Spannenannotation) oder ▶ Dependenzrelationen. Für jede Aufgabe bzw. Art der Annotation wird typischerweise ein eigenes Tagset verwendet, z. B. für ▶ Part-of-Speech-Tagging oder ▶ Named Entity Recognition.

**Tanimoto-Ähnlichkeit (Tanimoto index)** Die Tanimoto-Ähnlichkeit ist ein Maß für die Ähnlichkeit von Vektoren mit Einträgen 0 oder 1. Berechnet wird der Quotient aus der Anzahl der übereinstimmenden 1-Werte und der Anzahl der Positionen, die in mindestens einem der Vektoren den Wert 1 haben.

**Taxonomie (taxonomy)** Hierarchische Strukturierung von Fachbegriffen.

**TEI (Text Encoding Initiative)** Dokumentenformat auf der Basis von XML, entwickelt von der gleichnamigen Organisation. Eignet sich zur Auszeichnung komplexer Strukturen (z. B. in Wörterbüchern) oder editorischer Informationen.

**Template-basierte Informationsextraktion (template-based information extraction)** Eine Art Informationsextraktion, bei der die für das Ausfüllen vorgegebener Schemata (Templates) notwendigen Informationen automatisch aus dem Text extrahiert werden, im Gegensatz zur Open Information Extraction.

**Tensor Processing Unit (TPU)** Spezialhardware zur Beschleunigung der Berechnung von Operationen auf Tensoren, Matrizen und Vektoren. Diese Hardware wird zum Training und zur Inferenz großer neuronaler Netzwerke verwendet.

**Term (term)** Ein Term ist im Information Retrieval ein Wort in einem Text bzw. einer Kollektion von Texten.

**Term, diskriminierender (discriminating term)** Ein diskriminierender Term ist ein Wort, die nach einem vorgegebenen Kriterium für einen Text bzw. eine Kollektion von Texten charakteristisch ist und deswegen gut zur Unterscheidung von Texten eingesetzt werden kann. Bei der Differenzanalyse sind diskriminierende Terme

diejenigen Wörter, die im Fachtext wesentlich häufiger auftreten als in einem allgemeinsprachlichen Referenzkorpus.

**Term-Dokument-Matrix (term document matrix)** Eine Matrix für die Wortfrequenzen pro Dokument. Dabei repräsentiert jede Zeile ein Dokument und jede Spalte ein Wort des Vokabulars, sodass jedes Element der Matrix die Anzahl der Vorkommnisse eines spezifischen Worts in einem spezifischen Dokument beschreibt.

**Term-Frequenz/Inverse-Dokument-Frequenz (term frequency – inverse document frequency; tf-idf)** Maß im Information Retrieval zur Ermittlung statistisch signifikanter Unterschiede in der Verwendung von Token beim Vergleich eines Analysekorpus mit einem Referenzkorpus. Eine hohe Termfrequenz im Analysekörper deutet darauf hin, dass das Token im Analysekörper wichtig ist, die inverse Dokumentfrequenz bemisst die statistische Signifikanz in der unterschiedlichen Verwendung dieses Token in Bezug auf das Referenzkorpus.

**Terminologie (terminology)** Das Begriffs- und Benennungssystem eines Fachgebietes, das alle Fachausdrücke umfasst, die allgemein üblich sind. Die Terminologie eines Fachgebietes ist eine Sammlung von Fachausdrücken, die meist in Form eines Thesaurus beschrieben wird.

**Terminologielehre (terminology theory)** Beschreibung der Gesetzmäßigkeiten und die Erarbeitung der Terminologie von Fachsprachen.

**Testmenge (test set)** Bestandteil des Datensatzes aus Instanzen und Labels, bezüglich dessen ein Modell final evaluiert wird. Während der Entwicklung darf die Testmenge nicht zum Einsatz kommen. Siehe auch Trainingsmenge und Development-Menge.

**Text (text)** Ein Text ist strukturiert und besteht aus Sätzen, die nach Art und Zweck des Textes zu größeren Einheiten zusammengefasst werden (Absätze, Abschnitte, Kapitel). Ein Text besteht aus Wörtern, die ihrerseits aus den Buchstaben eines Alphabets bestehen. Text repräsentiert Wissen und stellt insofern eine wesentliche Grundlage der Wissensverarbeitung dar.

**Textabdeckung (text coverage)** Die Textabdeckung eines Textes durch eine vorgegebene Menge von Wörtern beschreibt den Anteil des Textes, welche diese Menge am Gesamttext hat. So haben in typischen deutschsprachigen Texten die häufigsten 10.000 Wörter eine Textabdeckung von rund 80 %.

