Dominik Kuropka

Modelle zur Repräsentation natürlichsprachlicher Dokumente

Ontologie-basiertes Information-Filtering und -Retrieval mit relationalen Datenbanken

Advances in Information Systems and Management Science

Publikationsjahr: 2004

ISBN 3-8325-0514-8
ISSN 1611-3101

Geleitwort

Die Verfügbarkeit von digitalen Informationen hat in den letzten Jahren stark zugenommen. Dieses wurde sowohl durch kostengünstige Massenspeicher als auch durch die rapide Entwicklung des Internets und der damit verbundenen weltweiten Vernetzung von Rechnern vorangetrieben. Somit ist es heutzutage relativ leicht auch auf große Mengen von natürlichsprachlichen Informationen zurückzugreifen und diese für Zwecke der Entscheidungsfindung, Forschung, Lehre oder lediglich zum Vergnügen zu verwenden. Die Kehrseite diese Medallie ist jedoch, dass aufgrund der großen Anzahl an verfügbaren Informationen die Suche nach oder das Herausfiltern der „richtigen“ Informationen zu einer zunehmend schwierigen Aufgabe geworden ist. Anwendungen und Verfahren, die die genannten Problemstellungen im Bereich der natürlichsprachlichen Informationen angehen, werden unter den Begriffen Information-Retrieval und Information-Filtering subsumiert.

Zur computergestützten Lösung der oben genannten Problemstellungen sind Modelle zur Repräsentation der natürlichsprachlichen Informationen erforderlich, anhand derer die Kriterien zum Filtern bzw. Suchen von „relevanten“ Informationen formal definiert werden können. Gängige in der Praxis eingesetzte Verfahren zeichnen sich im Allgemeinen durch die Verwendung von Modellen aus, die stark vereinfachend natürlichsprachliche Dokumente lediglich als eine Menge von voneinander unabhängigen Worten repräsentieren. Derartige Verfahren eignen sich für eine stichwortbasierte Filterung bzw. Suche, wie sie von den gängigen Suchmaschinen angeboten werden. Allerdings erreicht die Qualität der Such- bzw. Filterergebnisse häufig nicht das erwünschende Niveau. Insbesondere linguistische Phänomene wie z. B. Synonymie, Homographie oder thematische Beziehungen zwischen den einzelnen Wörtern werden von diesen Verfahren nicht berücksichtigt, was häufig zu falschen Ergebnissen führt. Neuere Verfahren versuchen daher die linguistischen Phänomene in den Modellen implizit zu erfassen, indem sie Wörter über die Häufigkeit ihres gemeinsamen Vorkommens in Dokumenten zueinander in Beziehung setzen. In der Praxis haben sich derartige Verfahren jedoch bis heute nicht durchsetzen können, weil bei der Qualität der Ergebnisse kein hinreichend großer Hinzugewinn belegt werden konnte, der den zusätzlichen Rechenaufwand gerechtfertigen würde. Dominik Kuropka untersucht und bewertet in seiner Arbeit eine Vielzahl von Modellen im Hinblick auf die Repräsentation von linguistischen Phänomenen und findet eine Erklärung, warum die neueren Verfahren, die linguistische Phänomene implizit zu erfassen versuchen, nicht den gewünschten Erfolg gebracht haben. Dabei zieht er den Schluss, dass nur

eine explizite Erfassung der linguistischen Phänomene zu dem gewünschten Erfolg führen kann und entwickelt daraufhin ein Modell (das enhanced Topic-based Vector Space Model), welches linguistische Phänomene unter Verwendung von Ontologien explizit abzubilden vermag.

Die Arbeit von Dominik Kuropka gibt nicht nur eine umfassende Übersicht über den Themenbereich und bekannte Ansätze des Information-Filtering und -Retrieval, sondern zeigt auch neue Ansätze und Modelle zur Lösung der beiden Problemstellungen und bringt somit neue Impulse in die wissenschaftliche Diskussion ein. Aufgrund der Vielzahl an Beispielen und Quelltextauszügen eignet sich diese Arbeit nicht nur für Studenten und Wissenschaftler, sondern auch für den Praktiker, der das enhanced Topic-based Vector Space Model für die Informationssuche oder -filterung in bestehende oder neue Anwendungen integrieren möchte.

Prof. Dr. Jörg Becker

Vorwort

Die Motivation zu dieser Arbeit entspringt der praktischen Problemstellung vor der ich zu Beginn meiner Tätigkeit im Jahre 1999 als wissenschaftlicher Mitarbeiter am Institut für Wirtschaftsinformatik der Westfälischen Wilhelms-Universität Münster stand: Wie decke ich mit einem möglichst geringen zeitlichen Aufwand meinen kontinuierlichen Bedarf an relevanten und aktuellen Informationen zu Entwicklungen und Ereignissen in den Bereichen meines Interessengebietes und Aufgabenfeldes ab? Ausgehend von dieser, zu Beginn doch recht persönlichen Problemstellung, haben mein Kollege Thomas Serries und ich während unseres dreiwöchigen Aufenthaltes als Dozenten an der Universität Tartu (Estland) u. a. darüber philosophiert, wie dieses Problem mit Hilfe einer geeigneten Software angegangen werden kann.

Das Ergebnis unserer Gedankenexkurse war die Idee zur Entwicklung eines Forschungsprototypen, eines Persönlichen Informations Agenten (π-*Agent*), der im Rahmen von insgesamt drei Projektseminaren und einigen Diplomhausarbeiten in mehreren unterschiedlichen Varianten entwickelt wurde. Ziel des Forschungsprototypen war es durch (Feld-)Versuche Erfahrungen zu sammeln, welche Verfahren zur Lösung der oben genannten Problemstellung geeignet sind und wie eine Oberfläche zu diesen Verfahren zu konzipieren ist, so dass sie einfach und intuitiv zu bedienen ist.[1] Der π-*Agent* wurde in drei aufeinanderfolgenden Jahren auf den CeBIT-Messen in Hannover vorgestellt und ist auf eine breite Resonanz sowohl beim Publikum als auch bei der Presse gestoßen. Diese Resonanz war es, die mich darin bestätigte, dass die von mir behandelte Problemstellung sich nicht nur mir, sondern auch vielen Anderen in den unterschiedlichsten Anwendungsbereichen der Forschung, Wirtschaft und Verwaltung stellt. Das hat mich darin bestärkt mich weiter dem Problemgebiet des Information-Filtering und -Retrieval zu widmen und diese Arbeit zu schreiben.

Die vorliegende Arbeit wurde als Dissertation von der Wirtschaftswissenschaftlichen Fakultät der Universität Münster angenommen. Mein besonderer Dank gilt meinem akademischen Lehrer und Doktorvater, Prof. Dr. Jörg Becker, nicht nur für seine fachlichen Hinweise und Anleitung, sondern insbesondere auch für das produktive Umfeld am Lehrstuhl und die Möglichkeit zur Promotion. Prof. Dr. Ulrich Müller-Funk danke ich für die Übernahme des Zweitgutachtens und für die kritischen, mich anspornenden, Diskurse zu meiner Arbeit.

[1] Aus diesem Grunde stand der π-*Agent* mehrere Jahre jedem, über das Internet frei zugänglich, unter der folgenden Adresse zur Verfügung: `http://www.pi-agent.com`

Des Weiteren Danke ich meinen Kollegen und insbesondere Thomas Serries und Ralf Knackstedt für die Zeit, die sie für fruchtbare Diskussionen mit mir aufgebracht haben. Ebenso bedanke ich mich bei den Studenten für die in den Projektseminaren geleistete Arbeit, besonders möchte ich mich für das große Engagement der Studenten Felix Wortmann, Christian Czarnecki und Kai Pastuch bedanken. Weiterer Dank geht an Sonja Hillebrand vom Arbeitsbereich Linguistik, für die anregenden Diskussionen zu Fragen der Linguistik. Für die „Bekämpfung“ der diversen Rechtschreib- und Grammatikfehler bedanke ich mich bei Ursula Essmann und Monika Rohe-Al Torman. Ebenfalls für erwähnenswert halte ich Prof. Dr. Mathias Weske, der mich und die anderen Studenten während meiner Studienzeit bei meinem Projektseminar und meiner Diplomhausarbeit vorbildlich betreut hat und von dem ich vieles gelernt habe, was mir bei meiner Tätigkeit als Wissenschaftler und Dozent später von Nutzen war.

Ein weiteres Fundament für diese Arbeit waren die beständigen und liebevollen Aufmunterungen meiner geliebten Ehefrau Kristina Kuropka. Besonderer Dank gebührt zudem meinen Eltern, Halina und Ludwig Kuropka, die mir während meiner Ausbildung und darüber hinaus stets Unterstützung in jeglicher Form gewährt haben.

Dominik Kuropka

Inhaltsverzeichnis

Geleitwort **i**

Vorwort **iii**

Abbildungsverzeichnis **xi**

Tabellenverzeichnis **xiii**

Abkürzungsverzeichnis **xv**

Symbolverzeichnis **xvii**

1 Einleitung **1**

1.1 Problemstellung 1

1.2 Zielsetzung der Arbeit 3

1.3 Aufbau der Arbeit 4

2 Grundlegende Definitionen und Methoden **7**

2.1 Information-Filtering und -Retrieval 7

2.1.1 Information-Retrieval 7

2.1.2 Information-Filtering 10

2.1.3 Gemeinsamkeiten und Unterschiede 11

2.1.4 Modell der Repräsentation von Dokumenten 12

2.1.5 Modell der Interaktion mit dem Benutzer 13

2.2 Datenmodelle 14

2.2.1 Entity-Relationship-Modelle 16

2.2.2 Relationale Datenbanken und SQL 19

2.3 Computerlinguistik 20
2.3.1 Phonologie 20
2.3.2 Morphologie 21
2.3.2.1 Flexion, Komposition und Derivation 21
2.3.2.2 Stemming (Normalisierung) 21
2.3.3 Syntax 23
2.3.3.1 Syntaktische Strukturen und formale Grammatiken 23
2.3.3.2 Automatisierte Analyse syntaktischer Strukturen 26
2.3.4 Semantik 29
2.3.4.1 Satz- und Diskurssemantik 29
2.3.4.2 Lexikalische Semantik 30
2.3.5 Pragmatik 33
2.3.6 Bedeutung für IF und IR 35
2.4 Ontologien 35
2.4.1 Ontologie-Modellierungssprachen 38
2.4.2 Anwendungsmöglichkeiten für IF- und IR-Systeme 41

3 Gängige IF/IR-Modelle 43
3.1 Fundamentale Konzepte 45
3.2 Modelle ohne Terminterdependenzen 49
3.2.1 Standard Boolean Model (SBM) 49
3.2.2 Vector Space Model (VSM) 50
3.2.3 Extended Boolean Model (EBM) 52
3.2.4 Binary Independence Retrieval (BIR) 54
3.2.5 Inference Network Model (INM) 56
3.2.6 Belief Network Model (BNM) 59
3.2.7 Language Model (LM) 60
3.3 Modelle mit immanenten Terminterdependenzen 63
3.3.1 Generalized Vector Space Model (GVSM) 68
3.3.2 Latent Semantic Index (LSI) 69
3.3.3 Spreading Activation Neuronal Network (SANN) 70
3.4 Modelle mit transzendenten Terminterdependenzen 72
3.4.1 Fuzzy Set Model (FSM) 73
3.4.2 Retrieval by Logical Imaging (RbLI) 76
3.4.3 Backpropagation Neuronal Network (BNN) 79

3.5 Bewertung der gängigen Modelle 81

4 Topic-based Vector Space Model (TVSM) 87

4.1 Motivation 87
4.2 Konzept 88
4.2.1 Vektorraum, Terme und Dokumente 89
4.2.2 Dokumente und Dokumentenähnlichkeiten 90
4.2.3 Berechnung der Dokumentenähnlichkeiten 93
4.2.4 Implementierung mit einer relationalen Datenbank 94
4.2.5 Einstellen neuer Dokumente / Durchführen von Anfragen 98
4.3 Stoppwort-Lemma 99
4.4 Stemming-Lemma 101
4.5 Synonym-Lemma 103
4.6 Vergleich mit anderen Modellen 104
4.7 Kritik am TVSM 107

5 Enhanced TVSM (eTVSM) 109

5.1 Konzept 110
5.1.1 Paarweise Themen-Ähnlichkeiten 115
5.1.1.1 Problemstellung 115
5.1.1.2 Repräsentationsform für Themenstrukturen 116
5.1.1.3 Herleitung der Themen-Ähnlichkeiten 120
5.1.1.4 Eigenschaften der Vektoren und Ähnlichkeiten 123
5.1.2 Interpretationen und ihre Beziehungen 128
5.1.2.1 Herleitung der Interpretations-Ähnlichkeiten 129
5.1.2.2 Repräsentation der (Totalen) Synonymie 131
5.1.2.3 Repräsentation der Homographie 132
5.1.2.4 Repräsentation der Partiellen Synonymie 134
5.1.2.5 Repräsentation der Metonymie 136
5.1.2.6 Definition der Dokumenten-Ähnlichkeiten 137
5.1.3 Wortstamm-Term-Zuordnung 139
5.1.4 Wort-Wortstamm-Zuordnung 140
5.2 Das eTVSM und der Ontologie-Begriff 143
5.2.1 Eine grafische Ontologie-Repräsentation für das eTVSM 143
5.2.2 Eine Beispiel-Ontologie 144

5.3 Umsetzung mit einer relationalen Datenbank 148
5.3.1 Datenmodell 148
5.3.2 Initialisierung 151
5.3.2.1 Initialisierung der Themenblätter 151
5.3.2.2 Initialisierung der Themenknoten 155
5.3.2.3 Ähnlichkeiten zwischen Themen 157
5.3.2.4 Skalarprodukte zwischen Interpretationen 158
5.3.3 Einstellen neuer Dokumente 159
5.3.3.1 Parsing 160
5.3.3.2 Zuordnung zu Interpretationen 162
5.3.3.3 Berechnung der Dokumentenbeträge 166
5.3.4 Anfrageausführung 166
5.4 Vergleich mit anderen Modellen / Kritik 168

6 Anwendung des eTVSM in der Praxis 171
6.1 Ontologien für das eTVSM 171
6.1.1 Erstellung einer Ontologie 171
6.1.2 Nutzung vorhandener Ontologien 172
6.1.2.1 Wortschatz-Lexikon 172
6.1.2.2 WordNet und GermaNet 174
6.2 Anwendung für das Information-Retrieval 182
6.3 Anwendung für das Information-Filtering 183
6.4 Quantitative Evaluierung 187
6.4.1 Evaluationsmaße 188
6.4.2 Evaluation von IR-Systemen 190
6.4.3 Evaluation von IF-Systemen 191

7 Zusammenfassung 193

A Datenbankeinträge der Beispiel-Ontologie 195
A.1 Vor der Initialisierung bekannt 195
A.1.1 Tabelle `Thema` 195
A.1.2 Tabelle `Themenstruktur` 196
A.1.3 Tabelle `Interpretation` 197
A.1.4 Tabelle `IT_Zuo` 197
A.1.5 Tabelle `Term` 198

A.1.6 Tabelle `TI_Zuo` 199
A.1.7 Tabelle `Supportterm` 200
A.1.8 Tabelle `Wortstamm` 200
A.1.9 Tabelle `WT_Zuo` 201
A.1.10 Tabelle `Wort` 202
A.2 Nach der Initialisierung 203
A.2.1 Tabelle `Themavektor` 203
A.2.2 View `ThemenAehnlichkeit` 205
A.2.3 Tabelle `Aehnlichkeit` 210
A.3 Dokumente einfügen 215
A.3.1 Tabelle `Dokument` vor der Betragsberechnung 215
A.3.2 Tabelle `DW_Zuo` 216
A.3.3 Tabelle `DI_Zuo` 216
A.3.4 Tabelle `Dokument` nach der Betragsberechnung 217
A.4 Dokumentenähnlichkeit 217
A.4.1 Ergebnisse der View `DokAehn` 217

B VSM: simuliert mit den eTVSM-Tabellen 219
B.1 View-Definitionen 219
B.1.1 Anzahl der Terme pro Dokument 219
B.1.2 Dokumentabhängige Termgewichte 220
B.1.3 Berechnung der Dokumentenähnlichkeit 221
B.2 View-Ergebnisse 221
B.2.1 View `VSM_a` 221
B.2.2 View `VSM_w` 222
B.2.3 View `VSM_dokaehn` 222

Literaturverzeichnis 225

Index 239

Abbildungsverzeichnis

1.1 Aufbau der Arbeit und empfohlene Lesereihenfolgen. 4

2.1 Ein allgemeines Modell zum Information-Retrieval. 9
2.2 Ein allgemeines Modell zum Information-Filtering. 10
2.3 Bestandteile eines Modells. 15
2.4 Beispiel für ein ERM. 17
2.5 Flexion, Komposition und Derivation im Überblick. 22
2.6 Ableitung zu der Hund bellt. 25
2.7 Ableitung zu der Hund sieht die Katze. 25
2.8 Ableitung zu der Hund bellt die Hund. 26
2.9 Verhältnis von Wörtern und (Wort-)Interpretationen 30
2.10 Beispiel für eine Taxonomie 39
2.11 Beispiel für einen Thesaurus 40
2.12 Ein Beispiel für eine logisch-mathematische Repräsentation einer Ontologie. 40

3.1 Eine Übersicht über gängige Modelle zur Repräsentation von natürlichsprachlichen Dokumenten. 44
3.2 Equidistanz-Linien und Ähnlichkeiten für den Fall, dass nur einer von zwei Termen in einem Dokument vorhanden ist. 53
3.3 Topologie eines Inference Network Model 57
3.4 Beispiel für ein einfaches Belief Network Model 59
3.5 Topologie eines SANN für IR. 71
3.6 Topologie eines BNN für IF/IR. 80

4.1 Veranschaulichung der Interpretation des TVSM-Vektorraums. 89
4.2 Dokumentenähnlichkeit im TVSM grafisch veranschaulicht. 91

4.3 Begründung für positive Achsenabschnitte. 92
4.4 Relationales Datenmodell für das TVSM. 95
4.5 Ein Beispiel für die schwache Transitivität beim TVSM. 106

5.1 Relationales Datenmodell für das eTVSM. 111
5.2 Konstrukte des eTVSM und ihre Bezug zu linguistischen Phänomenen. . . . 112
5.3 Transaktionen und ihre Ein-/Ausgabedaten. 114
5.4 Ein Beispiel für eine Themenstruktur. 117
5.5 Themen und ihre rekursive Strukturbeziehung. 117
5.6 Ein abstraktes Beispiel für eine Themenstruktur. 120
5.7 Themenstruktur, Themenvektoren und Themenähnlichkeiten. 124
5.8 Ähnlichkeitsmatrizen verschiedener Themenstrukturen. 126
5.9 Struktur und Ähnlichkeitsmatrix von Wasser, Eis und Schnee. 127
5.10 Die Term-Interpretation- und Supportterm-Zuordnung. 129
5.11 Beispiele für synonyme Terme und ihre Zuordnungen. 131
5.12 Der Homograph Maus und seine Zuordnungen. 133
5.13 Ableitung von Supporttermen aus einer beispielhaften Themenstruktur zu dem Term Maus. 135
5.14 Beispiel für partiell synonyme Terme und ihre Zuordnungen. 136
5.15 Auflösung von Metonymie für den Term Berlin. 137
5.16 Die Wortstamm-Term-Zuordnung und weitere, von der Repräsentation von Wortgruppen indirekt betroffene Entitäten/Beziehungen. 140
5.17 Beispiel für Beziehungen zwischen Wortstämmen und Termen. 141
5.18 Die Wort-Wortstamm-Zuordnung für das Stemming. 141
5.19 Beispiel für Zuordnungen zwischen Worten und Wortstämmen. 142
5.20 Die Elemente der Sprache. 144
5.21 Darstellung der (Totalen) Synonyme aus Abbildung 5.11. 145
5.22 Darstellung der Homographen aus Abbildung 5.12. 145
5.23 Darstellung der Metonyme aus Abbildung 5.15. 145
5.24 Darstellung der Partiellen Synonyme aus Abbildung 5.14. 146
5.25 Beispiel für eine Ontologie. 147
5.26 Vergleich der Dokumentenähnlichkeiten zwischen eTVSM und VSM. 169

6.1 Die Webseite des Wortschatz-Lexikons. 173
6.2 Online-Zugriff auf das WordNet. 174
6.3 eTVSM-Ontologie zu den GermaNet-Beispielen. 180

Tabellenverzeichnis

3.1 Co-Occurrenzen einiger Terme im WWW. 65
3.2 Bewertung der Modelle in Bezug auf die Abbildung von linguistischen Phänomenen. 82

5.1 Ähnlichkeitsmatrix zur Themenstruktur aus Abbildung 5.4. 119
5.2 Ähnlichkeitsmatrix zur Themenstruktur aus Abbildung 5.6. 123

6.1 Zuordnungsmatrix zwischen Wörtern und Interpretationen. 175
6.2 Ähnlichkeitsmatrix zu der Ontologie aus Abbildung 6.3. 181

Abkürzungsverzeichnis

ARIS	Architektur integrierter Informationssysteme
ASCII	American Standard Code for Information Interchange
BIR	Binary Independence Retrieval
BNM	Belief Network Model
BNN	Backpropagation Neuronal Network
bspw.	beispielsweise
DIN	Deutsches Institut für Normung
d. h.	das heißt
EBM	Extended Boolean Model
ERM	Entity-Relationship-Model
eTVSM	enhanced Topic-based Vector Space Model
evtl.	eventuell
FSM	Fuzzy Set Model
GVSM	Generalized Vector Space Model
HTML	Hypertext Markup Language
Hypon.	Hyponymie
IF	Information-Filtering
INM	Inference Network Model
IP	Internet Protocol
IR	Information-Retrieval

ISO	International Standards Organization
Komp.	Komposition
LM	Language Model
LSI	Latent Semantic Index
ML4UM	Machine Learning for User Modeling
RbLI	Retrieval by Logical Imaging
RFC	Request for Comments
RTF	Rich Text Format
SANN	Spreading Activation Neuronal Network
SBM	Standard Boolean Model
SQL	Standard Query Language
SVD	Singular Value Decomposition
tf-idf	term frequency – inverse document frequency
TVSM	Topic-based Vector Space Model
u. a.	unter anderen
vgl.	vergleiche
vs.	versus
VSM	Vector Space Model
WWW	World-wide Web
z. B.	zum Beispiel

Symbolverzeichnis

$=$	Gleichheit (ist gleich zu)
$\neq$	Ungleichheit (ist ungleich zu)
$>$	Vergleichsoperator (größer als)
$<$	Vergleichsoperator (kleiner als)
$\leq$	Vergleichsoperator (kleiner als oder gleich zu)
$\geq$	Vergleichsoperator (größer als oder gleich zu)
$\sim$	Proportionalität (verhält sich proportional zu)
$:=$	Zuweisung (im Sinne einer imperativen Programmierung)
$:$	Bedingung (für die gilt)
$\in$	Elementbeziehung (ist Element von)
$\subseteq$	unechte Teilmenge (ist Teilmenge von oder gleich zu)
$\subset$	echte Teilmenge (ist Teilmenge von)
$\cap$	Durchschittsmenge (geschnitten mit)
$\cup$	Vereinigungsmenge (vereinigt mit)
$\#M$	Kardinalzahl (Elementzahl der Menge M)
$\wp(M)$	Potenzmenge zu M
$\times$	Kreuzprodukt
$\mathbb{N}$	Menge der natürlichen Zahlen
$\mathbb{Z}$	Menge der ganzen Zahlen
$\mathbb{R}$	Menge der reellen Zahlen

$\{\}$	Nullmenge (leere Menge)
$\{a, b\}$	Mengendefinition (Zusammenfassung von a und b zu einer Menge)
$\{a; b\}$	Mengendefinition (Zusammenfassung von a und b zu einer Menge)[2]
$\{a : b\}$	Mengendefinition (Zusammenfassung aller a zu einer Menge, für die b gilt)
$[a...b]$	Mengendefinition (Intervall von a bis b inklusive a und b)
$\setminus$	Differenzmenge (minus bzw. ohne)
$\complement$	Komplementmenge (Komplementmenge von)
$\neg$	Negation (nicht)
$\wedge$	Konjunktion (und)
$\vee$	Disjunktion (oder)
$\forall x$	Allquantor (für alle x gilt)
$\exists x$	Existenzquantor (es existiert ein x)
$\nexists x$	Existenzquantor (es existiert kein x)
$\Rightarrow$	Folgerung (folgt)
$\Leftrightarrow$	Äquivalenz (ist äquivalent zu)
$\sum_a x$	Summe (summiere x über jedes existierende a)
$\sqrt{x}$	Quadratwurzel zu x
x^y	Potenz (x hoch y)
...	und so weiter
$+$	Addition (plus)
$-$	Subtraktion (minus)
$a \cdot b$	Multiplikation (a mal b)
ab	Multiplikation (a mal b)
∞	Unendlich
$x \to y$	x nähert sich y, x gegen y

[2] Ist für eine eindeutige Lesbarkeit erforderlich, da in der deutschen Sprache gebrochenrationale Zahlen mit Komma (z. B. $3,14$) dargestellt werden. Auf diese Weise können Mengen mit gebrochenrationalen Zahlen eindeutig dargestellt werden (z. B. $\{1,3;\ 3,4\}$).

$\lvert\vec{x}\rvert$	Betrag (Länge des Vektors $\vec{x}$)
(a, b)	Vektordefinition (Zusammenfassung von a und b zu einem Vektor)
$(a; b)$	Vektordefinition (Zusammenfassung von a und b zu einem Vektor)[3]
$\top$	Vertauschen von Zeilen und Spalten einer Matrix (transponiert)

[3] Hier gilt dieselbe Argumentation wie bei der Mengendefinition.

Kapitel 1

Einleitung

1.1 Problemstellung

Als die Computer in den 1980er Jahren hinreichend klein und gleichzeitig ausreichend leistungsstark geworden sind, so dass sie als Arbeitsplatzrechner im Büroumfeld genutzt werden konnten, nahm der Anteil der von den Rechnern gespeicherten und verarbeiteten natürlichsprachlichen Informationen[1] wie z. B. Briefe und Textdokumente stark zu. Dieser Trend hält bis heute an und wurde insbesondere durch die Anbindung der Arbeitsplatzrechner an das Internet verstärkt. Das Internet ermöglicht einen einfachen und schnellen Zugriff auf eine Vielzahl von Dokumenten bzw. Webseiten. Gleichzeitig vereinfacht das Internet die Publikation von Dokumenten und reduziert die Publikationskosten gegenüber den traditionellen Publikationswegen enorm. Die Folge davon ist ein rasanter Anstieg von Dokumenten, auf die einfach und kostengünstig zugegriffen werden kann bzw. die z. B. im Fall von E-Mails und insbesondere Werbe-E-Mails regelrecht um die Aufmerksamkeit der Computernutzer „ringen“. Einer im Jahre 2000 erstellten Studie der Berkeley-Universität zu Folge, enthält das WWW im Jahre 2000 ca. 2,5 Milliarden statische und ca. 550 Milliarden dynamische Webseiten, die zu 95 Prozent öffentlich zugänglich sind. [91] Ähnlich hohe Anzahlen von Dokumenten können auch für andere Bereiche der computergestützten Kommunikation, wie z. B. E-Mail und Usenet

[1] Der Begriff Information und insbesondere seine Abgrenzung zu dem Begriff Daten wird seit langem in der wissenschaftlichen Literatur kontrovers diskutiert. Unterschiedlichste Definitionsversuche kommen zu verschiedensten, sich teilweise widersprechenden Ergebnissen. (Vgl. dazu z. B. die Begriffsdiskussionen in BODE [21, S. 6ff], SCHÜTTE [133] und EHLERS [47, S. 13ff].) Dieses legt nahe, dass eine eindeutige Abgrenzung der beiden Begriffe (anwendungsübergreifend) nicht möglich ist oder nicht existiert. In dieser Arbeit wird daher die folgende pragmatische Definition für die beiden Begriffe Daten und Informationen verwendet: *Daten* bzw. *Informationen* sind alles das, was durch eine zeitliche oder räumliche Anordnung von Materie oder Energie dargestellt werden kann. Somit meinen bzw. beschreiben beide Begriffe dieselbe Sache. Der Begriff *Daten* wird verwendet, wenn die technischen Aspekte der Repräsentation (z. B. Datenformate und die algorithmische Verarbeitung oder ihre Speicherung) von Informationen im Vordergrund der Betrachtung bzw. Diskussion stehen. Im Unterschied dazu wird der Begriff *Information* bzw. *Informationen* verwendet, wenn die Bedeutung eines Datums bzw. von Daten, d. h. seine bzw. ihre „Aussage“ im Vordergrund steht.

aufgestellt werden. Mit der zunehmenden Kapazität von digitalen Speichermedien erreichen auch lokale Informationsbestände in Forschungseinrichtungen, Unternehmen, Behörden und Bibliotheken mittlerweile eine derartige Größe, dass niemand mehr in der Lage ist, den gesamten Dokumentenbestand zu überblicken. Die beschriebene Entwicklung hat dazu geführt, dass jeder Person in den Industrieländern heutzutage quantitativ sehr viele Informationen für die Problemlösung und die Entscheidungsfindung zur Verfügung stehen, dass aber ein Großteil dieser Informationen aufgrund der beschränkten Verarbeitungskapazität nicht adäquat verarbeitet werden kann und somit die relevanten Informationen nicht mehr in einem hinreichenden Ausmaß gefunden werden können. Es kann teilweise auch beobachtet werden, dass diese „Flut von Informationen“ die Arbeit sogar behindert. [58] Im allgemeinen Sprachgebrauch wird diese Problematik mit dem Begriff *Informationsüberflutung*[2] bezeichnet.

Aufgrund des Phänomens der Informationsüberflutung gehört die Entwicklung von geeigneten Methoden und Werkzeugen für die Suche (Information-Retrieval) und die Filterung (Information-Filtering) von natürlichsprachlichen Informationen zu den wichtigsten Herausforderungen der heutigen Zeit. Die Einsatzgebiete derartiger Werkzeuge sind mannigfaltig und überspannen verschiedenste Bereiche des gesellschaftlichen Lebens, wie z. B. Wirtschaft, Wissenschaft und öffentliche Verwaltung. Konkrete Aufgabenstellungen von Information-Retrieval Werkzeugen können z. B. die Suche nach Dokumenten im Internet, in Bibliotheken oder Nachrichten- bzw. Dokumentarchiven sein. Information-Filtering Werkzeuge können prinzipiell zum Filtern von Nachrichten- und Ereignisströmen, zum Filtern oder zielgruppengerechten Verteilen[3] von digitalen Dokumenten und insbesondere E-Mails verwendet werden.

Die Entwicklung von geeigneten Methoden und Werkzeugen für das Information-Filtering und -Retrieval ist bis heute nicht abgeschlossen, weil die Problemstellung von den bisherigen Verfahren nicht hinreichend gelöst wird. Aufgrund der hohen Komplexität von natürlichen Sprachen und der immer noch nicht ausreichenden Rechenkapazität von Rechnern können bis heute lediglich Heuristiken zur Lösung des Problems eingesetzt werden. Das heißt, dass der Rechner nicht in der Lage ist, den Inhalt der zu bearbeitenden Dokumente zu „verstehen“. Grundlage für die Heuristiken ist dabei immer ein (formales) Modell, welches zur Repräsentation von natürlichsprachlichen Dokumenten in einem Rechnersystem verwendet wird. Bei den meisten heute in der Praxis eingesetzten Heuristiken werden Dokumente als eine Menge von voneinander unabhängigen Wörtern modelliert. Das heißt, dass die komplexe Realität der natürlichen Sprachen durch ein stark vereinfachtes Modell abgebildet wird. Derartige Heuristiken sind geeignet, um Dokumente zu finden, die bestimmte Wortkombinationen oder Phrasen enthalten. Allerdings scheitern diese Heuristiken häufig dann, wenn das gesuchte Dokument anstatt der angegebenen Wortkombination eine andere, äquivalente oder bedeutungsähnliche Wortkombination enthält. Im Unterschied zu den sonst von Rechnersystemen verarbeiteten Informationen, die unter Verwendung von formalen Sprachen strukturiert sind, zeichnen sich natürliche Sprachen, die von Information-Filtering und -Retrieval Werkzeugen verarbeitet werden müssen u. a. durch Redundanzen und Ambiguitäten aus. BATES stellt da-

[2] Die in der englischen Literatur geläufigen Begriffe zur Benennung des Phänomens sind *Infoglut* [29] und *Information Overload*. [93]

[3] Z. B. können E-Mails an Unternehmen, die nicht an bestimmte Personen gerichtet sind, in Abhängigkeit von ihrem groben Themenbezug automatisch an unterschiedliche Unternehmensabteilungen weitergeleitet werden.

her für natürliche Sprachen die folgende Vermutung auf: Die Wahrscheinlichkeit dafür, dass zwei Personen denselben Begriff zur Beschreibung der selben Sache verwenden ist kleiner als 20 Prozent. [10]

Neuere, in der Forschung befindliche Heuristiken, versuchen der Problemstellung der Redundanz zu begegnen, indem sie die Ähnlichkeit von Wörtern in ihr Modell aufnehmen und über das gemeinsame Auftreten von Wörtern in einem Dokumentenbestand die Ähnlichkeit zwischen zwei Wörtern schätzen. Dieses Vorgehen hat sich bisher allerdings als nur wenig zielführend herausgestellt, weil alleine das gemeinsame Auftreten zweier Wörter nicht geeignet ist, auf eine Synonymie oder enge Verwandtschaft zweier Wörter zu schließen. Der Grund dafür ist, dass es verschiedene linguistische Phänomene gibt, die sich im gemeinsamen Auftreten von Wörtern ähneln, die aber mitunter nicht durch eine Verwandtschaft von Wörtern bedingt sind.[4] Zudem reicht das Berücksichtigen von Redundanz alleine nicht aus; es müssen vielmehr in dem Modell zur Repräsentation von Dokumenten auch die Ambiguitäten der natürlichen Sprachen berücksichtigt werden. Man kann somit sagen, dass die bisherigen Heuristiken für das Information-Filtering und -Retrieval daran kranken, dass sie linguistische Phänomene und Zusammenhänge zwischen Wörtern nicht hinreichend im Modell abbilden. Somit hat die Erforschung und Entwicklung derartiger Heuristiken bzw. insbesondere der Modelle zur Repräsentation von Dokumenten bis heute nicht an Aktualität eingebüßt.

1.2 Zielsetzung der Arbeit

Das Ziel dieser Arbeit ist die Konzeption und Entwicklung eines Modells zur Repräsentation von natürlichsprachlichen Dokumenten, das im Rahmen einer Heuristik für das Information-Filtering und -Retrieval eingesetzt werden kann. Im Vergleich zu den bisherigen Modellen und Heuristiken aus Forschung und Praxis werden sowohl linguistische Phänomene als auch Redundanzen und Ambiguitäten von natürlichen Sprachen in einem weit größerem Ausmaß als bisher in dem Modell abbildet. Somit ist das Modell in der Lage, thematische Zusammenhänge zwischen verschiedenen Wörtern zu berücksichtigen. Da einfache statistische Verfahren zur Erkennung derartiger Zusammenhänge nicht funktionieren, enthält weder das Modell noch die auf dem Modell aufbauende Heuristik Verfahren zur Erkennung derartiger Zusammenhänge. Vielmehr bietet das Modell eine Art Schnittstelle an, mit der thematische Zusammenhänge zwischen Wörtern von außen vorgegeben werden können. Somit ist das Modell beispielsweise in der Lage von Linguisten erstellte Ontologien[5] (wie z. B. die Wortnetze *WordNet* und *GermaNet*[6]), die „Wissen“ über die linguistischen und thematischen Zusammenhänge zwischen Wörtern enthalten, für das Information-Filtering und -Retrieval wiederzuverwenden. Zur Demonstration der Umsetzbarkeit des, in dieser Arbeit entwickelten, Modells wird neben dem Modell auch eine Implementierung des Modells unter Verwendung einer relationalen Datenbank vorgestellt.

[4] Diese Problematik wird in Abschnitt 3.3 ausführlich behandelt.
[5] Die Definition des Begriffs Ontologie findet sich in Abschnitt 2.4.
[6] Vgl. Abschnitt 6.1.2.2.

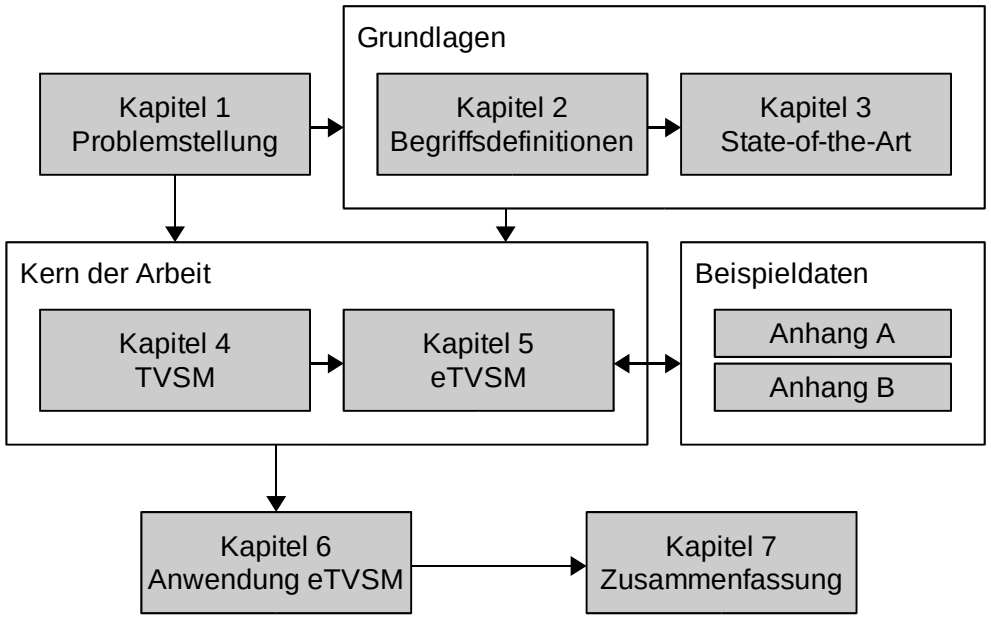


Abbildung 1.1: Aufbau der Arbeit und empfohlene Lesereihenfolgen.

1.3 Aufbau der Arbeit

Die Abbildung 1.1 zeigt eine schematische Darstellung des Aufbaus der Arbeit und deutet unter Verwendung von Pfeilen drei Lesereihenfolgen an. Ausgehend vom ersten Kapitel ist es bei einem konservativen Vorgehen sinnvoll, aber für den fachkundigen Leser nicht unbedingt notwendig, zunächst die beiden Grundlagenkapitel, Kapitel 2 und Kapitel 3, zu lesen. Im Kapitel 2 werden die zum Verständnis dieser Arbeit notwendigen Begriffe und Methoden u. a. aus den Fachgebieten der Datenmodellierung und Computerlinguistik vorgestellt. Aufbauend auf diesen Begriffen werden in Kapitel 3 die nach dem jetzigen Stand der Forschung und Praxis gängigen Modelle zur Repräsentation von natürlichsprachlichen Dokumenten vorgestellt und qualitativ (nach linguistischen Aspekten) bewertet. Diese Bewertung beantwortet die folgende Frage und motiviert somit den Kern der Arbeit: Warum sollte man angesichts der Vielzahl an bereits existierenden Modellen zur Repräsentation von natürlichsprachlichen Dokumenten ein neues Modell entwickeln?

Die Kapitel 4 und 5 bauen aufeinander auf und stellen den Kern dieser Arbeit dar. Ausgangspunkt des hier entwickelten Modells ist das von BECKER und KUROPKA in [12] eingeführte TVSM, dass in den Abschnitten 4.1 und 4.2 in einer überarbeiteten Form vorgestellt wird. Alle anderen Abschnitte der beiden Kapitel stellen neue bisher nicht publizierte Konzepte und Ergebnisse der Forschung dar. Zur Illustration der Funktionsweise und Implemen-

tierung des in Kapitel 5 vorgestellten eTVSM, wird ein durchgehendes Beispiel verwendet. Für Leser, die daran interessiert sind, die Verarbeitung aller Beispieldaten im Detail nachzuvollziehen, befinden sich in Kapitel 5 an den jeweils passenden Stellen Verweise auf die sich im Anhang befindlichen relevanten Auszüge aus Datenbanktabellen der Beispielimplementierung.

Abschließend wird in dem Kapitel 6 auf die Einsetzbarkeit, Evaluation und Integration des eTVSM mit vorhandenen Ontologien eingegangen. In Kapitel 7 wird ein Zusammenfassung der Arbeit und ein Ausblick auf noch ausstehende Forschungsarbeiten im Zusammenhang mit dem eTVSM gegeben.

Kapitel 2

Grundlegende Definitionen und Methoden

2.1 Information-Filtering und -Retrieval

Die Definitionen der beiden Begriffe *Information-Filtering* (IF) und *Information-Retrieval* (IR), die in dieser Arbeit verwendet werden, basieren auf den Begriffsdefinitionen von BELKIN und CROFT [17] und erweitern diese um den Bezug zu Modellen der Repräsentation von Dokumenten bzw. der Interaktion mit dem Benutzer. Vorab ist zu erwähnen, dass IR bzw. IF die Aufgaben eines IR-Systems bzw. IF-Systems sind. Insofern ist die Beschreibung der Aufgaben eines IR-Systems äquivalent mit der Definition des IR-Begriffes (dieses gilt analog für IF-Systeme).

2.1.1 Information-Retrieval

Es existieren verschiedene (zumeist vage formulierte) Definitionen von *Information-Retrieval*. So schreibt z. B. ROBERTSON in [123] (sinngemäß aus dem englischen übersetzt) folgendes:

> Die Aufgabe von IR-Systemen ist es, den Benutzer zu denjenigen Dokumenten zu führen, die seinen Bedarf an Informationen befriedigen.

Um den Begriff IR exakter zu definieren und damit eine Abgrenzung zwischen z. B. Datenbanken oder IF zu ermöglichen, ist es erforderlich, sich den folgenden Fragen im Detail zu widmen:

1. Welche Daten/Dokumente werden beim IR verarbeitet?

2. Wie sieht das Modell[1] von IR auf hoher Abstraktionsebene aus?

Bezugnehmend auf die erste Frage wird hier definiert, dass IR-Systeme ausschließlich in Textform vorliegende Schriftdokumente verarbeiten. Das heißt, im Folgenden wird unter dem Begriff *Dokument* ein digitales Schriftdokument (z. B. eine E-Mail, eine Web-Seite, ein Brief, etc.) verstanden, welches in einem Datenformat (z. B. ASCII-Text, HTML, RTF) vorliegt, das einen direkten Zugriff auf alle Zeichen des Dokuments ermöglicht.[2] Weitere Annahmen bezüglich einer tiefergehenden Strukturierung der Dokumente werden nicht getroffen, weshalb man diese Art von Daten häufig auch als *unstrukturierte* oder im Falle von z. B. E-Mail als *semistrukturierte*[3] Daten bezeichnet. Durch diese Definition wird der Unterschied zwischen IR-Systemen und gängigen (meist betrieblich genutzten) Datenbanksystemen deutlich: Datenbanksysteme verarbeiten im Gegensatz zu IR-Systemen Daten, die üblicherweise bis ins Detail über Datentypen, Attribute und Relationen/Beziehungen strukturiert bzw. gültige „Sätze“ einer formalen Sprache sind.