**Textkorpus (text corpus)** Siehe ▶ **Korpus**.

**Text Mining (Text Mining)** Computergestützte Verfahren für die semantische Analyse von Texten, welche die automatische bzw. semi-automatische Strukturierung von Texten, insbesondere sehr großen Mengen von Texten, unterstützen.

**Thesaurus (thesaurus)** Sammlung von Fachausdrücken eines Fachgebietes unter Nennung des Sachgebietes, der Synonyme, Übersetzungen und Übersetzungsvarianten, Oberbegriffe, Kohyponyme, Antonyme und Aufzählung von Beispielen.

**Token (token)** Im Text Mining Vorkommen einer nach den ▶ **Tokenregeln** definierten Zeichenkette in einem Text.

**Tokenisierung (tokenisation)** Prozess der Anwendung von ▶ **Tokenregeln** auf einen Text.

**Tokenregeln (token rules)** Regeln für die Zerlegung eines Textes in Wörter, Satzzeichen und Sonderzeichen.

**Topic-Modell (topic model)** Evidenzbasiertes, nicht-deterministisches Verfahren für die Generierung artifizieller Topics als einer Wahrscheinlichkeitsverteilung über das vorhandene Vokabular eines Textes. Jedes Dokument wird als eine Wahrscheinlichkeitsverteilung über eben diese Topics dargestellt, welche wiederum angibt, wie wahrscheinlich ein Topic für das aktuelle Dokument ist.

**Top-Level-Domain (TLD)** Meist länderspezifischer Teil einer Internetadresse, bestehend aus meist zwei oder drei Buchstaben nach dem letzten Punkt. Die Einschränkung auf die Top-Level-Domains de, at und ch ist eine sinnvolle Crawlingstrategie, um hauptsächlich deutschsprachige Webseiten zu besuchen.

**Trainingsmenge (training set)** Eine Menge von ▶ Instanzen, inklusive der jeweils erwünschten Ausgaben, die als Eingabe für einen maschinellen Lernalgorithmus verwendet werden und auf deren Basis ein Modell gelernt wird. Siehe auch Development-Menge und Testmenge.

**Transduktor (transductor)** Endlicher Automat, welcher eine Eingabe auf einem Eingabeband in eine Ausgabe auf einem Ausgabeband überführt. Also solche eignen sich Transduktoren zur Formalisierung von Regelsystemen der morphologischen Transformation, etwa um ein Wort auf seinen Stamm zurückzuführen.

**Transformation (transformation)** In der Transformationsgrammatik die Bezeichnung für die Ableitung von beobachteten Wortfolgen (sog. Oberflächenstrukturen) durch Veränderungen und Verschiebungen von Phrasen aus sog. Tiefenstrukturen, welche mithilfe einer kontextfreien PSG erzeugt worden sind.

**Transformer-Architektur/Transformer-Networks (transformer architecture/networks)** Neuronale Netzwerkarchitektur, welche auf Basis eines Encoders und eines Decoders sowohl zur Textgenerierung als auch für Klassifikationsaufgaben verwendet werden kann. Im Gegensatz zu anderen Architekturen wie ▶ Seq2Seq haben Transformer-Networks grundsätzlich eine feste Eingabegröße. So können keine Sequenzen beliebiger Länge behandelt werden, allerdings birgt die Architektur Vorteile bei effizienter Berechnung und der Modellierung weitreichender Abhängigkeiten.

**Translation Memory (translation memory)** System für die Erzeugung von Übersetzungshypothesen unter Einsatz maschinellen Lernens auf der Grundlage von bereits erstellten Übersetzungen eines Textes.

**Trainingskorpus (training corpus)** Ein Trainingskorpus ist eine Textmenge, die analysiert und ausgewertet wird, um die für einen Algorithmus notwendigen Daten und Parameter zu ermitteln.

**TREC (Text REtrieval Conference)** Seit 1992 Serie von Konferenzen im ▶ Information Retrieval und ▶ Natural Language Processing zur Evaluation von Verfahren für vorgegebene Retrieval- oder Verarbeitungsaufgaben wie beispielsweise das Erkennen von neuen Nachrichten im ▶ News Monitoring.

**Trend (trend)** Bezeichnung für eine sich langfristig abzeichnende Entwicklung. In der Statistik wird ein Trend als Zeitreihe dargestellt. Er beschreibt die quantitative Veränderung von Ereignissen, erklärt jedoch nicht die Hintergründe von Veränderungen.