Abbildung 2.1 zeigt ein allgemeines Modell zum IR und beantwortet somit die zweite Frage. Generell sind am IR zwei (sich unter Umständen überschneidende) Personenkreise involviert. Der erste Personenkreis sind die Autoren, die Dokumente in einem IR-System zur Verfügung stellen. Dieses kann sowohl aktiv geschehen, indem die Autoren die Dokumente selber in das System einstellen, oder auch passiv geschehen, indem das System über Kommunikationsmittel die Dokumente aus anderen verfügbaren Informationssystemen ausliest (wie es z. B. die Internet-Suchmaschinen praktizieren). Die in das System eingestellten Dokumente werden vom IR-System gemäß dem System-internen Modell der Repräsentation von Dokumenten in eine für die Verarbeitung günstige Form (Dokumentenrepräsentation) umgewandelt.[4]

Die zweite Benutzergruppe, die Anwender, haben bestimmte, zum Zeitpunkt der Arbeit am IR-System akute Ziele oder Aufgaben, für deren Lösung ihnen Informationen fehlen. Diese Informationsbedarfe beabsichtigen die Anwender mit Hilfe des Systems zu befriedigen. Dafür müssen sie ihre Informationsbedarfe in einer adäquaten Form als Anfragen formulieren. Die Form, in der die Informationsbedarfe formuliert werden müssen, hängt dabei von dem verwendeten Modell der Repräsentation von Dokumenten ab. Wie der Vorgang der Modellierung der Informationsbedarfe als Interaktion mit dem System abläuft (z. B. als einfache Eingabe von Suchbegriffen), wird vom Modell der Interaktion festgelegt. Sind die Anfragen formuliert, dann ist es die Aufgabe des IR-Systems, die Anfragen mit den im System eingestellten Dokumenten unter Verwendung der Dokumentenrepräsentationen zu vergleichen und eine Liste der zu den Anfragen passenden Dokumente an die Benutzer zurückzugeben. Der Benutzer steht nun vor der Aufgabe, die gefundenen Dokumente gemäß seiner Aufgabe

[1] Der Modell-Begriff wird in Abschnitt 2.2 im Detail behandelt. An dieser Stelle ist das „intuitive“ Verständnis des Begriffes Modell für die Erklärung der beiden Begriffe IR und IF ausreichend.

[2] Gegebenenfalls in den Dokumenten eingebettete Grafiken, Formatierungen etc. werden von den meisten IR-Systemen bei der Verarbeitung ignoriert.

[3] Semistrukturierte Daten sind Daten, die sowohl strukturierte als auch unstrukturierte Elemente enthalten. Der Text einer E-Mail ist unstrukturiert, wohingegen die Sender- und Empfängeradresse gemäß RFC 822 [74] strukturiert sind.

[4] Vgl. dazu den Abschnitt 2.1.4.

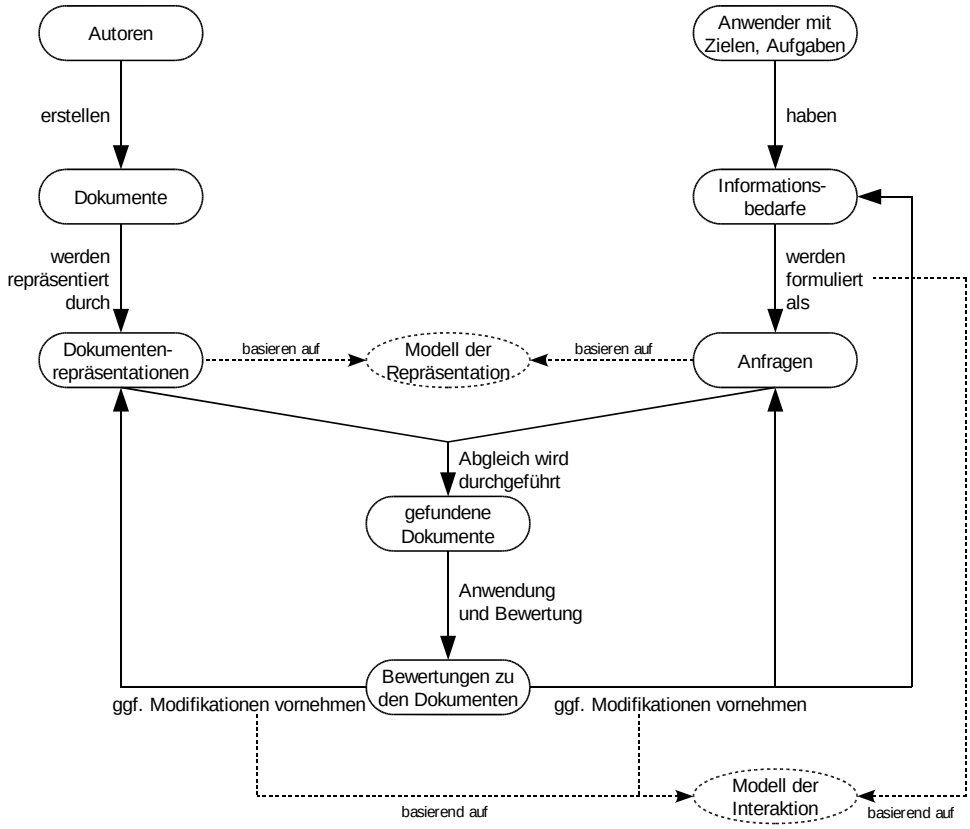


Abbildung 2.1: Ein allgemeines Modell zum Information-Retrieval.

auf die Lösungsrelevanz hin zu bewerten. Das Resultat ist die Bewertungen zu den Dokumenten. Anschließend haben die Benutzer drei Möglichkeiten: Erstens, sie können (meist nur in einem engen Rahmen) Modifikationen an den Repräsentationen der Dokumente vornehmen (z. B. indem sie neue Schlüsselwörter für die Indexierung eines Dokuments definieren). Zweitens, die Benutzer verfeinern ihre formulierten Anfragen (zumeist um das Suchergebnis weiter einzuschränken) und drittens, die Benutzer ändern ihre Informationsbedarfe, weil sie nach dem Durchführen der Recherche feststellen, dass sie zur Lösung ihrer Aufgaben weitere, zuvor nicht als relevant eingestufte Informationen benötigen. Der genaue Ablauf der drei Modifikationsformen wird vom Modell der Interaktion bestimmt. Z. B. gibt es Systeme, die den Benutzern bei der Reformulierung der Anfrage unterstützen, indem sie die Anfrage unter Verwendung von, vom Benutzer explizierter (d. h. dem System in irgendeiner Form mitgeteilter)

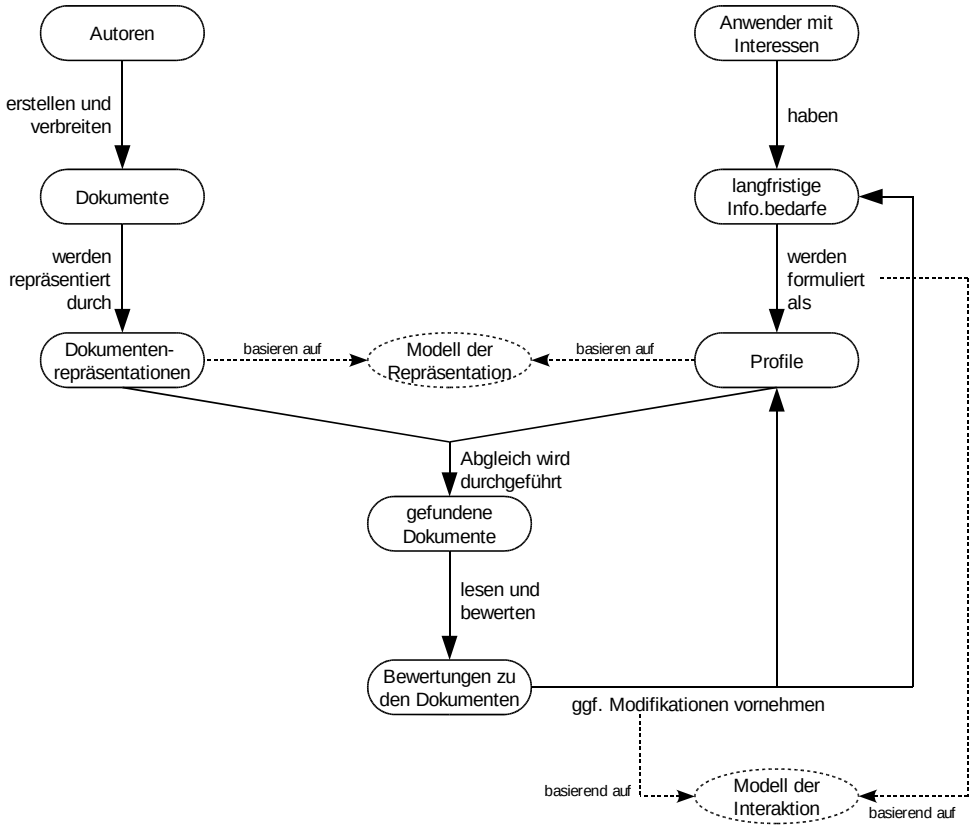


Abbildung 2.2: Ein allgemeines Modell zum Information-Filtering.

Dokumentenbewertungen, automatisiert reformulieren.[5]

2.1.2 Information-Filtering

Ebenso wie für IR-Systeme wird für *IF-Systeme* definiert, dass diese ausschließlich in Textform vorliegende digitale Schriftdokumente verarbeiten. Das allgemeine Modell des IF findet sich in Abbildung 2.2. Ein direkter Vergleich mit dem allgemeinen Modell des IR (Abbildung 2.1) offenbart die große, strukturelle Ähnlichkeit zwischen den beiden Aufgaben.

Wie beim IF gibt es auch beim IR zwei Anwendergruppen. Die erste Anwendergruppe sind die Autoren, die beim IF oft auch Institutionen (z. B. Nachrichtenagenturen, Werbe-Mailer,

[5] Vgl. dazu den Abschnitt 2.1.5.

etc.) sind und einen, über einen größeren Zeitraum gesehen kontinuierlichen „Strom von Dokumenten“ produzieren und die Dokumente aktiv oder passiv in das Filtering-System einstellen. Diese Dokumente werden auch beim IF gemäß dem internen Modell der Repräsentation von Dokumenten in eine, für die Maschine geeignete Form (Dokumentenrepräsentation) transformiert.[6]

Die zweite Anwendergruppe sind die Anwender mit langfristigen Interessen an Dokumenten zu bestimmten Themen. Diese langfristige Informationsbedarfe müssen für die Verarbeitung geeignet, in Form von Profilen formalisiert werden. Die Struktur dieser Profile wird vom Modell der Repräsentation von Dokumenten und der Erstellungsprozess wird vom Modell der Interaktion determiniert. Unter Anwendung der Profile werden neu eintreffende Dokumente anhand ihrer Dokumentenrepräsentationen bezüglich ihrer Relevanz für die Anwender bewertet und gegebenenfalls (in Abhängigkeit von den Filter-Einstellungen) an die Anwender weitergeleitet. Haben die Anwender ein Dokument gelesen und bewertet, dann haben sie die folgenden beiden Möglichkeiten: Erstens, sie aktualisieren ihr Profil, weil sie z. B. feststellen, dass das Profil nicht alle nicht relevanten Dokumente entfernt hat oder weil relevante Dokumente fälschlicherweise aussortiert[7] wurden. Dieses kann je nach System manuell geschehen oder in Abhängigkeit vom Modell der Interaktion vom System auch teilweise oder vollständig automatisiert umgesetzt werden. Beispielsweise kann ein System unter Verwendung von expliziten Benutzerbewertungen und von adaptiven Verfahren selbstständig Änderungen am Profil durchführen.[8] Zweitens, die Benutzer ändern ihre langfristigen Informationsbedarfe. In diesem Fall müssen, wie im ersten Fall auch, die betroffenen Profile der Benutzer überarbeitet werden. Allerdings ist das Ausmaß der notwendigen Modifikationen gegenüber dem ersten Fall im Allgemeinen größer. Daher ist es sinnvoll, wenn jeder der betroffenen Benutzer dem IR-System die Änderung seines Profils (evtl. mit einigen für die neuen Interessen relevanten Stichworten oder Beispieldokumenten) mitteilen kann, damit sich das System möglichst schnell an die veränderte Situation anpassen kann.

2.1.3 Gemeinsamkeiten und Unterschiede

Die *wichtigsten Gemeinsamkeiten* von IF und IR sind die strukturellen Ähnlichkeiten in den ihnen zu Grunde liegenden Modellen (vgl. Abbildungen 2.1 und 2.2). In beiden Aufgabenstellungen ist der grundsätzliche Ablauf des IF- bzw. IR-Prozesses hochgradig ähnlich und beide Aufgabenstellungen stützen sich bei ihrer Problemlösung auf einem Modell der Repräsentation von Dokumenten und einem Modell der Benutzerinteraktion. Die *wichtigsten Unterschiede* zwischen IF und IR lassen sich in den folgenden Punkten zusammenfassen:

- IR-Systeme dienen zur Befriedigung eines – aus zeitlicher Sicht gesehen – kurzfristigen Informationsbedarfs, meistens mit dem Ziel eine akute Aufgabe zu lösen. IF-Systeme

[6] Vgl. dazu den Abschnitt 2.1.4.

[7] Damit der Benutzer feststellen kann, welche Dokumente fälschlicherweise aussortiert wurden ist es erforderlich, dass das IF-System nicht relevante Dokumente nicht endgültig löscht. Zudem muss der Benutzer gelegentlich die aussortierten Dokumente bezüglich ihrer Relevanz überprüfen.

[8] Vgl. dazu die Abschnitte 2.1.5 und 6.3.

werden hingegen eingesetzt um ein langfristiges Ziel zu erreichen: Der Benutzer soll durch eine vom System vorab durchgeführte Bewertung dabei unterstützt werden, seinen langfristigen Informationsbedarf zu decken, ohne dass der Benutzer alle neuen Dokumente selber lesen muss.

- Aus Sicht der vom Benutzer formulierten Anfrage ist der Dokumentenbestand bei einem IR-System statisch – der Dokumentenbestand ändert sich zum Zeitpunkt der Anfrage normalerweise nicht. Aus Sicht des Profils ist der Dokumentenbestand bei einem IF-System dynamisch – es kommen laufend neue Dokumente dazu.

- Während eine Anfrage in einem IR-System nur eine kurze Gültigkeitsdauer hat und bei Bedarf vom Benutzer in kurzen Zeitintervallen modifiziert wird, hat das Profil in einem IF-System eine lange Gültigkeitsdauer und wird (nach einer Einführungsphase) eher selten und in einem geringem Ausmaß modifiziert, solange sich die langfristigen Benutzerinteressen nicht ändern.

- Während für IR-Systeme die zeitnahe Weitergabe von Dokumenten an den Benutzer eher eine untergeordnete Rolle spielt, ist es für ein IF-System von hoher Bedeutung, dass es neue Dokumente möglichst zeitnah evaluiert und gegebenenfalls an den Benutzer weiterleitet.

- Anwender von IR-Systemen legen weniger Wert darauf, dass ihre Anfragen vertraulich behandelt werden zumal die Anfragen häufig (relativ) anonym gestellt werden können.[9] IF ist umgekehrt dazu über die langfristige Profilbindung stark an einen Anwender oder eine Anwendergruppe gebunden und lässt sich nur schwer anonymisieren, ohne die Problemlösung zu beeinträchtigen. Zusätzlich enthalten die Anwenderprofile Informationen über die langfristigen Interessen eines Anwenders die unter Umständen auch seine politischen und persönlichen Interessen widerspiegeln können. Daher ist der Schutz der Profildaten vor unberechtigtem Zugriff für IF-Systeme von hoher Bedeutung.

2.1.4 Modell der Repräsentation von Dokumenten

Zur Verarbeitung von Dokumenten werden von IF- und IR-Systemen Modelle zur Repräsentation der Dokumente aus dem folgenden Grund benötigt: Ein Rechner ist ein Stück Hardware, welches in seiner ursprünglichen Intention zum Rechnen konzipiert wurde. Jede Aufgabe, die einem Rechner zur Lösung gegeben wird, muss daher (unter Verwendung einer Programmiersprache) derart formuliert werden, dass sie berechenbar ist – im Sinne des Berechenbarkeit-Begriffs von TURING [144]. Übertragen auf das Problem des IF bzw. IR heißt das, dass dem Rechner ein formales mathematisches Modell zur Repräsentation von Dokumenten zur Verfügung gestellt werden muss, anhand dessen der Rechner in der Lage ist, die Aufgabe des IF bzw. IR zu lösen. Konkret bedeutet das für das IR, dass der Rechner zu jeder vom Benutzer gestellten Anfrage in irgendeiner Form berechnen können muss, welches Dokument die Anfrage

[9] In der Tat werden von gängigen Internetsuchmaschinen (wie z. B. *Google*) alle Anfragen inklusive technischer Daten (wie z. B. Cookies, IP-Adressen, etc.) dauerhaft gespeichert. Vgl. dazu z. B. die Datenschutzbestimmungen von *Google* unter `http://www.google.de/intl/de/privacy.html`.

in wie weit erfüllt. Im Falle des IF bedeutet das, dass der Rechner für jedes neu eintreffende Dokument berechnen können muss, in wie weit dieses neue Dokument mit dem Benutzerprofil übereinstimmt.

Das Modell zur Repräsentation von Dokumenten umfasst demnach sowohl Dokumente als auch Anfragen und/oder Profile. Je nach Mächtigkeit und Ausgestaltung des Modells kann ein Modell somit entweder nur für IF, nur für IR oder für beides verwendet werden. Ein häufiger Grund dafür, ein Modell nur für eine Aufgabe zu konzipieren, ist die technische Sicht der Implementierung. Bei der Implementierung eines Modells können durch die Spezialisierung des Modells auf nur eine Aufgabe in vielen Fällen Optimierungen vorgenommen werden, die das Berechnen beschleunigen. Modelle zur Repräsentation von Dokumenten sind das Kerngebiet dieser Arbeit und werden ausführlich in den Kapiteln 3, 4 und 5 behandelt.

2.1.5 Modell der Interaktion mit dem Benutzer

Neben den eher bedienungsorientierten Kommunikationskonzepten kann das Modell zur Interaktion mit dem Benutzer auch mathematische Modelle zur automatisierten Modifikation von Anfragen oder Profilen enthalten. Im Falle des IR existieren z. B. Systeme, die den Benutzer bei der Reformulierung (und Erweiterung) einer Anfrage unterstützen. Eine Übersicht derartiger als *Query Expansion* bezeichneten Verfahren findet sich in [7, S. 117ff]. So werden beim *User Relevance Feedback* nach einer ersten, manuell eingegebenen Anfrage die Ergebnisse präsentiert, wobei der Benutzer die am meisten relevanten Dokumente kennzeichnen kann (Feedback). In einem nächsten Schritt verarbeitet das IR-System die gekennzeichneten Dokumente um die Anfrage zu reformulieren. Das Ziel ist, dass beim nächsten Ergebnis mehr Dokumente gefunden werden, die zu den vom Benutzer als relevant gekennzeichneten Dokumenten ähnlich sind. [127][124][129, S. 251ff][7, S. 118ff] Andere Verfahren zur automatisierten Modifikation von Anfragen sind z. B. die *Automatic Local Analysis* und die *Automatic Global Analysis*. Bei beiden Verfahren versucht das System, unter Verwendung einer größeren Anzahl von Dokumenten, Ähnlichkeiten zwischen Begriffen aufzudecken und die Anfrage des Benutzers um ähnliche Begriffe zu erweitern. Bei der Automatic Local Analysis werden Ähnlichkeiten in den gefundenen Dokumenten der vom Benutzer erstellten Anfrage gesucht, bei der Automatic Global Analysis hingegen werden alle Dokumente zur Bestimmung der Ähnlichkeiten verwendet. [7, S. 123ff]

Beim IF bemühen sich viele Systeme darum, dem Benutzer über geeignete Interaktionsmodelle das Erstellen und Warten von Profilen zu erleichtern. Ansätze dieser Art werden unter dem Stichwort *Machine Learning for User Modeling* (ML4UM) zusammengefasst. Ziel dieser Verfahren ist es die Benutzerprofile durch Beobachtung des Benutzerverhaltens oder durch Nutzen von, vom Benutzer gemachten Dokumentenbewertungen automatisiert zu erlernen. [20, 80] Aus diesem Grund greifen derartige Verfahren häufig auf bereits etablierte mathematische Lernverfahren zurück. Zu den im Bereich des IF am häufigsten eingesetzten Lernverfahren gehören (in alphabetischer Reihenfolge): Evolutionäre Optimierung [67], k-nearest neighbour [162, 40], Künstliche Neuronale Netze [22], Linear Least Squares Fit [161], Naiver

Bayes [162] und Support Vector Machines [153, S. 138ff].[10]

Modelle der Repräsentation von Dokumenten stehen mit Modellen der Interaktion im folgenden Zusammenhang: Die Modelle der Repräsentation bilden das Fundament, auf dem die Modelle der Interaktion aufbauen. Dieses Fundament spannt in übertragenem Sinne eine Art „Raum“ auf, in dem die realen Dokumente repräsentiert werden. Aufbauend auf den „Eigenheiten“ dieses Raumes können je nach Modell Anfragen, Profile oder Dokumente mit Dokumenten auf Ähnlichkeit hin verglichen werden. Die unterschiedlichen Modelle der Interaktion nutzen die Eigenheiten des Raumes meist in Kombination mit zusätzlichen Informationen vom Benutzer (z. B. den Benutzerbewertungen von Dokumenten), um die Anfragen bzw. Profile automatisiert zu modifizieren. Dabei ist zu beachten, dass ein Modell der Interaktion nur dann zu einem Modell der Repräsentation kompatibel ist, wenn die dem Modell der Interaktion zu Grunde liegenden Annahmen mit den Eigenheiten des Raumes vereinbar sind. Das heißt, dass nicht jedes Modell der Repräsentation mit jedem Modell der Interaktion kombinierbar ist. Der Schwerpunkt dieser Arbeit liegt auf den Modellen der Repräsentation von Dokumenten, weil diese die Grundlage zur Lösung der Problemstellung von IF und IR bilden und damit auch den theoretischen Rahmen für die maximal erreichbare Qualität eines Systems stellen. Modelle der Interaktion werden in dieser Arbeit nur am Rande behandelt.

2.2 Datenmodelle

Im Folgenden wird der Begriff *Modell*, unter Rückgriff auf die konstruktionsorientierte Definition von BECKER, wie folgt definiert:

> „Ein *Modell* ist die Repräsentation eines Objektsystems für Zwecke eines Subjekts. Es entsteht durch die Konstruktionsleistung eines Menschen (Modellierer) und ist durch ein System von Symbolen (Modellierungssprache) dargestellt.“ [11]

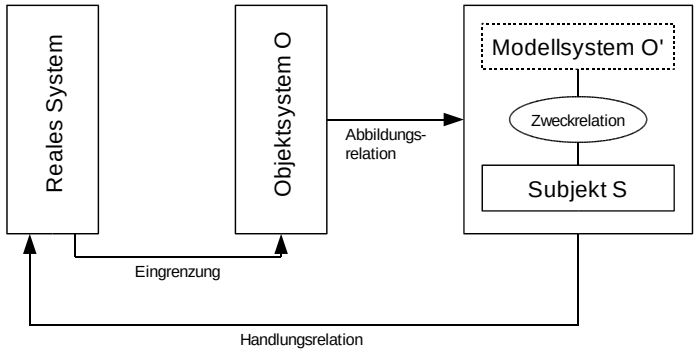
Eine grafische Illustration dieser Definition des Modell-Begriffs findet sich in der Abbildung 2.3. Eine detailliertere Darstellung und Herleitung des Begriffs findet sich in Publikationen von BECKER und SCHÜTTE [13, S. 21f] sowie von KUGELER [83, S. 92ff].

In Anlehnung an die *Architektur integrierter Informationssysteme* (ARIS) nach SCHEER [130] lassen sich Modelle in eine oder mehrere der folgenden vier Sichten einordnen:[11]

1. Die *Datensicht* trifft Aussagen über die in einem Informationssystem möglichen Daten (Zustände und Ereignisse). Darüberhinaus können in der Datensicht Konsistenzbedingungen für Daten und Datenkonstellationen definiert werden.

[10] Ein empirischen Vergleich bezüglicher der Eignung der unterschiedlichen Lernverfahren für IF (bzw. automatische Textkategorisierung im Allgemeinen) findet sich in [162].

[11] Die ursprüngliche Intention von SCHEER ist die Modellierung von betriebswirtschaftlichen Tatbeständen und Prozessen, dennoch kann ARIS auch in nicht unmittelbar betriebswirtschaftlichen Anwendungsfeldern (wie z. B. IF und IR) sinnvoll zur Strukturierung von Modellen eingesetzt werden. Daher ist die hier vorgestellte Definition der Sichten und der Modellierungsebenen gegenüber der Originaldefinition in [130] verallgemeinert worden, so dass sie auch ohne einem unmittelbareren Bezug zum betriebswirtschaftlichen Umfeld anwendbar ist.



Quelle: SINZ [136], HARS [68, S. 7ff] und STEINMÜLLER [141, S. 7ff].

Abbildung 2.3: Bestandteile eines Modells.

2. Auf den Daten auszuführende Funktionen bilden die *Funktionssicht*. Sie beinhaltet die Beschreibung der Funktionen, die Aufzählung einzelner Teilfunktionen und die zwischen den Funktionen evtl. bestehenden Anordnungsbeziehungen.

3. Die *Organisationssicht* wird über die Struktur und Beziehungen von Bearbeitern und Organisationseinheiten im Informationssystem gebildet.

4. Die Aufgabe der *Steuerungssicht* ist es, die ersten drei Sichten miteinander über Prozesse zu verknüpfen. Somit wird über die Steuerungssicht die zeitlich-sachlogische Reihenfolge der Ausführung von Funktionen auf Daten unter zur Hilfenahme von Organisationseinheiten bzw. Bearbeitern festgelegt.

Neben den vier Sichten werden in ARIS drei, zu den Sichten orthogonale Modellierungsebenen eingeführt:

1. Das *Fachkonzept* ist der Ausgangspunkt einer Systementwicklung. Die Beschreibung des Systems auf der Ebene des Fachkonzeptes ist sehr nahe an den fachlichen Zielsetzungen und an der fachlichen Sprachwelt orientiert. Sie ist im Allgemeinen nicht als unmittelbarer Ausgangspunkt einer Implementierung anwendbar.

2. Auf der Ebene des *DV-Konzeptes* wird die Begriffswelt des Fachkonzeptes in die Kategorien der DV-Umsetzung übertragen. Diese Ebene kann auch als Anpassung der Fachbeschreibung an generelle Schnittstellen der Informationstechnik bezeichnet werden.

3. Auf der Ebene der technischen *Implementierung* wird das DV-Konzept auf konkrete hardware- und softwaretechnische Komponenten übertragen. Hier wird damit die physische Verbindung zur Informationstechnik hergestellt.

Die in dieser Arbeit vorgestellten und entwickelten Modelle der Repräsentation von Dokumenten beziehen sich auf einen spezifischen Ausschnitt aus einem automatisierten Informationssystem. Daher wird in dieser Arbeit der Fokus auf die Datensicht des IF-/IR-Systems gelegt, weil die Komplexität der Funktions-, Organisations- und Prozesssicht im Bereich des betrachteten Ausschnitts eines IF- bzw. IR-Systems gering ist. So sind lediglich folgende Funktionen vorhanden, deren Ausführungsreihenfolge (Prozesssicht) und die beteiligten Personen (Organisationssicht) klar festgelegt sind: Dokumente erstellen, Dokumente einlesen bzw. geeignet im System repräsentieren, Dokument mit Profil bzw. Anfrage vergleichen, Ergebnis ausgeben und ggf. eine Bewertung des Ergebnisses notieren und auswerten.[12] Daher kann aus Gründen der Pragmatik auf die Verwendung formalisierter Modellierungssprachen für die Beschreibung dieser drei Sichten verzichtet werden. Im Gegensatz dazu steht die hohe Komplexität der Datensicht bei IF- und IR-Systemen, zu deren Bewältigung auf mathematische Modelle (Fachkonzept), Entity-Relationship-Modelle[13] (Fachkonzept und DV-Konzept) und relationale Modelle in SQL-Notation[14] (Implementierung) zurückgegriffen wird. Der Grund für die Verwendung von dreierlei Modellen für die Datensicht ist, dass einige Modelle der Repräsentation von Dokumenten bezüglich einer relationalen Implementierung spezielle Eigenschaften aufweisen, die eine komplexitätsreduzierende Implementierung ermöglichen. Diese lässt sich jedoch am besten unter Rückgriff auf implementierungsnahe Modellierungssprachen präsentieren. (Vgl. Kapitel 4 und 5.)

Die gleichzeitige Verwendung von mathematischen Modellen und Entity-Relationship-Modellen zur Beschreibung des Fachkonzeptes begründet sich darin, dass die beiden Modellierungssprachen unterschiedlich gut zur Repräsentation von mathematischen bzw. datenstrukturbezogenen Aspekten geeignet sind. Zusätzlich werden in einigen Fällen die Entity-Relationship-Modelle um einige wenige DV-Konzept spezifische Aspekte erweitert, die in den Modellen gestrichelt eingezeichnet werden.[15]

2.2.1 Entity-Relationship-Modelle

Eine weit verbreitete Modellierungssprache für die Modellierung der Datensicht ist das *Entity-Relationship-Model* (ERM) nach CHEN [32]. In dieser Arbeit wird diese Modellierungssprache mit Erweiterung um Minimal-Maximal-Kardinalitäten nach [131, S. 50] und [158, S. 193] für die fachkonzeptionelle (und DV-konzeptionelle) Modellierung verwendet. Im Folgenden werden die Modellierungskonzepte des ERM an einem einfachen Beispiel (Abbildung 2.4) erläutert. Ausführliche Beschreibungen der ERM-Notation und der gängigsten Erweiterungen finden sich in [130, S. 31ff] und in [13, S. 31].

Das Fundament der ERM-Notation bilden Entities (Dinge, Einheiten, Entitäten), Attribute und Relationships (Beziehungen), die sowohl auf der Ausprägungs- als auch auf der Typebene betrachtet werden. Die Typebene repräsentiert Mengen, wohingegen Ausprägungen Elemente

[12] Vgl. dazu die Abbildungen 2.1 und 2.2 auf den Seiten 9 und 10.
[13] Vgl. Abschnitt 2.2.1.
[14] Vgl. Abschnitt 2.2.2.
[15] Vgl. dazu z. B. das Modell in der Abbildung 5.1 auf Seite 111.

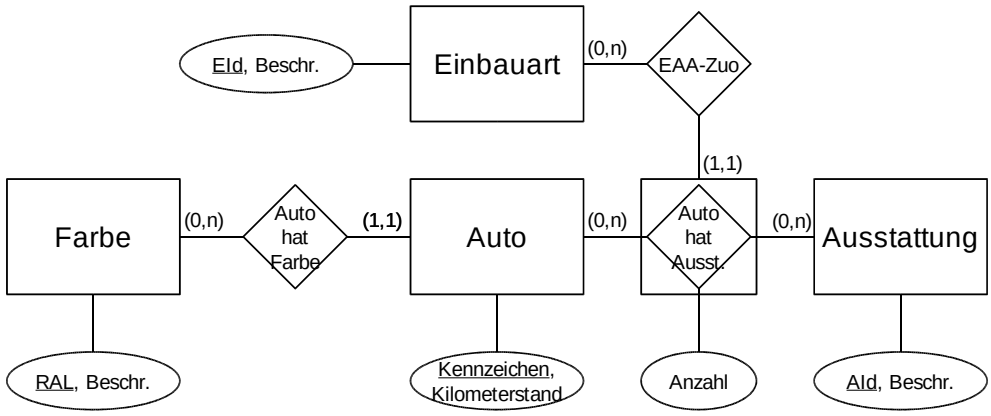


Abbildung 2.4: Beispiel für ein ERM.

dieser Mengen repräsentieren. *Entitytypen* werden im Modell durch Rechtecke dargestellt und repräsentieren reale oder abstrakte Dinge, die für das betrachtete Objektsystem von Interesse sind. Das Beispiel in Abbildung 2.4 enthält die vier Entitytypen Farbe, Auto, Ausstattung und Einbauart.

In der hier verwendeten ERM-Notation werden *Attributtypen* durch Ovale derart dargestellt, dass sich mehrere Attributtypen (jeweils durch Kommata getrennt) innerhalb eines Ovales befinden. Die Zuordnung zwischen Attributtypen und Entitytypen geschieht über eine Linie zwischen den jeweiligen grafischen Repräsentationen, also zwischen Oval und Rechteck. Im Beispiel sind dem Entitytypen Auto die beiden Attributtypen Kennzeichen und Kilometerstand zugeordnet. Demnach hat ein Entity vom Typ Auto zwei Attribute: Kennzeichen und Kilometerstand. Beispiele für Entities vom Typ Auto sind: (MS-DK-4711, 132365) und (HH-AB-1000, 1432). Für den Entitytyp Farbe mit den beiden Attributtypen RAL[16] und Beschreibung sind die folgenden Entities gültige Elemente: (9005, Tiefschwarz), (7011, Eisengrau), (3002, Karminrot) und (1015, Hellelfenbein). Gültige Entities des Typs Ausstattung sind z. B. (1, Radio), (2, Außenspiegel) und (3, Klimaanlage) und des Entitytyps Einbauart: (1, ab Werk) und (2, nachträglich in der Werkstatt). Jeder Entitytyp muss mindestens einen identifizierenden Attributtypen haben, der durch Unterstreichung im grafischen Modell gekennzeichnet wird. Für die Ausprägungen des identifizierenden Attributtypen gilt, dass es von jeder Ausprägung (bzw. Ausprägungskombination im Falle das mehrere Attributtypen unterstrichen sind) maximal eine existiert (somit sind z. B. beim Auto zwei Autos mit demselben Kennzeichen nicht erlaubt).

[16] Das Deutsche Institut für Gütesicherung und Kennzeichnung, das 1925 unter dem Namen Reichs-Ausschuss für Lieferbedingungen (RAL) gegründet wurde, hat einen weltweit anerkannten Maßstab für eindeutige Farbgebung entwickelt. Bei diesem Maßstab wird jeder RAL-Farbe eine eindeutige RAL-Nummer zugewiesen. Somit repräsentiert das Attribut RAL im Beispiel aus Abbildung 2.4 eine solche RAL-Nummer.

Relationshiptypen zwischen Entitytypen schaffen die Verbindungen und Zusammenhänge (Beziehungen) im ERM. Relationshiptypen werden durch Rauten dargestellt, die mit den jeweiligen Entitytypen, die zueinander in Beziehung stehen, über Linien verbunden sind. Von jeder Raute müssen mindestens zwei Kanten ausgehen, die aber auch zu ein und demselben Entitytypen führen können. So setzt im Beispiel (Abbildung 2.4) der Relationshiptyp Auto hat Farbe die beiden Entitytypen Auto und Farbe in Beziehung. Die an den Kanten notierten eingeklammerten Zahlen (inklusive dem Buchstaben n) geben die Kardinalität der Beziehung in Minimal-Maximal Notation an. Im Beispiel mit dem Auto hat Farbe Relationshiptyp sagt die Kardinalität (1,1) beim Auto folgendes aus: Jedes Auto hat minimal eine und maximal eine (also genau eine) Farbe. Die Kardinalität (0,n) hingegen auf Seiten der Farbe sagt aus: Jede Farbe ist minimal Null und maximal beliebig vielen (repräsentiert durch das n) Autos zugeordnet.

Ein Relationshiptyp kann nachträglich zu einem Entitytypen umdefiniert werden, indem um die Raute ein Rechteck gezeichnet wird. Ein solcher *umdefinierter Relationshiptyp* kann wie ein normaler Entitytyp in weitere Beziehungen eingehen. Im Beispiel in Abbildung 2.4 wird der Relationshiptyp Auto hat Ausst., der die beiden Entitytypen Auto und Ausstattung verbindet[17], zu einem Entitytypen umdefiniert. Somit kann Auto hat Ausst. mit dem Entitytypen Einbauart über den Relationshiptypen EAA-Zuo eine Beziehung eingehen.[18]

Ebenso wie Entitytypen haben auch Relationshiptypen Attributtypen, allerdings ist festgelegt, dass die identifizierenden Attributtypen von den identifizierenden Attributtypen der in Bezug gesetzten Entitytypen übernommen werden. In der hier verwendeten Notation werden die identifizierenden Attributtypen zu den Relationshiptypen aus Gründen der Übersichtlichkeit nicht explizit notiert. Im Beispiel mit dem Relationshiptypen Auto hat Farbe sind folgende Beziehungen gültige Elemente: (3002, HH-AB-1000) und (7011, MS-DK-4711). Daraus, dass die identifizierenden Attributtypen der Entitytypen zu den identifizierenden Attributtypen des Relationshiptypen werden folgt, dass eine Beziehung zwischen zwei Entities maximal einmal existieren kann. Zusätzlich zu den identifizierenden Attributtypen können Relationshiptypen weitere Attributtypen haben, die dann expliziert im Modell zu notieren sind. Im Beispiel hat der Relationshiptyp Auto hat Ausst. den nicht identifizierenden Attributtyp Anzahl. Gültige Instanzen dieses umdefinierten Relationshiptypen sind z. B.: (HH-AB-1000, 1, 1), (HH-AB-1000, 2, 2), (HH-AB-1000, 3, 1)[19]. Der Relationshiptyp EAA-Zuo hat drei identifizierende Attribute: Erstens, die beiden identifizierenden Attribute Kennzeichen und AId aus dem Relationshiptypen Auto hat Ausst. und zweitens, das identifizierende Attribut EId aus Einbauart. Gültige Instanzen dieses Relationshiptyp sind z. B.: (HH-AB-1000, 1, 2), (HH-AB-1000, 1, 2), (HH-AB-1000, 3, 1).

[17] Man beachte dabei, dass die Verbindung bis zur Raute durchgezogen ist.

[18] Hier bei ist zu beachten, dass die Verbindung zwischen EAA-Zuo und dem umdefinierten Relationshiptypen nur bis zum Rechteck geht.

[19] Das erste Attribut ist das Kennzeichen des Autos, das zweite die AId der Ausstattung und das dritte ist die Anzahl.

2.2.2 Relationale Datenbanken und SQL

Relationale Datenbanken wurden Anfang der 1980er Jahre kommerziell verfügbar. Sie zeichnen sich durch einen hohen Grad an *physischer Datenunabhängigkeit*, durch *mächtige Sprachen* und ein *einfaches konzeptionelles Modell* aus. Physische Datenunabhängigkeit heißt, dass die physische Speicherung nach außen transparent (im Sinne von „unsichtbar“) ist und dass man sowohl auf der logischen als auch auf der physischen Seite Veränderungen vornehmen kann, ohne dass die jeweils andere Seite davon betroffen ist. Die Mächtigkeit der Sprachen ergibt sich primär daraus, dass eine mengenorientierte Verarbeitung der Daten und nicht die, in den meisten Programmiersprachen übliche, datensatzorientierte Verarbeitung durchgeführt wird. Somit kann sich der Benutzer/Programmierer einer relationalen Datenbank auf sein Ziel (das „Was“) konzentrieren und wird von den meisten Problemen der Verwaltung der Daten (dem „Wie“) entlastet. (Vgl. dazu VOSSEN [155, S. 14f].)

Ein besonderes Merkmal relationaler Datenbanken ist, dass diese das *ACID-Prinzip* vollständig umsetzen.[20] Das ACID-Prinzip ist insbesondere für den Mehrbenutzerbetrieb von großer Bedeutung, weil es die Konsistenz der Daten auch bei einem gleichzeitigen Zugriff auf die Daten durch mehrere Benutzer sichert. Die grundsätzlichen Anforderungen hinter dem ACID-Prinzip sind nach VOSSEN [155, S. 525f]:

1. *Atomarität (atomicity)*: Ein Programm wird aus der Sicht des Benutzers vollständig oder gar nicht ausgeführt.

2. *Konsistenz (consistency)*: Integritätsbedingungen der Datenbank werden eingehalten. Eine Transaktion hinterlässt immer (auch im Fehlerfall) einen konsistenten Zustand, wenn sie in einem solchen gestartet wurde.

3. *Isolation (isolation)*: Mehrere Programme laufen aus logischer Sicht immer isoliert voneinander ab, auch dann, wenn sie parallel gestartet wurden.

4. *Persistenz (durability)*: Falls ein Programm erfolgreich abgeschlossen wurde, dann überleben die von ihm in der Datenbank erzeugten Effekte jeden danach auftretenden Hard- und Softwarefehler.[21]

Aufgrund der genannten Eigenschaften handelt es sich bei relationalen Datenbanken um sehr mächtige Werkzeuge zur Datenhaltung und Datenmanipulation. Daher ist es sinnvoll zu untersuchen, in wie weit sich die Aufgabenstellungen des IR bzw. IF mit Hilfe relationaler Datenbanken umsetzen lassen, mit dem Ziel den Implementierungsaufwand möglichst gering zu halten. Aus diesem Grunde wird diesem Aspekt in den Kapiteln 4 und 5 ein breiter Raum eingeräumt.

[20] Dieses gilt für fast alle kommerziellen und die meisten Open-Source Datenbanken.

[21] Diese Anforderung ist in der Praxis natürlich nicht für alle Umstände erreichbar, aber kommerzielle relationale Datenbanken kommen dieser Anforderung (insbesondere wenn die Daten weltweit verteilt gesichert werden) recht nahe.

Zur Modellierung der Datenstrukturen in relationalen Datenbanken werden auf den Konzepten der mathematischen Relationen und der Relationenalgebra [155, S. 302ff] basierende, relationale Datenbanksprachen verwendet. Eine der gängigsten relationalen Datenbanksprachen ist die *Standard Query Language* (SQL). Eine ausführliche Spezifikation der Sprache, die Standardvorgabe der International Standards Organization, die als *SQL92* bezeichnet wird, findet sich in [73]. Eine allgemeine Dokumentation für Einsteiger findet sich z. B. in [155, S. 339ff]. Datenbankspezifische Dokumentationen sind in den Anwendungshandbüchern der jeweiligen relationalen Datenbanken zu finden. In diesem Buch sind alle SQL-Quelltextauszüge im SQL-Dialekt der Open-Source Datenbank *PostgreSQL* (Version 7.2) formuliert. Eine ausführliche Dokumentation der Sprache findet sich in [115] und [116]. Der SQL-Dialekt von *PostgreSQL* lehnt sich stark an den Sprachstandard SQL92 an, bietet aber darüberhinaus einige zusätzliche Datentypen, wie z. B. den hier verwendeten Datentyp `TEXT` an, der im Unterschied zu den in SQL92 definierten Zeichenkettendatentypen nicht vorab auf eine maximale Anzahl von Zeichen beschränkt werden muss und zur Speicherung von (theoretisch) unbegrenzt langen Zeichenketten verwendet werden kann. [116, S. 27]

2.3 Computerlinguistik

In diesem Abschnitt werden dem Leser die für das Verständnis des Hauptteils der Arbeit notwendigen Grundlagen und Grundbegriffe der Computerlinguistik vorgestellt. Die Auswahl der vorgestellten Themen und der Grad der Detailliertheit erfolgt strikt nach pragmatischen Gesichtspunkten. Für eine ausführlichere Einführung in den Themenbereich empfiehlt sich die Lektüre des deutschsprachigen Lehrbuchs von CARSTENSEN ET AL. [30] und zur Vertiefung von Begriffsdefinitionen der Rückgriff auf das Lexikon der Sprachwissenschaft von BUSSMANN [28].

2.3.1 Phonologie

Die *Phonologie* oder *Lautlehre* ist eine alte Teildisziplin der Linguistik, die sich einerseits mit den Sprachlauten und ihrer Organisation innerhalb größerer linguistischer Einheiten beschäftigt, andererseits aber auch erforscht, wie die konkrete Realisierung der Sprachlaute in verschiedenen Kontexten variiert. Daneben wird auch untersucht, wie sich das lautliche Subsystem zur Morphologie, zum Syntax und zur Semantik verhält. Die *Computerphonologie* ist eine Variante der Phonologie, die die aus der Mathematik und Informatik stammenden Methoden der Computerlinguistik auf phonologische Probleme anwendet. [30, S. 136f]

Bei den in dieser Arbeit untersuchten Verfahren zum IF und IR wird davon ausgegangen, dass die zu verarbeitenden Dokumente in geschriebener Form vorliegen. Daher spielt die Phonologie für die weiteren Betrachtungen eine eher untergeordnete Bedeutung und wird nicht weiter thematisiert.

2.3.2 Morphologie

In vielen natürlichen Sprachen erscheinen Wörter in verschiedenen Formen (z. B. Hund: Hunde, Hunden, Hundes) und dienen gleichzeitig als Ausgangspunkt für neue Wörter (wie z. B. vom Hund ausgehend: Hündchen, Haushund). Der Bereich der (Computer-)Linguistik, der sich mit diesem Phänomen beschäftigt und es systematisiert, ist die *Morphologie*. Im Folgenden werden Begriffe aus der Morphologie in Anlehnung an [30, S. 175ff] vorgestellt und es wird anhand dieser Begriffe die wortformbedingte Problematik der Verarbeitung von natürlicher Sprache verdeutlicht.

2.3.2.1 Flexion, Komposition und Derivation

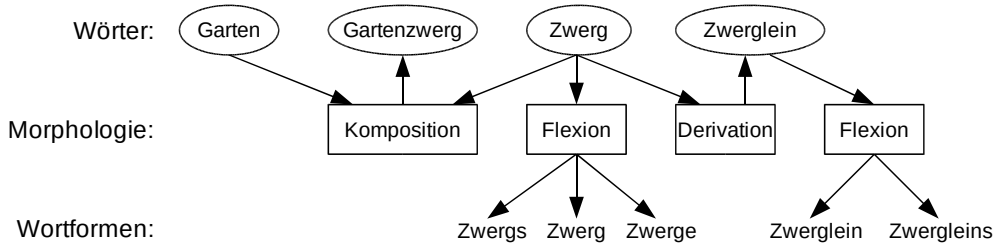
Ein *Wort* wird im Folgenden als eine abstrakte Einheit bezeichnet, die verschiedenen Formen zu Grunde liegt und die dem Eintrag eines Wörterbuchs entspricht. Häufig gebrauchte Synonyme für Wort sind *Lemma* oder *Lexem*. Beispiele für Wörter sind: Haus, gehen, schnell. Im Gegensatz dazu sind *Wortformen* die verschiedenen grammatikalischen Formen eines Wortes. Z. B. sind Häuser und Häusern Wortformen des Wortes Haus ebenso wie ging, gehe und gegangen die Wortformen des Wortes gehen und schneller und schnellsten die Wortformen von schnell sind. Die Bildung verschiedener Wortformen wird als *Flexion* (lateinisch für Beugung) bezeichnet und die Menge der Wortformen zu einem Wort heißt *Pradigma*.

Neben der Flexion gibt es in der Morphologie auch die *Wortbildung*, die in *Komposition* und *Derivation* spezialisiert werden kann. Bei der Komposition wird ein neues Wort auf der Basis mehrere Wörter gebildet. Das neue Wort (*Kompositum*) erhält dabei eine Bedeutung, die sich aus der Kombination der beiden Begriffe unter Berücksichtigung der sprachspezifischen Interpretationsregel für Komposita ergibt. Im Deutschen sieht die Interpretationsregel im Allgemeinen so aus, dass der rechts-stehende Begriff durch die links-stehenden Begriffe spezialisiert wird. So wird das Kompositum Haushund als ein Spezialfall eines Hundes interpretiert, der im Haus lebt. Donauschifffahrtsgesellschaft ist ein Spezialfall einer Gesellschaft, die sich mit Schifffahrt beschäftigt und die wiederum auf das Betätigungsgebiet der Donau spezialisiert wird.

Bei Derivationen handelt es sich um die Erstellung neuer Wörter unter Verwendung eines einzigen Wortes als Ausgangsbasis. Beispiele für Derivationen zu Zwerg sind zwergenhaft und Zwerglein. Im Allgemeinen handelt es sich bei der Derivation entweder um eine bestimmte Form der Spezialisierung (Zwerglein = kleiner Zwerg) oder um die Umgestaltung eines Worttypen (im Beispiel mit dem Zwerg von Substantiv zum Adjektiv zwergenhaft). Eine Übersicht über die Zusammenhänge zwischen Flexion, Komposition und Derivation gibt Abbildung 2.5.

2.3.2.2 Stemming (Normalisierung)

Bei der automatisierten Verarbeitung von Dokumenten zur Lösung von IF und IR Aufgaben, bereiten die verschiedenen Wortformen von Wörtern Probleme. So kann ein Rechner durch



Quelle: [30, S. 176].

Abbildung 2.5: Flexion, Komposition und Derivation im Überblick.

einen einfachen Zeichenkettenvergleich nicht feststellen, dass z. B. die in einem Dokument gefundene Wortform Hunde zu dem gesuchten Wort Hund passt. Aus diesem Grunde werden die Wortformen eines Dokuments und die Wortformen in Anfragen bzw. Profilen normalisiert. Unter *Stemming* (bzw. *Normalisierung*) wird im Folgenden das Zurückführen einer Wortform auf einen Wortstamm (*Strong-Stemming*) oder das Zurückführen der Wortform auf das jeweilige Wort in Grundform (als *Lemmatisierung* oder *Weak-Stemming* bezeichnet) verstanden. Bei der Normalisierung ist es jedoch wichtig, dass sie auf dieselbe, konsistente Weise für alle Worte durchgeführt wird: also entweder Strong-Stemming oder Weak-Stemming. Grundsätzlich gibt es drei Möglichkeiten, ein Verfahren für die Normalisierung zu konzipieren: Lexikon-basiert, Algorithmen-basiert und kombiniert.

Ein algorithmisch relativ einfaches Vorgehen zur Normalisierung ist die Verwendung eines *Lexikon-basierten* Verfahrens. Bei diesen Verfahren wird eine Tabelle angelegt, die in jeder Zeile einer Wortform das passende Wort bzw. den passenden Wortstamm zuordnet. Der Vorteil dieses Verfahren ist, dass es relativ einfach ist und dass es aus theoretischer Sicht fehlerfrei funktionieren kann. Das Problem bei diesem Vorgehen ist, dass die Tabelle manuell zu pflegen ist. Dieses ist ein aufwändiges Unterfangen, insbesondere in Kombination mit der Tatsache, dass die Tabelle dazu neigt, sehr groß zu werden. Besonders die deutsche Sprache zeigt sich in diesem Zusammenhang als problematisch, weil durch die starke Verwendung von Komposita im Deutschen die Anzahl der zu pflegenden Wortform durch die Zahl der möglichen Wort-Kombinationen relativ hoch ist.

Bei den *Algorithmen-basierten* Verfahren wird üblicherweise eine Menge von Ersetzungsregeln definiert, die auf einer Wortform (ggf. in mehreren Durchläufen) angewandt werden. Diese Ersetzungsregeln sind in etwa vom folgenden Schema: Wenn eine Wortform auf „e“ endet (z. B. Hunde), dann wird das „e“ abgeschnitten um den Wortstamm zu erhalten (also Hund). Das Problem bei den Algorithmen-basierten Verfahren ist, dass die Regeln sprachabhängig zu definieren sind und dass die Regeln bei Sprachen mit anspruchsvoller Morphologie schwer aufzustellen sind. Z. B. erfordert die Rückführung von Häuser in Haus einen erheblich höheren Regelaufwand als das Beispiel mit dem Hund. Zudem können Algorithmen-basierte Verfahren aus theoretischer Sicht nicht fehlerfrei sein, wenn die Sprache Worte hat,

die unregelmäßig gebeugt werden. In diesem Falle führen die Regeln häufig zu einem *Over-Stemming* bzw. *Under-Stemming*. Das heißt, dass die zu normalisierende Wortform gegenüber dem eigentlichen Wort entweder zu viele oder zu wenige Buchstaben hat.

Eine Möglichkeit, die Nachteile beider Verfahren abzumildern ist die Anwendung eines *kombinierten* Verfahrens. Bei diesem Verfahren wird (im einfachsten Fall) bei einer zu normalisierenden Wortform zuerst überprüft, ob ein passender Eintrag im Lexikon vorhanden ist. Ist das nicht der Fall, dann wird die Wortform unter Anwendung von Regeln normalisiert. Der Vorteil liegt auf der Hand: unregelmäßig gebeugte Wörter werden im Lexikon erfasst und die regelmäßig gebeugten werden über das Regelwerk normalisiert. Somit nimmt die Zahl der Einträge im Lexikon gegenüber dem Lexikon-basierten Verfahren ab, ebenso wie die Fehlerquote gegenüber dem Regel-basierten Verfahren.

Da die englische Sprache eine relativ regelmäßige Morphologie hat, werden in der Praxis für das englische Regel-basierte Verfahren bevorzugt. Klassische Vertreter dieser Verfahren sind der Successor Variety Stemmer [64], der n-Gram Stemmer [3] und die Affix Removal Stemmer [90, 126, 41, 109] zu denen auch das sehr weit verbreitete Verfahren von PORTER [114] gehört. Aufgrund der unregelmäßigen Morphologie der deutschen Sprache ist es empfehlenswert, für das Stemming Lexikon-basierte Verfahren anzuwenden, die auf vorhandenen Vollform-Lexikas wie z. B. dem *CISLEX*[22] oder dem *Wortschatz-Lexikon*[23] aufbauen. Ein bekanntes kombiniertes Lemmatisierungsverfahren für die deutsche Sprache ist *Morphy*[24]. Eine Beschreibung von rein Algorithmen-basierten Verfahren findet sich in [14] und in [1].

2.3.3 Syntax

Dieser Abschnitt gibt einen Überblick über das Vorgehen bei der Verarbeitung von syntaktischen Strukturen in natürlichsprachlichen Sätzen und begründet, warum die Syntax von Sätzen bei gängigen IF- und IR-Systemen nicht berücksichtigt wird. Unter *Syntax* wird in diesem Zusammenhang ein System von Regeln verstanden, die beschreiben, wie aus einem Inventar von Grundelementen durch spezifische (syntaktische) Mittel alle wohlgeformten Sätze einer Sprache abgeleitet werden können. [28, S. 676]

2.3.3.1 Syntaktische Strukturen und formale Grammatiken

Kontextfreie Grammatiken[25] (nach CHOMSKY auch Typ-2-Grammatiken genannt) sind – zumindest innerhalb der Tradition der Konstituentenstruktur-orientierten[26] Grammatikmodelle – nach wie vor das Basisinstrument für syntaktische Analysen, wenngleich sie heutzutage nur

[22] CISLEX: `http://www.cis.uni-muenchen.de/projects/CISLEX.htm`
[23] Vgl. Abschnitt 6.1.2.1.
[24] Morphy: `http://www-psycho.uni-paderborn.de/lezius`
[25] Vgl. HOPCROFT und ULLMANN [70, 71] sowie BUCHER und MAURER [27] bzw. SCHÖNING [132].
[26] Im Gegensatz zu den Dependenz- und Determinations-orientierten Grammatiken, bei denen syntaktische Strukturen ausschließlich als Relationen zwischen Wörtern aufgefasst werden, betrachten Konstituentenstruktur-orientierte Grammatiken neben Wörtern auch komplexere Einheiten (so genannte Konstituenten) und erlauben zusätzlich Relationen zwischen den komplexeren Einheiten. [30, S. 204]

noch selten in reiner Form verwendet werden. Zumeist bildet eine kontextfreie Grammatik das Grundgerüst (oder Skelett) eines Systems, das auch andere Elemente, z. B. statistische Bewertungen oder komplexe Kategorien enthält. [30, S. 205] Daher soll hier ein Einblick in das Thema am Beispiel einer einfachen, kontextfreien Grammatik gegeben werden. Eine kontextfreie Grammatik $G = (\Phi, \Sigma, R, S)$ zur Repräsentation von natürlichsprachlichen Sätzen besteht aus

1. einer Menge von Nichtterminalsymbolen Φ, die typischerweise syntaktische Kategorien wie z. B. NP (Nominalphrase), PP (Präpositionalphrase), DET (Determinierer) und Wortartenkategorien wie z. B. V (Verb) und N (Nomen) enthält.

2. einer Menge von Terminalsymbolen Σ. Diese Menge enthält alle nicht weiter zerlegbaren Ausdrücke der Grammatik. Typischerweise handelt es sich dabei um die Wörter der zu beschreibenden Sprache.

3. einer Regelmenge R, die endlich viele Regeln der Form

$$A \rightarrow \alpha \quad \text{mit} \quad A \in \Phi, \alpha \in (\Phi \cup \Sigma)^*$$

   enthält. Dabei ist A ein Nichtterminalsymbol und α eine Kette von Symbolen aus Φ oder Σ.

4. einem Startsymbol $S \in \Phi$ aus der Menge der Nichtterminalsymbole.

Ein Beispiel (in Anlehnung an [30, S. 206f]) für eine derartige Grammatik ist die folgende Grammatik G:

$$
\begin{aligned}
G = (&\{\mathrm{S}, \mathrm{NP}, \mathrm{VP}, \mathrm{DET}, \mathrm{N}, \mathrm{V}\},\\
&\{\mathsf{der}, \mathsf{Hund}, \mathsf{bellt}, \mathsf{sieht}, \mathsf{die}, \mathsf{Katze}\},\\
&\{\ \mathrm{S} \rightarrow \mathrm{NP\ VP},\\
&\quad \mathrm{NP} \rightarrow \mathrm{DET\ N},\\
&\quad \mathrm{VP} \rightarrow \mathrm{V},\\
&\quad \mathrm{VP} \rightarrow \mathrm{V\ NP},\\
&\quad \mathrm{DET} \rightarrow \mathsf{der},\\
&\quad \mathrm{DET} \rightarrow \mathsf{die},\\
&\quad \mathrm{N} \rightarrow \mathsf{Hund},\\
&\quad \mathrm{N} \rightarrow \mathsf{Katze},\\
&\quad \mathrm{V} \rightarrow \mathsf{bellt},\\
&\quad \mathrm{V} \rightarrow \mathsf{sieht}\ \},\\
&\mathrm{S})
\end{aligned}
$$

Mit dieser Grammatik können folgende, grammatikalisch korrekten Wortketten der deutschen Sprache abgeleitet werden (vgl. Abbildung 2.6 und 2.7):

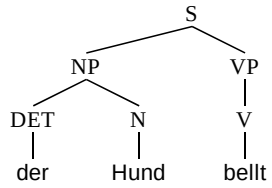
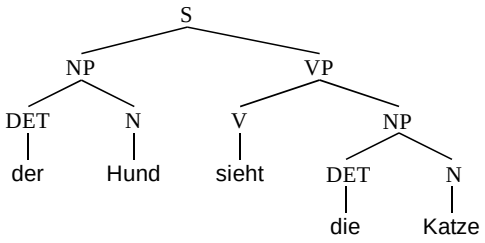


Abbildung 2.6: Ableitung zu der Hund bellt.



Quelle: [30, S. 207]

Abbildung 2.7: Ableitung zu der Hund sieht die Katze.

1. der Hund bellt
2. der Hund sieht die Katze

Allerdings ermöglicht die Grammatik auch die Ableitung von *grammatikalisch falschen* Wortketten wie z. B.

3. der Hund bellt die Hund (vgl. Abbildung 2.8)

Um ungrammatikalische Ausdrücke (wie z. B. die Wortkette 3) ausschließen zu können, müssen neue Nichtterminalsymbole zu unserer Beispielgrammatik G hinzugefügt werden, um sie in den folgenden (ausgewählten) Punkten differenzierter zu gestalten:

- Die einheitliche Repräsentation der Verben durch V muss in transitive und intransitive Verben V_t und V_i aufgespalten werden, weil nur die transitiven Verben (wie z. B. sehen) ein Subjekt mit einem Objekt in Verbindung setzen können.
- Um nicht-wohlgeformte Ausdrücke wie z. B. die Hund ausschließen zu können, müssen Genus-Informationen sowohl in den Artikeln (DET) als auch in die Nomina (N) kodiert werden. Insofern ist es erforderlich, drei Kategorien der Artikel und der Nomina zu unterschieden: jeweils eine für Maskulina, Feminina und Neutra. Also ergeben sich: DET_m, DET_f, DET_n und N_m, N_f, N_n.

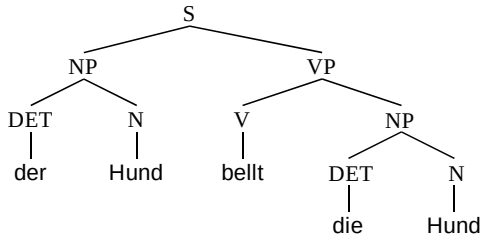


Abbildung 2.8: Ableitung zu der Hund bellt die Hund.

- Abschließend benötigen wir Kasus-Informationen. Somit muss z. B. N_m erneut in mehrere Kategorien aufgeteilt werden: je eine für Nominativ, Genitiv, Dativ und Akkusativ. Also: N_{mn}, N_{mg}, N_{md} und N_{ma}. Dieses ist analog für alle Genus-Fälle der Nomina und Artikel fortzuführen.

Passend zu den neuen Nichtterminalsymbolen müssen die vorhandenen Regeln angepasst und erweitert werden. Wie man sich leicht vorstellen kann, vergrößert sich die Zahl der Nichtterminalsymbole als auch die Anzahl der benötigten Regeln explosionsartig. Aus diesem Grunde wurde an neuen Grammatiktypen, den Unifikationsgrammatiken geforscht, die die kontextfreie Grammatik von CHOMSKY erweitern. Durch diese Grammatiken wird das Komplexitätsproblem bei der Erstellung und Repräsentation von natürlichsprachlichen Grammatiken reduziert. Zu den wichtigsten Vertretern dieser Grammatiken zählen die *Generalized Phrase Grammar* [54], die *Lexical Functional Grammar* [25], *PATR-II* [135] und die *Head-Driven Phrase Structure Grammar* [111]. Eine Übersicht über gängige Grammatiken und Systeme findet sich in [150].

2.3.3.2 Automatisierte Analyse syntaktischer Strukturen

Unter dem Begriff *Parsing* wird die automatische Analyse sprachlicher Ausdrücke verstanden. Als *Syntax-Parsing* wird daher die automatisierte Analyse von sprachlichen Ausdrücken in Bezug auf ihre Syntax, also z. B. die Ableitung eines Syntax-Baumes basierend auf einer kontextfreien Grammatik, bezeichnet. Dabei handelt es sich allgemein formuliert um einen Suchprozess, bei dem ein (graphentheoretischer) Suchraum durchlaufen wird. Ausgangspunkt des Suchprozesses ist die zu parsende Eingabekette von Zeichen (also z. B. ein Satz oder ein Dokument) und die der Eingabekette zu Grunde liegende Grammatik. Der Endpunkt dieses Prozesses ist idealerweise ein gültiger Ableitungsbaum (wie z. B. Abbildung 2.7) zu der Eingabekette, der die innere Struktur der Eingabekette offenbart. Eine ausführliche und mathematisch-formale Beschreibung dieses Suchprozesses ist in der gängigen Literatur zu den CHOMSKY-Grammatiken beschrieben: [70, 71, 27, 132].

Im Unterschied zu dem Syntax-Parsing von formalen Sprachen (wie z. B. den gängigen Programmiersprachen) treten bei dem Syntax-Parsing von natürlichen Sprachen drei bisher

nur unzureichend gelöste Probleme auf: *Ambiguität*, *Abdeckung* und *Effizienz*.

Ambiguität: Die Anzahl der syntaktischen Lesarten von ganz gewöhnlichen Sätzen, die von größeren Parsing-Systemen geliefert wird, ist zumeist erheblich höher als der Ambiquitätsgrad, den selbst geschulte Syntaktiker auf den ersten Blick erkennen. [30, S. 227] Ein klassiches Beispiel dafür ist der folgende (in der Literatur gerne gebrachte) englische Satz:

> Time flies like an arrow.

Diesen Satz würde ein Mensch (ohne weiteren Kontext) wortwörtlich als

> Zeit fliegt wie ein Pfeil.

oder sinngemäß und eingedeutscht als

> Die Zeit fliegt dahin.

übersetzen bzw. interpretieren. Die meisten Syntax-Parser würden neben dieser syntaktisch und semantisch korrekten Interpretation auch die beiden folgend genannten, syntaktisch korrekten aber semantisch (zumindest ohne entsprechenden Kontext) falschen Interpretationen liefern:

> Zeitfliegen mögen einen Pfeil.

und

> Bestimme die Geschwindigkeit von Fliegen so, wie es ein Pfeil tut.

Nicht nur ungewöhnliche Sätze sind ambig, sondern auch normale, häufig vorkommende Aussagen bereiten gängigen Syntax-Parsern Probleme. So liefert z. B. das Parsing-System des *ParGram*-Projektes für den Satz

> In der Stadt fehlen gemütliche Kneipen.

zwei Interpretationen und für den Satz

> Hinter dem Betrug werden die gleichen Täter vermutet, die während der vergangenen Tage in Griechenland gefälschte Banknoten in Umlauf brachten.

sogar 92 Interpretationen. (Vgl. [84, S. 6].)

Es gibt verschiedenste Ansätze, das Problem der Ambiguität zu mildern (vgl. dazu [30, S. 223ff]), so versucht z. B. das *VerbMobil*-System [156] die Ambiguität von Sätzen durch das Einbeziehen von Semantik, Pragmatik und Kontext zu reduzieren. Dennoch ist der derzeitige Stand der Technik der, dass die Ambiguität von Sätzen für umfassende domänenunabhängige Grammatiken nicht auszuschließen ist. [30, S. 230]

Abdeckung: Ein großes Problem, neben der vorkommenden Ambiguität von Sätzen ist der Abdeckungsgrad der in Syntax-Parsern verwendeten Grammatiken. Es gibt wenig Literatur, die Resultate zum Abdeckungsgrad von Parsing-Systemen vorstellt. Somit ist es häufig schwer abzuschätzen wie die Performanz der entwickelten Systeme in Bezug auf Realdaten ist. (Vgl. [30, S. 231]) Einen Eindruck über die Situation vermittelt jedoch WAUSCHKUHN in [157]. In diesem Artikel hat der Autor seinen partiellen Syntax-Parser an 72041 Sätzen der *Stuttgarter Zeitung* getestet. Der Parser hat für 85,7% aller Sätze zumindest einen unvollständigen Ableitungsbaum erstellen können. Ein vollständiger Ableitungsbaum konnte lediglich für 56,5% aller Sätze erstellt werden und nur für 50,2% aller Sätze war das Analyseergebnis eindeutig.

Effizienz: Die Effizienz eines Syntax-Parsers ist insbesondere in Bezug auf IF und IR Aufgaben von hoher Relevanz, weil viele Dokumente bestehend aus einer größeren Anzahl von Sätzen verarbeitet werden müssen. Für Typ-2 Grammatiken ist bekannt, dass die Komplexität des Parsingproblems im schwierigsten Fall (*worst case*) nicht schneller als mit dem kubischem Verlauf (n^3) zur Basislänge der Eingabekette (n) ansteigt. [70, 71, 27, 132]

In der Praxis sind mit gängigen Hard- und Softwaresystemen Parsingzeiten von mehreren Sekunden für einen Satz üblich. Der partielle Parser[27] von WAUSCHKUHN benötigt im Durchschnitt für einen Satz 1,3 Sekunden. [157] Im Unterschied dazu benötigt der *Gepard-Parser* von LANGER, der im Gegensatz zum Parser von WAUSCHKUHN ein vollständiger Parser ist, für einen Satz wie z. B.

> Es klappte gut, weil Maria die Freundin von Anna aus Osnabrück mit dem Auto von Petra aus Bielefeld abgeholt wurde.

etwas weniger als 7 Sekunden. [86] Somit ist damit zu rechnen, dass alleine das Parsing eines kürzeren Nachrichtenartikels mehrere Minuten in Anspruch nimmt. Dieses dürfte für IF und IR-Systeme mit einem hohem Dokumentenvolumen pro Zeiteinheit (wie z. B. Internet-Suchmaschinen oder personalisierte Nachrichtenfiltern für größere Benutzergruppen) ein nicht vertretbarer Zeitaufwand sein.

Fazit: Als Fazit aus der Betrachtung des gegenwärtigen Standes der Technik des Syntax-Parsing kann folgendes gezogen werden: Die Anwendung von *Syntax-Parsern* für das IF und IR, ist zum jetzigen Zeitpunkt aus zwei Gründen *nicht praktikabel*:

1. Die Qualität der Syntax-Parser in Bezug auf eine eindeutige Interpretation und den Abdeckungsgrad ist zu gering.
2. Die Hardware-Anforderungen (insbesondere der notwendige Bedarf an Rechenkapazität) des Syntax-Parsing sind für einen Einsatz für IR- und IF-Aufgaben, bei denen viele tausend Dokumente mit vielen Sätzen in möglichst kurzer Zeit geparst werden müssen, zu hoch.

[27] *Partielle Parser* sind im Unterschied zu *vollständigen Parsern* vom Grundsatz her derart konzipiert, dass sie nicht alle Feinheiten einer natürlichen Grammatik (wie z. B. Verbvalenzen oder Relativsatzanbindungen) – auf Kosten eines vollständigen Parsings – berücksichtigen.

2.3.4 Semantik

Der Begriff der *Semantik*[28] ist von BRÉAL [24] für diejenige Teildisziplin der Linguistik geprägt worden, die sich mit der Analyse und Beschreibung der sogenannten „wörtlichen“ Bedeutung von sprachlichen Ausdrücken beschäftigt. [28, S. 590] Je nach Forschungsinteresse können verschiedene Aspekte der Bedeutung im Vordergrund stehen, die die verschiedenen Teilgebiete der Semantik prägen. Man kann u. a. zwischen der *Satz-* und *Diskurssemantik*, sowie der *lexikalischen Semantik* unterscheiden. Im Folgenden werden die Satz- und Diskurssemantik nur kurz und lediglich der Vollständigkeit halber erwähnt, weil diese von Syntax-Parsern abhängen und somit derzeit in der Praxis nicht zielführend im Zusammenhang mit IF- und IR-Systemen eingesetzt werden können.[29] Auf die lexikalische Semantik wird hingegen ausführlicher eingegangen, wobei der Fokus auf der Definition von Begriffen liegt, die im Bereich der Linguistik zur Beschreibung von Zusammenhängen zwischen Wörtern verwendet werden und für das IF und IR von Bedeutung sind.

2.3.4.1 Satz- und Diskurssemantik

Die *Satzsemantik* versucht die Bedeutung von natürlichsprachlichen Sätzen zu erfassen. Das bis heute noch aktuelle Fundament dieser Teildisziplin ist in den 1970ern von dem amerikanischen Logiker und Sprachtheoretiker RICHARD MONTAGUE in drei Aufsätzen [102, 103, 104] niedergelegt worden. Die MONTAGUE-Semantik verbindet Syntax und Semantik natürlichsprachlicher Sätze systematisch mit der mathematischen Logik und ermöglicht somit die Darstellung von natürlichsprachlichen Aussagen in einem mathematischen Modell. Dieses ist möglich, weil – laut MONTAGUE – zwischen natürlichen und formalen Sprachen kein wesentlicher Unterschied besteht. Bei der Ableitung von Satzsemantik schränkt das *Kompositionalitätsprinzip* die Verbindung zwischen Syntax und Semantik ein. Demnach sind die syntaktische Ableitung eines Satzes und die semantische Ableitung parallel aufgebaut. Zu jeder syntaktischen Regel die einen komplexen Ausdruck aus einfacheren Ausdrücken bildet, gibt es eine entsprechende semantische Regel, die eine Bedeutung aus Teilbedeutungen zusammenstellt. (*rule-to-rule* Hypothese, vgl. [6]) Aus diesem Zusammenhang wird klar, dass die Anwendung von Verfahren zur Interpretation von Satzsemantik ohne die Anwendung von Syntax-Parsern nicht möglich ist. Die Anwendung von Syntax-Parsern im Bereich des IF und IR ist aber, wie bereits in Abschnitt 2.3.3.2 festgestellt wurde, mit der jetzigen Technik aufgrund des hohen Rechenaufwands in der Praxis nicht möglich.

Mit der Bedeutung von ganzen Dokumenten (Diskursen) beschäftigt sich die Teildisziplin der *Diskurssemantik*. Das Fundament bildet hier die *Diskursrepräsentationstheorie* die in den 1980ern von HANS KAMP [78, 79] entwickelt wurde. Diese Theorie postuliert zur semantischen Interpretation von Diskursen – im Unterschied zur der MONTAGUE-Semantik – nicht eine direkte Beziehung zwischen syntaktischer Analyse und dem semantischen Modell, sondern fügt eine Zwischenebene (die *Diskursrepräsentation*) ein. Dennoch wird die Diskurssemantik, wenn auch über den Umweg der Diskursrepräsentation, aus der syntaktischen Ana-

[28] Gelegentlich auch als *Semasiologie* bezeichnet.
[29] Vgl. dazu die Argumentation in Abschnitt 2.3.3.2.

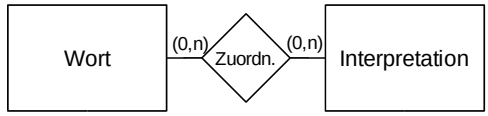


Abbildung 2.9: Verhältnis von Wörtern und (Wort-)Interpretationen

lyse abgeleitet. Dadurch ist die Diskurssemantik für praxisnahe IF- und IR-Systeme derzeit aus denselben Gründen wie die Satzsemantik nicht anwendbar.

2.3.4.2 Lexikalische Semantik

Die Frage, welche Bedeutung einzelnen Wörtern zu Grunde liegt, wird von der *lexikalischen Semantik* gestellt. Im Unterschied zu der Diskurs- und Satzsemantik hat die lexikalische Semantik für IF- und IR-Systeme eine praktische Relevanz, weil sie mit relativ geringem Rechenaufwand (im Vergleich zum Syntax-Parsing) die Such- und Filtering-Ergebnisse verbessern kann. Die grundlegende Erkenntnis hinter der lexikalischen Semantik ist die, dass es für einzelne Interpretationen (Bedeutungen) mehrere Wörter (*Synonymie*) und für einzelne Wörter mehrere Interpretationen (*Polysemie*, *Homonymie* und *Metonymie*) gibt. Abbildung 2.9 zeigt diesen Sachverhalt in ERM-Notation. Wie in der Abbildung gezeigt, kann es Wörter ohne direkten Bezug zu einer Interpretation geben. Dieser auf den ersten Blick verwunderliche Sachverhalt wird klar, wenn man sich gängige Präpositionen wie z. B. der, die und das bzw. im Englischen the, a und an anschaut. Diese Worte haben aus grammatikalischer Sicht eine Bedeutung beziehen sich aber nicht auf Objekte einer realen oder gedachten Welt. Insofern haben diese Wörter aus Sicht der lexikalischen Semantik keine explizite Bedeutung und werden in vielen IF- und IR-Systemen bei der Verarbeitung ignoriert. Dieses geschieht üblicherweise dadurch, dass diese Worte anhand von sogenannten *Stoppwortlisten* identifiziert (wie z. B. über die englischsprachige Liste von DROTT [46]) und vor der Verarbeitung aus den Dokumenten entfernt werden.

Umgekehrt ist es ebenfalls möglich, dass zu einer Interpretation noch kein Wort existiert. So haben Objekte kurz nach ihrer Entdeckung oder Erfindung häufig noch keinen eigenen Namen und müssen umschrieben werden. Es existiert somit eine Interpretation, aber dieser Interpretation ist (noch) kein Wort zugeordnet. Die im Folgenden genannten Phänomene der lexikalischen Semantik können in Ontologien[30] abgebildet und mit ihrer Hilfe erkannt werden. Dadurch können diese Phänomene von IF- und IR-Systemen in geeigneter Weise verarbeitet werden.

Synonymie bezeichnet die Eigenschaft, das mehrere Wörter dieselbe Interpretation haben können. Ein *Synonym* ist demnach ein Wort, das trotz unterschiedlicher Benennung mit ei-

[30] Der Begriff der Ontologie wird in Abschnitt 2.4 ausführlich definiert.

nem anderen Wort dieselbe Interpretation hat.[31] [147] Gängige Beispiele für Synonyme des täglichen Gebrauchs sind:

- Auto, Automobil, Wagen
- Computer, Rechner
- WWW, World-wide Web

Wie das letzte Beispiel zeigt, werden hier auch Abkürzungen den Synonymen zugerechnet. Dieses ist zum Einen möglich, weil das originale Wort (bzw. die Wortphrase) und die Abkürzung eine unterschiedliche Schreibweise haben, aber dasselbe meinen und somit die Definition für Synonymie erfüllen. Zum Anderen ist dieses sinnvoll, weil Abkürzungen in vielen Fällen (wie z. B. im Fall von WWW) sich sogar zu dem führenden Wort entwickeln, der im Sprachgebrauch häufiger genutzt wird als das Original.

Synonyme stellen für IF- und IR-Systeme ein Problem dar, weil im Falle einer nicht Beachtung von Synonymie-Eigenschaften von Begriffen, zwei Synonyme von einem System nicht als identisch in Bezug auf ihre Interpretation erkannt werden. Die daraus resultierende Konsequenz ist die, dass z. B. im Falle von IR die Anfrage suche Dokumente zum Thema Auto nicht korrekt beantwortet wird, weil z. B. Dokumente in denen nur von Wagen gesprochen wird, dem Benutzer nicht zurückgeliefert werden.

Polysemie und Homonymie: Ein weiteres Phänomen der lexikalischen Semantik, ist die Variabilität von Wortinterpretationen, die sich in zwei verschiedene, aber nicht immer eindeutig unterscheidbare Klassen unterteilt. Unter *Polysemie* wird die Eigenschaft von Wörtern bezeichnet, auf verschiedene Entitäten zu referenzieren, die aber semantisch zueinander in Bezug stehen. [30, S. 292] [28, S. 524] Klassische Beispiele für Polyseme sind die Nomen Schule und Zeitung (vgl. [18, 44]):

- Die Schule liegt an der Hauptstraße. (Schule als Gebäude)
- Die Schule prägt die Kinder von frühen Jahren an. (Schule als Prinzip der Erziehung und Wissensvermittlung)
- Die Schule erteilte einen Verweis. (Schule als Institution)
- Die Zeitung wurde im Jahre 1949 gegründet. (Zeitung als Institution)

[31] Generell kann man zwischen der *Totalen Synonymie* und der *Partiellen Synonymie* unterscheiden. Unter der Totalen Synonymie versteht man eine uneingeschränkte Austauschbarkeit der betreffenden Wörter in allen Kontexten. Allerdings spielt die Totale Synonymie in der praktischen Anwendung von Sprachen aufgrund des Prinzips der Sprachökonomie kaum eine Rolle, weil sie kaum existiert. Unter der Partiellen Synonymie ist zu verstehen, dass zwei Worte in bestimmten Kontexten synonym verwendet werden können und in anderen Kontexten nicht. [28, S. 673f] Z. B. sind Rechner und Computer synonym, wenn mit ihnen ein handelsüblicher Desktop-Rechner gemeint ist. Ist mit Rechner jedoch ein Taschenrechner gemeint, dann ist im allgemeinen Sprachgebrauch Computer in diesem Kontext der falsche Begriff und kann somit nicht synonym verwendet werden.

- Die Zeitung liegt auf dem Tisch. (Zeitung als Objekt)
- Die Zeitung wird gerne gelesen. (Zeitung als Informationsmedium)

Unter *Homonymie* versteht man Wörter, die sich hinsichtlich ihrer Orthographie und Aussprache gleichen und die mehrere unterschiedlichen Interpretationen, die in keinem semantischen Zusammenhang zueinander stehen, haben. [28, S. 283f] [30, S. 293] [147] Spezialfälle der Homonymie sind die *Homographie*, bei der lediglich die gleiche Orthographie zweier Wörter betrachtet wird und die *Homophonie*, bei der nur die gleiche Aussprache der Wörter untersucht wird. Klassische Beispiele für deutsche Homonyme sind die Wörter Maus als Computereingabegerät bzw. kleines Nagetier und Bank als Sitzgelegenheit bzw. Geldinstitut. Im Englischen gibt es zahlreiche Homophone die zwar unterschiedlich geschrieben aber gleich oder sehr ähnlich ausgesprochen werden. Beispiele dafür sind ads (Plural der geläufigen Abkürzung ad für Werbung) und adds (das Ausführen der Additionsoperation) sowie sole (nur) und soul (Seele).[32]

Zur Abgrenzung von Polysemie und Homonymie wird traditionell das Kriterium der etymologischen Verwandtschaft herangezogen. Das heißt, dass die historischen Wurzeln von Wörtern zu untersuchen sind. Homonyme Worte lassen sich dabei auf verschiedene und polyseme Worte auf nur eine Wurzel zurückführen. [28, S. 524] Das etymologische Kriterium ist insofern problematisch, als nicht zu klären ist, wie weit die historische Herkunft zweier Worte für eine eindeutige Abgrenzung zwischen Polysemie und Homonymie zurückverfolgt werden soll. [28, S. 283] Aus pragmatischer Sicht ist für IF- und IR-Systeme lediglich relevant, ob einem Wort zwei unterschiedliche Interpretationen zugeordnet werden. Es ist dabei nicht von Interesse, ob den Worten eine etymologische Verwandtschaft nachgewiesen werden kann oder nicht. Da zudem lediglich die Orthographie zur Unterscheidung von Wörtern herangezogen werden kann, wird in der Arbeit folgende vereinfachende Arbeitsdefinition für den Begriff Homographie verwendet: *Homographie* liegt genau dann vor, wenn einem Wort, unabhängig von etymologischen Betrachtungen, mehr als eine Interpretation zugeordnet ist. Ein *Homograph* ist demnach ein Wort mit mindestens zwei verschiedenen Interpretationen. Eine Unterscheidung zwischen Polysemen und Homographen wird somit nicht vorgenommen. Ein Verfahren, das in der Lage ist anhand des Kontextes zu entscheiden, welche der verschiedenen Interpretationen eines Homographen im jeweiligen Kontext gemeint ist, wird als *Disambiguierung* bezeichnet.

Metonymie liegt genau dann vor, wenn eine nicht-wörtliche Verschiebung der begrifflichen Interpretation vorgenommen wird. [28, S. 434] Die grundlegenden Relationen, bei denen die Metonymie angewendet wird, sind häufig Teil-Ganzes und Verursacher-Effekt Relationen. Dabei wird vom Sprecher/Autor z. B. anstelle der eigentlich gemeinten Entität diejenige Entität genannt, der die gemeinte Entität unter- oder übergeordnet ist:

- Berlin entschied sich gegen einen Eingriff in Irak. (Hier steht Berlin stellvertretend für Deutschland oder Bundestag)

[32] Eine Liste mit einer Vielzahl von englischen Homophonen wird von ALAN COOPER im WWW unter folgender Adresse publiziert: `http://www.cooper.com/alan/homonym_list.html`

- Das Institut hat mich angerufen. (Institut anstelle des Namen der konkreten Person)

Beispiele für eine Verursacher-Effekt Relation sind:

- Peter hört gerne Bach.
- Der Picasso steht im Museum.

Die Variabilität von Wortinterpretationen (Homographie und Metonymie) stellt IF- und IR-Systeme ebenso vor Probleme wie die Synonymie: Ein System, dass die verschiedenen Interpretationen eines Wortes nicht berücksichtigt, hat Probleme bei der thematischen Zuordnung von Dokumenten. Am Beispiel des IR bedeutet das, dass das System auf eine Anfrage wie suche mir Dokumente zum Thema Maus alle Dokumente zurückliefert, die das Wort Maus enthalten. Wesentlich sinnvoller wäre es, wenn das System vor der Suche den Benutzer darauf aufmerksam machen würde, dass Maus unterschiedlich interpretiert werden kann und wenn das System vom Benutzer die gewünschte Interpretation erfragen würde.

Antonymie, Hyponymie und Meronymie: Die folgenden linguistischen Beziehungen zwischen Interpretationen bilden strukturelle Zusammenhänge ab, die sich z. B. in Ontologien abbilden lassen. Die *Antonymie* ist der Oberbegriff für semantische Gegensatzrelationen. Zwei Ausdrücke sind zueinander antonym, wenn folgendes gilt: Wenn der erste Ausdruck zutrifft, dann trifft der zweite Ausdruck nicht zu. [28, S. 85] Beispiele für derartige antonyme Worte sind: heiß ↔ kalt, wahr ↔ falsch, Mann ↔ Frau.

Als *Hyponymie* wird die semantische Relation der Unterordnung (Subordination oder ist-ein Beziehung) bezeichnet. [28, S. 287] [99, S. 8] So sind beispielsweise Apfel und Birne (als Unterbegriffe) hyponym zu dem Oberbegriff Frucht. Ähnlich zu der Hyponymie ist die *Meronymie*, die sich jedoch auf die semantische Teil-Ganzes-Relation bezieht. [99, S. 8] Beispielsweise sind Reifen und Motor Meronyme von Auto, weil ein Auto u. a. aus mehreren Reifen und einem Motor besteht.

2.3.5 Pragmatik

Pragmatik ist der Bereich der Linguistik, der sich mit dem sprachlichen Handeln beschäftigt. Im Unterschied zur Semantik betrachtet die Pragmatik die Bedeutungsaspekte, die über reine Wahrheitsbedingungen (vom Typ: „die gemachte Aussage ist wahr/falsch.“) hinausgehen. [30, S. 305] Ein Beispiel für einen Teilbereich der Pragmatik sind die, den meisten Sätzen zu Grunde liegenden Präsuppositionen. So sagt der Satz „Das Buch liegt neben dem Rechner.“ nicht nur aus, dass irgendein Buch sich in lokaler Nähe zu irgendeinem Rechner befindet, sondern auch, dass es im betrachteten Kontext genau ein Buch und genau einen Rechner gibt. Neben den Präsuppositionen gehören auch die Deixis, die Implikatur, der Sprechakt und die Konversationsstruktur zu dem Kernbereichen der Pragmatik. Einen Überblick über diese Gebiete findet sich in MEIBAUER [97].

Für den Bereich des IF und IR ist insbesondere ein Nebenbereich der Pragmatik, die Benutzermodellierung von Bedeutung. Mit *Benutzermodellierung* (englisch *user modelling*) bezeichnet man die Methoden, die interaktive Software-Systeme in die Lage versetzen, ihr Verhalten an ihren jeweiligen Benutzer anzupassen. Dieses geschieht mit Hilfe der Erstellung und Ausnutzung eines Benutzermodells, das die Eigenschaften des Benutzers beinhaltet. Unter Verwendung der Benutzermodellierung können Computersysteme benutzerfreundlicher werden, wodurch die Benutzer ihre Ziele besser erreichen können. [30, S. 316] Insbesondere beim IF spielt die Benutzermodellierung eine sehr große Rolle: das Benutzerprofil, anhand dessen ein IF-System die Relevanz von Nachrichten bewertet, ist ein Benutzermodell. Hierbei ist insbesondere für manuell zu erstellende Benutzermodelle (aber auch für erlernte) sinnvoll, wenn diese leicht zu erstellen und gleichzeitig robust sind. Eine leichte Erstellbarkeit und eine hohe Robustheit von Benutzerprofilen bei IF-Systemen ist nur möglich, wenn die Ambiguität der natürlichen Sprache[33] berücksichtigt wird und das System über ein Mindestmaß an lexikalischen „Wissen“ über die Zusammenhänge von Dingen (Metonymie, Hyponymie und Meronymie) in der Welt und deren Bezeichnungen (Synonymie und Homographie) verfügt. Angenommen, ein Benutzer möchte von seinem IF-System alle eingehenden Dokumente zum Thema Autos zum Lesen präsentiert bekommen. Bei einer manuellen Programmierung seines Profils würde der unerfahrene Benutzer folgenden Profileintrag erstellen:

Auto

Dieser Eintrag bedeutet, dass das System alle Dokumente an den Benutzer weiterleiten soll, in denen des Wort Auto vorkommt. Nach kurzer Zeit wird der Benutzer feststellen, dass nicht alle vom Benutzer intendierten Dokumente an ihn weitergeleitet wurden. Der Grund dafür sind einerseits die verschiedenen Beugungsformen von Auto und die verschiedenen Synonyme des Wortes. Somit muss der Profileintrag wie folgt modifiziert werden:

Auto, Autos, Wagen, Automobil, Automobile

Aber auch mit diesem Profil wird der Benutzer feststellen müssen, dass immer noch nicht alle relevanten Dokumente an ihn weitergeleitet werden. Es gibt Dokumente in denen von Audi, VW oder Jaguar gesprochen wird, aber in denen das Wort Auto oder eines seiner Synonyme nicht vorkommt. Somit muss der Benutzer wieder sein Profil überarbeiten:

Auto, Autos, Wagen, Automobil, Automobile, Audi, VW, Jaguar,...