**Trie (Trie)** Ein Trie (digitaler Mehrwegebaum) ist ein m-Wege-Baum, wobei m die Anzahl der Zeichen des Alphabets ist. Jeder Knoten im Baum repräsentiert einen Buchstaben. Die Buchstaben auf dem Weg von der Wurzel zu einem Knoten stehen für den Anfang eines Wortes. Die Nachfolgerknoten stehen für die entsprechenden Fortsetzungsmöglichkeiten. Syn.: digitaler Mehrwegebaum.

**Tri-Gramm (tri-gram)** Ein Tri-Gramm ist ein spezieller Typ von ▶ **n-Grammen**, das aus drei aufeinander folgenden Wörtern besteht. Siehe n-Gramm und Bi-Gramm.

**t-Test (t-test)** Statistischer Hypothesentest zur Überprüfung, ob zwei Stichproben sich statistisch signifikant unterscheiden.

**Type (type)** Äquivalenzklasse derjenigen Zeichenketten, die entsprechend den Tokenregeln für eine Anwendung als gleich betrachtet werden; Eintrag im Vokabular.

**Übersetzung, automatische (machine translation)** Übersetzung, die durch ein Computerprogramm ohne menschliche Hilfe erstellt wurde; das Forschungsgebiet der automatischen Übersetzung.

**Unicode (unicode)** Genormter 16-Bit-Zeichensatz (65.469 Positionen), der die Schriftzeichen aller Verkehrssprachen der Welt aufnehmen soll.

**Unigramm (uni-gram)** Ein Unigramm ist ein spezieller Typ von n-Grammen, das aus einem Wort oder einem Buchstaben besteht.

**Unterbegriff (subordinate term)** Ein nicht-leerer Begriff A ist ein nicht-leerer Unterbegriff eines Begriffs B, wenn die Extension von A eine echte Teilmenge der Extension von B ist. Syn. Hyponym.

**Urheberrecht (copyright law)** Eigentümer bzw. Eigentümerin aller Verwertungsrechte eines Textes ist sein Urheber bzw. seine Urheberin. Das Urheberrecht umfasst die gesetzlichen Regelungen zur Verwertung und zum Schutz der Urheberrechte.

**URL (uniform resource locator; uniform resource locator)** Im Web verwendete standardisierte Darstellung von Internetadressen; Aufbau: protokoll://domain-Name/Dokumentpfad.

**UTF-8** Beliebtes Kodierungsschema für den ▶ **Unicode** Standard. Hier werden durch die variable Länge ▶ **ASCII** Zeichen in nur 8 Bit kodiert, wohingegen Symbole aus anderen Sprachen (etwa deutsche Umlaute, aber z. B. auch kyrillische Zeichen und Emoji) in 16 oder mehr Bit kodiert werden.

**Valenzstruktur (valence structure)** Fähigkeit eines Wortes oder einer Wortgruppe, andere Wörter oder Wortgruppen als Ergänzungen zu fordern oder zu erlauben. Speziell können einzelne Verben Akkusativobjekte und/oder Dativobjekte fordern (*schenken*), erlauben (*lernen*) oder auch nicht erlauben (*regnen*).

**Vektor, dichtbesetzter (dense vector)** Reellwertiger Vektor, in dem die meisten Zahlenwerte ungleich null sind. Solche Vektoren ermöglichen effiziente mathematische Operationen wie Addition oder Multiplikation und bilden damit die Basis für meisten Verarbeitungsschritte in neuronalen Netzwerken.

**Vektorraummodell (vector space model)** Das Vektorraummodell ist ein Modell zur Repräsentation von Informationen in einem hochdimensionalen metrischen Vektorraum. Es wird im Text Mining für die Repräsentation von Wörtern und von Dokumenten eingesetzt; auch ▶ **Embeddings** werden im Vektorraum repräsentiert.

**Verteilung von Wörtern im Text (word distribution)** Unter Verteilung der Wörter im Text versteht man die Häufigkeit der einzelnen Wörter und Wortkombinationen.

**Verfahren, evidenzbasiertes (evidence-based method)** Statistisches Verfahren fürs Text Mining auf Grundlage der Bayesschen Statistik, bei dem Wahrscheinlichkeitsverteilungen verwendet werden, in denen Vorwissen und A-Priori-Annahmen explizit im Modell ausgedrückt werden, beispielsweise Topic-Modelle. Syn.: Bayessche Verfahren.