Wie man an diesem Beispiel leicht erkennt, ist die Konzeption eines derartigen Benutzerprofils mühsam. Es ist daher sinnvoll, wenn das IF-System unter Anwendung seines linguistischen und weltbezogenen „Wissens“ das Wort Auto von sich aus, dank eines geeigneten Modells der Repräsentation, „richtig“ interpretieren kann. Ein weiterer Vorteil eines solchen Systems ist die höhere Robustheit. So würde das System dem Benutzer aufgrund seines gespeicherten lexikalischen „Wissens“ auch Dokumente über Automarken präsentieren, die der Benutzer vorher nicht gekannt hat.

[33] Vgl. dazu die Abschnitte 2.3.2 und 2.3.4.2.

Bei IR-System ist der Aspekt der Benutzermodellierung nicht so ausgeprägt wie bei den IF-Systemen. Allerdings ist es sinnvoll, wenn ein IR-System im Laufe einer Anfragesitzung ein Benutzermodell konstruiert. Dieses kann z. B. dadurch geschehen, dass das System eine Anfrage nach einem Homographen (bspw. Maus) identifiziert und den Benutzer nach der intendierten Interpretation (Computereingabegerät bzw. kleines Nagetier) fragt.

2.3.6 Bedeutung für IF und IR

Ein ideales IF- bzw. IR-System ist in der Lage, die zu bearbeitenden Dokumente gemäß allen genannten Teilgebieten der Linguistik zu verarbeiten und somit den Inhalt des Geschriebenen zu verstehen. Wie in Abschnitt 2.3.3.2 jedoch gezeigt wurde, sind derzeitige Syntax-Parser weder in Bezug auf die Parsingqualität (Ambiguität und Abdeckung), noch in Bezug auf die Effizienz hinreichend, um in der Praxis eingesetzt zu werden. Dieses impliziert, wie wir in Abschnitt 2.3.4.1 gesehen haben, dass auch die Satz- und Diskurssemantik von Dokumenten nicht betrachtet werden kann. Somit erlaubt der gegenwärtige Stand der Technik für einen Rechner kein „echtes Verstehen“ von Dokumenten und somit ist ein ideales IF- bzw. IR-System jenseits des derzeit technisch Realisierbaren.

Dennoch besteht ein Bedarf an IF- und IR-Systemen.[34] Um diesem Bedarf nachzukommen, muss auf heuristische Verfahren ausgewichen werden, die nicht in der Lage sind, Dokumente zu verstehen, aber die dennoch akzeptable Ergebnisse liefern. Hierbei ist die Betrachtung der beherrschbaren Aspekte der Linguistik von hoher Bedeutung, um trotz der für Computer in ihrer Ursprungsform schwer verarbeitbaren natürlichen Sprache, Strukturen zu identifizieren, die eine sinnvolle Repräsentation und eine algorithmische sowie effiziente Verarbeitung ermöglichen. Moderne IF- bzw. IR-Systeme sollten dabei folgende Aspekte der Linguistik berücksichtigen:

- Morphologie: Flexion, Komposition und Derivation
- Lexikalische Semantik: Synonymie, Homographie, Metonymie, Hyponymie und Meronymie[35]
- Pragmatik: Benutzermodellierung

2.4 Ontologien

Ein grundsätzliches Problem des Begriffes Ontologie ist, dass dieser ursprünglich aus der Philosophie stammende Begriff in unterschiedlichen Fachbereichen – zu denen auch die Informatik zählt – polysem verwendet wird. Dieses hat zur Folge, dass dem Begriff verschiedene,

[34] Vgl. dazu Abschnitt 1.1.

[35] Die Antonymie ist in dieser Liste nicht enthalten, weil die Verwendung eines konkreten Begriffs von zwei antonymen Begriffen in vielen Fällen von der Perspektive des Autors abhängt und somit nicht sinnvoll für das IF und IR verwendet werden kann. Eine ausführliche Darlegung dieser Problematik wird in Abschnitt 4.2.2 gegeben.

wenn auch im Grundsatz oft ähnliche Definitionen zu Grunde gelegt werden. Daher ist es sinnvoll zunächst die ursprüngliche (die philosophische) Interpretation der Ontologie zu betrachten, bevor wir uns der Ontologie im Sinne der Informatik zuwenden.

In der Philosophie wird als Ontologie diejenige Wissenschaft bezeichnet, die sich mit dem Sein, den Arten von Objektstrukturen, den Eigenschaften, Ereignissen, Prozessen und Beziehungen in allen Bereichen der Realität beschäftigt. Der Begriff der *Ontologie* (lateinisch: *ontologia*) ist im Jahre 1613 unabhängig von den beiden Philosophen RUDOLF GÖCKEL (Goclenius) in seinem „Lexicon philosophicum“ und von JACOB LORHARD (Lorhardus) in seinem „Theatrum philosophicum“ ins Leben gerufen worden. [137, S. 1] Das Ziel der philosophischen Ontologie (als Wissenschaft) ist es, die Wahrheit zu ergründen. [170, S. 188] Dieses bedeutet, dass es im Sinne der Philosophie nur eine einzige Ontologie geben kann, weil es nur eine einzige „wahre“ Wahrheit gibt. Somit verbietet sich im philosophischen Kontext die Verwendung des Plurals für den Begriff der Ontologie.

In den letzten Jahren hat der Ontologie-Begriff große Aufmerksamkeit in vielen Teilbereichen der Informatik, wie z. B. der Wissensverarbeitung, dem Wissensmanagement, der Verarbeitung natürlicher Sprache sowie im Bereich der kooperativen Informationssysteme, zu denen auch die Software-Agenten zählen, erfahren. Dabei können zwei Dinge festgestellt werden: Erstens, bezüglich der genauen Interpretation des Begriffs Ontologie herrscht Uneinigkeit über einige Teilbereiche der Definition. Zweitens, es besteht Konsens darüber, dass eine Ontologie im Sinne der Informatik ein Repräsentation eines domänenspezifischen Wissensbereiches ist, die auf einer formalen Sprache basiert. [170, S. 187] Damit bestehen zum Ontologie-Begriff der Philosophie zwei grundsätzliche Unterschiede: Erstens, eine Ontologie im Sinne der Informatik hat einen pragmatischen Anwendungsbezug (Domäne) und zweitens, macht es somit durchaus Sinn von Ontologie im Plural zu sprechen, da es beliebig viele Repräsentationen von Wissensbereichen geben kann. Eine viel zitierte Ontologie-Definition in der Informatik ist die Definition von GRUBER:

> „An *ontology* is an explicit specification of a conceptualization.“ [59, 60]

Unter einer Konzeptualisierung (englisch: conceptualization) versteht GRUBER eine abstrakte, vereinfachte Sichtweise auf denjenigen Weltausschnitt, den man zu repräsentieren beabsichtigt, um einen Zweck zu erreichen. An der Definition können zwei Aspekte kritisiert werden: zum Einen ist sie relativ vage formuliert und zum Anderen versteht GRUBER unter „explicit specification“ eine mengentheoretische Repräsentation, die nach Ansicht von GUARINO zu restriktiv ist. [61]

Aus diesem Grunde verwenden wir zur Definition des Ontologie-Begriffes die Definition von ZELEWSKI, die auf der Definition von GRUBER aufbaut und diese erweitert und inhaltlich präzisiert:

> „Eine *Ontologie* ist eine explizite und formalsprachliche Spezifikation der ‚sinnvollen‘ sprachlichen Ausdrucksmittel für eine von mehreren Akteuren gemeinsam verwendete Konzeptualisierung von realen Phänomenen, die in einem Subjekt- und Zweck-abhängig einzugrenzenden Realitätsausschnitt als wahrnehmbar oder

> vorstellbar gelten und für die Kommunikation zwischen Akteuren benutzt oder benötigt werden.“ [165]

Bei genauerer Betrachtung der Definition von ZELEWSKI fällt die starke Nähe des Ontologie-Begriffs zu dem Modell-Begriff von BECKER (vgl. Abschnitt 2.2) auf. So spricht ZELEWSKI von einer

> „... expliziten und formalsprachlichen Spezifikation...“

dieser Teil ist äquivalent zu folgendem Ausschnitt in der Definition des Modell-Begriffs von BECKER, weil eine explizite und formalsprachliche Spezifikation ein System von Symbolen und somit eine Sprache zur Modellierung und Explizierung von Sachverhalten ist:

> „...ein System von Symbolen (Modellierungssprache)...“

Ebenso ist der Ausschnitt

> „... eine von mehreren Akteuren gemeinsam verwendete Konzeptualisierung von realen Phänomenen, die in einem Subjekt- und Zweck-abhängig einzugrenzenden Realitätsausschnitt als wahrnehmbar oder vorstellbar gelten...“

in ZELWESKIS Definition, bis auf die Verwendung des Plurals bei den Akteuren, mit BECKERS Definition von Modell inhaltlich identisch:

> „...ist die Repräsentation eines Objektsystems für Zwecke eines Subjekts.“

Der Grund dafür ist der, dass eine Konzeptualisierung von realen Phänomenen eine Eingrenzung eines realen Systems auf ein Objektsystem mit anschließender Abbildung in einem Modell ist, oder in GRUBERS Worten formuliert: eine abstrakte, vereinfachte Sichtweise auf denjenigen Weltausschnitt, den man zu repräsentieren beabsichtigt. Die Subjekt- und Zweckabhängigkeit der Konzeptualisierung ist sogar wörtlicher Bestandteil beider Definitionen. Der einzige Unterschied ist der, dass beim Modell-Begriff nur von einem Subjekt und beim Ontologie-Begriff von mehreren Subjekten (Akteuren) gesprochen wird. Nimmt man die inhaltlichen Gemeinsamkeiten der beiden Definitionen auf, dann kann der Ontologie-Begriff von ZELEWSKI mit anderen Worten wie folgt definiert werden:

> „Eine *Ontologie* ist ein *Modell* von sprachlichen Ausdrucksmitteln, auf die sich mehrere Akteure geeinigt haben und die für eine Kommunikation zwischen den Akteuren benutzt werden.“

Die Mächtigkeit und Einsetzbarkeit von Ontologien ist gemäß dieser Definition abhängig von zwei Faktoren: Von dem modellierten Inhalt der Ontologie und der verwendeten Modellierungssprache. Verschiedene Autoren haben sich mit der Evaluation [65] von oder mit Gütekriterien [59, 60, 23, 5, 34, 43] für Ontologien beschäftigt. Diese Kriterien werden in dieser Arbeit nicht weiter behandelt, da der Fokus dieser Arbeit auf der prinzipiellen Integration von

bestehenden Ontologien in IF- und IR-Systeme liegt. Im Fokus liegen also weder die Aspekte der Erstellung und Verbreitung neuer Ontologien, noch irgendwelche Begründungsschemata für konkrete Auswahlempfehlungen.

Der Vollständigkeit halber ist zu erwähnen, dass manche Autoren, wie z. B. MAEDCHE [92], eine Unterscheidung zwischen Ontologien und Wissensbasen vornehmen. Im Allgemeinen wird dabei definiert, dass Ontologien zur Abbildung von generellen oder langfristig gültigen Zusammenhängen (wie z. B. „Ein Wissenschaftler ist eine Person.“) verwendet werden, wohingegen Wissensbasen zur Speicherung von Fakten, speziellen oder nur kurzfristig gültigen Zusammenhängen (wie z. B. „Albert Einstein ist ein Wissenschaftler“) verwendet werden. Allerdings ist eine genaue Abgrenzung von Ontologien und Wissensbasen bzw. eine eindeutige Entscheidung auf die Frage, ob ein bestimmter Zusammenhang in einer Ontologie oder einer Wissensbasis zu hinterlegen ist, nicht immer eindeutig möglich, wie MAEDCHE [92, S. 21] feststellt. Es ist vielmehr von der Domäne und dem Anwendungsziel eines Informationssystems abhängig, ob eine derartige Unterscheidung sinnvoll ist. Im Anwendungsbereich des IF und IR ist eine Unterscheidung zwischen Ontologien und Wissensbasen aus logischer Sicht nur wenig hilfreich, weil kein eindeutiges Entscheidungskriterium definiert werden kann, ob ein Zusammenhang in einer Ontologie oder einer Wissensbasis gespeichert werden soll. Aus diesem Grunde werden die beiden Begriffe Wissensbasis und Ontologie in dieser Arbeit synonym behandelt.

2.4.1 Ontologie-Modellierungssprachen

Zur Formulierung einer Ontologie bedarf es einer Modellierungssprache, die über eine Syntax festlegt, wie die verschiedenen Elemente einer Ontologie miteinander verbunden werden können und welche Bedeutung (Semantik) diese Verbindung hat. Da es sich beim Ontologie-Begriff der Informatik um eine formale Sprache handeln muss, sind natürlichsprachliche Modelle/Spezifikationen wie z. B. Enzyklopädien keine Ontologien im Sinne der Informatik. Die verschiedenen Modellierungssprachen für Ontologien lassen sich in Abhängigkeit von ihrer Ausdruckskraft und somit auch in ihrer Komplexität in die drei folgenden Klassen einteilen:[36]

Taxonomien, Klassifikationen und Systematiken werden insbesondere in den Naturwissenschaften zur Klassifikation von Objekten verwendet. Ein Beispiel aus der Biologie ist die Klassifikation von Lebewesen gemäß ihrer historischen Entwicklung in Form eines Stammbaumes. Sie zeichnen sich durch eine *strikt hierarchische* Klassifikation von Objekten aus. Das heißt: zu jeder Subklasse gibt es maximal eine Superklasse (oder umgekehrt). Eine gän-

[36] Da sich viele Ontologien historisch vor und zudem in anderen Fachbereichen entwickelt haben als die Ontologie-Definition der Informatik, gibt es keine fachübergreifend einheitliche Benennung für die verschiedenen Klassen von Ontologien. Es werden hier daher die geläufigsten Benennungen für die grundsätzlichen Typen von Modellierungssprachen auf einer hohen Abstraktionsebene vorgestellt. Die verschiedenen Ausprägungen eines Typs unterscheiden sich häufig nur im Detail, in fachspezifischen Restriktionen und in ihrer grafischen oder textuellen Repräsentation. In ihrem grundsätzlichen Aufbau und ihrer prinzipiellen Mächtigkeit, sind die Modellierungssprachen einer Klasse jedoch (nahezu) identisch.

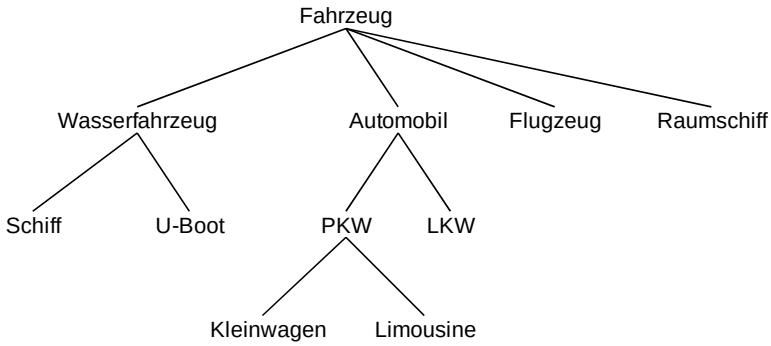


Abbildung 2.10: Beispiel für eine Taxonomie

gige, grafische Visualisierungsform für Taxonomien sind „Bäume“ (vgl. Beispiel in Abbildung 2.10).

Thesauren und Wortnetze zeichnen sich dadurch aus, dass diese im Unterschied zu den Taxonomien, Klassifikationen und Systematiken *keine strikt hierarchische* Klassifikation von Objekten vornehmen und somit eine höhere Ausdruckskraft und Komplexität haben. Thesauren und Wortnetze erlauben zwischen Objekten *beliebige Beziehungen*, wobei auch unterschiedliche *Beziehungstypen* verwendet werden können. Klassische Vertreter für derartige Modellierungssprachen sind die Norm für Thesauren nach DIN 1462[37] bzw. ISO 2788 und Topic Maps[38], aber auch die bei der Informationssystemmodellierung gängigen Fachbegriffsmodelle[39]. Ein Beispiel für einen Thesaurus zeigt Abbildung 2.11.

Zu den konkreten Ontologien, die mit den genannten Sprachen entwickelt wurden gehört z. B. das WordNet[40]. Dieses Netz bildet die Bedeutungen und Beziehungen (Synonyme, Homographen, ...) zwischen Wörtern der englischen Sprache ab. Zwei deutsche Projekte mit ähnlichem Ziel sind das GermaNet[40] der Universität Tübingen und das Wortschatzlexikon[41] der Universität Leipzig.

[37] Nach DIN 1463 ist ein *Thesaurus* im Bereich der Information und Dokumentation eine geordnete Zusammenstellung von Begriffen und ihrer (vorwiegend natürlichsprachlichen) Bezeichnung, die in einem Dokumentationsgebiet zum Indexieren, Speichern und Wiederauffinden dient. Neben der Definition des Begriffs legt die Norm auch die möglichen Beziehungen zwischen Begriffen fest. Z. B.: BF – Benutzt für Synonym, OB – Oberbegriff, ...

[38] Vgl. auch ISO 9075 und [143].

[39] Vgl. dazu ROSEMANN [125, S. 74–87] und SPECK [139, S. 139–148].

[40] Das WordNet wird in Abschnitt 6.1.2.2 zusammen mit dem GermaNet ausführlich vorgestellt.

[41] Eine Kurzdarstellung des Wortschatzlexikons findet sich in Abschnitt 6.1.2.1.

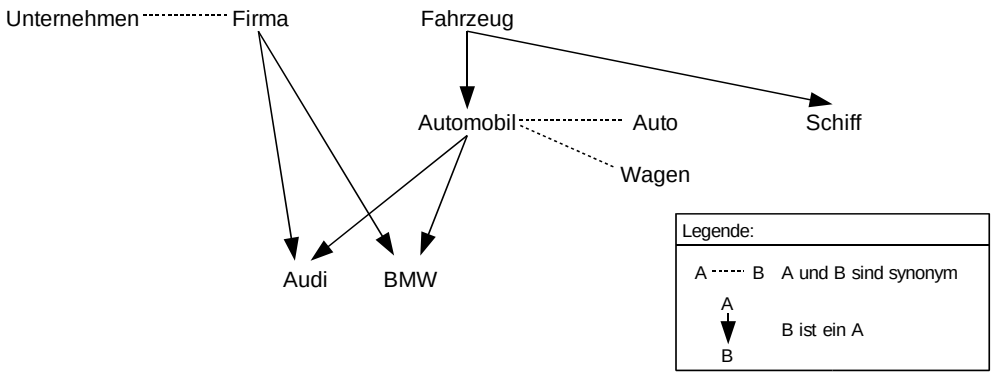


Abbildung 2.11: Beispiel für einen Thesaurus

Logisch-mathematische Repräsentationen und semiotische Thesauren greifen für eine Darstellung von Ontologien auf Formalismen und Notationsformen der Logik oder Mathematik zurück. Die mögliche Ausdruckskraft logisch-mathematischer Modelle ist prinzipbedingt sehr hoch und ermöglicht die Anwendung von Inferenz zur deduktiven Ableitung neuer, impliziter und vorher nicht direkt ersichtlicher Zusammenhänge aus den Ontologien. Nachteile logisch-mathematischer Repräsentationen sind zumeist ein hoher Rechenaufwand für die Inferenz bzw. im Extremfall das Problem der Nicht-Berechenbarkeit von Ausdrücken. Ein Beispiel für eine logisch-mathematische Repräsentation einer Ontologie zeigt Abbildung 2.12.

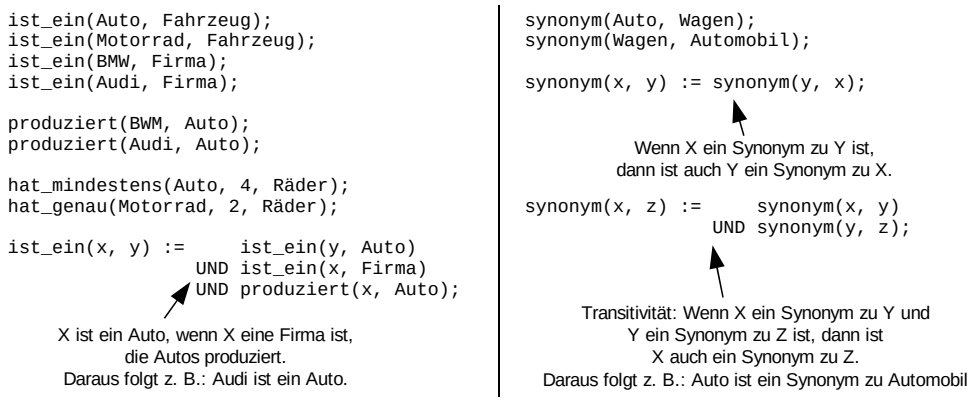


Abbildung 2.12: Ein Beispiel für eine logisch-mathematische Repräsentation einer Ontologie.

Bekannte Modellierungssprachen in dieser Klasse sind u. a. KIF [55], GOL [42], OIL[42], DAML[43], DAML+OIL[44], RDF[45] und Ontolingua[46]. Bekannte Ontologien aus dieser Klasse sind z. B. die FIPA Agent Management Ontology [52], die als Kommunikationsstandard eine wichtige Rolle bei der Inter-Agentenkommunikation spielt. Neben Ontologien für die Agentenkommunikation existieren auch domänenspezifische Ontologien für andere Bereiche wie z. B. die Enterprise-Ontology[47] von der Universität Stanford. Die Ontologien CYC[48] und OpenCYC[49] versuchen hingegen das „allgemeine Weltwissen“ formal und für Rechner auswertbar zu repräsentieren. Eine Liste mit aktuell in der Entwicklung befindlichen Ontologien und Modellierungssprachen für Ontologien findet sich auf den Webseiten der Universität Texas.[50]

2.4.2 Anwendungsmöglichkeiten für IF- und IR-Systeme

Wie aus den Beispiel-Abbildungen 2.10, 2.11 und 2.12 leicht ersichtlich, eignen sich Ontologien dazu, einen begrenzten Umfang von fachspezifischen oder allgemeinen lexikalischen „Wissen“ in formaler Weise abzubilden. Eine Integration dieses „Wissens“ in IF- bzw. IR-Systeme kann nutzbringend eingesetzt werden, um die Suche bzw. das Filtering zu verbessern. Durch die Verwendung einer Ontologie, in der hinterlegt ist, dass ein BMW auch ein Auto sein kann und dass ein Auto ein Wagen ist, kann das System bei der Auswertung einer Anfrage bzw. eines Dokuments mit diesen „intelligenter“ umgehen. Das heißt z. B., dass das System auf die Anfrage Wagen auch Dokumente zu BMW's findet in denen das Wort Wagen nicht vorkommt. Zusammengefasst heißt das, dass das System in der Lage ist Synonyme, Homographen, Hyponyme etc. sinnvoll zu nutzen um dem Anwender bessere Suchergebnisse oder Filtering-Ergebnisse zu präsentieren.

Es gibt jedoch zwei Problembereiche, die einer Integration von Ontologien in IF- und IR-Systeme im Wege stehen können: der Aufwand einer Ontologie-Erstellung und der Rechenaufwand bei der Ontologie-Anwendung. Bezogen auf den ersten Problembereich ist es sinnvoll, möglichst vorhandene Ontologien von (Computer-)Linguisten und Bibliothekaren für das IF und IR wiederzuverwenden und ggf. fachspezifisch zu erweitern. Gute Kandidaten für die Verwendung im IF und IR deutschsprachiger Dokumente sind das GermaNet und das Wortschatzlexikon, für englischsprachige Dokumente verdient das WordNet einer genauere Untersuchung.[51]

Sowohl für das IF als auch IR ist die Rechenzeit ein kritischer Faktor, der u. a. dazu

[42] OIL: `http://www.ontoknowledge.org/oil`
[43] DAML: `http://www.daml.org`
[44] DAML+OIL: `http://www.daml.org/2001/03/daml+oil-index.html`
[45] RDF: `http://www.w3.org/RDF/`
[46] Ontolingua: `http://www.ksl.stanford.edu/software/ontolingua`
[47] Enterprise-Ontology: `www-ksl-svc.stanford.edu`
[48] CYC: `http://www.cyc.com`
[49] OpenCYC: `http://www.opencyc.org`
[50] Liste von Ontologien: `http://www.cs.utexas.edu/users/mfkb/related.html`
[51] Die Eignung des Wortschatzlexikons und des GermaNet für die Anwendung mit dem, hier in Kapitel 5 vorgestellten Modell zur Repräsentation von Dokumenten (eTVSM), wird in Abschnitt 6.1.2 untersucht.

führt, dass – wie wir bereits in Abschnitt 2.3.3.2 festgestellt haben – das Syntax-Parsing derzeit im Zusammenhang mit IF und IR nicht sinnvoll einsetzbar ist. Daher muss die Auswahl einer Ontologie (bzw. einer Ontologie-Modellierungssprache) ebenfalls nach Effizienz-Gesichtspunkten und evtl. auf Kosten der Vollständigkeit und Mächtigkeit durchgeführt werden. Von diesem Standpunkt aus, erscheinen logisch-mathematische Repräsentationsformen für Ontologien aufgrund ihres hohen Rechenaufwands nur bedingt geeignet.

Kapitel 3

Gängige IF/IR-Modelle

Dieses Kapitel gibt (ohne Anspruch auf Vollständigkeit zu erheben) eine Übersicht über gängige Modelle zur Repräsentation von natürlichsprachlichen Dokumenten, die zur Lösung von IF- bzw. IR-Aufgaben bereits eingesetzt werden oder derzeit in der Forschung diskutiert werden. Die vorgestellten Modelle werden gemäß der Dimension des verwendeten mathematischen Fundamentes und der Dimension der modellinhärenten Eigenschaften der Terminterdependenzen kategorisiert (vgl. Abbildung 3.1).[1] Zusätzlich wird in Abschnitt 3.5 eine Bewertung der einzelnen Modelle und Modellklassen bezüglich ausgewählter Aspekte der Linguistik durchgeführt. Diese Bewertung dient gleichzeitig als Motivation für das in Kapitel 4 vorgestellte TVSM bzw. für das in Kapitel 5 vorgestellte eTVSM. Zur Benennung der Modelle werden in dieser Arbeit die englischen Bezeichnungen verwendet, um eine bessere Vergleichbarkeit mit der (üblicherweise englischen) Literatur sicherzustellen. Die meisten der hier vorgestellten Verfahren sind im (und zumeist auch für den) englischen Sprachraum entwickelt worden.

Die Abbildung 3.1 zeigt u. a. die Klassifikation der Modelle bezüglich ihres mathematischen Fundamentes (vertikale Achse) an. Konkret können in dieser Dimension drei verschiedene Modellkategorien unterschieden werden:

1. *Mengentheoretische Modelle* zeichnen sich dadurch aus, dass sie natürlichsprachliche Dokumente auf Mengen abbilden und die Ähnlichkeitsbestimmung von Dokumenten (in erster Linie) auf die Anwendung von Mengenoperationen zurückführen.

[1] Die Präsentation der verschiedenen IF- bzw. IR-Modelle lehnt sich an einigen Stellen an die Arbeit von BAEZA-YATES und RIBEIRO-NETO [7] an, allerdings werden hier auch Modelle präsentiert, die in der genannten Arbeit nicht enthalten sind. Dazu gehören das Language Model, das Backpropagation Network Model, das Retrieval by Logical Imaging und natürlich die in dieser Arbeit neu vorgestellten Modelle (TVSM und insbesondere das eTVSM). Zusätzlich zu [7] werden die Modelle in dieser Arbeit bezüglich der Dimension der modellinhärenten Eigenschaften der Terminterdependenzen kategorisiert und der Autor vermeidet die in [7] vielfach zu findende Mehrfachbelegung von Variablen, die das Nachvollziehen der mathematischen Ausdrücke unnötig erschwert.

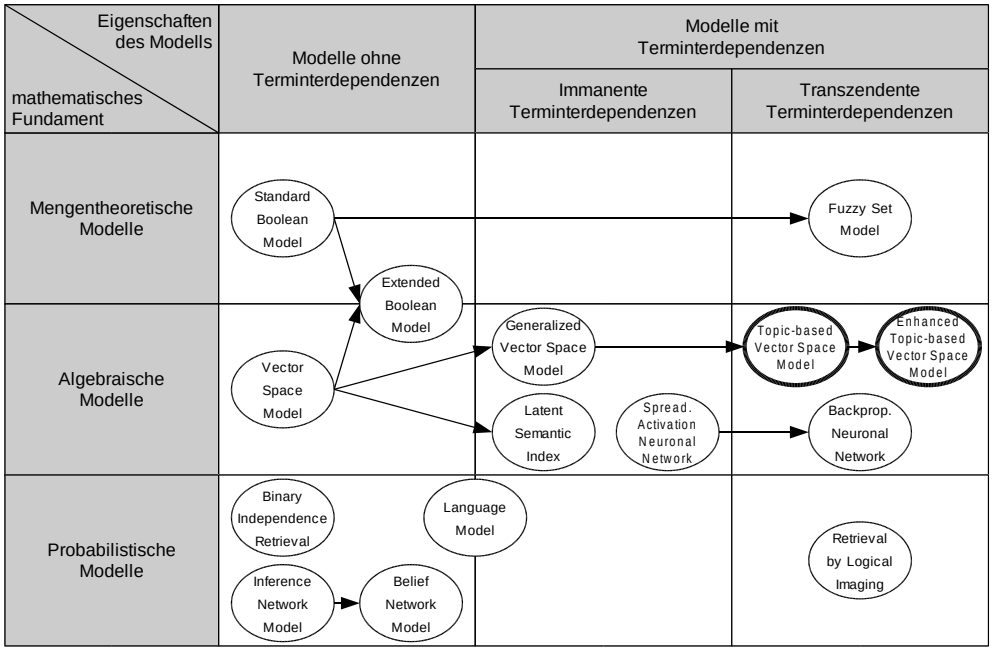


Abbildung 3.1: Eine Übersicht über gängige Modelle zur Repräsentation von natürlichsprachlichen Dokumenten.

2. *Algebraische Modelle* stellen Dokumente und Anfragen als Vektoren, Matrizen oder Tupel dar, die zur Berechnung von paarweisen Ähnlichkeiten über eine endliche Anzahl algebraischer Rechenoperationen in ein eindimensionales Ähnlichkeitsmaß überführt werden.[2]

3. *Probabilistische Modelle* sehen den Prozess der Dokumentensuche bzw. der Bestimmung von Dokumentenähnlichkeiten als ein mehrstufiges Zufallsexperiment an. Zur Abbildung von Dokumentenähnlichkeiten wird daher auf Wahrscheinlichkeiten und probabilistische Theoreme (insbesondere auf das Theorem von BAYES [26, S. 660]) zurückgegriffen.

Auf der horizontalen Achse ist in Abbildung 3.1 die Klassifikation der Modelle bezüglich

[2] Das Backpropagation Neuronal Network stellt in Bezug auf die algebraischen Rechenoperationen eine Ausnahme dar, weil in Abhängigkeit von dem verwendeten Verfahren die Aktivierungsfunktion der Neuronen evtl. eine nicht algebraische Funktion sein kann. Von dieser Ausnahme allerdings abgesehen kommen auch beim Backpropagation Neuronal Network üblicherweise nur algebraische Rechenoperationen zum Einsatz und eventuell verwendete nicht algebraische Funktionen werden – zumal die Berechnung in einem Computer stattfindet – durch eine endliche Anzahl algebraischer Rechenoperationen geschätzt. Das grundsätzliche Vorgehen zur Bestimmung von Ähnlichkeiten beim Backpropagation Neuronal Network entspricht dem der anderen algebraischen Modelle.

ihrer modellinhärenten Eigenschaften der Terminterdependenzen aufgetragen. Hierbei können folgende Hauptklassen unterschieden werden: *Modelle ohne Terminterdependenzen* und *Modelle mit Terminterdependenzen*. Die zuletzt genannte Hauptklasse unterteilt sich dabei wiederum in zwei Unterklassen: in Modelle mit *immanenten* bzw. *transzendenten* Terminterdependenzen. Die drei Klassifikationen dieser Dimension werden zusammen mit einer kompakten Vorstellung der den einzelnen Klassen zugeordneten Modelle in den Abschnitten 3.2, 3.3 und 3.4 im Detail vorgestellt und diskutiert. Einzige Ausnahme ist das TVSM und seine Erweiterung, die in den Kapiteln 4 und 5 einzeln vorgestellt werden, weil diese durch die Bewertung der in diesem Kapitel vorgestellten Modelle in Abschnitt 3.5 motiviert werden und den Kern dieser Arbeit ausmachen.

Zusätzlich zu der Einordnung der Modelle in die einzelnen Klassen, kann anhand der Pfeile in der Abbildung 3.1 entnommen werden, ob die einzelnen Modelle in einem Zusammenhang zueinander stehen. Ein Pfeil zwischen zwei Modellen sagt aus, dass das Modell, auf das die Pfeilspitze zeigt, eine (vom Autor intendierte oder sich auch „zufällig“ ergebende) Verallgemeinerung oder Erweiterung des jeweils anderen Modells darstellt. Bevor nun die modellinhärenten Eigenschaften der Terme und die einzelnen Modelle im Detail vorgestellt werden, werden in Abschnitt 3.1 die fundamentalen und den meisten Modellen gemeinsamen Konzepte bei der Verarbeitung und Interpretation von natürlichsprachlichen Dokumenten, als Ausgangsbasis für die weitere Betrachtung, vorgestellt.

3.1 Fundamentale Konzepte

Zur Verarbeitung von natürlichsprachlichen Dokumenten ist allen hier genannten Verfahren gemeinsam, dass diese die Dokumente in einzelne Terme als atomare Bestandteile eines Dokuments zerstückeln. Ein digitales, natürlichsprachliches Dokument liegt einem Rechner als eine lange Zeichenkette vor. Das Mustervorgehen zur Gewinnung von Termen aus der Zeichenkette sieht wie folgt aus: Ein Parser untersucht die Zeichenkette des Dokuments (ggf. in mehreren Durchläufen) systematisch Zeichen für Zeichen und entfernt alle evtl. vorhandenen Formatierungen (z. B. HTML-Befehle) und Sonderzeichen (wie z. B.: `..;:?!`) und ersetzt diese durch Leerzeichen. Das Ergebnis dieser Transformation ist eine Zeichenkette, die eine Vielzahl von durch Leerzeichen getrennten Wörtern enthält, die im Folgenden – wie in der IF/IR-Literatur üblich – als *Terme* bezeichnet werden. Zur besseren Verarbeitung wird pro Dokument das Vorkommen aller Terme in dem Dokument gezählt und gespeichert. Somit ergeben sich folgende Variablen:[3]

[3] Je nach Anwendungsbereich kann es sinnvoll sein, das hier genannte Mustervorgehen zu modifizieren. So kann man sich überlegen, dass die Formatierung von Wörtern und Sätzen eine besondere Bedeutung hat (z. B. bei Überschriften) und es daher als sinnvoll erachten, die formatierten Wörter höher zu gewichten als den Rest. Ähnliche Diskussionen können in Bezug auf Sonderzeichen, Zahlen und aus Sonderzeichen, Zahlen und Buchstaben kombinierte Begriffe geführt werden. Des Weiteren kann es sinnvoll sein beim Parsing sprachenspezifische Überlegungen anzustellen. So erlaubt die deutsche Sprache gerade bei Bindestrichen verschiedene Schreibweisen für dieselben Wörter (z. B. Workflow-Management vs. Workflowmanagement). Daher könnte es evtl. sinnvoll sein durch Bindestriche getrennte Wörter zu verschmelzen.

- D bezeichnet die Menge aller Dokumente.
- T bezeichnet die Menge aller Terme, die in den Dokumenten aus D vorkommen.
- $a_{d,t} \in \mathbb{Z}_{\geq 0}$ ist die Anzahl des Vorkommens des Terms $t \in T$ in dem Dokument $d \in D$. Sollte der Term t im Dokument d nicht vorkommen, dann ist $a_{d,t} = 0$.

Anwendung von Stoppwortlisten: Wie bereits in Abschnitt 2.3.4.2 erläutert, existieren Wörter, die keine explizite themenbezogene Bedeutung haben, die Stoppwörter genannt werden. Aus diesem Grunde ist es intuitiv nachvollziehbar diese Stoppwörter bei der Verarbeitung nicht zu berücksichtigen, weil sie zur Suche bzw. zum Vergleich von Dokumenten keinen Beitrag leisten. In der Praxis hat sich dieses Vorgehen für die meisten Modelle in vielen Situationen bewährt. Sowohl die Vergleichsgeschwindigkeit als auch die Vergleichsqualität lässt sich durch die Anwendung von Stoppwortlisten steigern. In Abschnitt 4.3 wird dieses bisher nur intuitiv motivierte Vorgehen mit Hilfe des TVSM auf ein theoretisches Fundament gestellt.

Zur Anwendung einer Stoppwortliste $T_{\oslash}$ (wie z. B. der Liste von DROTT [46]) ist folgendes Vorgehen erforderlich: Allen Vorkommen von Termen in Dokumenten die Stoppwörter betreffen, wird vor der weiteren Verarbeitung der Wert Null zugewiesen (wodurch der alte Wert überschrieben wird, was durch die folgende Verwendung von $:=$ im Unterschied zu $=$ festgelegt wird).

$$a_{d,t} := 0 \quad \forall d \in D, t \in T_{\oslash}$$

Zusätzlich ist es sinnvoll, alle Stoppwörter aus der Menge der Terme T zu löschen um die Verarbeitungsgeschwindigkeit zu erhöhen:

$$T := T \setminus T_{\oslash}$$

Durchführen des Stemming: Die Flexion von Wörtern erschwert den Vergleich von Dokumenten, weil ein und dasselbe Wort in unterschiedlichen Flexionsformen vorkommen kann (vgl. Abschnitt 2.3.2). Daher ist intuitiv leicht zu begründen, warum Wörter vor der weiteren Verarbeitung auf ihre Stammform zurück geführt werden sollten. In der Praxis beobachtet man bei der Anwendung des Stemming gute Erfolge; eine theoretische Fundierung für die Anwendung des Stemming wird in Abschnitt 4.4 unter Zuhilfenahme des TVSM vorgestellt.

Zur Umsetzung des Stemming ist eine Stemmingfunktion $\perp(t) = t_{\perp}$ zu definieren, die zu jedem beliebigen Term $t \in T$ den dazugehörigen Wortstamm $t_{\perp} \in T_{\perp}$ aus der Menge aller Wortstämme $T_{\perp} \subseteq T$ liefert. Zusätzlich liefert die Funktion zu einem Wortstamm den eingegebenen Wortstamm zurück:

$$\perp(t_{\perp}) = t_{\perp} \quad \forall t_{\perp} \in T_{\perp}$$

Zur besseren Handhabbarkeit wird zu der Stemmingfunktion $\perp()$ eine Umkehrrelation $\perp^{-1}()$ zu definiert. Diese Umkehrrelation liefert zu jedem Wortstamm $t_{\perp} \in T_{\perp}$ die Menge aller Terme (inklusive des Wortstammes), die zu diesem Wortstamm gehören:

$$\perp^{-1}(t_{\perp}) = \{t \in T : \perp(t) = t_{\perp}\} \quad \forall t_{\perp} \in T_{\perp}$$

Beim Aufstellen der Stemmingfunktion und ihrer Umkehrrelation kann es (in Abhängigkeit von der Sprache der Dokumente) vorkommen, dass ein Term zu mehreren Wortstämmen gehört. Da dieser Fall bei den meisten Sprachen selten ist, wird dieses Problem in der Praxis ignoriert. Beim Auftreten des Problems wird in der Praxis vielmehr willkürlich eine Entscheidung getroffen, so dass die Funktion $\perp()$ eindeutig ist. Gängige Algorithmen zum Berechnen der Stemmingfunktion $\perp()$ sind bereits im Abschnitt 2.3.2.2 kurz vorgestellt worden.

Das Stemming wird umgesetzt, indem alle Terme, die keine Wortstämme sind, durch ihren Wortstamm ersetzt werden und alle Nicht-Wortstämme aus der Menge der Terme gelöscht werden:

$$\begin{aligned} a_{d,t_\perp} &:= \sum_{t \in \perp^{-1}(t_\perp)} a_{d,t} \quad \forall t_\perp \in T_\perp \\ a_{d,t} &:= 0 \quad \forall t \in T \setminus T_\perp \\ T &:= T \setminus T_\perp \end{aligned}$$

Anwendung von Synonymersetzungen: Seltener angewandt als die zuvor genannten Verfahren ist das Ersetzen von synonymen Begriffen durch einen führenden Begriff. Da üblicherweise eine Totale Synonymie unterstellt wird, ist das Vorgehen analog zu dem Vorgehen beim Stemming. Es wird eine Funktion $S(t) = t_s$ definiert, die zu jedem Term $t \in T$ (z. B. Auto, Automobil oder Wagen) den dazu passenden, synonymen und führenden Term t_s aus der Menge der führenden Terme $T_s \subseteq T$ liefert (z. B. Wagen). Des Weiteren gilt (analog zum Stemming):

$$S(t_s) = t_s \quad \forall t_s \in T_s$$

Auch hier definieren wir zur besseren Handhabung eine Umkehrrelation $S^{-1}()$ als

$$S^{-1}(t_s) = \{t \in T : S(t) = t_s\} \quad \forall t_s \in T_s$$

Die Anwendung erfolgt analog zum Stemming:

$$\begin{aligned} a_{d,t_s} &:= \sum_{t \in S^{-1}(t_s)} a_{d,t} \quad \forall t_s \in T_s \\ a_{d,t} &:= 0 \quad \forall t \in T \setminus T_s \\ T &:= T \setminus T_s \end{aligned}$$

Eine gängige Methode zur Implementierung der Synonymersetzungsfunktion $S()$ ist die Verwendung einer Tabelle. Die einzelnen Tabelleneinträge (Term und führender Term) werden dabei entweder von Hand eingefügt oder mit Hilfe von statistischen auf Co-Occurrenz basierenden Verfahren ermittelt. Die Anwendung von derartigen Verfahren und die damit verbundenen Probleme in diesem Zusammenhang werden in Abschnitt 3.3 ausführlich dargestellt.

Bestimmung von Ähnlichkeiten: Je nach Anwendungsgebiet sind unterschiedliche Ähnlichkeiten von Interesse. Beim IR gibt der Anwender eine Anfrage q vor. Somit ist es erforderlich für alle Dokumente $d \in D$, die Ähnlichkeit $\text{sim}(d, q)$ zwischen den Dokumenten und der Anfrage zu berechnen, um die Dokumente gemäß dieser Ähnlichkeit zu ordnen und dem Benutzer zu präsentieren.

Im Unterschied dazu ist beim IF ein anderes Vorgehen erforderlich: Beim IF werden neue Dokumente vom System in verschiedene Klassen (z. B. im einfachsten Fall in die Klassen relevant und nicht relevant) eingeordnet. Dazu ist es erforderlich, ein neues Dokument $d \in D$ mit den Profilen der einzelnen Klassen zu vergleichen. Zur Implementierung der einzelnen Klassenprofile wird häufig eine der beiden folgenden genannten Möglichkeiten gewählt:

1. Zu jeder Klasse k_i wird eine eigene Anfrage q_i definiert. Zur Einordnung eines Dokuments d wird die Ähnlichkeit $\text{sim}(d, q_i)$ des Dokuments zu den Anfragen aller Klassenprofile berechnet und das Dokument in diejenige Klasse eingeordnet, die die größte Ähnlichkeit zu dem Dokument aufgewiesen hat. Dieses Verfahren ist somit analog zu dem Vorgehen beim IR.

2. Zu jedem Klassenprofil k_i wird eine Menge von Vergleichsdokumenten $D_i \subseteq D$ definiert, die dieser Klasse bereits angehören, wobei alle Klassenprofile disjunkt sind (also $D_i \cap D_j = \{\}$ für alle i, j). Ein neues Dokument d wird einer Klasse zugeordnet, indem das Dokument zunächst mit allen Dokumenten $d_{i,j} \in D_i$ aller Klassen i verglichen wird: $\text{sim}(d, d_{i,j})\ \forall i, d_{i,j} \in D_i$. Das Dokument d wird anschließend derjenigen Klasse zugeordnet, die die meisten Vergleichsdokumente unter den k ähnlichsten Dokumente zu d hatte. Dieses Verfahren wird auch als *k-nearest neighbour* bezeichnet. (Eine ausführliche Beschreibung dieses etablierten und in vielen Bereichen eingesetzten Verfahrens findet sich in [40].)

Die meisten der hier vorgestellten Verfahren sind für das IR konzipiert worden, können aber auch für das IF mit der ersten Methode angewendet werden. Einige Verfahren ermöglichen auch die direkte Anwendung der zweiten Methode beim IF, weil diese Verfahren Anfragen als virtuelle Dokumente betrachten und eigentlich nur Dokumentenähnlichkeiten berechnen können. Bei anderen Verfahren kann die Ähnlichkeit zwischen zwei Dokumenten erst dann berechnet werden, wenn eines der beiden Dokumente in eine Anfrage umgewandelt wurde. In diesem Fall ist allerdings vor der Anwendung dieses Vorgehens zu prüfen, ob die Kommutativität bei der Ähnlichkeitsberechnung erhalten bleibt.[4]

[4] Wird die Kommutativität bei der Ähnlichkeitsberechnung außer Kraft gesetzt, dann ist es möglich, dass die Ähnlichkeit zweier Dokumente $d_1, d_2 \in D$ davon abhängt, in welcher Reihenfolge diese in die Ähnlichkeitsfunktion eingehen. Demnach ist dann folgende Situation nicht ausgeschlossen: $\text{sim}(d_1, d_2) \neq \text{sim}(d_2, d_1)$. Diese Situation ist aber üblicherweise mit der Vorstellung von Dokumentenähnlichkeit nicht vereinbar.

3.2 Modelle ohne Terminterdependenzen

Die Modelle ohne Terminterdependenzen zeichnen sich dadurch aus, dass jeweils zwei verschiedene Terme als vollkommen unterschiedlich und in keinster Weise miteinander verbunden angesehen werden. Dieser Sachverhalt wird in der Literatur häufig auch als *Orthogonalität von Termen* – bei einer grafischen Interpretation wie sie bei den algebraischen Modellen üblich ist – bzw. als *Unabhängigkeit von Termen* – bei einer probabilistischen Interpretation – bezeichnet.

Das Fehlen von Terminterdependenzen stellt gegenüber der Realität der natürlichen Sprachen eine starke Vereinfachung dar, die zunächsteinmal dazu führt, dass morphologische und lexikalisch-semantische Zusammenhänge zwischen Termen nicht erfasst werden können. Einige dieser fehlenden Zusammenhänge lassen sich durch die Anwendung der in Abschnitt 3.1 genannten Verfahren des Stemming und der Synonymersetzung abfangen. Die Anwendung dieser Verfahren lässt sich aus den Modellen ohne Terminterdependenzen jedoch nicht theoretisch ableiten, weil diese eben davon ausgehen, dass alle Terme keinen Bezug zueinander haben. Somit eignen sich diese Modelle nicht als Erklärungsmodelle, was aus theoretisch-wissenschaftlicher Sicht unbefriedigend ist. Zudem lassen sich Beziehungen zwischen Termen der folgenden linguistischen Phänomene mit Modellen ohne Terminterdependenzen (trotz der Anwendung von Stemming und Synonymersetzung) nicht abbilden: Komposita-Beziehungen (wie z. B. zwischen Chip, Fabrik und Chipfabrik), Derivationsbeziehungen (wie z. B. zwischen Zwerg und Zwerglein) sowie Metonymie, Hyponymie und Meronymie (wie z. B. die ist-ein Beziehung zwischen Linux und Betriebssystem).

Zu den Modellen ohne Terminterdependenzen gehören neben den drei Klassikern unter den IR/IF-Modellen – Standard Boolean Model, Vector Space Model und Binary Independence Retrieval – auch die folgenden neueren Modelle: Inference Network Model, Belief Network Model und Extended Boolean Model. Das Language Model, dass derzeit auf Konferenzen und in der Literatur viel diskutiert wird, ist ein Grenzgänger. Theoretisch ist das Modell in der Lage, Terminterdependenzen modellimmanent zu berücksichtigen, in der Forschungspraxis wird es aber derzeit überwiegend terminterdependenzfrei (bei Verwendung von Unigram-Modellen) oder seltener mit nur geringen Terminterdependenzen (bei Bigram- bzw. Trigram-Modellen) verwendet, weshalb es hier den Modellen ohne Terminterdependenzen zugerechnet wird. Allgemein kann man über Modelle ohne Terminterdependenzen sagen, dass diese in Bezug auf den Rechenaufwand geringere Anforderungen stellen als Modelle mit Terminterdependenzen. Zudem sind einige Vertreter dieser Klasse algorithmisch wenig anspruchsvoll, was die Implementierung und Anwendung in der Praxis stark vereinfacht.

3.2.1 Standard Boolean Model (SBM)

Das SBM gehört zu den Klassikern unter den mengentheoretischen IR/IF-Modellen. Es ist intuitiv, relativ leicht nachzuvollziehen und insbesondere mit relationalen Datenbanken ohne großem Aufwand zu implementieren. Aus didaktischen Gründen wird das Modell im Folgenden auf Mengenoperationen basierend und anhand einer Beispielanfrage erklärt. Eine alterna-

tive Darstellung findet sich in [7, S.25ff].

Eine Anfrage wird beim SBM als eine logische Verknüpfung von Termen formuliert. Die folgende Anfrage q sucht diejenigen Dokumente heraus, die den Term t_a enthalten und entweder den Term t_b enthalten oder den Term t_c nicht enthalten:

$$q = t_a \wedge (t_b \vee \neg t_c)$$

Zum Aufstellen der Ähnlichkeitsfunktion benötigen wir folgende Variablendefinitionen: D_t sei diejenige Menge von Dokumenten, die den Term t enthalten:

$$D_t = \{d \in D : a_{d,t} > 0\}$$

Zusätzlich definieren wir die Komplementmenge $\complement D_t$ als diejenige Menge von Dokumenten, die den Term t nicht enthalten:

$$\complement D_t = D \setminus D_t$$

Zur Berechnung der Ähnlichkeit eines Dokuments d mit der Anfrage q braucht die Anfrage lediglich derart umgeformt zu werden, dass t_a, t_b, t_c durch $D_{t_a}, D_{t_b}, D_{t_c}$ und $\wedge, \vee, \neg$ durch $\cap, \cup, \complement$ ersetzt werden. Das Ergebnis dieser Umformung ist eine Reihe von Mengenoperationen, die im Resultat diejenigen Dokumente selektieren, die die gestellte Anfrage erfüllen:

$$\text{sim}(d, q) = \begin{cases} 1 & \text{falls } d \in D_{t_a} \cap (D_{t_b} \cup \complement D_{t_c}) \\ 0 & \text{sonst} \end{cases}$$

Ein großer Vorteil dieses Modells ist die hohe Effizienz bei der Berechnung. Es brauchen nur diejenigen Dokumente betrachtet zu werden, die einen der in der Anfrage genannten Terme enthalten. Ein großer Nachteil ist, dass dieses Modell lediglich einen binären Ähnlichkeitswert berechnet. Somit können partielle Treffer zu einer Anfrage nicht gefunden werden. In der Praxis führt dieses dazu, dass eine Anfrage entweder kein Dokument oder gleich eine Vielzahl von Dokumenten (die nicht weiter sortiert werden können) liefert.

Für die Anwendung im IF unter Verwendung des k-nearest neighbour Verfahrens ist das SBM kaum zu verwenden: zum Ersten, weil Dokumente nicht direkt miteinander verglichen werden können und zum Zweiten aufgrund der binären Ähnlichkeitswerte.[5]

3.2.2 Vector Space Model (VSM)

Das VSM ist 1968 von SALTON in [126, 127] vorgestellt worden und gehört zu den Klassikern unter den algebraischen IF/IR-Modellen, dass sich bis heute (zumeist in einer mehr oder

[5] Eine Möglichkeit, ein Dokument in eine Anfrage umzuwandeln, sieht wie folgt aus: Es wird eine Anfrage konstruiert, die alle Terme über alle Dokumente hinweg mit und ($\wedge$) verknüpft, wobei die im Dokument nicht enthaltenen negiert ($\neg$) werden. Ein derartiges Vorgehen erfüllt die Bedingung der Kommutativität. In Kombination mit den binären Ähnlichkeitswerten ist dieses Verfahren jedoch sehr restriktiv, weil nur Dokumente, die dieselben Terme enthalten (und keinen Term mehr oder weniger), als ähnlich zueinander definiert werden.

weniger abgewandelten Form) einer großen Beliebtheit in der Praxis erfreut. Gegenüber dem SBM bietet das VSM abgestufte Ähnlichkeitswerte und eine geometrische Interpretation von Dokumenten, die leicht zu vermitteln und anzuwenden ist und daher die große Popularität des Modells begründet.

Das VSM repräsentiert alle Dokumente $d \in D$ über einen Dokumentenvektor $\vec{d} \in \mathbb{R}^{\#T}$. Jede Dimension des Vektors entspricht dabei einem Term $t_i \in T$. Da alle Dimensionen zueinander orthogonal sind, werden die Terme beim VSM somit als frei von Interdependenzen modelliert. Die Ausprägung einer jeden Dimension eines Dokumentenvektors ist über das Gewicht w_{d,t_i} festgelegt:

$$\vec{d} = (w_{d,t_1}, w_{d,t_2}, ..., w_{d,t_{\#T}}) \quad \text{mit} \quad t_i \in T$$

Zur Berechnung von Gewichten werden in der Literatur verschiedenste Verfahren diskutiert. [77, 124] Zu der bekanntesten Klasse von Gewichtungsverfahren gehören die *tf-idf* (term frequency – inverse document frequency) Verfahren. Ein Vertreter dieser Verfahrensklasse ist das folgende Gewichtungsschema nach [7, S. 29]:

$$w_{d,t_i} = \frac{a_{d,t_i}}{\max_{t \in T} a_{d,t}} \log \frac{\#D}{\#\{e \in D : \ a_{e,t_i} > 0\}} \tag{3.1}$$

Die Ähnlichkeit zwischen zwei Dokumenten $d_i, d_j \in D$ wird beim VSM üblicher Weise unter Anwendung des normierten Skalarproduktes als der Kosinus des Winkels zwischen den Vektoren $\vec{d_i}$ und $\vec{d_j}$ der beiden Dokumente berechnet:[6]

$$\begin{aligned} \text{sim}(d_i, d_j) &= \frac{\vec{d_i}\vec{d_j}}{|\vec{d_i}||\vec{d_j}|} \\ &= \frac{\sum_{t \in T} w_{d_i,t} w_{d_j,t}}{\sqrt{\sum_{t \in T} w_{d_i,t}^2} \sqrt{\sum_{t \in T} w_{d_j,t}^2}} \end{aligned}$$

Die Berechnung der Ähnlichkeit zwischen einem Dokument und einer Anfrage q verläuft analog zur Berechnung der Ähnlichkeit zwischen zwei Dokumenten. Eine Anfrage wird beim VSM als virtuelles Dokument aufgefasst, dass lediglich die in der Anfrage enthaltenen Terme hat, und bei dem die Gewichte analog zu den realen Dokumenten bestimmt werden:

$$\vec{q} = (w_{q,t_1}, w_{q,t_2}, ..., w_{q,t_{\#T}}) \quad \text{mit} \quad t_i \in T$$

Gegenüber dem SBM hat das VSM den Nachteil, dass es keine Verknüpfungsoperationen zwischen den Termen bei Anfragen erlaubt. Da das VSM sowohl die Berechnung von Ähnlichkeiten zwischen Dokumenten als auch die Berechnung von Ähnlichkeiten zwischen Dokumenten

[6] Andere bekannte Maße sind das Pseudo-Kosinus-Maß, das Dice-Maß, das Overlap-Maß und das Jaccard-Maß. Eine ausführliche Beschreibung dieser Maße findet sich in FERBER [50, S. 46ff].

und Anfragen unterstützt, kann das VSM problemlos für IF und IR (auch mit k-nearest neighbour Verfahren) verwendet werden.

Bei der Berechnung von Ähnlichkeiten werden immer alle Terme der betroffenen Dokumente einbezogen. Daher ist die Anwendung von Stoppwortlisten und Stemming für akzeptable Ergebnisse unumgänglich. Zur Behandlung von Synonymen sollte eine Synonymersetzung Verwendung finden, oder es sollten alternativ bei der Anwendung des VSM für das IF *Query-Expansion-Methoden* benutzt werden. Bei diesen Methoden werden Anfragen vor oder während der Verarbeitung um zusätzliche synonyme Terme erweitert.[7]

3.2.3 Extended Boolean Model (EBM)

Das EBM ist 1983 von SALTON, FOX und WU in [128] als eine Kombination des VSM mit dem SBM vorgestellt worden, die die Stärken der beiden Modelle miteinander verbindet. Gegenüber dem SBM hat das EBM den Vorteil, dass es ein fein abgestuftes Ähnlichkeitsmaß bietet. Gegenüber dem VSM hat es den Vorteil, dass es logische Verknüpfungen (Und-, Oder-Verknüpfung) unterstützt. Da das Modell sowohl Konzepte aus der Mengentheorie als auch Konzepte aus der Algebra kombiniert, lässt sich dieses Modell keiner der beiden Klassen eindeutig zuordnen.

Das EBM ist in erster Linie für die Berechnung von Ähnlichkeiten zwischen Dokumenten und Anfragen und – wie das SBM – nicht für die Berechnung von Ähnlichkeiten zwischen Dokumenten konzipiert worden. Im Folgenden wird die Berechnung der Ähnlichkeit zwischen einer Anfrage und einem Dokument an einigen Beispielen erläutert. Zunächst sind jedoch einige grundlegende Variablen zu definieren: Wie beim VSM werden Dokumente $d \in D$ beim EBM als Vektoren $\vec{d}$ interpretiert.

$$\vec{d} = (w_{d,t_1}, w_{d,t_2}, ..., w_{d,t_{\#T}}) \quad \text{mit} \quad t_i \in T$$

Zum besseren Vergleich mit dem SBM verwenden wir hier das folgende – in der Praxis neben dem in Abschnitt 3.2.2 beschriebenen tf-idf Schema durchaus übliche – Gewichtungsschema:

$$w_{d,t_i} = \begin{cases} 1 & \text{für } a_{d,t_i} > 0 \\ 0 & \text{für } a_{d,t_i} = 0 \end{cases}$$

Die Ähnlichkeitsfunktion $\text{sim}()$ für das EBM wird nach dem Baukastenschema aus den beiden folgenden elementaren Verknüpfungsbausteinen $q_\vee, q_\wedge$ zusammengebaut:

$$q_\vee = t_a \vee^p t_b \quad \Rightarrow \quad \text{sim}(d, q_\vee) = \left(\frac{w_{d,t_a}^p + w_{d,t_b}^p}{2} \right)^{\frac{1}{p}}$$

$$q_\wedge = t_a \wedge^p t_b \quad \Rightarrow \quad \text{sim}(d, q_\wedge) = 1 - \left(\frac{(1 - w_{d,t_a})^p + (1 - w_{d,t_b})^p}{2} \right)^{\frac{1}{p}}$$

[7] Eine Übersicht über Query-Expansion-Methoden findet sich in [7, S. 117ff].

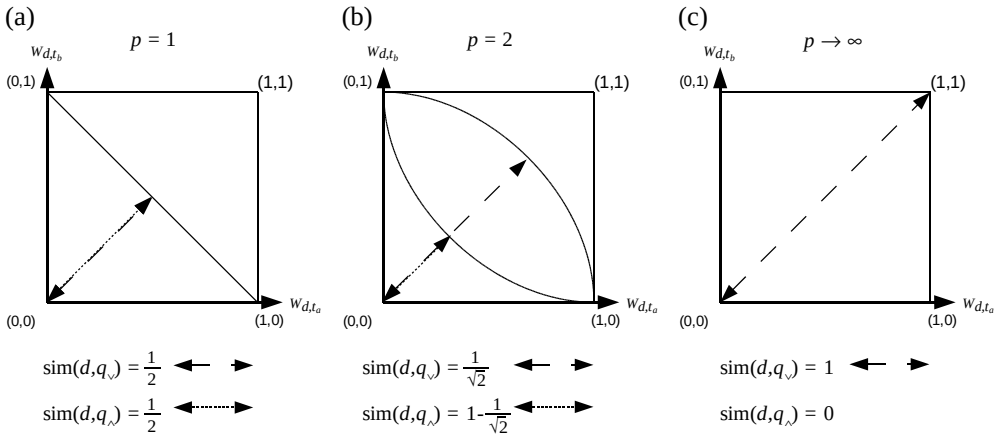


Abbildung 3.2: Equidistanz-Linien und Ähnlichkeiten für den Fall, dass nur einer von zwei Termen in einem Dokument vorhanden ist.

Diese elementaren Verknüpfungsbausteine sind so konzipiert, dass die Ähnlichkeiten genau dann gleich Null sind, wenn in dem Dokument keiner der gesuchten Terme vorhanden ist. Umgekehrt sind die Ähnlichkeiten gleich Eins, wenn beide gesuchten Terme vorhanden sind. Das Verhalten für den Fall, dass nur einer der beiden Terme im Dokument d vorhanden ist, wird durch die gewählte Verknüpfungsoperation und den Parameter p festgelegt. Der Parameter p mit $1 \leq p < \infty$ definiert, wie streng die Und- bzw. wie schwach die Oder-Verknüpfung ausgelegt wird. Die Abbildung 3.2 illustriert diesen Sachverhalt:

- Für den Fall $p = 1$ werden die Und- und die Oder-Verknüpfung identisch interpretiert. Das heißt, dass das Fehlen eines der gesuchten Terme bei beiden Verknüpfungen gleich negativ auf die Ähnlichkeit angelastet wird. Es lässt sich zeigen, dass sich das EBM in diesem Falle genauso verhält wie das VSM. (Vgl. dazu die Herleitung in [128, S. 1025].)
- Im Falle, dass $p > 1$ ist, wird das Fehlen eines Terms bei der Und-Verknüpfung stärker negativ auf die Ähnlichkeit angelastet als bei der Oder-Verknüpfung. Diese Differenz nimmt mit zunehmenden p zu.
- Läuft der Parameter $p \rightarrow \infty$, dann kann gezeigt werden, dass sich das EBM genauso verhält wie das SBM. Das heißt, dass nur noch binäre Ähnlichkeiten gemäß der binären Logik für Und- bzw. Oder-Verknüpfungen zurückgegeben werden. (Vgl. dazu die Herleitung in [128, S. 1025].)

Zur Berechnung von komplizierteren Anfragen wie z. B.

$$q = (t_a \wedge^p t_b) \vee^q t_c$$

muss die Berechnungsvorschrift für die Ähnlichkeitsberechnung lediglich, analog zur Verschachtelung der elementaren Anfragebestandteile, verschachtelt werden. Somit ergibt sich die Berechnung der Ähnlichkeit zwischen einem Dokument d und der genannten Anfrage q als:

$$\mathrm{sim}(d,q) = \left(\frac{\left(1 - \left(\frac{(1-w_{d,t_a})^p + (1-w_{d,t_b})^p}{2}\right)^{\frac{1}{p}}\right)^q + w_{d,t_c}^q}{2} \right)^{\frac{1}{q}}$$

Wie bereits erwähnt, eignet sich dieses Modell weniger für die Berechnung von Ähnlichkeiten zwischen Dokumenten – aus demselben Grund wie das SBM. Wie beim VSM, ist es bei dem EBM empfehlenswert, Stoppwörter über Stoppwortlisten vor der Verarbeitung abzufangen, um gute Ergebnisse zu erzielen.

3.2.4 Binary Independence Retrieval (BIR)

Der Klassiker unter den probabilistischen Modellen ist das BIR-Modell, welches ursprünglich 1976 von ROBERTSON und SPARK JONES in [124, S. 140ff] unter dem Namen *Probabilistic Model* eingeführt wurde.[8] Das BIR-Modell ist lediglich in der Lage die Ähnlichkeit zwischen einem Dokument und einer Anfrage zu ermitteln. Dabei geht das Modell jedoch im Unterschied zu den anderen Modellen einen ganz speziellen Weg: Der Benutzer kann seine Anfrage nicht explizit formulieren, sondern die Anfrage wird von dem Modell in mehreren Zyklen unter Verwendung von Benutzer-Feedback erlernt/erraten.

Wie oben erwähnt wird die Benutzeranfrage beim BIR nicht explizit modelliert. Die Variable q wird daher lediglich als Platzhalter für die Anfrage verwendet. Die innere Struktur von q ist nicht bekannt. Die fundamentale Annahme des BIR ist die Existenz einer Menge $R \subseteq D$ von Dokumenten, die in Bezug auf die Anfrage q für den Benutzer relevant sind. Entsprechend bezeichnet $\complement R = D \setminus R$ das Komplement, also die Menge der nicht relevanten Dokumente. Da es sich beim BIR um ein probabilistisches Modell handelt, wird die Ähnlichkeit von Dokumenten und Anfragen aus (geschätzten) Eintrittswahrscheinlichkeiten $P(e)$ für spezifische Ereignisse e abgeleitet:

- $\delta \in R$ wird definiert als das Ereignis, bei dem bei einer zufälligen Ziehung eines Dokuments δ aus der Menge von Dokumenten D das gezogene Dokument relevant ist. Bei der Variable δ handelt es sich somit um eine *Zufallsvariable*. $\delta \in \complement R$ ist entsprechend das Gegenereignis, also das Ziehen eines Dokuments, welches nicht relevant ist.
- $d = \delta$ ist das Ereignis, bei dem es sich bei einem zufällig gewählten Dokument δ aus D um das explizit vorgegebene Dokument d handelt.

Die Ähnlichkeit zwischen einem Dokument d und einer Anfrage q wird definiert als die Wahrscheinlichkeit dafür, dass das Dokument d gezogen wird (Vorbedingung) und dass es sich bei

[8] Der Name Probabilistic Model hat sich im Nachhinein als ungeeignet herausgestellt, weil dieser Begriff eher mit einer ganzen Klasse von Modellen assoziiert wurde, als mit der Zeit weitere probabilistische Modelle vorgestellt wurden.

dem gezogenen Dokument um ein relevantes Dokument handelt (Nachbedingung), dividiert durch die Wahrscheinlichkeit dafür, dass das Dokument d gezogen wird (Vorbedingung) und dass es sich dabei *nicht* um ein relevantes Dokument handelt (Nachbedingung):

$$\text{sim}(d,q) = \frac{P(\delta \in R | d = \delta)}{P(\delta \in \complement R | d = \delta)}$$

Unter Anwendung des Theorems von BAYES [26, S. 660] kann folgende Umformung vorgenommen werden:

$$\text{sim}(d,q) = \frac{P(d = \delta | \delta \in R) P(\delta \in R)}{P(d = \delta | \delta \in \complement R) P(\delta \in \complement R)}$$

Dabei ist $P(d = \delta | \delta \in R)$ die Wahrscheinlichkeit dafür, dass das Dokument d zufällig aus der Menge der relevanten Dokumente $\delta \in R$ gezogen wird.

Da die Dokumente lediglich gemäß der Ähnlichkeiten geordnet werden sollen und $P(\delta \in R)$ bzw. $P(\delta \in \complement R)$ über alle Dokumente konstant sind, reicht die proportionale Vereinfachung $\text{sim}'()$ zur Berechnung der Ähnlichkeiten aus:

$$\text{sim}'(d,q) = \frac{P(d = \delta | \delta \in R)}{P(d = \delta | \delta \in \complement R)} \sim \text{sim}(d,q)$$

$P(d = \delta | \delta \in R)$ kann berechnet werden, indem folgende Annahmen über die Beschaffenheit von Dokumenten getroffen werden: Ein Dokument wird als Binärvektor über alle Terme aufgefasst: $\vec{d} \in \{0;1\}^{\#T}$. Ein Eintrag für ein Dokument d hat im Vektor $\vec{d}$ an der Stelle eine Eins, an der der korrespondierende Term t im Dokument vorkommt (also $a_{d,t} > 0$). Ansonsten ist der Eintrag Null. Ein solcher Vektor und somit die Wahrscheinlichkeit dafür, dass ein Dokument zufällig gezogen wird, kann auch als eine Verkettung von $\#T$ binären Zufallsexperimenten aufgefasst werden. Die Wahrscheinlichkeit dafür, dass ein Term t in einem beliebigen Dokument d aus der Menge der relevanten Dokumente vorhanden ist, ist definiert als $P(a_{\delta,t} > 0 | \delta \in R)$ (daraus folgt: $P(a_{\delta,t} = 0 | \delta \in R) = 1 - P(a_{\delta,t} > 0 | \delta \in R)$). Unter der Annahme, dass das Auftreten der Terme *voneinander unabhängig* ist, kann $P(d = \delta | \delta \in R)$ (und analog $P(d = \delta | \delta \in \complement R)$) wie folgt berechnet werden:

$$P(d = \delta | d \in R) = \prod_{t \in T:\, a_{d,t} > 0} P(a_{\delta,t} > 0 | \delta \in R) \prod_{t \in T:\, a_{d,t} = 0} P(a_{\delta,t} = 0 | \delta \in R)$$

Daraus folgt für die Berechnung der Ähnlichkeit:

$$\text{sim}'(d,q) = \frac{\prod_{t \in T:\, a_{d,t} > 0} P(a_{\delta,t} > 0 | \delta \in R) \prod_{t \in T:\, a_{d,t} = 0} P(a_{\delta,t} = 0 | \delta \in R)}{\prod_{t \in T:\, a_{d,t} > 0} P(a_{\delta,t} > 0 | \delta \in \complement R) \prod_{t \in T:\, a_{d,t} = 0} P(a_{\delta,t} = 0 | \delta \in \complement R)}$$

Wie bereits oben erwähnt, wird beim BIR die Suche nach relevanten Dokumenten in mehreren Schritten unter Verwendung von Benutzer-Feedback durchgeführt. Im ersten Schritt wird

dem Benutzer zunächst eine Menge von Dokumenten präsentiert. Diese kann entweder dadurch ermittelt werden, dass vermutlich relevante Dokumente unter Verwendung eines anderen IR-Modells gesucht werden, oder dass das BIR zur Selektion von Dokumenten verwendet wird, wobei folgende Annahmen zur Initialisierung gemacht werden:

$$P(a_{\delta,t} > 0 | \delta \in R) = \frac{1}{2}$$
$$P(a_{\delta,t} > 0 | \delta \in \complement R) = \frac{\#D_t}{\#D}$$

Wobei $D_t \subseteq D$ die Menge derjenigen Dokumente ist, die den Term $t \in T$ enthalten. Aus den präsentierten Dokumenten selektiert der Benutzer eine Menge $V \subseteq D$ von Dokumenten, die seinem Informationsbedarf am nächsten kommen und somit seine nicht explizierte Anfrage q am besten erfüllen.

Unter Verwendung der vom Benutzer selektierten Menge V können die Wahrscheinlichkeiten für $P(a_{\delta,t} > 0 | \delta \in R)$ und $P(a_{\delta,t} > 0 | \delta \in \complement R)$ wie folgt geschätzt werden: [7, S. 33f]

$$P(a_{\delta,t} > 0 | \delta \in R) = \frac{\#V_t + \frac{\#D_t}{\#D}}{\#V + 1}$$
$$P(a_{\delta,t} > 0 | \delta \in \complement R) = \frac{\#D_t - \#V_t + \frac{\#D_t}{\#D}}{\#D - \#V + 1}$$

Dabei ist $V_t \subseteq V$ die Menge derjenigen vom Benutzer selektierten Dokumente, die den Term $t \in T$ enthalten. Mit den neu geschätzten Wahrscheinlichkeiten können Dokumente in einem neuen Durchlauf vom System selektiert und dem Benutzer als relevant präsentiert werden. Bei Bedarf kann dieser aus den präsentierten Dokument diejenigen auswählen, die seinem Informationsbedarf am nächsten kommen. Das Verfahren wird dann mit dem zweiten Schritt solange wiederholt, bis der Benutzer mit dem Ergebnis zufrieden ist.

Da das BIR-Modell die Terme eines Dokuments bezüglich ihrer Auftrittswahrscheinlichkeit als unabhängig ansieht, fällt das BIR-Modell in die Klasse der Modelle ohne Terminterdependenzen. Durch die Verwendung von vom Benutzer selektierten Dokumenten zur Berechnung von Auftrittswahrscheinlichkeit von Termen wird das Manko der nicht interdependenten Terme jedoch ein wenig gemildert. Es besteht eine gewisse Wahrscheinlichkeit, dass ein relevantes Dokument mehrere synonyme Terme enthält. Dadurch erhöht sich beim nächsten Durchlauf die Wahrscheinlichkeit für andere Dokumente, die nur einen der beiden Terme enthalten, gefunden zu werden. Damit können Terminterdependenzen – analog zu den Query-Expansion-Techniken beim VSM – unter Umständen indirekt erfasst werden.

3.2.5 Inference Network Model (INM)

Das INM ist 1990 von TURTLE und CROFT entwickelt und in [145, 146] ausführlich vorgestellt worden. Es würde den Rahmen dieses Kapitels sprengen, das INM in allen seinen Detail

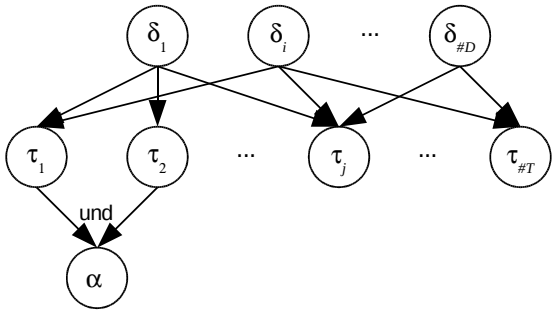


Abbildung 3.3: Topologie eines Inference Network Model

vorzustellen, daher wird hier zum besseren Verständnis eine vereinfachte Fassung (mit binären Termvorkommen in Dokumenten ohne tf-idf Termgewichte und ohne komplex zusammengesetzte Anfragen aus Schlüsselwort-basierten und bool'schen Anfragen) vorgestellt, die aber die fundamentalen Prinzipien des INM vermittelt.

Das INM basiert auf den *Bayesian Networks*[9] zur Modellierung von Dokumenten und Anfragen. Abbildung 3.3 illustriert die Topologie des verwendeten Netzwerkes. Die Wurzeln des Netzwerkes sind die Ereignisse bzw. Zufallsvariablen $\delta_i \in \{0; 1\}$ (wobei eine 1 bedeutet, dass das Ereignis aufgetreten ist. Eine 0 bedeutet folglich, dass das Ereignis nicht aufgetreten ist). Jedes δ_i repräsentiert dabei das Ereignis, bei dem durch ein zufälliges Ziehen eines Dokuments aus der Dokumentenmenge D das Dokument $d_i \in D$ gezogen wird. Die Wahrscheinlichkeit für das Auftreten des Ereignisses ($\delta_i = 1$) wird folglich als $P(\delta_i)$ bezeichnet. Von den Ereignissen δ_i hängen die Ereignisse $\tau_j \in \{0; 1\}$ ab, wobei das Auftreten spezifischer δ_i eine Bedingung für das Auftreten eines spezifischen τ_j ist. Jedes τ_j repräsentiert das Ereignis, bei dem der konkrete Term $t_j \in T$ beobachtet wird, wobei dieser zufällig aus einem Dokument gezogen wird, welches zuvor ebenfalls zufällig aus der Menge der Dokumente gezogen wurde. Das $\alpha \in \{0; 1\}$ repräsentiert das Ereignis des Beobachten der Beispielanfrage $q = t_1 \wedge t_2$. Das Ereignis ist direkt von den Ereignissen τ_j und indirekt von den Ereignissen δ_i abhängig.

Beim INM wird die Ähnlichkeit eines Dokuments zu einer Anfrage als die Wahrscheinlichkeit für das gemeinsame Auftreten einer Anfrage q und eines Dokuments d_i interpretiert:

[9] Bayesian Networks [110] sind gerichtete azyklische Graphen, welche durch ihre Knoten Zufallsvariablen repräsentieren. Die Kanten stellen kausale Zusammenhänge zwischen den Zufallsvariablen her. Die Stärke des Einflusses dieser kausalen Zusammenhänge wird durch bedingte Wahrscheinlichkeiten dargestellt. Die Vorgängerknoten zu einem Knoten werden als direkte Ursache oder Vorbedingung für den Knoten betrachtet. Die Knoten ohne Vorgänger bilden die Wurzel des gesamten Netzwerkes und sind von allen anderen Knoten unabhängig.

$$\begin{aligned}\text{sim}(d_i, q) &= P(\delta_i \wedge \alpha) \\ &= \sum_{\forall \vec{\tau}} P(\alpha \wedge \vec{\tau} \wedge \delta_i) \\ &= \sum_{\forall \vec{\tau}} P(\alpha|\vec{\tau}) P(\vec{\tau}|\delta_i) P(\delta_i)\end{aligned}$$

$\vec{\tau}$ ist dabei ein Zufallsvektor über alle τ_j:

$$\vec{\tau} = (\tau_1, \tau_2, ..., \tau_{\#T}) = \{0; 1\}^{\#T}$$

Unter der Annahme, dass die Terme t_j und somit die Ereignisse τ_j innerhalb eines Dokuments *unabhängig voneinander* sind, kann $P(\vec{\tau}|\delta_i)$ wie folgt berechnet werden:

$$P(\vec{\tau}|\delta_i) = \prod_{\forall t_j \in T:\, a_{d_i,t_j} > 0} P(\tau_j|\delta_i) \prod_{\forall t_j \in T:\, a_{d_i,t_j} = 0} P(\neg\tau_j|\delta_i)$$

Somit berechnet sich die Ähnlichkeit $\text{sim}(d_i, q)$ wie folgt:

$$\text{sim}(d_i, q) = \sum_{\forall \vec{\tau}} P(\alpha|\vec{\tau}) \left(\prod_{\forall t_j \in T:\, a_{d_i,t_j} > 0} P(\tau_j|\delta_i) \prod_{\forall t_j \in T:\, a_{d_i,t_j} = 0} P(\neg\tau_j|\delta_i) \right) P(\delta_i)$$

Die gängigen Annahmen für die noch nicht spezifizierten Wahrscheinlichkeiten sind dabei die folgenden:

$$\begin{aligned}P(\delta_i) &= \frac{1}{\#D} \\ P(\tau_j|\delta_i) &= \begin{cases} 1 & \text{falls } a_{d_i,t_j} > 0 \\ 0 & \text{sonst} \end{cases} \\ P(\neg\tau_j|\delta_i) &= 1 - P(\tau_j|\delta_i) \\ P(\alpha|\vec{\tau}) &= \begin{cases} 1 & \text{falls } \vec{\tau} \text{ eine gültige Termkombination für } q \text{ ist.} \\ 0 & \text{sonst} \end{cases}\end{aligned}$$

Für die Beispielanfrage $q = t_1 \wedge t_2$ ist die einzig gültige Termkombination von $\vec{\tau}$ diejenige Kombination bei der $\tau_1 = \tau_2 = 1$ und $\tau_j = 0$ für alle $j \geq 2$ ist.

Dieses Modell geht davon aus, dass Terme nicht interdependent sind (vgl. Unabhängigkeitsannahme bei der Herleitung von $P(\vec{\tau}|\delta_i)$). Im Original von TURTLE und CROFT erlaubt das INM neben den bool'schen Anfragen auch Schlüsselwort-basierte Anfragen (und Kombinationen aus den beiden Anfragetypen) und ermöglicht auch die Anwendung von nicht binären Termgewichten, wie z. B. tf-idf. [7, S. 54f]

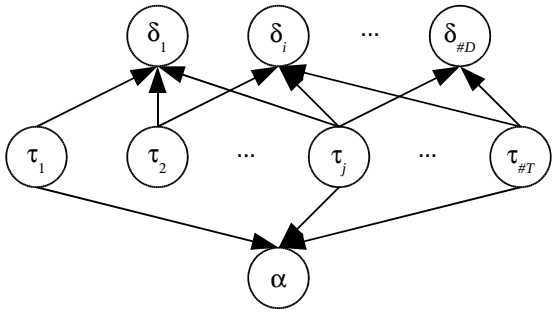


Abbildung 3.4: Beispiel für ein einfaches Belief Network Model

3.2.6 Belief Network Model (BNM)

Das 1996 von RIBEIRO-NETO und MUNTZ [122] vorgestellte BNM basiert ebenso wie das INM auf dem Bayesian Network Konzept. Allerdings unterscheidet sich das BNM Modell durch seine Topologie vom INM (vgl. Abbildung 3.4). Beim BNM sind die Wurzeln des Netzwerkes die Ereignisse bzw. Zufallsvariablen $\tau_j \in \{0;1\}$ (wobei $\tau_j = 1$ das Eintreten des Ereignisses bedeutet), die das zufällige Ziehen des Terms $t_j \in T$ aus der Menge aller Terme T repräsentieren. Die einzelnen Zufallsvariablen τ_j werden zu dem Zufallsvektor $\vec{\tau}$ zusammengefasst:

$$\vec{\tau} = (\tau_1, \tau_2, ..., \tau_{\#T}) = \{0;1\}^{\#T}$$

$\alpha \in \{0;1\}$ bezeichnet im Folgenden das Ereignis, dass zufällig eine Anfrage q beobachtet werden konnte. $\delta_i \in \{0;1\}$ beschreibt das zufällige Beobachten des Dokuments $d_i \in D$. Im Unterschied zum INM hängen beim BNM sowohl das Anfrageereignis α als auch die Dokumentereignisse δ_i von den Termereignissen τ_j kausal ab. Daher wird als Ähnlichkeitskriterium beim BNM die Wahrscheinlichkeit $P(\delta_i|\alpha)$, also das Beobachten eines Dokuments d_i unter der Bedingung, dass die Anfrage q gestellt wurde, gewählt.

$$\text{sim}(d_i, q) = P(\delta_i|\alpha)$$

Unter Anwendung des Theorems von BAYES [26, S. 660] ist dies äquivalent zu:

$$\text{sim}(d_i, q) = \frac{P(\delta_i \wedge \alpha)}{P(\alpha)}$$

Da $P(\alpha)$ über alle Dokumente hinweg konstant ist und lediglich die relativen Größenunterschiede der Wahrscheinlichkeiten für die einzelnen Dokumente von Interesse sind, kann auf die Betrachtung von $P(\alpha)$ verzichtet werden und $\text{sim}'(d_i, q)$ zur Berechnung der Ähnlichkeit

verwendet werden:

$$\begin{aligned}\text{sim}'(d_i, q) &= P(\delta_i \wedge \alpha) \\ &= \sum_{\forall \vec{\tau}} P(\delta_i \wedge \alpha | \vec{\tau}) P(\vec{\tau}) \\ &= \sum_{\forall \vec{\tau}} P(\delta_i | \vec{\tau}) P(\alpha | \vec{\tau}) P(\vec{\tau}) \\ &\sim \text{sim}(d_i, q)\end{aligned}$$

Unter der Annahme, dass die Terme *zueinander unabhängig* sind, kann die Wahrscheinlichkeit für das Beobachten einer bestimmten Termkombination $P(\vec{\tau})$ wie folgt geschätzt werden:

$$P(\vec{\tau}) = \left(\frac{1}{2}\right)^{\#T}$$

Die Wahrscheinlichkeiten für $P(\delta_i|\vec{\tau})$ und $P(\alpha|\vec{\tau})$ können unterschiedlich festgelegt werden. Eine Möglichkeit ist die hier präsentierte und an das Vector Space Model angelehnte Variante, die mit dem BNM das VSM (hier jedoch ohne tf-idf Termgewichten) simuliert: [7, S. 59]

$$P(\alpha|\vec{\tau}) = \begin{cases} \frac{a_{q,t_j}}{\sqrt{\sum_{t_k \in T} a_{q,t_k}^2}} & \text{falls } (\exists\, \tau_j = 1) \wedge (\forall k \neq j :\ \tau_k = 0) \wedge a_{q,t_j} > 0 \\ 0 & \text{sonst} \end{cases}$$

$$P(\delta_i|\vec{\tau}) = \begin{cases} \frac{a_{d_i,t_j}}{\sqrt{\sum_{t_k \in T} a_{d_i,t_k}^2}} & \text{falls } (\exists\, \tau_j = 1) \wedge (\forall k \neq j :\ \tau_k = 0) \wedge a_{d_i,t_j} > 0 \\ 0 & \text{sonst} \end{cases}$$

wobei $a_{q,t}$ = Anzahl des Vorkommens des Terms $t \in T$ in der Anfrage q ist.

Aus theoretischer Sicht ist dass BNM ein Übermodell zu dem INM – es ist in der Lage jedes von INM generierte Ranking abzubilden, was umgekehrt für das INM nicht gilt. [122] [7, S. 60] Des Weiteren bildet das BNM Dokumente und Anfragen in derselben Weise ab, wodurch es neben dem Anfrage-basierten IF auch für das k-nearest neighbour IF geeignet ist.

3.2.7 Language Model (LM)

Beim LM handelt es sich nicht um ein konkretes Modell, sondern vielmehr um eine derzeit viel diskutierte Klasse von Modellen [112, 138, 87, 75, 69, 168][10], die auf dem *Statistical Language Model* aufbauen. Das Statistical Language Model wurde bereits erfolgreich im Bereich der Spracherkennung, dem Tagging als auch dem Syntax-Parsing angewandt. [31] Die

[10] Bei der aufgezählten Literatur handelt es sich lediglich um Beispiele. Die Aufzählung erhebt keinen Anspruch auf Vollständigkeit.