**Verfahren, frequentistisches (count-based method)** Statistisches Verfahren fürs Text Mining, bei dem die Häufigkeit und statistische Verteilung von Sprachdaten untersucht wird. Hierzu gehören bspw. Korpusvergleiche zur Ermittlung von statistisch signifikanten Unterschieden in der Verwendung von Vokabularen sowie Kookkurrenzanalysen.

**Verfahren, neuronales (neural method)** Text-Mining-Verfahren unter Verwendung von reellwertigen Vektoren für die Repräsentation von Sprachdaten und neuronalen Netzen, bei denen Feature-Repräsentationen für die Lösung einer Aufgabe nicht vorgegeben, sondern vom neuronalen Netz selbst gelernt werden, um eine optimale Lösung zu finden.

**Verfahren, regelbasiertes (rule-based method)** Text-Mining-Verfahren zur Erkennung von Mustern von Wörtern in Texten. Die Muster werden in Form von Regeln definiert, welche es ermöglichen, Wörter und Wortfolgen in Texten zu finden. Hierfür werden eigene, vollständig definierte Kunstsprachen wie z. B. reguläre Ausdrücke definiert.

**Verfahren, statistisches (statistical method)** Ausnutzung sprachstatistischer Gesetzmäßigkeiten für das Text Mining. Abhängig davon, wie der Begriff der Wahrscheinlichkeit gefasst wird, kann zwischen frequentistischen und evidenzbasierten Ansätzen unterschieden werden.

**Verfahren, überwachtes (supervised method)** maschinelles Lernverfahren, welches auf einem Trainingsdatensatz trainiert wird und eine vorgegebene Klassifizierung lernt.

**Verfahren, unüberwachtes (unsupervised method)** maschinelles Lernverfahren, welches auf nicht annotierten Datensätzen operiert, Ergebnis ist typischerweise eine ▶ **Cluster-Analyse**.

**Verkettung (concatenation)** Siehe ▶ **Konkatenation**.

**Viterbi-Algorithmus (Viterbi algorithm)** ist ein effizienter Algorithmus für das Bestimmen der wahrscheinlichsten Sequenz von Labels und wird z. B. bei ▶ **Hidden-Markov-Modellen** und ▶ **Conditional Random Fields** für die ▶ **Sequenzklassifikation** eingesetzt.

**Vokabular (vocabulary)** Das Vokabular eines Textes besteht aus allen Types, die in einem Text auftreten. Tritt ein Type mehrfach im Text auf, fließt er trotzdem nur einmal in das Vokabular ein. Der Umfang des Vokabulars entspricht der Anzahl der in einem Text vorkommenden Types.

**Vokabular, Umfang (size of vocabulary)** Der Umfang des Vokabulars ist gleich der Anzahl der in einem Text vorkommenden Types.

**Vollform (full form)** Das Vorkommen eines flektierten Wortes im Text, vgl. ▶ **Wortform**.

**Vorverarbeitung (preprocessing)** Arbeitsschritt im Text Mining, bei dem Textdaten inkrementell angereichert werden, um sie für Text-Mining-Aufgaben vorzubereiten. Dies fängt mit dem Einlesen der Daten und deren Überführung in eine geeignete interne Repräsentation an und umfasst Arbeitsschritte wie das Erkennen der Sprache, die Textsegmentierung und verschiedene Bereinigungsschritte.

**Wahrscheinlichkeit (probability)** Die Wahrscheinlichkeit eines Ereignisses ist nach frequentistischer Auffassung die relative Häufigkeit, mit der es in einer großen Anzahl gleicher, wiederholter Experimente auftritt. Die relative Häufigkeit eines Wortes in einem Text ist gleich dem Quotienten aus der (absoluten) Häufigkeit dieses Wortes und der Gesamtzahl der Wörter im Text. Betrachtet man immer längere, gleichartige Texte, so nähert sich diese relative Häufigkeit einem konstanten Wert, der Wahrscheinlichkeit dieses Wortes.

**Wahrscheinlichkeit, bedingte (conditional probability)** Die bedingte Wahrscheinlichkeit  $P(A|B)$  gibt die Wahrscheinlichkeit des Ereignisses A an, unter der Voraussetzung, dass das Ereignis B bereits eingetreten ist.

**Web (web; World Wide Web)** Informationssystem im Internet, das auf der Hypertext-Technik basiert; ermöglicht außerdem den Zugriff auf die anderen Internet-Dienste. Der Zugang zum Web erfolgt über **Web-Browser**. Syn.: W3, WWW.