Grundidee des Statistical Language Model ist die Zuordnung von Wahrscheinlichkeiten für einzelne Wordsequenzen $w_1, w_2, ..., w_n$ anhand eines statistischen Sprachmodells:

$$w_1, w_2, ..., w_n \quad \Rightarrow \quad P(w_1, w_2, ..., w_n) \in [0...1]$$

Das Bestimmen von Wahrscheinlichkeiten für beliebig lange Wordsequenzen ist aufwändig, weil aufgrund der großen Anzahl an verschiedenen Worten ein großer Textkorpus zum Ableiten einer stabilen Schätzung für die Wahrscheinlichkeiten erforderlich ist. Aus diesem Grund approximiert man die statistischen Sprachmodelle in der Praxis häufig durch sogenannte n-Gram Modelle: [138]

- Unigram: $P(w_1, w_2, ..., w_n) = P(w_1)P(w_2) \cdots P(w_n)$
- Bigram: $P(w_1, w_2, ..., w_n) = P(w_1)P(w_2|w_1)P(w_3|w_2) \cdots P(w_n|w_{n-1})$
- Trigram:

$$\begin{aligned} P(w_1, w_2, ..., w_n) =& P(w_1)P(w_2|w_1)P(w_3|w_1, w_2)P(w_4|w_2, w_3) \cdots \\ & P(w_n|w_{n-2}, w_{n-1}) \end{aligned}$$

Das Unigram Modell geht von der Annahme aus, dass die einzelnen Wörter voneinander unabhängig sind. Beim Bigram bzw. Trigram Modell ist ein Wort von seinem Vorgänger bzw. von seinen beiden Vorgängern abhängig. Somit eignen sich diese beiden Modelle gut, um aus mehreren Wörtern zusammengesetzte Begriffe abzubilden. Dennoch können auch mit dem Bi- bzw. Trigram Modell keine beliebigen Interdependenzen zwischen zwei Termen dargestellt werden, vielmehr sind die Interdependenzen sehr restriktiv auf den umgebenden Kontext (auf das vorangehende bzw. auf die beiden vorangehenden Worte) beschränkt. Aus diesem Grunde wird das LM an dieser Stelle diskutiert und nicht im Abschnitt 3.3 der Modelle mit immanenten Terminterdependenzen.

Für die Übertragung des Statistical Language Model in den Anwendungsbereich des IF/IR ist die, auf der SIGIR Konferenz von 1998 erschienene Publikation von PONTE und CROFT [112] von entscheidender Bedeutung, weil sie „den Stein ins Rollen brachte“. Das hier vorgestellte LM basiert auf dieser Publikation und zeichnet sich durch die Verwendung eines Unigram Modells aus.

Die grundlegende Idee des LM ist es, die Relevanz eines Dokuments $d \in D$ bezüglich einer Anfrage q als die Wahrscheinlichkeit dafür aufzufassen, dass q eine zufällige Wortsequenz des Statistical Language Model M_d für das Dokument d ist. Aus Gründen der Vereinfachung betrachten wir hier – analog zur Arbeit von PONTE und CROFT – die Anfrage als Menge von Termen $q \subseteq T$ und nicht als Termsequenz.[11]

Die größte Herausforderung beim LM ist die Erstellung der Sprachmodelle M_d für die einzelnen Dokumente. Da wir hier ein Unigram Modell anwenden, müssen alle bedingten Wahrscheinlichkeiten $P(t|M_d)$ für alle Terme $t \in T$ über alle Modelle M_d der Dokumente

[11] Andere LM – insbesondere Bi- und Trigram basierte – arbeiten mit Termsequenzen. (Vgl. z. B. [138].)

$d \in D$ geschätzt werden. Der erste, intuitive Ansatz diese Wahrscheinlichkeiten zu schätzen, ist die Verwendung der *Maximum-Likelihood Methode*:

$$P_{ml}(t|M_d) = \frac{a_{d,t}}{\sum_{u \in T} a_{d,u}}$$

Dieser Ansatz funktioniert jedoch in der Praxis nicht, weil ein einziges Dokument zu wenig Daten (Terme) enthält, um eine statistisch zuverlässige Schätzung ableiten zu können. So führt die Maximum-Likelihood Methode zu dem Problem, dass Terme die in einem Dokument nicht enthalten sind, eine Wahrscheinlichkeit von Null zugewiesen bekommen. Dieses hat zur Folge, dass eine Anfrage (bzw. Wortsequenz) mit einem nicht im Dokument enthaltenen Term ebenfalls eine Auftrittswahrscheinlichkeit von Null hat. Dieses ist im Rahmen des IF und IR jedoch nicht erwünscht. Aus diesem Grunde ist es erforderlich, die Wahrscheinlichkeitsschätzungen unter Verwendung von Korpusdaten, die sich auf den gesamten Dokumentenbestand beziehen, zu stabilisieren (vgl. [113, 112, 138]). Dieses Vorgehen wird als *Smoothing* bezeichnet.

PONTE und CROFT schlagen in [112] für das Smoothing folgendes Vorgehen vor: Es wird angenommen, dass ein in einem Dokument nicht vorkommender Term die gleiche Wahrscheinlichkeit hat, wie der Term bezogen auf den gesamten Dokumentenbestand. Zusätzlich wird die Wahrscheinlichkeit für Terme, die im Dokument vorkommen, mit der durchschnittlichen Vorkommenswahrscheinlichkeit über alle Dokumente hinweg, in denen der Term vorkommt, „geglättet“. Konkret sind folgende Daten zu erheben: Die Wahrscheinlichkeit $P(t)$ für den Term t über alle Dokumente hinweg, die wie folgt geschätzt wird:

$$P(t) = \frac{\sum_{d \in D} a_{d,t}}{\sum_{d \in D} \sum_{u \in T} a_{d,u}}$$

Zur Glättung der Maximum-Likelihood Wahrscheinlichkeiten wird die durchschnittliche Auftrittswahrscheinlichkeit $P_{avg}(t)$ eines Terms über alle Maximum-Likelihood Wahrscheinlichkeiten des Terms über diejenigen Dokumente berechnet, in denen der Term vorkommt:

$$P_{avg}(t) = \frac{\sum_{d \in D:\ a_{d,t} > 0} P_{ml}(t|M_d)}{\#\{d \in D :\ a_{d,t} > 0\}}$$

Zum Aufstellen des Schätzers sind noch folgende Daten zu erheben: Erstens, das durchschnittliche Vorkommen $\bar{a}_t$ eines Terms t über alle Dokumente hinweg, die den Term enthalten.

$$\bar{a}_t = \frac{\sum_{d \in D:\ a_{d,t} > 0} a_{d,t}}{\#\{d \in D :\ a_{d,t} > 0\}}$$

Zweites, das Risiko $R_{t,d}$ für einen Term t in Bezug auf ein Dokument d:

$$R_{d,t} = \frac{1}{1 + \bar{a}_t} \left(\frac{\bar{a}_t}{1 + \bar{a}_t} \right)^{a_{d,t}}$$

Der von PONTE und CROFT vorgeschlagene Schätzer $P(t|M_d)$ für die Termwahrscheinlichkeiten in Bezug auf ein Dokumentenmodell sieht wie folgt aus:

$$P(t|M_d) = \begin{cases} P_{ml}(t|M_d)^{(1-R_{d,t})} P_{avg}(t)^{R_{d,t}} & \text{falls} \quad a_{d,t} > 0 \\ P(t) & \text{sonst} \end{cases}$$

Neben diesem Smoothing Verfahren werden in der aktuellen Literatur noch viele weitere Verfahren, wie z. B. der Good-Turing Schätzer [138] oder das Term-Specific Smoothing [69], diskutiert.

Zur Berechnung von Dokument/Anfrage Ähnlichkeiten gibt es in der Literatur ebenfalls verschiedenste Vorschläge. Zu den einfacheren Verfahren gehört das folgende, von PONTE und CROFT vorgeschlagene, Verfahren, welches sich aus der Unigram-Annahme ergibt:

$$\begin{aligned}\text{sim}(d,q) &= P(q|M_d) \\ &= \prod_{t\in q} P(t|M_d) \prod_{t\notin q} 1 - P(t|M_d)\end{aligned}$$

Die Diskussion um die LM-Verfahren läuft noch, daher lässt sich das Potential dieser Verfahren noch nicht abschließend beurteilen. Fest steht auf jeden Fall, dass das LM in seiner jetzigen Form die Beugung von Worten und Terminterdependenzen nicht im vollen Umfang berücksichtigt. Insofern ist die Verwendung von Stemming und Synonym-Ersetzung vorteilhaft. Bezüglich der Stoppwörter gibt es bereits einen Vorschlag von HIEMSTRA [69], wie diese in das LM unter Anwendung von Termgewichten aufgenommen werden können. Der Anwendung der LM-Verfahren im IF (mit und ohne k-nearest neighbour) neben dem IR steht nichts im Wege, zumal je nach Modell, Dokumente und Anfragen analog als Wortsequenzen oder Wortmengen modelliert werden können.

3.3 Modelle mit immanenten Terminterdependenzen

Modelle mit immanenten Terminterdependenzen zeichnen sich dadurch aus, dass sie vorhandene Interdependenzen zwischen Termen berücksichtigen und ihnen somit – im Unterschied zu den Modellen ohne Terminterdependenzen aus Abschnitt 3.2 – *nicht* die implizte Annahme zu Grunde liegt, dass Terme orthogonal bzw. unabhängig voneinander sind. Die Modelle mit den *immanenten* Terminterdependenzen grenzen sich von den Modellen mit den transzendenten Terminterdependenzen dadurch ab, dass das Ausmaß einer Interdependenz zwischen zwei Termen aus dem Dokumentenbestand, in einer vom Modell bestimmten Weise, abgeleitet wird – also dem Modell innewohnend (immanent) ist.

Die Interdependenz zwischen zwei Termen wird bei dieser Klasse von Modellen direkt oder indirekt aus der *Co-Occurrenz* der beiden Terme abgeleitet. Unter Co-Occurrenz versteht man dabei das gemeinsame Auftreten zweier Terme in einem Dokument. Dieser Modellklasse liegt somit die Annahme zu Grunde, dass zwei Terme zueinander interdependent sind, wenn sie häufig gemeinsam in Dokumenten vorkommen. Ein Maß für die Co-Occurrenz ist z. B. das in der IF/IR-Literatur oft verwendete Jaccard-Maß:

$$\text{sim}(t_i,t_j) = \frac{\#D_{t_i\wedge t_j}}{\#D_{t_i} + \#D_{t_j} - \#D_{t_i\wedge t_j}} \in [0...1] \quad \forall t_i,t_j \in T \tag{3.2}$$

mit

$$\begin{aligned} D_{t_i} &= \{d \in D : \ a_{d,t_i} > 0\} \\ D_{t_j} &= \{d \in D : \ a_{d,t_j} > 0\} \\ D_{t_i \wedge t_j} &= D_{t_i} \cap D_{t_j} \\ &= \{d \in D : \ a_{d,t_i} > 0 \ \wedge \ a_{d,t_j} > 0\} \end{aligned}$$

Andere geläufige Maße sind z. B. das Dice-Maß[12] und das Kosinus-Maß[13], die eine geringere Anzahl gemeinsamer Termauftritte weniger hart „bestrafen“ als das Jaccard-Maß (vgl. auch FERBER [50] oder MANNING und SCHÜTZE [95]).

Die Co-Occurrenz zweier Terme ist ein Maß für die Ähnlichkeit der beiden Terme und gibt somit das Ausmaß der Interdependenz der beiden Terme wieder. Idealerweise sollten diese statistisch beobachteten Termähnlichkeiten in etwa mit dem Übereinstimmen, was man intuitiv – durch die linguistische Erfahrung bedingt – an Ähnlichkeit für die jeweils betrachteten Terme erwarten würde. Konkret heißt das, dass immer dann, wenn zwei Terme über ein linguistisches Phänomen miteinander verknüpft sind, sich dieses in der aus der Co-Occurrenz abgeleiteten Termähnlichkeit widerspiegeln sollte. Dieses kann jedoch in der Praxis tendenziell eher nicht beobachtet werden, wie die Tabelle 3.1 es an einigen einfachen Beispielen der deutschen und englischen Sprache zeigt. Die Tabelle zeigt drei unterschiedlich Maße für die paarweise Co-Occurrenz zweier Terme. Als Dokumentenbasis werden alle, von der Suchmaschine *alltheweb*[14] indizierten, Internet-Dokumente verwendet.[15] Die einzelnen Häufigkeiten der jeweiligen Terme $\#D_{t_i}$ bzw $\#D_{t_j}$ sowie die paarweisen, gemeinsamen Häufigkeiten beider Terme $\#D_{t_i \wedge t_j}$ wurden unter Verwendung der *Advanced-Search* Funktion der Suchmaschine durch das Erstellen einer Suche nach Dokumenten, der jeweils betrachtete Sprache (Deutsch bzw. Englisch), die entweder einen bestimmten oder zwei vorgegebene Terme enthalten, ermittelt. Aufgrund der vielen von der Suchmaschine indizierten Dokumente kann man davon ausgehen, dass die hier berechneten Co-Occurrenzen signifikant sind.

Man kann der Tabelle 3.1 entnehmen, dass die Co-Occurrenz-basierten Ähnlichkeitsmaße bei den linguistischen Phänomenen der Flexion, Synonymie, Komposition, Hyponymie und Meronymie dazu tendieren, die Ähnlichkeit zwischen zwei Termen stark zu unterschätzen. Bei Wortgruppen hingegen besteht eher eine Tendenz, dass die Ähnlichkeit durch die Co-Occurrenz-basierten Schätzer überschätzt wird. Die Ursachen für diese Fehlschätzungen

[12] Dice-Maß:

$$\text{sim}(t_i, t_j) = \frac{2 \cdot \#D_{t_i \wedge t_j}}{\#D_{t_i} + \#D_{t_j}} \in [0...1] \quad \forall t_i, t_j \in T$$

[13] Kosinus-Maß:

$$\text{sim}(t_i, t_j) = \frac{\#D_{t_i \wedge t_j}}{\sqrt{\#D_{t_i} \cdot \#D_{t_j}}} \in [0...1] \quad \forall t_i, t_j \in T$$

[14] Web-Adresse der Suchmaschine *alltheweb* lautet: `http://www.alltheweb.com`

[15] Zum Zeitpunkt der Durchführung der Auswertung waren nach Angaben der Suchmaschine insgesamt über 3 Mrd. Dokumente indiziert.

Sprache	t_i	t_j	$\#D_{t_i}$	$\#D_{t_j}$	$\#D_{t_i \wedge t_j}$	Jaccard-Maß	Dice-Maß	Cosinus-Maß	ling. Phäno.	Erw. Ähn.
Deutsch	Auto	Autos	7.343.643	1.865.435	1.819.318	0,246	0,395	0,492	Flexion	nahe 1
Deutsch	Haus	Häuser	13.665.590	1.516.628	1.491.391	0,109	0,196	0,328	Flexion	nahe 1
Deutsch	Computer	Rechner	8.318.750	2.471.993	661.192	0,065	0,123	0,146	Synonymie	nahe 1
Deutsch	Auto	Automobil	7.343.643	668.413	226.075	0,029	0,056	0,102	Synonymie	nahe 1
Deutsch	Zwerg	Gartenzwerg	329.340	45.529	5.651	0,015	0,030	0,046	Komp. + Hypon.	hoch
Deutsch	Aktie	Stammaktie	1.005.054	15.189	13.226	0,013	0,026	0,107	Komp. + Hypon.	hoch
Deutsch	New	York	5.058.672	1.899.504	1.827.011	0,356	0,525	0,589	Wortgruppe	gering
Deutsch	Albert	Einstein	1.129.866	248.290	133.015	0,107	0,193	0,251	Wortgruppe	gering
Deutsch	Eingabegerät	Tastatur	640.075	1.698.352	53.653	0,023	0,046	0,051	Hyponymie	hoch
Deutsch	Computer	Festplatte	8.318.750	2.176.211	844.407	0,087	0,161	0,198	Meronymie	hoch
Deutsch	Haus	Computer	13.665.590	8.318.750	1.403.237	0,068	0,128	0,132	kein	nahe 0
Deutsch	Haus	Physik	13.665.590	857.634	168.065	0,012	0,023	0,049	kein	nahe 0
Englisch	New	York	551.899.093	87.699.589	84.452.150	0,152	0,264	0,384	Wortgruppe	gering
Englisch	OS	Windows	12.698.254	59.239.524	6.375.286	0,097	0,177	0,232	Hyponymie	hoch
Englisch	Computer	Harddisk	125.481.851	286.558	135.409	0,001	0,002	0,023	Meronymie	hoch
Englisch	Icebox	Fridge	160.686	3.702.573	13.462	0,003	0,007	0,017	Synonymie	nahe 1
Englisch	House	Houses	133.807.980	23.190.586	11.029.299	0,076	0,141	0,198	Flexion	nahe 1
Englisch	House	Computer	133.807.980	125.481.851	18.056.162	0,075	0,139	0,139	kein	nahe 0

Diese Auswertung ist mit Hilfe der Advanced-Search Funktion der Suchmaschine www.alltheweb.com am 17.10.2003 erstellt worden.

Tabelle 3.1: Co-Occurrenzen einiger Terme im WWW.

lassen sich durch folgende Zusammenhänge und Gepflogenheiten beim Schreiben von Dokumenten erklären:

- *Flexion*: Verschiedene Flexionsformen eines Nomen wie z. B Auto und Autos sollten dem linguistischen Verständnis nach, eine sehr hohe Ähnlichkeit (auf einer Skala von Null bis Eins also sehr nahe oder, wenn man nur den Themenbezug als Ähnlichkeitsmaß wählt, sogar gleich Eins) aufweisen. In der Praxis wird man jedoch feststellen, dass gerade in kürzeren Dokumenten nur eines der beiden Worte Verwendung findet. Der Grund dafür liegt darin, dass, wenn z. B. über ein konkretes Auto gesprochen wird, der Plural des Wortes in den meisten Fällen nicht gebraucht wird. Somit wird zwar von den Co-Occurrenz-Maßen eine Ähnlichkeit erkannt, diese liegt aber in der Regel deutlich unter dem aus linguistischer Sicht zu erwartenden Wert. Aus diesem Grund sollte auch bei Modellen mit immanenten Terminterdependenzen auf die gängigen Stemming-Verfahren[16] nicht verzichtet werden.

- *Synonymie*: Für zwei Synonyme erwartet man aus linguistischer Sicht ebenfalls eine sehr hohe Termähnlichkeit. In der Praxis ist die Erkennung von Synonymen mit Hilfe von Co-Occurrenz-Maßen stark von der Art der Dokumente abhängig. Bei narrativen Texten gehört es durchaus zum guten Stil, Synonyme häufig zu verwenden, um Wortwiederholungen zu vermeiden. Somit ist bei derartigen Texten die Chance sehr gut, Synonyme unter Verwendung von Co-Occurrenz-Maßen zu erkennen. Anderseits ist die unbegründete Verwendung von Synonymen bei wissenschaftlichen Texten[17] aufgrund von vorgegebenen Begriffsdefinitionen und zur Vermeidung von Interpretationsspielräumen eher weniger gerne gesehen. Besteht der Dokumentenkorpus überwiegend aus solchen Dokumenten, dann ist die Erkennung von Synonymen auf Basis von Co-Occurrenz-Maßen stark gefährdet.

- *Komposition*: Da die meisten IF/IR-Verfahren im englischsprachigen Raum konzipiert wurden und die englische Sprache nur wenige Komposita aufweist, ist das nun geschilderte Problem in der Literatur häufig vernachlässigt worden. Komposita, wie z. B. Gartenzwerg, bestehen aus mindestens zwei Wörtern, im genannten Beispiel aus Garten und Zwerg. Aus linguistischer Sicht muss der Begriff Gartenzwerg somit als Spezialfall (Hyponymie) von Zwerg eine Ähnlichkeit zu dem Begriff Zwerg aufweisen. Ebenfalls sollte Gartenzwerg eine thematische Ähnlichkeit zu Garten aufweisen. Zudem sollte Gartenzwerg bedeutungsidentisch mit der Wortfolge ein Zwerg für den Garten sein und somit eine sehr hohe Ähnlichkeit zu der Wortfolge haben. Man kann

[16] Vgl. dazu die Abschnitte 2.3.2.2 und 3.1.

[17] Eine Begründung für die Verwendung von Synonymen sind gleiche/ähnliche Definitionen von Begriffen in unterschiedlichen Fachgebieten bei Fachgebiet-übergreifender Literatur. So werden die Begriffe Wort und Term in dieser Arbeit – wenn diese nicht an einer Stelle explizit anders definiert werden – synonym verwendet. Der Grund für dieses Vorgehen liegt darin, dass die aus der Linguistik stammende Definition für Wort sich in den meisten Fällen mit dem in der IF/IR-Literatur verwendeten Begriff Term deckt. Um eine leichte Wiedererkennbarkeit der Begriffe für Fachleute des jeweiligen Fachgebiets zu gewährleisten, ist es sinnvoll im jeweiligen fachspezifischen Kontext den fachspezifischen Begriff zu verwenden und in den Ausnahmesituationen auf die speziellen Unterschiede zu verweisen.

feststellen, dass Autoren von Dokumenten dem kürzeren Kompositum häufig den Vorzug gegenüber der längeren Wortfolge geben. Dieses begründet sich in der geringeren Komplexität und damit besseren Lesbarkeit des Kompositums gegenüber der Wortfolge. Durch das systematische Vorziehen des Kompositums ist die Wahrscheinlichkeit, dass in kürzeren Texten sowohl das Kompositum als auch die Wortfolge vorkommt, relativ gering. Das hat zur Folge, dass die Ähnlichkeit zwischen den drei genannten Worten mit Co-Occurrenz-Verfahren systematisch unterschätzt wird.

- *Hyponymie und Meronymie*: Es ist tendenziell unüblich, alle Bestandteile oder Über- und Unterbegriffe eines Wortes in Dokumenten aufzuzählen, wenn dieses nicht gerade der Inhalt und das Ziel des Dokuments ist. Somit ist zu erwarten, dass die Co-Occurrenz zweier Worte, die über Hyponymie oder Meronymie miteinander verbunden sind, insbesondere in themenübergreifenden Dokumentensammlungen eher gering ist und somit nicht der linguistisch motivierten Erwartung entspricht.

- *Wortgruppen*: Viele der hier vorgestellten Verfahren messen Wortgruppen keine besondere Bedeutung bei. Dieses hat zur Folge, dass Wortgruppen, die sehr bekannte Eigennamen wie z. B. New York repräsentieren, aus Sicht der Co-Occurrenz-Maße Ähnlichkeiten suggeriert, die in Wirklichkeit nicht vorhanden sind. Da die Wortgruppe New York aufgrund der hohen Bekanntschaft des Ortes in vielen Dokumenten vorkommt, kommen auch die Worte New und York für sich betrachtet häufig gemeinsam vor. Somit ergibt sich gemäß der Co-Occurrenz eine hohe Ähnlichkeit zwischen den beiden Worten, die aus linguistischer Sicht nicht existiert.

Folgendes Fazit kann für die Modelle mit immanenten Terminterdependenzen gezogen werden: Die Idee, Terminterdependenzen alleine aus einer großen Menge von Dokumenten ohne einer weiteren menschlicher Eingabe abzuleiten, ist verführerisch. Leider zeigt sich in der Praxis, dass einfache auf Co-Occurrenz-basierende statistische Verfahren nicht immer in der Lage sind, Terminterdependenzen gemäß dem linguistischen Verständnis korrekt abzuleiten. Es lässt sich vielmehr zeigen, dass die Ableitung systematische Fehler aufweist, die von den linguistischen Phänomen, die zwei Begriffe verbinden, abhängen. Die Zuordnung zweier Begriffe zu einen linguistischen Phänomen bedarf allerdings einer menschlichen Eingabe (oder eines wesentlich komplexeren und die Linguistik umfassenden Modells). Insofern ist es nicht verwunderlich, dass im Experiment und in der Praxis die erwartete Qualitätssteigerung von Modellen mit immanenten Terminterdependenzen gegenüber den Modellen ohne Terminterdependenzen trotz des höheren Rechenaufwands nicht beobachtet werden konnte.[18] Unabhängig von dieser ernüchternden Erkenntnis werden im Folgenden die bekanntesten Modelle dieser Klasse – das Generalized Vector Space Model, das Modell des Latent Semantic Index und das Spreading Activation Neuronal Network – kurz vorgestellt.

[18] Vgl. z. B. [7, S. 44] und [107].

3.3.1 Generalized Vector Space Model (GVSM)

Das GVSM ist 1987 von S. WONG, ZIARKO, RAGHAVAN und P. WONG in [160] vorgestellt worden und stellt eine echte Erweiterung des VSM[19] dar, bei der Terme auch nicht orthogonal zueinander sein dürfen. Um dieses zu realisieren, wird beim GVSM ein Term (im Unterschied zum VSM) nicht durch eine eigene Dimension im Vektorraum repräsentiert, sondern ein Term besteht aus einer Menge von kleineren Vektoren, den sogenannten *Mintermen*.

Ein Minterm $m_i = (m_{i,1}, m_{i,2}, ..., m_{i,\#T}) \in M$ ist (zunächst unabhängig von den konkret existierenden Dokumenten) eine Kombination aus dem möglichen Vorhandensein bzw. Nicht-Vorhandensein von Termen in einem Dokument. Es gilt $m_{i,j} \in \{0; 1\}$. So bedeutet $m_{i,1} = 1$, dass beim Minterm m_i der Term t_1 vorhanden ist. Entsprechend bedeutet $m_{i,2} = 0$, dass beim Minterm m_i der Term t_2 nicht vorhanden ist. Aus der Kombinatorik ergibt sich, dass die Menge aller Minterme M genau $\#M = 2^{\#T}$ Elemente bzw. Minterme hat. Alle Minterme m_i sind von $i = \{1, 2, ..., 2^{\#T}\}$ eindeutig ($m_i \neq m_j$ für alle $i \neq j$) durchnummeriert.

Zur besseren Handhabbarkeit werden die beiden folgenden Funktionen definiert: Die Funktion g ist definiert als diejenige Funktion, die zu einem Minterm m_i und einem Term t_j den passenden, für den Minterm gültigen Eintrag $m_{i,j}$ zurückgibt:

$$g_{t_j}(m_i) = m_{i,j} \in \{0; 1\}$$

Die zweite Funktion $h(d)$ liefert zu jedem Dokument d einen passenden Minterm, der der Kombination aus Vorhandensein und Nicht-Vorhandensein von Termen in d entspricht:

$$h(d) = m_i \in M \quad \Leftrightarrow \quad \forall t_j \in T: \; m_{i,j} = \begin{cases} 1 & \text{falls } a_{d,t_j} > 0 \\ 0 & \text{falls } a_{d,t_j} = 0 \end{cases}$$

Jedem der Minterme $m_i \in M$ ist ein Vektor $\vec{m}_i$ derart zugeordnet, dass alle Minterme zusammen einen Raum mit $2^{\#T}$ Dimensionen aufspannen. Dabei gilt:

$$\begin{aligned} \vec{m}_1 &= (1, 0, ..., 0, 0) \\ \vec{m}_2 &= (0, 1, ..., 0, 0) \\ &\vdots \\ \vec{m}_{2^{\#T}} &= (0, 0, ..., 0, 1) \end{aligned}$$

Jedem Term $t_i \in T$ ist beim GVSM ein eigener Termvektor $\vec{t_i}$ zugeordnet. Dieser Termvektor ist normiert und setzt sich aus einer Summe von Minterm-Vektoren zusammen:

[19] Vgl. Abschnitt 3.2.2.

$$\vec{t_i} = \frac{\sum_{\forall r:\, g_{t_i}(m_r)=1} c_{i,r}\vec{m}_r}{\sqrt{\sum_{\forall r:\, g_{t_i}(m_r)=1} c_{i,r}^2}}$$

mit

$$c_{i,r} = \sum_{d_j \in D:\, g_l(h(d_j))=g_l(m_r)\ \forall l \in T} a_{d_j,t_i}$$

Diese Definition von Termvektoren impliziert, dass der Winkel zwischen zwei Termvektoren abhängig ist von der Häufigkeit des gemeinsamen Vorkommens der beiden Terme im Dokumentenbestand. Kommen die beiden Terme häufig gemeinsam vor, dann ist der Winkel klein. Kommen die Terme selten gemeinsam in Dokumenten vor, dann ist der Winkel groß. Somit ist der Winkel ein Maß für die Co-Occurrenz.[20]

Der Dokumentvektor $\vec{d_i}$ zum Dokument $d \in D$ berechnet sich als die, mit dem Termvorkommen gewichtete Summe der Termvektoren:

$$\vec{d_i} = \sum_{t_j \in T} a_{d_i,t_j}\vec{t_j}$$

Die Ähnlichkeit zwischen zwei Dokumenten $d_i, d_j \in D$ berechnet sich wie beim VSM über das normierte Skalarprodukt der Dokumentenvektoren und entspricht somit dem Kosinus-Maß:

$$\text{sim}(d_i, d_j) = \frac{\vec{d_i}\vec{d_j}}{|\vec{d_i}||\vec{d_j}|}$$

Wie beim VSM werden Anfragen als virtuelle Dokumente betrachtet, wodurch dieses Modell auch für ein k-nearest-neighbour basiertes IF geeignet ist. Da Stoppwörter nicht explizit im GVSM modelliert werden, ist die Anwendung einer Stoppwortliste von Vorteil.

3.3.2 Latent Semantic Index (LSI)

Wie das GVSM basiert das 1988 von FURNAS ET AL. [53] vorgestellte LSI Modell von seiner grundsätzlichen Konzeption auf dem VSM, allerdings arbeitet das LSI nicht mit einer Dekomposition von Termvektoren in Minterm-Vektoren, sondern die Zahl der betrachteten Dimensionen des Vektorraumes wird unter Verwendung der *Singular Value Decomposition* (SVD) [51] reduziert.

Ausgangspunkt des LSI ist die Dokument-Term Matrix $M \in \mathbb{R}^{\#T \times \#D}$, bei der die Spalten die Dokumente mit ihrem Termvorkommen (= Dokumentvektoren) und die Zeilen die

[20] Eine ausführliche Herleitung dieses Sachverhaltes findet sich in [160].

Terme mit ihrem Vorkommen in den verschiedenen Dokumenten repräsentieren:

$$M = \begin{pmatrix} a_{d_1,t_1} & a_{d_2,t_1} & \cdots & a_{d_{\#D},t_1} \\ a_{d_1,t_2} & a_{d_2,t_2} & \cdots & a_{d_{\#D},t_2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{d_1,t_{\#T}} & a_{d_2,t_{\#T}} & \cdots & a_{d_{\#D},t_{\#T}} \end{pmatrix}$$

Um die Dimension des Vektorraumes zu reduzieren, greift das LSI auf das SVD zurück. Gemäß dem SVD kann jede Matrix M in zwei orthonormale Matrizen T, D und eine pseudo-diagonale Matrix S mit den singulären Werten der Matrix M zerlegt werden:[21]

$$M = TSD^\top$$

Alle vier Matrizen der Gleichung haben denselben Rang $r \leq \min(\#T, \#D)$. Zur Dimensionsreduktion wird der Rang der Matrizen T, S, D auf den Wert $s < r$ reduziert, indem nur diejenigen s Zeilen oder Spalten der Matrizen beibehalten werden, so dass in der Diagonal-Matrix S die s-größten Werte erhalten bleiben. Das Ergebnis dieser Dimensionsreduktion sind die Matrizen T_s, S_s, D_s. Die sich daraus errechnende Matrix $M_s = T_s S_s D_s^\top$ ist die beste dimensionsreduzierte Annährung zur Matrix M im Sinne der kleinsten Quadrate Methode.

Die paarweisen Ähnlichkeiten zwischen allen Dokumenten kann durch die Multiplikation der Matrix mit sich selbst (transponiert) berechnet werden:

$$M_s M_s^\top$$

Natürlich reicht es aus, nur einzelne Zeilen/Spalten der Matrix auszurechnen, um ein bestimmtes Dokument mit allen anderen zu vergleichen. Anfragen sind beim LSI wie auch beim VSM und GVSM als virtuelle Dokumente zu behandeln.

FURNAS ET AL. formulieren die dem LSI zu Grunde liegende Annahme bezüglich der Dimensionsreduktion wie folgt:

> „... some closely related documents should contain nearly identical patterns of terms, and synonymous terms should have heighly similar pattern of occurrence across documents“ [53, S. 467]

Konkret bedeutet dieses, dass das LSI zur Bestimmung der Terme, die über die Dimensionsreduktion zusammengefasst werden, implizit auf das Maß der Co-Occurrenz zurück greift.

3.3.3 Spreading Activation Neuronal Network (SANN)

In der Literatur[22] werden verschiedene SANN-Topologien für das IR diskutiert. Das Grundkonzept von SANN's für den Anwendungsbereich des IR wird hier im Folgenden in Anlehnung an WILKINSON und HINGSTON [159] vorgestellt. Die Topologie dieses SANN zeigt

[21] Eine Übersicht über algorithmische Verfahren zur SVD-Zerlegung findet sich in [56].
[22] Siehe z. B. [16, 85, 159].

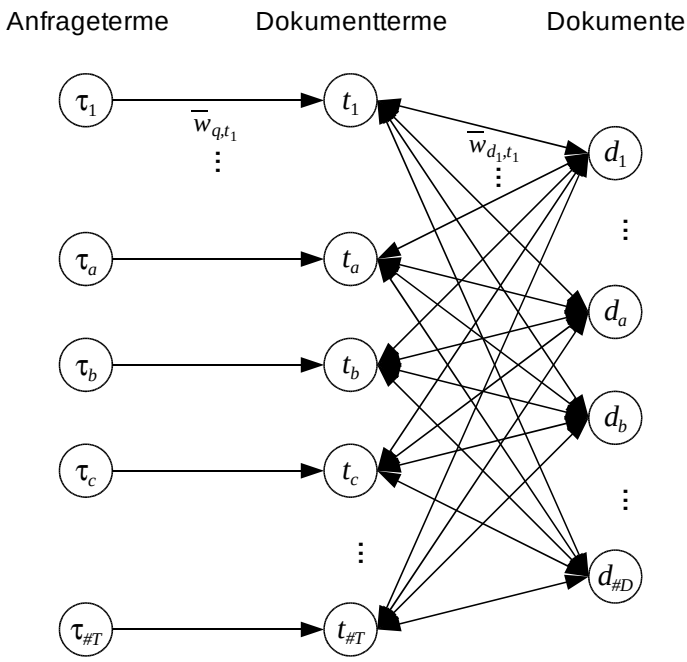


Abbildung 3.5: Topologie eines SANN für IR.

Abbildung 3.5. Es besteht aus drei neuronalen Schichten: einer Anfrageschicht, in der die Terme der Anfrage repräsentiert werden, einer Zwischenschicht, in der die Neuronen einzelne Dokumentterme repräsentieren und der Dokumentschicht, bei der jedes Neuron ein einzelnes Dokument verkörpert und aus der die Ähnlichkeiten für die einzelnen Dokumente zu der gestellten Anfrage ausgelesen werden. Während die Neuronen der Anfrageterme mit den Dokumenttermen unidirektional und paarweise verbunden sind, sind die Neuronen der Dokumentterme mit den Dokumentneuronen bidirektional und vollständig miteinander vernetzt.

Eine Anfrage $q \subseteq T$ wird als eine Menge von Termen aufgefasst. Jedem Term $t \in T$ wird relativ zur Anfrage q ein Gewicht $w_{q,t}$ z. B. über tf-idf (vgl. Gleichung 3.1 auf Seite 51) zugeordnet. Des Weiteren ist jedem Term auch ein Gewicht $w_{d,t}$ in Relation zu allen Dokumenten $d \in D$ (z. B. gleichfalls unter Verwendung von tf-idf) zugewiesen. Die Gewichte $w_{q,t}$ bzw. $w_{d,t}$ haben den Wert Null, wenn der Term t nicht in der Anfrage q bzw. dem Dokument d enthalten ist. Zum Durchführen einer Anfrage werden *alle* Neuronen τ_i der Anfrageschicht (unabhängig von den Termen in der Anfrage q) mit dem Aktivierungspotential von Eins initiiert. Dieses Aktivierungspotential wird mit $\bar{w}_{q,t}$ gewichtet und an die Neuronen t_i, die die Dokumentterme repräsentieren, weitergeleitet. $\bar{w}_{q,t}$ kann dabei als normalisiertes Gewicht aus

den $w_{q,t}$ wie folgt berechnet werden:

$$\bar{w}_{q,t} = \frac{w_{q,t}}{\sqrt{\sum_{t \in T} w_{q,t}^2}}$$

Daraus folgt, dass jedes Neuron der Dokumentterme das Aktivierungspotential $1 \cdot \bar{w}_{q,t}$ hat. Diese Neuronen gewichten das Potential mit $\bar{w}_{d,t}$ und senden es an die Dokumentneuronen weiter. Das Gewicht $\bar{w}_{d,t}$ berechnet sich dabei wie folgt:

$$\bar{w}_{d,t} = \frac{w_{d,t}}{\sqrt{\sum_{t \in T} w_{d,t}^2}}$$

Da die Dokumentneuronen vollständig mit allen Termneuronen vernetzt sind, ergibt sich das Aktivierungspotential der Dokumentneuronen – und damit die Ähnlichkeit $\text{sim}_1(d, q)$ der Dokumente d zu der gestellten Anfrage q nach dem ersten Schritt – aus der Summe aller in ein Dokumentneuron eingehenden, gewichteten Aktivierungspotentiale der Termneuronen:

$$\text{sim}_1(d, q) = \sum_{t \in T} \bar{w}_{q,t} \bar{w}_{d,t}$$

Es kann gezeigt werden, dass unter Verwendung des hier vorgestellten Gewichtungsschemas, die vom SANN berechneten Ähnlichkeiten zwischen Dokumenten und Anfragen nach dem ersten Schritt zu den von einem VSM (Abschnitt 3.2.2) berechneten Ähnlichkeiten äquivalent sind.[23]

Um das Suchergebnis weiter zu verbessern, werden in den nächsten Schritten die Aktivierungspotentiale der Dokumentneuronen über eine einer Feedback-Schleife an die Termneuronen geleitet, die dann wieder zurück geleitet werden. Dadurch können Dokumentneuronen derjenigen Dokumente aktiviert werden, die keinen der in der Anfrage enthaltenen Terme enthalten. Somit wird indirekt eine Interdependenz zwischen Termen aufgebaut, die häufig in Dokumenten co-occurieren. Um den Abbruch der Feedback-Schleife sicherzustellen, sind geeignete Abbruchkriterien, wie z. B eine minimale Aktivierungsschwellwertänderung für das Feedback oder eine maximale Anzahl an Schritten, festzulegen. Weitere Details dazu finden sich in [159].

Da dieses Verfahren Stoppwörter und verschiedene Flexionsformen von Wörtern nicht explizit berücksichtigt, ist die Anwendung von Stoppwortlisten und Stemming-Verfahren sinnvoll.

3.4 Modelle mit transzendenten Terminterdependenzen

Wie bei den Modellen mit immanenten Terminterdependenzen liegt auch den Modellen mit transzendenten Terminterdependenzen *keine* Annahme über die Orthogonalität oder Unabhängigkeit von Termen zu Grunde. Im Unterschied zu den Modellen mit immanenten Terminterdependenzen können die Interdependenzen zwischen den Termen bei den Modellen mit

[23] Vgl. dazu [7, S. 47f] und [105].

transzendenten Terminterdependenzen nicht ausschließlich aus dem Dokumentenbestand und dem Modell abgeleitet werden. Das heißt, dass die den Terminterdependenzen zu Grunde liegende Logik als über das Modell hinausgehend (transzendent) modelliert wird.

Das bedeutet, dass in den Modellen mit transzendenten Terminterdependenzen das Vorhandensein von Terminterdependenzen explizit modelliert wird, aber dass die konkrete Ausprägung einer Terminterdependenz zwischen zwei Termen direkt oder indirekt von außerhalb (z. B. von einem Menschen) vorgegeben wird. Zu den Verfahren bei denen die Terminterdependenzen direkt vorgegeben werden, gehören das Topic-based Vector Space Model (Kapitel 4) sowie die hier vorgestellte Erweiterung (Kapitel 5) als auch das Retrieval by Logical Imaging (Abschnitt 3.4.2). Bei diesen Verfahren werden die konkreten Ausprägungen der Terminterdependenzen direkt, z. B in Form einer Tabelle, Matrix oder einer Ontologie von außen vorgegeben. Im Unterschied dazu greifen die Modelle, bei denen die Interdependenzen indirekt vorgegeben werden, auf Lernverfahren der künstlichen Intelligenz zurück, um die indirekten Interdependenzinformationen in eine nutzbare Form umzuwandeln. Das Backpropagation Neuronal Network (Abschnitt 3.4.3) ist ein solches Verfahren, dass die Interdependenzen anhand von Trainingsdaten erlernt. Bei dem Fuzzy Set Model (Abschnitt 3.4.1) handelt es sich um ein Mischverfahren, dass zunächst mit einer direkten Vorgabe für Terminterdependenzen beginnt und diese direkte Vorgabe später mit Trainingsdaten (indirekten Vorgaben) verfeinert.

Der Vorteil von Modellen mit transzendenten Terminterdependenzen gegenüber Modellen mit immanenten Terminterdependenzen ist der, dass diese Modelle durch die externe Vorgabe von Terminterdependenzen die linguistischen Phänomene besser erfassen können, sofern die Vorgabe geeignet ist. Da die Vorgabe zum Zeitpunkt der Anfrageberechnung statisch ist, kann ein höherer Aufwand bei der Generierung der Vorgabe getrieben werden und es ist somit möglich, linguistische Phänomene besser zur berücksichtigen. Im Extremfall können die Terminterdependenzen von menschlichen Experten vorgegeben bzw. geprüft werden, um eine hohe Qualität der Vorgabe sicherzustellen. Der Nachteil dieser Modelle ist der bereits erwähnte höhere Aufwand bei der Generierung der Terminterdependenzen und der für die explizite Speicherung der Terminterdependenzen notwendige Speicherplatzbedarf.

3.4.1 Fuzzy Set Model (FSM)

Die Fuzzy-Mengen Theorie [163, 164, 169] handelt von der Repräsentation von Mengen, deren Grenzen nicht scharf, sondern *unscharf* (engl. fuzzy) definiert sind. Eine Fuzzy-Menge $\tilde{F}$ ist eine Menge von Tupeln. Jedes Tupel enthält ein Element e und den Zugehörigkeitswert (engl. Membership) des Elementes zu der Menge. Die Zugehörigkeitswerte von Elementen werden üblicher Weise über eine Zugehörigkeitsfunktion $\mu_{\tilde{F}}(e) \in [0...1]$ festgelegt:

$$\tilde{F} = \{(e, \mu_{\tilde{F}}(e)) : \forall e\}$$

Ein Zugehörigkeitswert von Null bedeutet, dass das entsprechende Element nicht Bestandteil der Fuzzy-Menge ist. Alle Werte größer Null bedeuten, dass das Element zu einem gewissen Grad Bestandteil der Fuzzy-Menge ist, wobei der Wert Eins den maximalen Grad der Zugehörigkeit bedeutet. Eine Fuzzy-Menge $\tilde{F}$, deren Zugehörigkeitswerte $\mu_{\tilde{F}}(e)$ entweder nur die

Werte Null oder Eins annehmen, ist äquivalent zu der normalen Menge F, die nur aus den Elementen besteht, deren Zugehörigkeitswert in $\tilde{F}$ gleich Eins ist ($F = \{e : \ \mu_{\tilde{F}}(e) = 1\}$).

Es gibt verschiedene Ansätze Fuzzy-Mengen für das IF nutzbar zu machen.[24] Zu den bekanntesten Ansätzen zählt der Ansatz von OGAWA, MORITA und KOBAYASHI [107], dessen Konzept hier kurz und in einer modifizierten Form unter Verwendung eines Beispiels erläutert wird, um eine bessere Vergleichbarkeit mit dem SBM in Abschnitt 3.2.1 zu ermöglichen. Dabei definieren wir q als folgende Beispielanfrage, bei der diejenigen Dokumente gesucht werden, die den Term t_a und den Term t_b oder nicht den Term t_c enthalten:

$$q = t_a \wedge (t_b \vee \neg t_c) \quad \text{mit} \quad t_a, t_b, t_c \in T$$

T_d sei die Menge der Terme, die in einem Dokument $d \in D$ enthalten sind:

$$T_d = \{t \in T : a_{d,t} > 0\}$$

Die Fuzzy-Menge $\tilde{D}_t$ definiert diejenigen Dokumente, die den Term $t \in T$ enthalten:

$$\tilde{D}_t = \{(d, \mu_{\tilde{D}_t}(d)) : d \in D\}$$

Die Zugehörigkeitsfunktion $\mu_{\tilde{D}_t}(d)$ wird dabei derart definiert, dass Dokumente, die den Term t enthalten, einen Zugehörigkeitswert von Eins haben. Dokumente, die den Term t nicht enthalten, erhalten einen geringeren Wert als Eins. Der konkrete Wert dieser Dokumente hängt davon ab, ob das Dokument irgendwelche anderen Terme enthält, die mit dem Term t interdependent sind:

$$\mu_{\tilde{D}_t}(d) = 1 - \prod_{u \in T_d} (1 - c_{t,u}) \in [0...1]$$

Die Interdependenz zwischen zwei Termen wird bei diesem Ansatz in Form einer *Keyword-Connection-Matrix* vorgegeben. Jedes Element dieser Matrix $c_{t,u}$ ist ein Maß für die Interdependenz zwischen den beiden Termen t und u. Eine Null bedeutet keine Interdependenz, eine Eins bedeutet maximale Interdependenz – also (totale) Synonymie der beiden Terme. Folgendes gilt für die Elemente der Matrix:

$$\begin{aligned} & c_{t,u} \in [0...1] && \forall t, u \in T \\ \wedge \quad & c_{t,u} = 1 && \forall t = u \end{aligned}$$

Folgende Mengenoperationen lassen sich auf der Fuzzy-Menge $\tilde{D}_t$ definieren: Ein Element ist in Fuzzy-Mengen nicht einfach enthalten, sondern es hat einen Zugehörigkeitswert. Daher liefert die Fuzzy-Enthalten Operation keinen boolschen Rückgabewert, sondern den Zugehörigkeitswert des zu prüfenden Elements:

$$(d \in \tilde{D}_t) = \mu_{\tilde{D}_t}(d)$$

[24] Vgl. z. B. [81, 120, 121, 106, 100, 101].

Die Schnittmenge zwischen beliebig vielen Fuzzy-Mengen $\tilde{D}_t, \tilde{D}_u, ..., \tilde{D}_v$ wird dadurch gebildet, indem die Zugehörigkeitsfunktionen der einzelnen Fuzzy-Mengen miteinander nach folgendem Schema kombiniert werden:

$$\mu_{\tilde{D}_t \cap \tilde{D}_u \cap \cdots \cap \tilde{D}_v}(d) = \mu_{\tilde{D}_t}(d)\mu_{\tilde{D}_u}(d) \cdots \mu_{\tilde{D}_v}(d)$$

Auch die Vereinigung von Fuzzy-Mengen wird durch die Kombination der Zugehörigkeitsfunktionen gebildet. Dabei schlagen OGAWA, MORITA und KOBAYASHI folgendes Schema vor:

$$\mu_{\tilde{D}_t \cup \tilde{D}_u \cup \cdots \cup \tilde{D}_v}(d) = 1 - (1 - \mu_{\tilde{D}_t}(d))(1 - \mu_{\tilde{D}_u}(d)) \cdots (1 - \mu_{\tilde{D}_v}(d))$$

Abschließend wird das Komplement $\complement\tilde{D}_t$ zu $\tilde{D}_t$ durch die folgende Modifikation der Zugehörigkeitsfunktion realisiert:

$$\mu_{\complement\tilde{D}_t}(d) = 1 - \mu_{\tilde{D}_t}(d)$$

Die Ähnlichkeit $\text{sim}(d, q)$ zwischen einem Dokument $d \in D$ und der Beispielanfrage q lässt sich im ersten Schritt analog zum SBM in Abschnitt 3.2.1 berechnen, indem die Anfrage derart umgeformt wird, dass t_a, t_b, t_c durch $\tilde{D}_{t_a}, \tilde{D}_{t_b}, \tilde{D}_{t_c}$ ersetzt werden und $\wedge, \vee, \neg$ durch $\cap, \cup, \complement$. Das Ergebnis dieser Umformung ist eine Reihe von Fuzzy-Mengenoperationen, die im Ergebnis zu jedem Dokument einen Zugehörigkeitswert liefern, der als Ähnlichkeit des Dokuments in Bezug auf die Anfrage interpretiert wird. Zur einfacheren Verknüpfung der Zugehörigkeitsfunktionen wird die Anfrage in die konjunktive Normalform umgewandelt:

$$\begin{aligned}
\text{sim}(d, q) &= d \in (\tilde{D}_{t_a} \cap (\tilde{D}_{t_b} \cup \complement\tilde{D}_{t_c})) \\
&= d \in ((\tilde{D}_{t_a} \cap \tilde{D}_{t_b} \cap \tilde{D}_{t_c}) \cup (\tilde{D}_{t_a} \cap \tilde{D}_{t_b} \cap \complement\tilde{D}_{t_c}) \cup (\tilde{D}_{t_a} \cap \complement\tilde{D}_{t_b} \cap \complement\tilde{D}_{t_c})) \\
&= 1 - (1 - \mu_{\tilde{D}_{t_a}}(d)\mu_{\tilde{D}_{t_b}}(d)\mu_{\tilde{D}_{t_c}}(d))(1 - \mu_{\tilde{D}_{t_a}}(d)\mu_{\tilde{D}_{t_b}}(d)(1 - \mu_{\tilde{D}_{t_c}}(d))) \\
&\quad (1 - \mu_{\tilde{D}_{t_a}}(d)(1 - \mu_{\tilde{D}_{t_b}}(d))(1 - \mu_{\tilde{D}_{t_c}}(d)))
\end{aligned}$$

Zur Initialisierung der Keyword-Connection-Matrix verwenden OGAWA, MORITA und KOBAYASHI die Co-Occurrenz von Termen in dem Dokumentenbestand unter Anwendung des Jaccard-Maßes (vgl. Gleichung 3.2 auf Seite 63). Unter Verwendung von Trainingsdaten und einer Fehlerfunktion haben die Autoren ein Gradienten-Abstiegsverfahren realisiert, bei dem durch Modifikation der Keyword-Connection-Matrix der Fehler des von ihnen implementierten IR-Systems in Bezug auf die verwendeten Trainingsdaten reduziert wurde. Die Autoren konnten zeigen, dass sich die Qualität des IR-Systems durch das Lernverfahren deutlich steigern lässt. Gleichzeitig kann man [107] entnehmen, dass sich die Werte der Terminterdependenzen in der Keyword-Connection-Matrix teilweise in einem deutlichen Ausmaß ändern. Dieses ist ein Hinweis darauf, dass die durch Co-Occurrenz gewonnen Initialwerte sich bei weitem nicht in der Nähe des Optimums befinden. Diese Beobachtung stützt die in Abschnitt 3.3 geäußerte These, dass Co-Occurrenz-Maße für die Bestimmung von Terminterdependenzen wenig geeignet sind.

Da das FSM Terme nicht gewichtet, ist die Verwendung von Stoppwortlisten zu empfehlen. Des Weiteren haben die Autoren von [107] ein Stemming-Verfahren eingesetzt, um Wörter in ihre Stammform zu überführen. Dieses ist sinnvoll, weil das Jaccard-Maß nicht geeignet ist, verschiedene Flexionsformen eines Wortes zu erkennen.[25] In [107] ist für die Optimierung dem FSM lediglich eine einfache, nur aus dem Wort CAD bestehende Anfrage mit den dazugehörigen gewünschten Dokumenten als Trainingsdaten zur Verfügung gestellt worden. Das daraus resultierende Training war somit stark auf dieses eine Wort und die engeren „Verwandten“ dieses Wortes fokussiert. Um eine gleichmäßige Optimierung der Keyword-Connection-Matrix zu gewährleisten, ist es aber erforderlich, eine Vielzahl von Anfragen (und den dazugehörigen Dokumenten) für das Training zu verwenden. Idealerweise sollte zumindest jeder Term eine eigene Anfrage erhalten, um ein gewisse Gleichmäßigkeit der Optimierung sicherzustellen. Dieses Unterfangen dürfte sich in der Praxis bei der Vielzahl von Termen und Dokumenten, mit denen gängige IF/IR-Systeme umgehen müssen – mehrere Zehntausend Terme und mehrere Tausend bis mehrere Millionen Dokumente sind da ein realistischer Wert – aufgrund des hohen Aufwands als problematisch erweisen.

3.4.2 Retrieval by Logical Imaging (RbLI)

Das RbLI ist 1995 von CRESTANI anf RIJSBERGEN in [38] vorgestellt werden. Im Unterschied zu den anderen probabilistischen Modellen (BIR, LM, INM und BNM) basiert das RbLI nicht auf dem Theorem von BAYES, sondern auf der Methode des *Logical-Imaging* [140, 88], deren Fundament die von KRIPKE konzipierte *Possible-World-Semantics* [82] ist. [35] Die Possible-World-Semantics postuliert die Existenz einer Menge von möglichen Welten W in denen verschiedene Aussagen x, y in Abhängigkeit von der betrachteten Welt wahr oder falsch sein können. Gemäß dieser Semantik ist der Wahrheitswert der Aussage, dass in der Welt $w \in W$ aus y ein x folgt ($y \rightarrow x$), äquivalent zu dem Wahrheitswert der Aussage x, in der zu w ähnlichsten Welt, in der y wahr ist. Formal lässt sich dieser Sachverhalt wie folgt darstellen:

$$\tau(w, y \rightarrow x) = \tau(\sigma(w, y), x) \tag{3.3}$$

Dabei liefert die Funktion $\sigma(w, y) \in W$ diejenige Welt zurück, die der Welt w am ähnlichsten ist und in der y wahr ist. Die Funktion $\tau(w, y)$ ist eine Wahrheitsfunktion mit folgenden Eigenschaften:

$$\tau(w, y) = \begin{cases} 1 & \text{falls } y \text{ in der Welt } w \text{ wahr ist,} \\ 0 & \text{sonst} \end{cases}$$

$$\tau(w, \neg y) = 1 - \tau(w, y)$$

Die Anwendung des durch die Gleichung 3.3 beschriebenen Sachverhaltes wird als Logical-Imaging bezeichnet und bildet die Grundlage für das RbLI.

[25] Vgl. Abschnitt 3.3.

Das RbLI geht von der Annahme aus, dass ein Term t eine Welt im Sinne der Possible-World-Semantics darstellt. Die Menge aller Welten ist somit die Menge aller Terme T. $P(t)$ ist die Wahrscheinlichkeit dafür, dass die Welt $t \in T$ beobachtet wird. Da alle Welten voneinander verschieden sind, gilt folgendes:

$$\sum_{t \in T} P(t) = 1$$

CRESTANI bezeichnet $P(t)$ auch als ein Maß für die Wichtigkeit eines Terms in Relation zu den anderen Termen. [35, S. 257] Dokumente $d \in D$ und Anfragen q werden beim RbLI als Bestandteile bzw. Aussagen einer Welt bzw. eines Terms t angesehen. Demnach ist die Aussage d bzw. q genau dann wahr, wenn d bzw. q den Term t enthält. Somit ergeben sich folgende Wahrheitsfunktionen $\tau(t, d)$ bzw. $\tau(t, q)$:

$$\tau(t, d) = \begin{cases} 1 & \text{falls der Term } t \text{ im Dokument } d \text{ enthalten ist,} \\ 0 & \text{sonst} \end{cases}$$

$$\tau(t, q) = \begin{cases} 1 & \text{falls der Term } t \text{ in Anfrage } q \text{ enthalten ist,} \\ 0 & \text{sonst} \end{cases}$$

Die Ähnlichkeit $\text{sim}(d, q)$ zwischen einem Dokument d und einer Anfrage q ist beim RbLI die Wahrscheinlichkeit dafür, dass aus dem Dokument die Anfrage gefolgert werden kann: $P(d \to q)$. Diese berechnet sich wie folgt:

$$\begin{aligned} \text{sim}(d, q) &= P(d \to q) \\ &= \sum_{t \in T} P(t)\tau(t, d \to q) \\ &= \sum_{t \in T} P(t)\tau(\sigma(t, d), q) \end{aligned}$$

Die zweite Zeile ergibt sich daraus, dass die Wahrscheinlichkeit dafür, dass eine Aussage unabhängig von den Welten wahr ist, der Summe der Wahrscheinlichkeiten derjenigen Welten entspricht, in denen die Aussage wahr ist. Die dritte Zeile ist das Ergebnis der Anwendung des Logical-Imaging (Gleichung 3.3).

Weil die exakten Wahrscheinlichkeiten für das IR nicht von Interesse sind und die Wahrscheinlichkeiten lediglich zu einem Ranking benutzt werden, verwenden CRESTANI und RIJSBERGEN [38] in ihrer Implementierung für $P(t)$ die Inverse-Document-Frequency $idf(t)$:

$$P(t) = idf(t) = -\log \frac{\#\{d \in D : \ a_{d,t} > 0\}}{\#D}$$

In diesem Zusammenhang wird eine theoretische Schwäche des RbLI-Ansatzes deutlich: Wie oben bereits erwähnt, bezeichnet CRESTANI die Wahrscheinlichkeit $P(t)$ auch als Maß für die Wichtigkeit eines Terms in Relation zu den anderen Termen. [35, S. 257] Aus diesem

Grunde ist es verständlich, dass CRESTANI und RIJSBERGEN $P(t)$ mit der Inverse-Document-Frequency gleichsetzen, weil somit seltene Terme die Dokumente gut gegeneinander abgrenzen, eine hohe Wahrscheinlichkeit[26] und somit eine hohes Maß an Wichtigkeit erhalten und häufige Terme (zu denen insbesondere die häufig vorkommenden Stoppwörter zählen) ein geringes Maß an Wichtigkeit/Wahrscheinlichkeit erhalten. Das Problem ist aber, dass diese Wahrscheinlichkeitsschätzung nicht mit der, aus probabilistischer Sicht erwarteten Abschätzung von Wahrscheinlichkeiten über statistische Beobachtungen vereinbar ist. Gemäß den statistischen Beobachtungen ist festzustellen, dass Stoppwörter und diejenigen Terme, die nach CRESTANI und RIJSBERGEN nur eine geringe Wichtigkeit/Wahrscheinlichkeit haben, in Dokumenten häufig vorkommen und umgekehrt. Insofern ist das RbLI es aus theoretischer Sicht nicht konsistent, weil das RbLI zwar auf probabilistischen Methoden aufbaut, aber zur Schätzung von Termwahrscheinlichkeiten willkürliche, nicht auf Eins normierte und der probabilistischen Sicht zuwiderlaufende Wahrscheinlichkeitsschätzungen für $P(t)$ vornimmt.

Die Funktion $\sigma(t, d)$ definiert zu jedem Term t einen Term, der im Dokument d enthalten ist und der zu t die größte Ähnlichkeit hat. Als Ähnlichkeitsmaß verwenden CRESTANI und RIJSBERGEN das Expected-Mutual-Information-Measure [151]. Dieses Maß nutzt die Co-Occurrenz von Termen zur Bestimmung der Ähnlichkeit. Dieses ist bedauerlich, weil Co-Occurrenz-basierte Ähnlichkeitsmaße für Dokumententerme die Phänomene der Linguistik nur unzureichend erfassen (vgl. Abschnitt 3.3). Somit werden die theoretisch möglichen Vorzüge von Modellen mit transzendenten Terminterdependenzen beim RbLI nicht vollständig umgesetzt.[27]

Das hier vorgestellte Verfahren wird auch als das *Standard Imaging* Verfahren bezeichnet. Zu diesem Verfahren werden in der Literatur folgende Erweiterungen bzw. Modifikationen diskutiert: *General Imaging* [39], *Proportinal Imaging* [134, 37], *Mixed Imaging* [36]. Bei diesen Verfahren wird das RbLI derart modifiziert, dass in einem Dokument nicht vorhandene Terme nicht nur durch den ähnlichsten Term ersetzt werden.

---

[26] Von der Tatsache, dass die Wahrscheinlichkeiten über das Inverse-Document-Frequency Maß nicht normiert sind, wird jetzt einmal abgesehen. Diese Normierung kann mit geringem Aufwand hergestellt werden, indem $P(t)$ wie folgt definiert wird:

$$P(t) = \frac{idf(t)}{\sum_{u \in T} idf(u)}$$

Das Fehlen der Normierung hat keinen schwächenden Einfluss auf die Argumentation, weil lediglich die relativen Verhältnisse der verschiedenen $P(t)$ in die Argumentation eingehen.

[27] In [35, S. 263ff] stellt CRESTANI fest, dass das RbLI in bestimmten Fällen dazu neigt denjenigen Dokumenten den Vorzug zu geben, die die Terme einer Anfrage in einem „unüblichen“ Kontext enthalten. Im von CRESTANI vorgestellten Beispiel werden auf eine Anfrage mit den zwei Termen bat (engl. Homograph für Fledermaus oder Schlagholz/Schläger) und cricket (engl. Homograph für Grille oder den Sport Kricket) diejenigen Dokumente vom RbLI bevorzugt, die sich mit Fledermäusen beschäftigen. Eigentlich würde man bei einer derartigen Anfrage erwarten, dass Dokumente bezüglich des Kricket-Sports (bei dem Schlaghölzer verwendet werden) bevorzugt werden. Die Ursache dieses Phänomens liegt zum einen darin, dass Termähnlichkeiten über Co-Occurrenz-Maße, wie in Abschnitt 3.3 dargelegt, problematisch sind, was durch unausgeglichene Dokumentenverteilungen noch verstärkt werden kann. Zum Anderen ist die Funktion $\sigma(t, d)$ im RbLI die Ursache. Dadurch, dass nur die ähnlichsten Terme bei der Ähnlichkeitsberechnung berücksichtigt werden, ist das RbLI sehr empfindlich gegenüber falschen Termähnlichkeiten, die einen hohen Betrag haben (Ausreißer).

3.4.3 Backpropagation Neuronal Network (BNN)

Ohne auf eine konkrete Anwendung Bezug zu nehmen, können über BNN's folgende Aussagen getroffen werden: Sie sind in der Lage eine (fast) beliebigen Funktion $\vec{a} = f(\vec{e})$ mit einem Eingabevektor $\vec{e}$ einer bestimmten Dimensionalität und einem Ausgabevektor $\vec{a}$ einer vorher festgelegten Dimensionalität zu simulieren. Im Unterschied zu einer herkömmlichen Programmierung von Funktionen in einem Rechner wird das Verfahren zur Berechnung des Ausgabevektors bei BNN's nicht direkt vorgegeben, sondern das Netzwerk wird anhand von vorgegebenen Trainingsdaten angelernt. Dabei lernt das Netz die Trainingsdaten nicht „auswendig", sondern es ist bei geeigneter Parameterauswahl in der Lage die Trainingsdaten zu generalisieren und auch zu vorher unbekannten Eingabevektoren den korrekten Ausgabevektor zu liefern.

Ein BNN besteht aus einer Vielzahl von *Neuronen*, die miteinander vernetzt sind. Im Allgemeinen hat ein BNN mehrere Schichten von Neuronen: eine Eingabeschicht, ein oder mehrere versteckte Schichten und eine Ausgabeschicht. Üblicherweise sind die Neuronen einer Schicht mit den Neuronen der anliegenden Schichten vollständig vernetzt. Ein Eingabevektor wird direkt an die Neuronen der Eingabeschicht eingegeben, indem diese als Aktivierungspotential die einzelnen Werte des Eingabevektors (ggf. über eine Aktivierungsfunktion modifiziert) übernehmen. Diese Aktivierungspotentiale werden von jedem Neuron der nächsten (versteckten) Schicht individuell gewichtet, aufsummiert und über eine Aktivierungsfunktion zum Aktivierungspotential des verarbeitenden Neurons transformiert. Dieser Vorgang wird Schicht für Schicht für alle Neuronen ausgeführt. Am Ende kann aus den Neuronen der Ausgabeschicht der Ausgabevektor aus den Aktivierungspotentialen ausgelesen werden.

Zum Training des Netzwerkes werden dem Netz sukzessive Trainingsdaten (Eingabevektoren) und die dazu passenden Musterlösungen (Ausgabevektoren) präsentiert und das Ergebnis des Netzwerkes wird mit der vorgegebenen Musterlösung vergleichen. Üblicherweise wird der Fehler des Netzwerkes dadurch verkleinert, dass die Gewichte des Netzwerkes nach dem Gradienten-Abstiegsverfahren derart modifiziert werden, dass der Fehler sich verkleinert. Da die Gewichte dabei, beginnend mit der letzten Schicht – der Ausgabeschicht – modifiziert werden, wird dieses Verfahren als das *Backpropagation*-Verfahren bezeichnet. Eine ausführlichere und mathematische Beschreibung von BNN's findet sich z. B. in PRINCIPE ET AL. [117] und in ZELL [166].

Grundsätzlich haben die BNN's den großen Vorteil, dass sie Funktionen alleine aus Trainingsdaten erlernen können, wodurch sie in vielen Bereichen nutzbringend eingesetzt werden können. Allerdings stehen diesem Vorteil auch einige Probleme gegenüber, die hier inklusive einiger bekannter Lösungsansätze kurz vorgestellt werden:

- Das Training von BNN's ist sowohl in Bezug auf die Anzahl der benötigten Trainingsdaten als auch auf die benötigte Rechenzeit oft sehr aufwendig. Demnach sollen z. B. laut einer Faustregel von BIGUS pro Verbindung mindestens zwei Trainingsdaten zur Verfügung stehen. [19] Aufgrund der vollständigen Vernetzung der verschiedenen Schichten steigt somit der Bedarf an Trainingsdaten mit der Anzahl der Neuronen schnell an. Bezüglich der benötigten Rechnenzeit gibt es Ansätze zum Beschleunigen des Lernens,

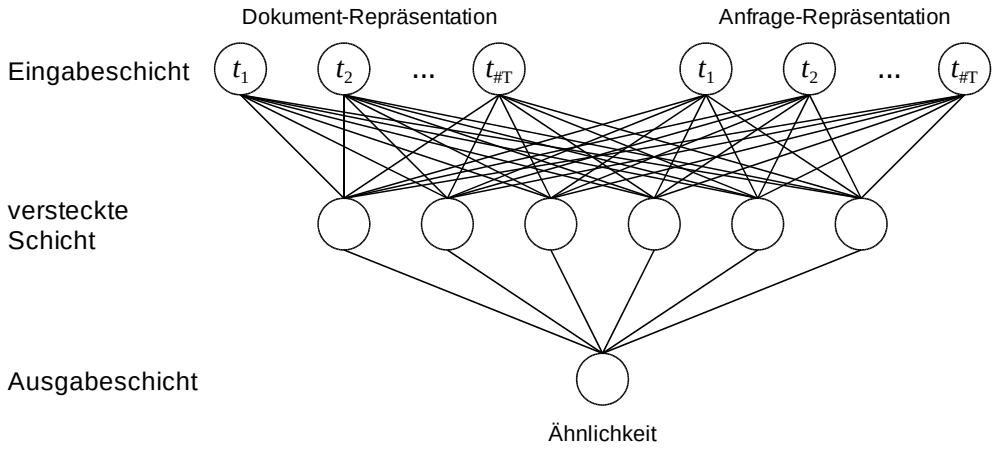


Abbildung 3.6: Topologie eines BNN für IF/IR.

wie z. B. die *Learning Rate Adaption* [33, S. 144] und die *Momentum*-Methode [167, S. 115].

- Es ist nicht garantiert, dass das Backpropagation-Verfahren das globale Fehlerminimum findet. Es ist möglich, dass nur ein lokales Minimum erreicht wird. [166, S. 112f] Eine Möglichkeit diesem Problem zu begegnen, ist es mehrere Netzwerke, die unterschiedlich initialisiert wurden, parallel zu trainieren oder zufallsgesteuerte Elemente in das Lernverfahren einzubringen, um lokalen Minima zu entkommen (vgl. dazu [45, S. 271]).
- Die Anzahl der Neuronen der versteckten Schicht muss experimentell optimiert werden. Sie hängt von der Komplexität der anzunähernden Funktion ab. Es gibt Verfahren, die dieses automatisiert vornehmen können, wie z. B. die *Cascade Correlation* [166, S. 161ff].
- Bei einem zu langen Training besteht die Gefahr des *Overlearning*. In diesem Fall wird der Fehler des Netzes in Bezug auf die Trainingsdaten zwar zunehmend geringer, aber die Generalisierungsfähigkeit des Netzes für unbekannte Daten nimmt ab. Eine gängige Möglichkeit, diesem Problem zu begegnen, ist die Verwendung von Testdaten, die nicht in der Trainingsmenge enthalten sind. Wird der Fehler anhand dieser Testdaten gemessen, dann sollte das Training genau dann abgebrochen werden, wenn der Fehler nach einiger Zeit ansteigt.

Zu den bekanntesten Vertretern von BNN's im Anwendungsbereich des IR gehört das COSIMIR[28]-Modell, dass MANDL in [94] ausführlich vorstellt. Die Topologie des neuronalen Netzwerks von COSIMIR zeigt die Abbildung 3.6. Das Netzwerk besteht aus (mindestens)

[28] Die Abkürzung COSIMIR steht für Cognitive Similarity Learning in Information Retrieval.

drei Schichten. Die Eingabeschicht hat insgesamt $2 \cdot \#T$ Neuronen und repräsentiert gleichzeitig ein einzelnes Dokument und eine Anfrage mit jeweils $\#T$ Neuronen. Jedes Neuron entspricht dabei dem Vorkommen eines Terms in einem Dokument bzw. einer Anfrage. Die Ausgabeschicht besteht lediglich aus einem Neuron, aus dem die Ähnlichkeit zwischen dem Dokument und der Anfrage ausgelesen wird. Alle anderen Neuronen gehören zu der versteckten Schicht.[29] Zum Training des Netzwerkes sind Trainings- und Testdatensätze mit folgender Struktur erforderlich: Jeder Datensatz besteht aus einem Dokument, einer Anfrage sowie dem dazugehörigen Ähnlichkeitswert.

Der große Vorteil des COSIMIR-Modells ist, dass das Modell in Bezug auf seine Ähnlichkeitsfunktion fast vollkommen frei ist und dass diese anhand der Trainingsdaten erlernt werden kann. Somit sollte zumindest aus theoretischer Sicht ein sehr gutes Retrieval-Ergebnis erreicht werden können, weil das Modell für fast alle linguistischen Phänomene offen ist.[30] In der Praxis stellt sich jedoch das Problem, dass die benötigte Zahl an Trainingsdaten und der benötigte Rechenaufwand sehr hoch ist. So zeigen die Experimente von MANDL mit der relativ kleinen *Cranfield*-Kollektion[31] die Probleme des COSIMIR-Modells in Bezug auf eine praktische Anwendung: Die Kollektion umfasst 1400 Dokumente mit 225 Anfragen. Insgesamt können aus den Dokumenten 3763 Terme extrahiert werden, von denen aber lediglich 585 in den Anfragen vorkommen. Zum Testen verwendete MANDL ein neuronales Netzwerk mit 1170 Eingabeneuronen (jeweils 585 für Dokument und Anfrage), 50 Neuronen in der ersten verdeckten Schicht und 10 Neuronen in der zweiten. Es ergaben sich somit 292.510 Verbindungen für die nach das Faustregel von BIGUS ca. 600.000 Trainingsbeispiele von Nöten wären. Somit ist es nicht verwunderlich, dass selbst nach einer mehrwöchigen Trainingszeit, nach dem künstlichem Aufblähen der Trainingsdaten und auch nach dem Durchführen einer künstlichen Dimensionsreduktion, das CORIMIR-Modell keine mit herkömmlichen Verfahren vergleichbar guten Ergebnisse liefern konnte. [94, S. 227ff]

3.5 Bewertung der gängigen Modelle

Im Folgenden werden die bisher vorgestellten Modelle bewertet, allerdings nicht, wie in der IF/IR-Literatur üblich, nach quantitativen Kriterien[32], sondern nach qualitativen Kriterien. Bei den qualitativen Kriterien handelt es sich dabei um linguistische Phänomene und die Fragestellung, in wie weit diese Phänomene in den Modellen berücksichtigt bzw. von den Modellen im Sinne der Modell-Begriffsdefinition aus Abschnitt 2.2 repräsentiert werden.

Die Tabelle 3.2 zeigt das Bewertungsschema und die einzelnen Bewertungen für die verschiedenen Modelle. Neben den Phänomenen der Morphologie sind auch die Phänomene der lexikalischen Semantik Kriterien für die Bewertung der Modelle. Die meisten der in Tabelle 3.2 genannten Begriffe sind bereits in den Abschnitten 2.3.2 und 2.3.4.2 definiert und erklärt

[29] Es sind auch mehrere versteckte Schichten möglich.

[30] Lediglich Wortgruppen können nicht berücksichtigt bzw. erkannt werden, weil jedes Neuron genau ein Term repräsentiert und die Reihenfolge der Terme im Eingabevektor nicht abgebildet wird.

[31] `ftp://ftp.cs.cornell.edu/pub/smart/cran`

[32] Wie z. B. Precision, Recall und Fehlerrate, die in Abschnitt 6.4 noch vorgestellt werden.

IF/IR-Modelle / linguistische Phänomene		ohne Terminterdependenzen							mit immanenten Terminterdep.			mit transzendenten Terminterdep.				
		SBM	VSM	EBM	BIR	INM	BNM	LM	GVSM	LSI	SANN	FSM	RbLI	BNN	**TVSM**	**eTVSM**
Morphologie	Flexion	1	1	1	1	1	1	1	1	1	1	●	●	●	●	●
	Komposition	-	-	-	-	-	-	-	(C)	(C)	(C)	●	●	●	●	●
	Derivation	-	-	-	-	-	-	-	(C)	(C)	(C)	●	●	●	●	●
lexikalische Semantik	Synonymie	2	2,4	2	2,4	2	2	2	(C)	(C)	(C)	●	●	●	●	●
	Homographie	-	-	-	-	-	-	-	-	-	-	-	-	●	-	●
	Metonymie	-	-	-	-	-	-	-	-	-	-	-	-	●	-	●
	Hyponymie	-	4	-	4	-	-	-	(C)	(C)	(C)	●	●	●	●	●
	Meronymie	-	4	-	4	-	-	-	(C)	(C)	(C)	●	●	●	●	●
	Wortgewichte	3	●	●	3	●	●	3 ○	3	3	●	3	●	●	●	●
Wortgruppen		-	○	-	-	-	-	○	-	-	-	-	-	-	-	●
Kommentar													*A*	*B*		

Legende:

- ● = wird repräsentiert
- (C) = wird unter Verwendung von Co-Occurenz-basierten Verfahren repräsentiert
- ○ = es existieren bekannte Erweiterungen zur Repräsentation des Phänomens
- \- = wird nicht repräsentiert
- *A* = Modell ist nicht konsistent
- 1 = nur über externe Stemming-Verfahren
- 2 = nur über externe Synonymersetzung
- 3 = nur über eine externe Stoppwortliste
- 4 = durch Query-Expansion-Verfahren (i.A. Co-Occurenz-basiert)
- *B* = benötigt sehr viele Trainingsdaten

Tabelle 3.2: Bewertung der Modelle in Bezug auf die Abbildung von linguistischen Phänomenen.

worden. Zwei Begriffe sind jedoch noch offen und werden daher im Folgenden erläutert:

- Bei den Wortgewichten wird die Frage untersucht, in wie weit ein Modell in der Lage ist, einzelnen Termen ein Gewicht zuzuweisen. Da die Syntax von Sätzen und Dokumenten nicht berücksichtigt wird, sind Worte ohne Themenbezug (wie z. B. Präpositionen) beim IF und IR hinderlich.[33] Daher werden diese üblicherweise über Stoppwortlisten entfernt. Ein IF/IR-Modell sollte idealerweise in der Lage sein, dieses Phänomen geeignet abzubilden, indem einzelnen Termen individuelle Termgewichte zugewiesen werden.

- Da Wortgruppen (insbesondere bei Eigennamen wie z. B. New York, X Windows[34] oder Windows XP[35]) spezielle Bedeutungen haben können, die sich nicht alleine aus den einzelnen Worten ableiten lassen, sollten Wortgruppen in einem IF/IR-Modell explizit berücksichtigt werden. SONG und CROFT [138] zeigen in ihrer Arbeit, dass das Berücksichtigen von Wortgruppen die Retrieval-Qualität ihres Modells steigert.

Die beiden Modelle TVSM und eTVSM sind hervorgehoben, weil diese erst in den Kapiteln 4 und 5 vorgestellt werden und weil diese durch das Ergebnis, der nun folgend vorgestellten Bewertung der gängigen Modelle, motiviert werden:

1. Wortgruppen werden grundsätzlich von fast allen IF/IR-Modellen nicht berücksichtigt. (Ausnahmen sind spezielle Versionen des VSM, Bi- und Trigram LM-Modelle und das eTVSM.) Wie oben bereits erwähnt, ist die Berücksichtigung von Wortgruppen zur Erfassung von speziellen Bedeutungen einzelner Wortkombinationen (wie z. B. Eigennamen) von Vorteil.

2. Die insbesondere in der Praxis gängigen Modelle ohne Terminterdependenzen bilden linguistische Phänomene kaum ab. So wird von den morphologischen Phänomenen lediglich die Flexion berücksichtigt, wobei dieses außerhalb des eigentlichen Modells durch vorgeschaltete Stemming-Verfahren geschieht. Das heißt, dass diese Modelle das Flexions-Phänomen selbst nicht repräsentieren. Um aber die Flexion in irgendeiner Form zu berücksichtigen, wird das reale Objektsystem (bestehend aus Sprache, Dokumenten, Termen etc.) über das vorgeschaltetes Stemming-Verfahren an das Modell angepasst, anstatt dass das Modell an das reale Objektsystem angepasst wird. Dieses in der Praxis zwar funktionierende Vorgehen muss aus der theoretisch-wissenschaftlichen Perspektive als problematisch bewertet werden, weil ein Modell gemäß der Begriffsdefinition in Abschnitt 2.2 eine Repräsentation eines Objektsystems sein soll und nicht vom Objektsystem losgelöst sein darf.

3. Lexikalische Phänomene werden von Modellen ohne Terminterdependenzen entweder gar nicht oder nur über externe Erweiterungen indirekt abgebildet. Die in Feststellung 2

[33] Vgl. dazu die Diskussion in Abschnitt 2.3.4.2.
[34] Ein Protokoll aus der Unix-Welt zur Realisierung von Client/Server-basierten grafischen Oberflächen.
[35] Ein Betriebssystem der Firma Microsoft.

genannte Kritik an den Modellen ohne Terminterdependenzen gilt auch für das lexikalische Phänomen der Synonymie, wenn dieses über die Synonymersetzung realisiert wird. Bei Anwendung von Query-Expansion-Verfahren werden die Probleme der Synonymie, Hyponymie und Meronymie implizit vom Anwender des Systems gelöst, indem dieser in mehreren Durchläufen die bisher relevantesten Dokumente selektiert. Dadurch ist ein IR-System in der Lage, durch Co-Occurrenz von Termen auf Synonymien bzw. sonstige Termbeziehungen zu schließen und diese durch Aufnahme von neuen Termen in die Anfrage (VSM) bzw. durch Rekalibierierung der Termwahrscheinlichkeiten (BIR) in eine neue Anfrage umzusetzen.

4. Modelle mit immanenten Terminterdependenzen berücksichtigen deutlich mehr linguistische Phänomene (insbesondere Komposition, Derivation, Synonymie, Hyponymie und Meronymie), allerdings verwenden diese Modelle Co-Occurrenz-basierte Verfahren zur Erkennung dieser Phänomene. Diese Verfahren sind aber, wie in Abschnitt 3.3 gezeigt, nur wenig zur Erkennung der genannten linguistischen Phänomene geeignet. Daher müssen auch die Modelle mit immanenten Terminterdependenzen aus der theoretisch-wissenschaftlichen Perspektive als problematisch bezeichnet werden.

5. Aus linguistischer Sicht sind die Modelle mit transzendenten Terminterdependenzen am besten geeignet, sofern qualitativ hochwertige Terminterdependenzen vorgegeben werden und diese nicht lediglich unter Verwendung von Co-Occurrenz-basierten Verfahren hergeleitet werden. In Tabelle 3.2 fällt auf, dass bei den drei Modellen FSM, RbLI und BNN neben der Synonymie auch die Flexion, Komposition, Derivation, Hyponymie und Meronymie als repräsentiert gekennzeichnet sind, obwohl diese nicht in den Modellen explizit erwähnt werden. Der Grund dafür ist, dass sich diese Phänomene über Termähnlichkeiten abbilden lassen.[36] Man kann feststellen, dass von den bisher vorgestellten Modellen mit transzendenten Terminterdependenzen (FSM, RbLI und BNN) Wortgruppen nicht berücksichtigt werden. Bezüglich des BNN ist anzumerken, dass seine Anwendbarkeit in der Praxis trotz seiner theoretisch guten Abdeckung linguistischer Phänomene aufgrund der vielen notwendigen Trainingsdaten, die für das Erlernen von Wortbeziehungen notwendig sind, problematisch ist.[37] Beim RbLI ist aus theoretisch-wissenschaftlicher Sicht die in Abschnitt 3.4.2 genannte Abschätzung der Termwahrscheinlichkeiten über die inverse Dokumentenhäufigkeit als problematisch zu nennen, weil diese das Modell in Bezug auf die probabilistische Grundannahme des Modells inkonsistent gestaltet.

Unter Einbeziehung linguistischer Phänomene, der verschiedenen Modelldefinitionen und der Tabelle 3.2 kann für dieses Kapitel das folgende Fazit gezogen werden:

1. Zur Abbildung einer möglichst großen Anzahl von linguistischen Phänomenen sind Modelle mit transzendenten Terminterdependenzen die vielversprechendsten Kandidaten.

[36] Wie Synonyme, Flexion, Komposition, Derivation, Hyponymie und Meronymie unter Verwendung von Termähnlichkeiten repräsentiert werden können, wird in den Kapiteln 4 und 5 besprochen.

[37] Vgl. Abschnitt 3.4.3.

2. Die Anzahl der Terminterdependenzen zwischen jeweils zwei Termen wächst quadratisch mit der Anzahl der Terme und ist somit relativ hoch. Es ist daher von Vorteil, wenn die Terminterdependenzen in irgendeiner Form automatisch abgeleitet werden können. Da aber, wie in Abschnitt 3.3 gezeigt, die Co-Occurrenz-basierten Verfahren auf Dokumentenbeständen nur wenig zur Herleitung von Terminterdependenzen geeignet sind, sind neue Quellen für die Herleitung der Terminterdependenzen zu erschließen. Mögliche Quellen dieser Art wären Ontologien, wie z. B. WordNet, GermaNet oder das Wortschatz-Lexikon.[38]

3. Worte können in verschiedenen Kontexten unterschiedliche Bedeutungen haben (Homographie und Metonymie). Dieses Phänomen wird von den vorgestellten Modellen (bis auf das BNN) nicht berücksichtigt, obwohl es häufig Missverständnisse zwischen Anwendern und einem IF/IR-System verursacht und damit die Ergebnisqualität mindert. Daher sollte Forschung betrieben werden, wie dieses Phänomen in IF/IR-Modellen repräsentiert werden kann.

4. Von den gängigen IF- und IR-Modellen werden Wortgruppen selten (siehe Erweiterungen von VSM und LM) repräsentiert. Wortgruppen können aber eine eigene Bedeutung haben, die sich nicht aus den einzelnen Wörtern ableiten lässt. Daher sollten zukünftige Modelle Wortgruppen repräsentieren können.

Wie man erkennen kann, besteht somit durchaus der Bedarf – trotz der Vielzahl an bereits bestehenden IF- bzw. IR-Modellen – neue Modelle zu entwickeln bzw. bestehende Modelle derart zu erweitern, dass die linguistischen Phänomene besser abgebildet werden. Daher beschäftigt sich der Kern dieser Arbeit (die Kapitel 4 und 5) mit der Entwicklung neuer Modelle für das IF und IR zur Repräsentation von natürlichsprachlichen Dokumenten.

[38] WordNet, GermaNet und das Wortschatz-Lexikon werden in Abschnitt 6.1.2 vorgestellt.

Kapitel 4

Topic-based Vector Space Model (TVSM)

4.1 Motivation

Die Motivation zur Entwicklung des TVSM ist die bisher noch immer unzureichend gelöste Problemstellung des IF und IR. Das Ziel der Entwicklung des TVSM ist die Schaffung einer theoretischen Plattform (oder genauer eines Modells), die möglichst viele linguistische Phänomene (konkret sind es zunächst Flexion, Komposition, Derivation, Synonymie, Hyponymie, Meronymie und Wortgewichte) unter Verwendung von transzendent[1] modellierten Terminterdependenzen abbilden kann. Zu den wichtigsten Nebenzielen gehört dabei, dass die Terminterdependenzen über eine sauber spezifizierte Schnittstelle[2] von außen bezogen werden können und dass sich die Komplexität und der Rechenaufwand des Modells in einem überschaubaren Rahmen[3] bewegt, um das Modell beherrschbar zu gestalten. Daraus folgt, dass die Berücksichtigung der Syntax wegen der Probleme der Abdeckung, Effizienz und Ambiguität nicht kompatibel zu den genannten Anforderungen ist.[4] Somit wird ebenfalls die Betrachtung von Satz- und Diskurssemantik obsolet.[5]

[1] Ein Ergebnis von Abschnitt 3.5 ist, dass Modelle mit transzendenten Terminterdependenzen am besten für die Modellierung von linguistischen Phänomenen im Rahmen von IF/IR-Modellen geeignet sind.

[2] Ein weiteres Ergebnis von Abschnitt 3.5 ist, dass neue Quellen für die Herleitung von Terminterdependenzen zu erschließen sind, weil die gängigen Co-Occurrenz-basierten Verfahren nicht in der Lage sind linguistische Phänomene hinreichend zu erfassen. Es ist daher sinnvoll, eine wohlstrukturierte und -definierte Schnittstelle für Terminterdependenzen zur Verfügung zu stellen, so dass Quellen für Terminterdependenzen möglichst leicht mit dem Modell verknüpft werden können.

[3] Soll heißen, dass sich der Rechenaufwand für den Vergleich zweier Dokumente auf Ähnlichkeit im polynomiellen Bereich mit geringem Grad bewegen sollte, weil ansonsten die Anwendbarkeit des Modells in der Praxis für längere Dokumente und eine größere Anzahl von Dokumenten in Frage gestellt ist.

[4] Vgl. dazu die Diskussion in Abschnitt 2.3.3.2.

[5] Vgl. dazu Abschnitt 2.3.4.1.

4.2 Konzept

Das hier vorgestellte TVSM lehnt sich an die Arbeit von BECKER und KUROPKA [12] an, die das Modell im Jahre 2003 zum ersten Mal vorgestellt haben. Bei dem TVSM handelt es sich um ein Vektor-basiertes Modell, dass eine Erweiterung und Verallgemeinerung des VSM und des GVSM ist. Gründe, die für die Anwendung von Vektor-basierten Modellen sprechen, sind folgende:

- Vektor-basierte Modelle sind anschaulich (weil grafisch gut darstellbar) und daher auch für Laien leicht nachzuvollziehen. Dieses ist im Allgemeinen für den interessierten Anwender von IF- bzw. IR-Systemen von Vorteil, auch wenn bei der Anwendung des Systems das zu Grunde liegende Vektor-basierte Konzept nicht sichtbar ist.
- Anfragen werden bei Vektor-basierten Modellen als virtuelle Dokumente aufgefasst. Dieses wirkt komplexitätsreduzierend, weil somit Anfragen und Dokumente identisch behandelt werden. Zudem lassen sich Vektor-basierte Modelle somit auch problemlos mit k-nearest neighbour Verfahren einsetzen.