**Web-Browser (web browser)** Software, über die Benutzer die Dienstleistungen des Internets, insbesondere des Webs, in Anspruch nehmen können. Durch Angabe der URL wird das Computersystem, das die jeweilige Dienstleistung anbietet, eindeutig adressiert. Syn: Browser.

**Webcrawler** Siehe ▶ **Crawler**.

**Webseite (web page)** Ein Internet-Dokument mit Text- und multimodalen Inhalten und ▶ **Hyperlinks** zur Navigation.

**Whitespace (white space)** Sammelbegriff für verschiedene unsichtbare ▶ **Zeichen** (z. B. Leerzeichen, Tabulator, Zeilenumbruch), die nur als Leerraum dargestellt werden und typischerweise zur Trennung von ▶ **Wörtern(Tokens)** oder Formatierung von Texten verwendet werden.

**Wildcard (wildcard)** In regulären Ausdrücken und in Anfragesprachen sind Wildcards Zeichen, welche stellvertretend für andere Zeichen stehen.

**Wissen (knowledge)** Wissen ist die meist auf Erfahrung beruhende und objektiv nachprüfbare Kenntnis von Fakten und Zusammenhängen eines Weltausschnitts, die

Personen zur Lösung von Problemen einsetzen. Wissen ermöglicht die Vernetzung von Informationen.

**Wissensmanagement (knowledge management)** Wissensmanagement ist eine Organisationsform, die das Wissen der in einem Unternehmen beschäftigten Personen, das für einen Erfolg des Unternehmens relevant ist, erfasst, strukturiert und zum Nutzen des Unternehmens einsetzt.

**Wohlgeformtheit (well-formedness)** Eine ▶ XML-Datei heißt wohlgeformt, wenn sie entsprechend den Regeln von ▶ XML aufgebaut ist.

**Wort (word)** Ein Wort im engeren Sinne ist eine Äquivalenzklasse von Wortformen (z. B. alle flektierten Formen des Wortes *sprechen*). Im weiteren Sinne kann Wort für ▶ Token, ▶ Type, ▶ Term, ▶ Wortform und ▶ Grundform verwendet werden.

**Wortart (part of speech)** Siehe ▶ Parts of Speech.

**Wortartentagging (part of speech tagging)** Siehe ▶ Part-of-Speech-Tagging.

**Wortbedeutungsdisambiguierung (word sense disambiguation)** Bezeichnet das Auflösen lexikalischer Mehrdeutigkeiten von Wörtern im Kontext; eine der Bedeutungen wird explizit für das Wortvorkommen zugewiesen.

**Wortform (word form)** Eine Wortform ist die flektierte Erscheinungsform eines Wortes, wie sie in einem syntaktischen Kontext vorkommt.

**Wortklasse, geschlossene (closed word class)** Eine geschlossene Wortklasse ist eine Klasse von Wörtern, die nur in sehr begrenztem Maße Veränderungen unterliegt. Bei Wörtern aus geschlossenen Wortklassen steht ihre grammatische Bedeutung im Vordergrund. Sie werden auch als Funktionswörter bezeichnet. Geschlossene Wortklassen: Artikel, Hilfsverb, Interjektion, Konjunktion, Numeral, Präposition, Pronomen.

**Wortklasse, offene (open word class)** Eine offene Wortklasse ist eine Klasse von Wörtern, die Veränderungen durch die morphologischen Prozesse Flexion, Derivation und Komposition unterliegt. Offene Wortklassen: Adjektiv, Nomen, Verb, Adverb.

**Wort-Kontext-Matrix (word-context matrix)** Anordnung von ▶ Wörtern (▶ Types) und ihren Kontexten in Form einer Matrix, aus der ersichtlich ist, in welchen Kontexten ein Wort vorkommt bzw. welche Kontexte welche Wörter enthalten.

**Wortnetz (word net)** Ein Wortnetz ist ein Graph mit Wörtern als Knoten. Die Kanten beschreiben Beziehungen zwischen den entsprechenden Wörtern und können manuell oder automatisch erzeugt worden sein. Im Falle von signifikanten Kookkurrenzen beschreiben die Kanten häufig syntagmatische und paradigmatische Zusammenhänge.

**Wort-n-Gramm (token n-gram)** Bezeichnung für n aufeinanderfolgende Wörter. Siehe ▶ n-Gramm.

**XML (Extensible Markup Language)** Sprache, mit der sich Auszeichnungssprachen definieren lassen. XML ist eine verkürzte Version der Standard Generalized Markup Language (SGML). Die Version 1.0 von XML wurde im November 1998 vom WWW Consortium (W3C) verabschiedet.

**Zeichen (character)** Ein Zeichen ist ein Element eines endlichen geordneten Zeichenvorrats.

**Zeichenkette (string of character)** Eine Zeichenkette entsteht durch die Aneinanderfügung (Konkatenation) von Zeichen.

**Zeta (Zeta)** Verfahren in der Stylometrie, welches auf dem ▶ **t-Test** aufbaut und diesen verfeinert. Das Grundprinzip dieses Maßes ist es, vor der Berechnung die Texte in kleinere Segmente aufzuteilen, wobei die Segmentlänge ein wichtiger Parameter ist. Dann wird für jedes Merkmal der Anteil der Segmente erhoben, in denen das Merkmal mindestens einmal vorkommt (die „document proportion“). Von diesem Anteil in der untersuchten Gruppe wird der entsprechende Anteil in der Vergleichsgruppe subtrahiert, woraus sich der Zeta-Wert ergibt.

**Zielfunktion (loss function)** Die Zielfunktion gibt in neuronalen Netzen die Abweichung vom gewünschten Ergebnis an, das kann etwa durch die Euklidische Distanz passieren. Die Zielfunktion wird im Rahmen von Gradientenverfahren als Basis verwendet um, mit Hilfe von ▶ **Backpropagation**, die Gewichte des Netzes anzupassen.

**Zipfsches Gesetz (Zipf's law)** Bezeichnung für den von George Kingsley Zipf formulierten Zusammenhang: Wenn Wörter aus einem Korpus ihrer Häufigkeit nach absteigend dass für ein geordnet sind, ergibt das Produkt aus dem Rang eines Wortes (innerhalb der häufigkeitssortierten Liste) mit seiner Häufigkeit für alle Wörter in etwa denselben Wert. Die häufigsten Wörter, also die Wörter, die in der häufigkeitssortierten Liste den niedrigsten Rang haben, sind meist sehr kurze Funktionswörter/▶ **Stoppwörter**.

**Zufallsstichprobe, geschichtete (stratified sampling)** Im Gegensatz zur reinen Zufallsstichprobe stellt die geschichtete Zufallsstichprobe sicher, dass die Verteilung der Kategorien der in der Probe enthaltenen Instanzen die Verteilung in der Grundgesamtheit approximiert. Wird oft zur Erstellung von ▶ **Trainings-** und ▶ **Validationsmengen** bei der Klassifikation eingesetzt.

---

# Stichwortverzeichnis

## A

Accuracy, 296  
Affektwörterbuch, 324  
Allomorph, 41  
Alphabet, 13, 23  
Analysekorpus, 4, 242  
Annotation, 3, 85  
Annotator, 88  
Ansatz, korpusbasierter, 131  
Ansatz  
  evidenzbasierter, 3  
  frequentistischer, 3  
Antonym, 60  
Äquivalenzklasse, 11  
ASCII, 14  
Asset, 14  
Attention, 225  
Attention-Mechanismus, 294  
Attribut  
  extrinsisches, 167  
  intrinsisches, 167  
Ausdruck, regulärer, 3, 43, 76, 331  
Automat, endlicher, 45  
Automatentheorie, 29

## B

Backpropagation, 83, 221  
Bag-of-Words, 283  
Bag-of-Words-Modell, 215, 218, 325  
Basismorphem, 40  
Bayesscher Ansatz, 29  
Bayessche Statistik, 3  
Big Data, 8  
BiLSTM, 294

## BIO-Schema, 295

Bootstrapping, 316  
Buchstabe, 36

## C

Chomsky-Grammatiken, 28  
Chunk, 102  
Chunking, 102  
Clusteranalyse, 3  
Clustering, 6, 261, 277, 331  
Codepage, 90  
Cohens Kappa, 287  
Compiler, 53  
Content, 14  
Content-Management-System, 90, 154  
Continuous Bag-of-Words (CBOW) Modell, 220  
Convolutional Neural Network, 83  
Crawler, 137  
Crawlingstrategie, 135

## D

Data Mining, 2  
Daten, 13  
Deklination, 41  
Dendrogramm, 265  
Dependenz, 103  
Dependenz-Struktur, 47  
Derivation, 41  
Derivative, 40  
Development-Menge, 281  
Differenzanalyse, 242, 316  
Dirichlet-Verteilung, 273

Disambiguierung, 84  
 Dispersionsmaß, 244  
 Distanzmaß, 262  
 Distributionelle Hypothese, 208  
 Dokumentfrequenz, 246  
 Dokumentkorpus, 134  
 Drift, 282  
 Dublin Core, 11

**E**

Early-Stopping-Methode, 282  
 Ebene, linguistische, 23, 37, 85  
 Eigenname (Named Entity), 6  
 Eigenname, 196  
 Eigennamenerkennung (Named Entity Recognition), 106, 170  
 Embedding, 57, 83, 217, 247  
     kontextualisiertes, 84  
 End-to-End-System, 89  
 Entity Linking, 106  
 Escape-Sequenz, 77  
 Evaluationsmaß, 295  
 Explosion, kombinatorische, 88  
 Extension, 57  
 Extraktion, 160

**F**

Facettensuche, 317  
 Fachausdruck, 6, 63  
 Fachtext, 62  
 Feed-Forward-Netz, 82  
 Fehlerfortpflanzung, 89  
 Fine-Tuning, 285, 302  
 Fleiss' Kappa, 287  
 Flexion, 41  
 Flexive, 40  
 Fügung, feste, 196  
 F-Wert, 277, 296

**G**

Gated Recurrent Units, 82  
 Gazetteer, 106, 164  
 Gegensatz, 60  
 Gibbs Sampling, 274  
 Gold-Daten, 295  
 Grammatik, probabilistische kontextfreie, 51

Ground Truth, 15  
 Grundform, 12, 44  
 Grundformreduktion, 44  
 Gültigkeitsregel, 24

**H**

Hashverfahren, 141  
 Häufigkeit, relative, 214  
 Häufigkeitsklasse, 172, 339  
 Heuristik, 92  
 Hidden-Markov-Modell, 292  
 Hintergrundkorpora, 84  
 HTML (Hypertext Markup Language), 90  
 HTML-Dokument, 137, 331  
 Hyperparameter, 75, 281  
 Hypothesentest, 243

**I**

Information, 2, 14  
 Information Retrieval, 2, 328  
 Informationsextraktion, 74  
     template-basierte, 113  
 Instanz, 74  
 Interrater-Reliabilität (Inter-Annotator-Agreement, IAA), 287

**J**

Join, 156, 159  
 JSON, 154

**K**

Klassifikation, 3, 74  
 Klassifikator, 279, 325  
 Kohyponym, 58  
 Komplementärbegriff, 60  
 Komponente, 85  
 Komposita, 314  
 Komposition, 41  
 Kompositionsprinzip, 56  
 Kompositum, 43  
 Konfidenz, 79  
 Konjugation, 41  
 Konkatenation, 23  
 Konstituente, 48  
 Konstituentenparsing, 105

Konstituenten-Struktur, 47  
Kontext, 37  
  globaler, 38  
  lokaler, 37  
Kontextvolatilität, 249, 339  
Konverse, 61  
Kookkurrenz, 29, 247, 338  
  signifikante, 196  
Kookkurrenzanalyse, 3  
Kookkurrenzmatrix, 221  
Koreferenzauflösung, 108  
Korpusvergleich, 3, 331  
Kosinusähnlichkeit, 222, 249, 262  
Kosinus-Maß, 340  
Kreuzvalidierung, 281

**L**

Lemma, 12, 168  
Lernverfahren  
  maschinelles, 74  
  unüberwachtes, 29  
Levenshtein-Distanz, 262  
Lexikologie, 24  
Lexikon, 25  
Liste, inverse, 157  
Log-Likelihood-Maß, 198, 331  
Log-Likelihood-Ratio, 243, 313  
Long Short-Term Memories, 82

**M**

Makro-Sicht, 6  
Markdown, 90  
Markierung, 325  
Maschinelles Lernen, 29  
Mehrdeutigkeit, 25  
Mehrworteinheit, 331  
Merkmal (Feature), 4, 74, 258  
Metadaten, 11, 133, 317  
Mikro-Sicht, 6  
Modell, distributionales semantisches, 84  
Morphem, 23, 36, 40  
Morphologie, 24, 40  
Multitask-Lernen, 302  
Mundart, 203  
Muster, 27, 150  
Mutual Information, 265

**N**  
Nachricht, 13  
Natural Language Processing (NLP), 2  
Netz, lexikalisch-semantisches, 164  
Netzwerk  
  neuronales, 285  
  rekurrentes, 83  
Netzwerktopologie, 82  
Neuron, 82  
News Monitoring, 327  
N-Gramm-Modell, 213  
nicht-transparent, 171  
Nominalphrase, 202

**O**

Oberbegriff, 58, 315  
OCR, 91  
Office-Format, 90  
One-Hot-Kodierung, 220  
Optical Character Recognition (OCR), 4  
out of vocabulary (OOV), 46, 214  
Overfitting, 259, 282

**P**

paradigmatisch, 36  
Parlsen, 47  
  semantisches, 110  
Parsing, 101  
Part-of-Speech Tagging, 331  
PDF (Portable Document Format), 91  
Perplexität, 215, 265  
Perzeptron, 82  
Phrase, 23, 36  
Phrasen-Struktur-Grammatik, 48  
Pointer-Generator-Network, 83  
Pointwise Mutual Information (PMI), 201  
Polarität, 322  
Porter-Stemmer, 45  
POS-Muster, 316  
POS-Tag, 50  
Precision, 296

**R**

Rang, 190  
  höchster, 192

Ranking, 298  
Recall, 296  
Rechtschreibreform, 332  
Redundanz, 14  
Referenzkorpus, 4, 242  
Relation, semantische, 6  
ROC-Kurve, 297

## S

Satz, 23, 36, 136, 147  
Satzkorpus, 134  
Semantik, 53, 56  
Sentimentanalyse, 7, 322  
    aspekt-basierte, 322  
Sentiment-Orientierung, 324  
Seq2Seq, 83  
Sequenzklassifikation, 291  
Signal, 13  
    “schwaches“, 339  
Signifikanzmaß, 197  
Skalierung, 114  
Skip-Gram Modell, 220  
Sliding-Window, 340  
Softmax-Funktion, 216  
Spam, 90  
Spannenannotation, 86  
Sprachdynamik, 25  
Sprache, formale, 24, 47  
Spracherkennung, 30  
Sprachklassifikation, 331  
Sprachmodell, 75  
Sprachstatistik, 25  
Stamm, 40  
Stammformreduktion, 44  
Stemming, 101  
Stimulus-Response-Experiment, 200  
Stopwort, 62, 188  
Strukturalismus, 36, 220  
Stylometrie, 244  
Subwort-Tokenisierung, 99  
Synonym, 59  
    “schwaches“, 167  
syntagmatisch, 36  
Syntax, 24, 46  
System, hybrides, 75

## T

Tagset, 85  
Tanimoto-Ähnlichkeit, 201  
Taxonomie, 315  
TEI, 154  
Term-Dokument-Matrix, 209  
Term-Frequenz, 246  
Terminologie, 63  
Terminologielehre, 63  
Testmenge, 281  
Text, 1  
Textabdeckung, 182  
Textkorpus, 6  
Text Mining, 2, 189  
tf·idf, 243, 313  
Thesaurus, 63  
Token, 11, 26, 96  
Tokenisierung, 11  
Tokenregel, 11, 26  
Topic-Modell, 4, 29, 270, 330, 342  
Top-Level-Domain, 139  
Trainingsdaten, 167  
Trainingsmenge, 74, 281  
Transduktor, 78  
Transformation, 48  
Transformer-Architektur, 225  
Transformer-Network, 83  
Translation Memories, 30  
TREC-Konferenz, 328  
Trend, 327  
t-Test, 246  
Type, 11  
Type-Token-Verhältnis, 11, 180

## U

Übersetzung, automatische, 178  
Unicode, 90, 137  
Unterbegriff, 58  
Urheberrecht, 16  
UTF-8, 90, 137

## V

Valenzstruktur, 162  
Vektorraummodell, 208

Verarbeitungspipeline, 330

Verfahren, 323

frequentistisches, 29

neuronales, 4, 30, 73

regelbasiertes, 3, 73

statistisches, 3, 73, 313

Viterbi-Algorithmus, 293

V-Modell, 8

Vokabular, 3, 189, 193

Vorverarbeitung, 312

linguistisches, 24

Wohlgeformtheitsregel, 24

Wort, 12, 36

Wortbedeutungsdisambiguierung, 109

Wortform, 12, 23, 40

Wort-Kontext-Matrix, 218

Wort-n-Gramm, 29

## X

XML, 92

## W

Webbrowser, 90

Webcrawler, 154

Webseite, 90

Whitespace, 93

Wildcard, 27, 77, 156

Wissen, 14

## Z

Zeichen, 13

Zeta, 247

Zielfunktion, 74

Zipfsches Gesetz, 26, 189, 190

Zufallsstichprobe, 281