Dem TVSM liegen zwei Ideen zu Grunde: Erstens gibt es Terme (Worte) die gut geeignet sind, den thematischen Bezug eines Dokuments zu erschließen, und es gibt andere Terme (z. B. Stoppwörter), die weniger gut geeignet sind für diese Aufgabe. Daher werden beim TVSM alle Terme mit einem Gewicht versehen, welches die Eignung eines Terms für die genannte Aufgabe widerspiegelt. Die zweite Idee des TVSM ist die, dass verschiedene linguistische Phänomene durch Termähnlichkeiten von Termen in Bezug auf das ihnen zu Grunde liegende Thema abgebildet werden können. Konkret handelt es sich um folgende Phänomene der Morphologie und der lexikalischen Semantik:

- *Flexion*: In Bezug auf das Thema wird für verschiedene Flexionsformen eines Wortes (z. B. Haus und Häuser) angenommen, dass die Termähnlichkeiten zwischen den verschiedenen Flexionsformen maximal sind (also Identität bedeuten).
- *Komposition*: Für Komposita (wie z. B. Telekomaktie wird angenommen, dass diese ein gewisses Maß an Ähnlichkeit zu den Einzelworten des Kompositums aufweisen, da Komposita im Allgemeinen thematisch mit den Einzelworten verwandt sind.
- *Derivation*: Auch den Derivaten (wie z. B. Zwerglein) liegt die Annahme zu Grunde, dass sich die Beziehung zwischen dem Derivat und dem ursprünglichen Wort durch eine hohe Ähnlichkeit abbilden lässt.
- *Synonymie*: Für Synonyme (wie z. B. Rechner und Computer) wird angenommen, dass diese eine maximale Ähnlichkeit (Identität) zueinander aufweisen.
- *Hyponymie und Metonymie*: Das TVSM nimmt an, dass sich generell alle Wortbeziehungen vom Typ ist-ein, besteht-aus, etc. durch Termähnlichkeiten ausdrücken lassen. So ist es intuitiv nachvollziehbar, dass z. B. der Term BMW eine hohe Ähnlichkeit zum Term Auto haben muss.

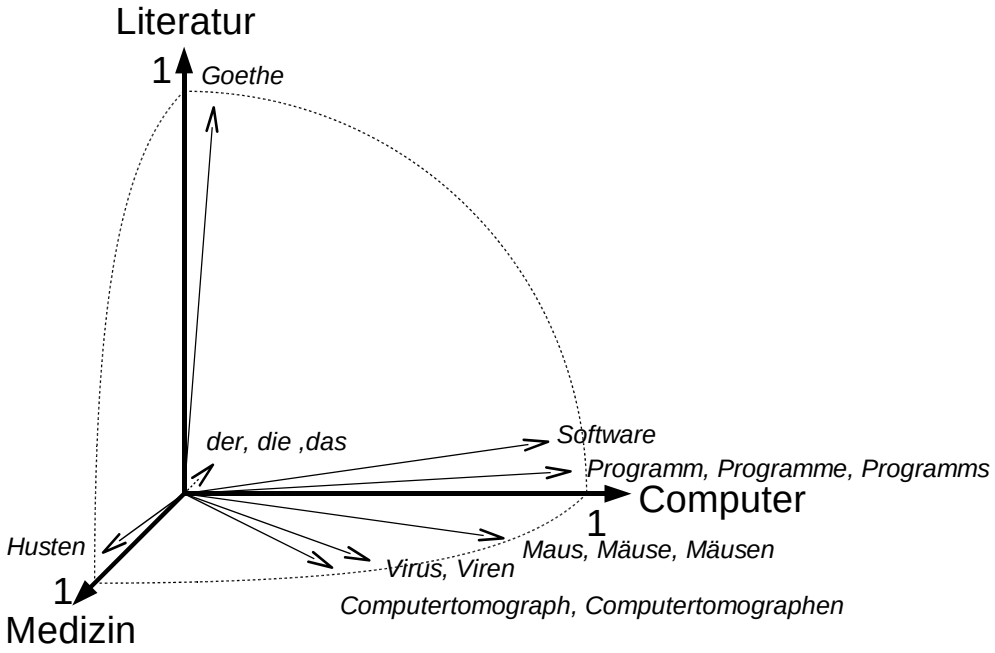


Abbildung 4.1: Veranschaulichung der Interpretation des TVSM-Vektorraums.

Im weiteren Verlauf dieses Abschnitts werden in Abschnitt 4.2.1 die grundlegenden Annahmen des Modells und in Abschnitt 4.2.2 die Definition der Dokumentenähnlichkeit vermittelt, deren Algorithmus zur konkreten Berechnung in Abschnitt 4.2.3 hergeleitet wird. Zum Abschluss wird eine mögliche Implementierung des Modells unter Verwendung einer relationalen Datenbank vorgestellt (Abschnitt 4.2.4).

Neu gegenüber der Arbeit von BECKER und KUROPKA [12] ist, dass hier unter Verwendung des TVSM die Anwendung von Stoppwortlisten und die Anwendung von Stemming-Verfahren durch das Stoppwort- (Abschnitt 4.3) bzw. das Stemming-Lemma (Abschnitt 4.4) theoretisch fundiert werden. Im weiteren Verlauf der Arbeit werden diese Kenntnisse und die im Abschnitt 4.7 geführte Kritik am TVSM aufgenommen und in Form des eTVSM in Kapitel 5 in einem neuen Modell umgesetzt.

4.2.1 Vektorraum, Terme und Dokumente

Die fundamentale Annahme des TVSM ist die Existenz eines d-dimensionalen Vektorraums R mit $d \in \mathbb{N}$, der in jeder Dimension nur positive Achsenabschnitte[6] aufweist. Jede Dimen-

[6] Die Begründung dafür, warum der Raum R nur positive Achsenabschnitte hat, wird in Abschnitt 4.2.2 gegeben.

sion bzw. jeder Achsenabschnitt dieses Raumes repräsentiert ein elementares Themengebiet (englisch: „topic“).

$$R = \mathbb{R}^d_{\geq 0} \quad \text{mit} \quad d \in \mathbb{N} \tag{4.1}$$

Diese Interpretation impliziert die Annahme, dass die elementaren Themengebiete zueinander orthogonal (also voneinander thematisch unabhängig) sind. Die Abbildung 4.1 zeigt die Interpretation des Vektorraums anhand eines grafischen Beispiels.

Jeder Term i aus der Menge aller möglichen Terme T wird im Vektorraum R durch einen Termvektor $\vec{t_i}$ repräsentiert, wobei die Länge (der Betrag) des Termvektors auf einen maximalen Wert von eins beschränkt ist:

$$\forall i \in T: \quad \vec{t_i} \in R \quad \wedge \quad |\vec{t_i}| = [0...1] \tag{4.2}$$

Ein Term i wird somit über den Termvektor $\vec{t_i}$ einem oder mehreren elementaren Themengebieten zugeordnet. So ist beispielsweise der Term Goethe in Abbildung 4.1 nahezu[7] ausschließlich dem elementaren Themengebiet Literatur zugeordnet. Im Unterschied dazu sind die beiden Terme Virus und Viren den beiden Themengebieten Medizin und Computer zugeordnet. Die drei Stoppwörter der, die und das haben einen Betrag (eine Länge) von Null[8], weil sie keinen Themenbezug haben.

Als Maß für Ähnlichkeit zwischen zwei Termen i und j (Termähnlichkeit) ist beim TVSM der Kosinus des Winkels $\omega_{i,j}$ zwischen den beiden Termen definiert. Da der Vektorraum R auf positive Achsenabschnitte eingeschränkt ist, sind die Winkel bzw. die Termähnlichkeiten wie folgt beschränkt:

$$\begin{aligned} & \omega_{i,j} \in [0°...90°] && \forall i,j \in T \\ \Rightarrow \quad & \cos\omega_{i,j} \in [0...1] && \text{(Termähnlichkeit)} \end{aligned}$$

4.2.2 Dokumente und Dokumentenähnlichkeiten

Ein Dokument k aus der Menge aller Dokumente D wird im TVSM durch einen Dokumentenvektor $\vec{d_k} \in R$ repräsentiert:

$$\forall k \in D: \quad \vec{d_k} = \frac{1}{|\vec{\delta_k}|}\vec{\delta_k} \quad \Rightarrow \quad |\vec{d_k}| = 1 \tag{4.3}$$

mit

$$\vec{\delta_k} = \sum_{i \in T} a_{k,i}\vec{t_i} \quad \text{wobei} \quad a_{k,i} = \text{Vorkommensanzahl des Terms } i \text{ in Dokument } k \tag{4.4}$$

[7] Aus Gründen der besseren Lesbarkeit sind die Terme in Abbildung 4.1 den elementaren Themengebieten derart zugeordnet, dass Überdeckungen zwischen Termen mit unterschiedlichen Wortstämmen und zwischen Termen und Achsenabschnitten nicht vorkommen.

[8] Aus Gründen der besseren Erkennbarkeit sind die Vektoren der Stoppwörter in Abbildung 4.1 mit einer Länge größer als Null eingezeichnet.

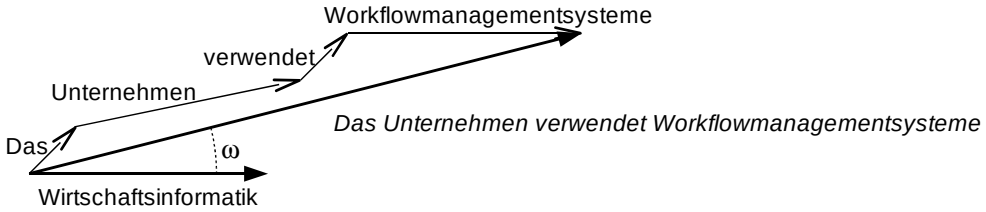


Abbildung 4.2: Dokumentenähnlichkeit im TVSM grafisch veranschaulicht.

Jeder Dokumentenvektor definiert sich als die gewichtete Summe über alle Terme, wobei die Gewichtung der jeweiligen Vorkommensanzahl der Terme im Dokument entspricht. Somit werden nicht vorkommende Terme (wegen ihrer Gewichtung mit dem Wert Null) bei der Aufsummierung ignoriert und mehrfach vorkommende Terme werden entsprechend ihrer Häufigkeit stärker berücksichtigt. Damit die Zahl der Terme in einem Dokument (und damit die Dokumentenlänge) keinen Einfluss auf die spätere Berechnung von Dokumentenähnlichkeiten hat, ist der Dokumentenvektor in seiner Länge auf den Betrag von Eins normiert. Diese Definition des Dokumentenvektors bedingt, dass jedes Dokument mindestens einen Term mit einem Betrag größer Null enthält:[9]

$$\forall k \in D: \quad \exists (a_{k,i} > 0 \ \wedge \ |t_i| > 0) \text{ mit } i \in T \tag{4.5}$$

Das Maß der Ähnlichkeit $\text{sim}(k,l)$ zwischen zwei Dokumenten $k, l \in D$ wird als das Skalarprodukt der beiden Dokumentenvektoren definiert, welches durch die in Definition 4.3 vorgenommene Normierung der Dokumentenvektoren bedingt, dem Kosinus des Winkels $\omega_{k,l}$ zwischen den beiden Dokumentenvektoren $\vec{d_k}$ und $\vec{d_l}$ entspricht:

$$\begin{aligned} \text{sim}(k,l) &= \vec{d_k}\vec{d_l} \\ &= |\vec{d_k}||\vec{d_l}| \cos \omega_{k,l} \\ &= \cos \omega_{k,l} \end{aligned} \tag{4.6}$$

Das Skalarprodukt $\vec{d_k}\vec{d_l}$ und somit die Dokumentenähnlichkeit $\text{sim}(k,l)$ weisen folgende Eigenschaften auf:

$\text{sim}(k,l) \leq 1$	wegen $\cos \omega_{k,l} \leq 1$
$\text{sim}(k,l) \geq 0$	wegen (4.1) $\Rightarrow \omega_{k,l} \in [0°; 90°]$
$\text{sim}(k,l) = 1$	$\forall k = l$
$\text{sim}(k,l) = \text{sim}(l,k)$	$\forall k, l \in D$

[9] Diese Bedingung wird üblicherweise von allen „normalen“ Dokumenten erfüllt. Der thematische Inhalt eines Dokuments, dass keine Terme oder lediglich nur Stoppwörter enthält dürfte so gering sein, dass man auf das Dokument problemlos verzichten kann.

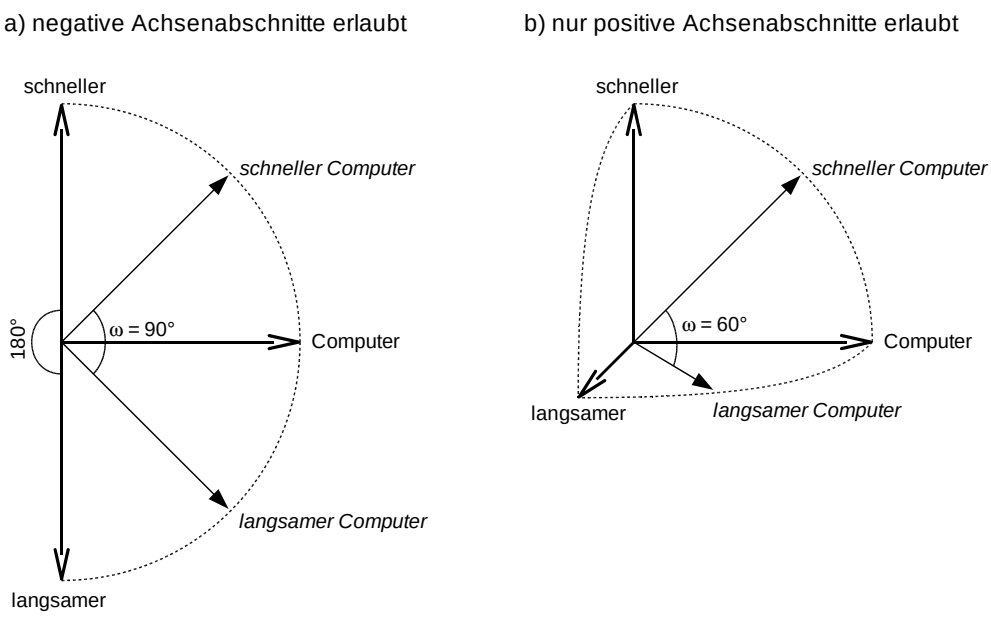


Abbildung 4.3: Begründung für positive Achsenabschnitte.

Aus dieser Definition folgt, dass Dokumente, deren Dokumentenvektoren in eine ähnliche Richtung weisen (also einen geringen Winkel $\omega_{k,l}$ zueinander haben) eine hohe Ähnlichkeit ($\text{sim}(k, l)$ nahe Eins) zueinander haben und umgekehrt. Im Ergebnis heißt das beispielsweise, dass ein Dokument zum Thema Wirtschaftsinformatik mit mehreren Wirtschaftsinformatik-nahen Begriffen (vgl. Abbildung 4.2), jedoch *ohne* dem Begriff Wirtschaftsinformatik, eine hohe Ähnlichkeit zu einem Dokument bzw. einer Anfrage (welche als virtuelles Dokument aufzufassen ist) aufweisen kann, das bzw. die *nur* den Begriff Wirtschaftsinformatik (und keine weiteren) enthält, sofern die Termvektoren der Wirtschaftsinformatik-nahen Begriffe in eine ähnliche Richtung weisen wie der Begriff Wirtschaftsinformatik.[10]

Der Grund dafür, dass der Raum R nur positive Achsenabschnitte hat, ist der, dass negative Achsenabschnitte zu Themengebieten oder Termen so etwas wie Anti-Themengebiete bzw. Anti-Terme ermöglicht bzw. postulieren. Auf den ersten Blick erscheint das Konzept von Termen und Anti-Termen (Antonymie) schlüssig, weil es für viele Begriffe einen gegenteiligen Begriff gibt: schnell ↔ langsam oder Freund ↔ Feind. Das Problem an den Anti-Termen bzw. Anti-Themengebieten ist, dass die Bindung dieser Terme/Themen an ein Objekt von der Perspektive des Erzählers abhängig ist. Ein Rechner, der 1990 als schnell attributiert

[10] Aufgrund der besseren Lesbarkeit sind in Abbildung 4.2 die Termvektoren der beiden Stoppwörter Das und verwendet überproportional lang dargestellt. Des Weiteren sind die Dokumentenvektoren nicht, wie es von Definition 4.3 gefordert wird, normiert dargestellt, weil der Winkel ω auch ohne Normierung erklärt werden kann und eine grafische Darstellung ohne Normierung besser lesbar ist.

wurde, wird mit hoher Wahrscheinlichkeit im Jahre 2000 als langsam attributiert. Auch für Freund und Feind gibt es eine Abhängigkeit von der Perspektive, die durch den folgenden Spruch ausgedrückt wird: „Der Freund meines Feindes ist mein Feind." Dieses hat zur Folge, dass der Vergleich von Dokumenten erschwert wird, wenn Anti-Terme zugelassen werden, weil in einem größeren Dokumentenbestand die Wahrscheinlichkeit hoch ist, dass die Autoren unterschiedliche Perspektiven auf denselben Sachverhalt haben. Abbildung 4.3a) zeigt die Problematik, die sich ergeben würde, wenn negative Achsenabschnitte erlaubt wären. In diesem Fall müssten die Terme schneller und langsamer zueinander eine Anti-Term-Beziehung eingehen und somit einen Winkel von 180° haben. Daraus würde aber folgen, dass die beiden Phrasen schneller Computer und langsamer Computer zueinander orthogonal (90°) wären, was bedeuten würde, dass diese thematisch nichts miteinander gemein haben. Dieses kann aber definitiv verneint werden. Abbildung 4.3b) zeigt dieselbe Situation mit lediglich positiven Achsenabschnitten. Hier sind schneller und langsamer zueinander orthogonal. Daraus ergibt sich, dass die beiden vorhin genannten Phrasen einen Winkel von 60° haben und somit einen gewissen thematischen Bezug zueinander haben, was mit dem intuitiven Empfinden eher übereinstimmt.

4.2.3 Berechnung der Dokumentenähnlichkeiten

Zur Berechnung der paarweisen Ähnlichkeiten von Dokumenten werden folgende Annahmen bezüglich der Variablen D, T, $a_{k,i}$, $|\vec{t_i}|$ und $\omega_{i,j}$ getroffen:

- Die Menge aller Terme T ist gegeben.
- Die Menge der Dokumente D, sowie die Anzahl $a_{k,i}$ der Terme $i \in T$ in den Dokumenten $k \in D$ ist für alle Kombinationen aus i und k gegeben.
- Der Betrag aller Termvektoren $|\vec{t_i}|$ ist bekannt. Der Betrag von Stoppwort-Termvektoren hat einen Wert gleich Null. Der Betrag von allen anderen Termen mit einer hohen Aussagekraft bezüglich ihrer Themenzugehörigkeit ist hingegen nahe (gleich) Eins.
- Der Winkel $\omega_{i,j}$ zwischen allen möglichen Kombinationen von Termvektoren $\vec{t_i}$ und $\vec{t_j}$ mit $i, j \in T$ ist bekannt. Wie in Abbildung 4.1 illustriert und in Abschnitt 4.2 postuliert, haben thematisch identische Terme (verschiedene Flexionsformen eines Terms und synonyme Terme) einen Winkel von 0° zueinander. Verwandte Terme (Komposita und ihre Einzelworte, Derivate und ihre ursprünglichen Worte sowie Worte, die zueinander meronyme oder hyponyme Wortbeziehungen haben) haben einen geringen Winkel und thematisch verschiedene Terme (die in keiner Beziehung zueinander stehen) haben einen großen Winkel (in der Nähe von oder genau gleich 90°).

Basierend auf diesen Annahmen kann für alle Termkombinationen das Termskalarprodukt aus jeweils zwei Termen ($\vec{t_i}\vec{t_j}$) wie folgt berechnet werden:

$$\vec{t_i}\vec{t_j} = |\vec{t_i}||\vec{t_j}| \cos \omega_{i,j} \tag{4.7}$$

Für die Implementierung des TVSM folgt daraus, dass die Winkel zwischen allen Termvektoren, bei denen einer der beiden Termvektoren einen Betrag von Null aufweist (z. B. Stoppwörter), vernachlässigt werden kann. Dieses begründet sich darin, dass die im Folgenden vorgestellte Berechnung von Dokumentenähnlichkeiten vollständig auf den Termskalarprodukten und den in den Dokumenten vorkommenden Termen basiert und die Termskalarprodukte gemäß Gleichung 4.7 u. a. immer dann gleich Null sind, wenn einer der in das Skalarprodukt eingehenden Terme einen Betrag von Null aufweist.

Bevor die Ähnlichkeit $\text{sim}(k,l)$ zwischen zwei Dokumenten $k,l \in D$ berechnet werden kann, muss der Betrag der beiden unnormierten Dokumentvektoren $\vec{\delta}_k$ bzw. $\vec{\delta}_l$ (vgl. auch Definition 4.4) einmalig berechnet werden:[11]

$$\begin{aligned}
|\vec{\delta}_k| &= |\sum_{i \in T} a_{k,i}\vec{t}_i| \\
&= \sqrt{|\sum_{i \in T} a_{k,i}\vec{t}_i|^2} \\
&= \sqrt{(\sum_{i \in T} a_{k,i}\vec{t}_i)^2} \\
&= \sqrt{\sum_{i \in T}\sum_{j \in T} a_{k,i}a_{k,j}\vec{t}_i\vec{t}_j}
\end{aligned} \tag{4.8}$$

Abschließend kann mit dem Gegebenen die Dokumentenähnlichkeit $\text{sim}(k,l)$ zwischen zwei beliebigen Dokumenten k und l aus dem Dokumentenbestand D wie folgt bestimmt werden:

$$\begin{aligned}
\text{sim}(k,l) &= \vec{d}_k\vec{d}_l \\
&= \frac{1}{|\vec{\delta}_k|}\vec{\delta}_k\frac{1}{|\vec{\delta}_l|}\vec{\delta}_l \\
&= \frac{1}{|\vec{\delta}_k||\vec{\delta}_l|}\vec{\delta}_k\vec{\delta}_l && \text{Assoz.Ges.} \\
&= \frac{1}{|\vec{\delta}_k||\vec{\delta}_l|}\sum_{i \in T} a_{k,i}\vec{t}_i \sum_{j \in T} a_{l,j}\vec{t}_j \\
&= \frac{1}{|\vec{\delta}_k||\vec{\delta}_l|}\sum_{i \in T}\sum_{j \in T} a_{k,i}a_{l,j}\vec{t}_i\vec{t}_j && \text{Distrib.Ges.}
\end{aligned} \tag{4.9}$$

4.2.4 Implementierung mit einer relationalen Datenbank

Das TVSM kann mit relativ geringem Aufwand unter Verwendung einer relationalen Datenbank implementiert werden. Abbildung 4.4 zeigt das logische Datenmodell für das TVSM in

[11] Die Berechnung von $|\vec{\delta}_l|$ verläuft analog zu der Berechnung von $|\vec{\delta}_k|$.

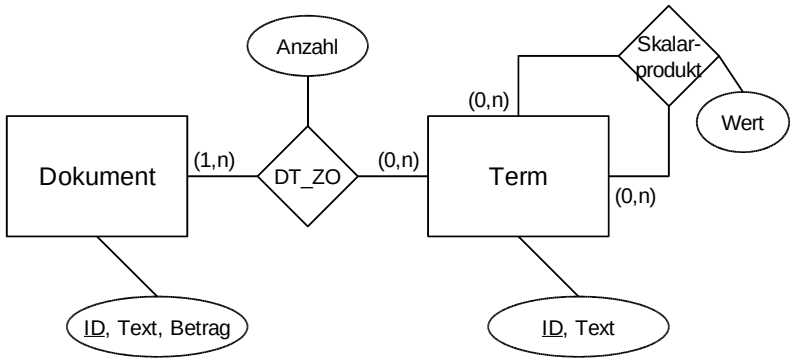


Abbildung 4.4: Relationales Datenmodell für das TVSM.

Anlehnung an die ERM-Notation nach CHEN (vgl. Abschnitt 2.2.1). Die Dokumentenmenge D wird bei der Umsetzung im ERM durch den Entitytyp Dokument und die Termmenge T durch Term repräsentiert. Die Vorkommenshäufigkeit $a_{k,i}$ (vgl. Definition 4.4) eines bestimmten Terms in einem Dokument ist beim Einfügen eines jeden Dokuments, für jeden Term einmalig, im Attribut Anzahl des Relationshiptyps DT_ZO zu speichern. Aus Gründen der höheren Performanz und aufgrund des geringeren Speicherplatzbedarfs sollte dieser Relationshiptyp jedoch nur diejenigen Dokument-Term-Kombinationen beinhalten, bei denen ein Term mindestens einmal im Dokument vorkommt (also bei denen das Attribut Anzahl bzw. $a_{k,i}$ größer als Null ist).

Zur Repräsentation der Termvektorlängen und der Winkel zwischen jeweils zwei verschiedenen Termvektoren ist es ausreichend, lediglich die Termskalarprodukte zu speichern, weil die Berechnung der Dokumentenähnlichkeiten ausschließlich auf den Termskalarprodukten $\vec{t}_i\vec{t}_j$ und den in den Dokumenten vorkommenden Termen basiert.[12] Die Termskalarprodukte werden in Abbildung 4.4 durch den Relationshiptyp Skalarprodukt repräsentiert. Auch hier sollten aus Gründen der höheren Performanz und des geringeren Speicherplatzbedarfs lediglich diejenigen Termskalarprodukte gespeichert werden, die größer als Null sind.

Erwähnenswert ist, dass beim Einfügen eines neuen Dokuments in die Datenbank der Betrag des dazugehörigen Dokumentenvektors aus Performanz-Gründen im Attribut Betrag des Entitytypen Dokument zu hinterlegen ist. Die Berechnung des Wertes erfolgt gemäß der Gleichung 4.8. Der einmal berechnete Betrag eines Dokuments braucht solange nicht erneut berechnet werden, solange nicht ein Termskalarprodukt eines im Dokument enthaltenen Terms zu einem beliebigen anderen im Dokument enthaltenen Term geändert wird. Da die Termskalarprodukte im Zeitverlauf eines sich im Einsatz befindlichen Systems als relativ stabil anzusehen sind, ist eine nachträgliche Anpassung der Beträge der Dokumentenvektoren als eher selten anzunehmen.

[12] Vgl. Abschnitt 4.2.3.

Der folgende, in der Datenbankanfrage-, -manipulations- und -definitionssprache SQL[13] formulierte Quelltextauszug erzeugt die für eine Implementierung des TVSM in einer relationalen Datenbank notwendigen Tabellen:

```
CREATE TABLE Dokument (
       ID        INTEGER,
       Text      TEXT NOT NULL,
       Betrag    DOUBLE PRECISION,
       PRIMARY KEY(ID));

CREATE TABLE Term (
       ID        INTEGER,
       Text      TEXT UNIQUE NOT NULL,
       PRIMARY KEY(ID));

CREATE TABLE DT_ZO (
       DokID     INTEGER NOT NULL REFERENCES Dokument(ID),
       TermID    INTEGER NOT NULL REFERENCES Term(ID),
       Anzahl    INTEGER NOT NULL,
       PRIMARY KEY(DokID, TermID));

CREATE TABLE Skalarprodukt (
       Term1     INTEGER NOT NULL REFERENCES Term(ID),
       Term2     INTEGER NOT NULL REFERENCES term(ID),
       Wert      DOUBLE PRECISION NOT NULL,
       PRIMARY KEY(Term1, Term2));
```

Die folgende View definiert eine Datenbankabfrage zur Bestimmung des Dokumentenbetrags gemäß Gleichung 4.8. Das Ergebnis der View wird zum Zeitpunkt der Anfrage berechnet und kann aus der virtuellen Tabelle `Dok_Betrag` ausgelesen werden. Nach dem Einfügen eines neuen Dokuments in die Datenbank muss das Attribut `Betrag` mit dem passenden Wert aus der View `Dok_Betrag` initialisiert werden.[14]

```
CREATE VIEW Dok_Betrag AS
       SELECT dtzo1.DokID,
              SQRT(SUM(dtzo1.Anzahl * dtzo2.Anzahl * s.Wert))
                AS Betrag
       FROM   DT_ZO dtzo1, DT_ZO dtzo2, Skalarprodukt s
       WHERE  dtzo1.DokID  = dtzo2.DokID
          AND dtzo1.TermID = s.Term1
          AND dtzo2.TermID = s.Term2
       GROUP BY dtzo1.DokID;
```

[13] Vgl. Abschnitt 2.2.2.

[14] Eine mögliche Alternative zum Kopieren des Dokumentenbetrags in die Tabelle `Dokument` ist die Verwendung von materialisierten Views. Diese Views werden (sofern im Zeitverlauf keine Änderungen an den Quelldaten vorgenommen werden) nur einmalig berechnet und das Ergebnis wird datenbankintern gespeichert. Materialisierte Views werden von den meisten professionellen Datenbanken wie z. B. *Oracle9i* (vgl. [108]) unterstützt.

Die Berechnung der Ähnlichkeiten zwischen Dokumenten nach Gleichung 4.9 ermöglichen die beiden folgenden Views. Die erste View berechnet dabei die Doppelsumme aus der Gleichung und somit die unnormierte Dokumentenähnlichkeit. Die zweite nimmt das Ergebnis der ersten auf und normiert es mit den passenden Beträgen der jeweiligen Dokumentenvektoren.[15]

```
CREATE VIEW Unnorm_Dok_Aehn AS
       SELECT dtzo1.DokID AS DokID1,
              dtzo2.DokID AS DokID2,
              SUM(dtzo1.Anzahl * dtzo2.Anzahl * s.Wert)
                AS Unnorm_Aehn
       FROM   DT_ZO dtzo1, DT_ZO dtzo2, Skalarprodukt s
       WHERE  s.Term1 = dtzo1.TermID
          AND s.Term2 = dtzo2.TermID
       GROUP BY dtzo1.DokID, dtzo2.DokID;

CREATE VIEW Dok_Aehn AS
       SELECT uda.DokID1,
              uda.DokID2,
              uda.Unnorm_Aehn / (d1.Betrag * d2.Betrag)
                AS Aehn
       FROM   Unnorm_Dok_Aehn uda, Dokument d1, Dokument d2
       WHERE  uda.DokID1 = d1.ID
          AND uda.DokID2 = d2.ID;
```

Unter Verwendung der oben vorgestellten Views lassen sich zwei Dokumente relativ einfach auf ihre Ähnlichkeit hin vergleichen. Das folgende Beispiel gibt neben den Dokumentennummern die Ähnlichkeit zwischen den beiden (fiktiven) Dokumenten 3 und 7 aus:

```
SELECT *
FROM   Dok_Aehn
WHERE  DokID1 = 3
   AND DokID2 = 7;
```

Dank der Mächtigkeit von SQL lässt sich das Dokument 3 ebenfalls relativ einfach mit allen anderen Dokumenten in der Datenbank (inklusive Dokument 3) vergleichen und in absteigender Reihenfolge nach Ähnlichkeiten sortieren:

```
SELECT *
FROM   Dok_Aehn
WHERE  DokID1 = 3
ORDER BY Aehn DESC;
```

Abschießend ist darauf hinzuweisen, dass die hier vorgestellte relationale Implementierung mit den genannten Optimierungen (kein Speichern von Zeilen mit Null-Werten beim

[15] Aus didaktischen Gründen ist auf eine Kombination der beiden Views in nur einer einzigen View verzichtet worden, weil diese aufgrund der sich ergebenden Verschachtelung unübersichtlich ist.

Attribut `Anzahl` bzw. `Wert` in der Tabelle `DT_ZO` bzw. `Skalarprodukt`) im folgend genannten Extremfall ein zur Gleichung 4.9 unterschiedliches Ergebnis liefert: Werden mit der hier vorgestellten relationalen Implementierung zwei Dokumente verglichen, bei denen zu keiner einzigen Termkombination aus Termen der beiden Dokumente ein Eintrag in der Tabelle `Skalarprodukt` existiert, dann liefert die View `Dok_Aehn` für die Kombination aus den beiden Dokumenten keine Ergebniszeile – im Unterschied zur Gleichung 4.9, die als Ergebnis eine Ähnlichkeit von Null liefert.

Dieser kleine Unterschied zwischen Implementierung und Definition der Dokumentenähnlichkeit ist in der Praxis nicht problematisch. Wird die genannte Implementierung für das IR verwendet, dann sind für den Benutzer Dokumente mit einer Ähnlichkeit Null zu der von ihm gestellten Anfrage nicht von Interesse und können somit aus dem Ergebnis problemlos gestrichen werden. Gleiches gilt bei der Anwendung der Implementierung im IF. Insofern wirkt sich dieser Unterschied zwischen der Implementierung und der Definition der Dokumentenähnlichkeit in Gleichung 4.9 lediglich positiv auf die Geschwindigkeit des Systems aus, weil zu Dokumenten, die offensichtlich keine Gemeinsamkeiten haben, die Ähnlichkeit nicht berechnet wird.

4.2.5 Einstellen neuer Dokumente / Durchführen von Anfragen

Das TVSM integriert im Unterschied zu den in Kapitel 3 vorgestellten Modellen Stoppwörter und Flexionsformen, indem der Termvektorbetrag $|\vec{t_i}|$ für Stoppwörter gleich Null gesetzt wird und der Winkel $\omega_{i,j}$ zwischen den Termvektoren verschiedener Flexionsformen eines Wortes als 0° definiert wird. Daher kommt das TVSM ohne externe Stoppwortlisten und externe Stemmingverfahren aus. Bei Implementierung des TVSM mit einer relationalen Datenbank muss ein Parser folgende Aufgaben durchführen, um neue Dokumente in das Modell einzustellen:

1. Neue Dokumente sind in einzelne Terme zu zerlegen. Dabei sind eventuell vorhandene Formatierungen, Sonderzeichen etc. zu entfernen.[16]

2. Es ist in der Tabelle `Dokument` ein neuer, das neue Dokument repräsentierender Eintrag zu erstellen. Die Anzahlen der verschiedenen Terme im neuen Dokument sind zu zählen und unter Verwendung von SQL-Befehlen in die Tabelle `DT_ZO` einzutragen. Sollte in dem Dokument ein Term vorkommen, der noch nicht in der Tabelle `Term` vorhanden ist, dann ist dieser Term (zuzüglich aller relevanten Einträge in der Tabelle `Skalarprodukt`) anzulegen.

3. Zum Abschluss ist unter Verwendung der View `Dok_Betrag` der Betrag des Dokuments zu berechnen und im Attribut `Betrag` der Tabelle `Dokument` zu dem neuen Dokument zu hinterlegen.

[16] Vgl. dazu auch Abschnitt 3.1, wobei die Paragraphen zur Anwendung von Stoppwortlisten, zum Durchführen des Stemming und zur Anwendung von Synonymerersetzung für das TVSM obsolet sind.

Beim Einsatz des TVSM für IR-Aufgaben werden Anfragen als virtuelle Dokumente aufgefasst. Das Vorgehen bei der Bearbeitung von Anfragen ist daher ähnlich zu dem Einstellen neuer Dokumente:

1. Eine Anfrage ist in einzelne Terme zu zerlegen.

2. Es ist ein neues Dokument in Tabelle `Dokument` zu erstellen, dass die Anfrage repräsentiert. Zu jedem Term der Anfrage sind passende Einträge in `DT_ZO` zu erstellen. Sollte ein Term der Anfrage nicht in der Tabelle `Term` vorhanden sein, dann kann er ignoriert werden, wenn der Term zu allen anderen Termen orthogonal ist (Skalarprodukt gleich Null).[17]

3. Unter Verwendung der View `Dok_Betrag` ist der Vektorbetrag der Anfrage zu berechnen und in der Tabelle `Dokument` zu speichern.

4. Abschließend ist das Anfrage-Dokument mit den restlichen Dokumenten unter Verwendung der View `Dok_Aehn` zu vergleichen und das Ergebnis ist dem Benutzer zu präsentieren.[18]

Um die Anfragebearbeitung zu beschleunigen, ist es sinnvoll, häufig gestellte Anfragen in der Datenbank zwischenzuspeichern. In diesem Fall werden die Schritte 2 und 3 für in der Datenbank schon vorhandenen Anfragen obsolet. Zusätzlich ist es sinnvoll, zu Anfragen gehörende, virtuelle Dokumente entweder durch ein zusätzliches Attribut in der Tabelle `Dokument` zu kennzeichnen oder vollständig in eine eigene Tabelle auszulagern. Damit kann sichergestellt werden, dass dem Benutzer keine virtuellen Dokumente als Ergebnis einer Anfrage präsentiert werden.

4.3 Stoppwort-Lemma

In diesem Abschnitt wird das Stoppwort-Lemma hergeleitet, welches die gängige Vorgehensweise der Praxis, Stoppwörter in Dokumenten bei der Bestimmung der Dokumentenähnlichkeiten zu ignorieren (vgl. Abschnitte 2.3.4.2 und 3.1), auf eine theoretisch fundierte Basis stellt. Dazu wird unter Verwendung des TVSM hergeleitet, dass das Ignorieren von Stoppwörtern keine Auswirkungen auf Dokumentenähnlichkeiten hat. Dieser Herleitung liegt dabei die folgende, in der Praxis verbreitete Annahme zu Grunde: Ein Stoppwort ist ein Term, der bezüglich seiner Bedeutung keinem Thema zugeordnet werden kann. Daraus ergibt sich, dass die Menge aller Stoppworte $T_{\oslash}$ eine echte Teilmenge zu der Menge aller Terme T ist und dass der Betrag des Termvektors eines jeden beliebigen Stoppworts den Wert Null hat. Eine weitere Folge dieser Annahme ist, dass die Termskalarprodukte zwischen zwei Termen immer dann genau Null sind, wenn einer der beiden involvierten Terme ein Stoppwort ist.

[17] Dieses dürfte die in der Praxis übliche Annahme für nicht in der Datenbank enthaltene Terme sein.

[18] Sinnvollerweise sollte der Vergleich des Anfragedokumentes mit sich selbst im Ergebnis, das dem Benutzer präsentiert wird, nicht enthalten sein.

$$
\begin{aligned}
& & |\vec{t_i}| = 0 & \qquad \forall i \in T_\oslash \subset T \\
\Rightarrow & & \vec{t_i}\vec{t_j} = |\vec{t_i}||\vec{t_j}| \cos\omega_{i,j} = 0 & \qquad \text{falls} \quad i \in T_\oslash \ \vee \ j \in T_\oslash
\end{aligned} \tag{4.10}
$$

Der Betrag des unnormierten Dokumentenvektors basiert gemäß Gleichung 4.8 auf Termskalarprodukten. Durch das Ausnutzen von Gleichung 4.10 kann die Gleichung zur Berechnung des Betrages eines unnormierten Dokumentenvektors derart umgeformt werden, dass lediglich Terme, die keine Stoppwörter sind ($T \setminus T_\oslash$), für die Berechnung benötigt werden. Diese Umformung wird im Folgenden für die Berechnung des unnormierten Dokumentenvektors $|\vec{\delta_k}|$, welche analog auf $|\vec{\delta_l}|$ übertragen werden kann, präsentiert (Gleichung 4.11):

$$
\begin{aligned}
|\vec{\delta_k}| &= \sqrt{\sum_{i \in T} \sum_{j \in T} a_{k,i} a_{k,j} \vec{t_i}\vec{t_j}} \\
&= \sqrt{\sum_{i \in T \setminus T_\oslash} \sum_{j \in T} a_{k,i} a_{k,j} \vec{t_i}\vec{t_j} + \underbrace{\sum_{i \in T_\oslash} \sum_{j \in T} a_{k,i} a_{k,j} \vec{t_i}\vec{t_j}}_{=0}} \\
&= \sqrt{\sum_{i \in T \setminus T_\oslash} \left(\sum_{j \in T \setminus T_\oslash} a_{k,i} a_{k,j} \vec{t_i}\vec{t_j} + \underbrace{\sum_{j \in T_\oslash} a_{k,i} a_{k,j} \vec{t_i}\vec{t_j}}_{=0} \right)} \\
&= \sqrt{\sum_{i \in T \setminus T_\oslash} \sum_{j \in T \setminus T_\oslash} a_{k,i} a_{k,j} \vec{t_i}\vec{t_j}}
\end{aligned} \tag{4.11}
$$

$$
\begin{aligned}
\text{sim}(k,l) &= \vec{d_k}\vec{d_l} \\
&= \frac{1}{|\vec{\delta_k}||\vec{\delta_l}|} \sum_{i \in T} \sum_{j \in T} a_{k,i} a_{l,j} \vec{t_i}\vec{t_j} \\
&= \frac{1}{|\vec{\delta_k}||\vec{\delta_l}|} \sum_{i \in T \setminus T_\oslash} \sum_{j \in T} a_{k,i} a_{l,j} \vec{t_i}\vec{t_j} + \underbrace{\sum_{i \in T_\oslash} \sum_{j \in T} a_{k,i} a_{l,j} \vec{t_i}\vec{t_j}}_{=0} \\
&= \frac{1}{|\vec{\delta_k}||\vec{\delta_l}|} \sum_{i \in T \setminus T_\oslash} \left(\sum_{j \in T \setminus T_\oslash} a_{k,i} a_{l,j} \vec{t_i}\vec{t_j} + \underbrace{\sum_{j \in T_\oslash} a_{k,i} a_{l,j} \vec{t_i}\vec{t_j}}_{=0} \right) \\
&= \frac{1}{|\vec{\delta_k}||\vec{\delta_l}|} \sum_{i \in T \setminus T_\oslash} \sum_{j \in T \setminus T_\oslash} a_{k,i} a_{l,j} \vec{t_i}\vec{t_j}
\end{aligned} \tag{4.12}
$$

Abschließend wird in Gleichung 4.12, wie bei der Berechnung des Betrages des unnormierten Dokumentenvektors, die Gleichung 4.10 auf die Berechnung der Dokumentenähnlichkeiten aus Gleichung 4.9 angewandt. Das Ergebnis dieser Umformung kann in der folgenden Aussage zusammengefasst werden: Die Ähnlichkeit zweier Dokumente $\vec{d_k}\vec{d_l}$ kann alleine unter Verwendung von Termen, die keine Stoppwörter sind ($T \setminus T_{\oslash}$), berechnet werden.

4.4 Stemming-Lemma

Das hier im Folgenden vorgestellte und auf dem TVSM basierende Stemming-Lemma fundiert das in der Praxis übliche Vorgehen, Terme auf ihre Stammformen oder Worte in Grundform zurückzuführen und im späteren Verlauf nur mit den Stammformen bzw. Worten in Grundform weiterzuarbeiten.[19] Zur Herleitung des Lemmas ist es zunächst notwendig, folgende Definitionen bezüglich der Bedeutung von Wortstämmen zu treffen: Die Menge aller Wortstämme $T_\perp$ ist eine Teilmenge der Menge aller Terme T. Des Weiteren sei eine Funktion $\perp : T \rightarrow T_\perp$ definiert, die jedem Term den dazugehörigen Wortstamm[20] zuordnet. Die Umkehrrelation $\perp^{-1} : T_\perp \rightarrow \wp(T)$ zur Funktion $\perp()$ liefert zu jedem Wortstamm die Menge aller zu diesem Wortstamm gehörenden Terme inklusive des Wortstamms selbst. Formal gilt somit Folgendes:

$$
\begin{aligned}
T_\perp &\subseteq T & \\
\perp(i) &\in T_\perp & \forall i \in T \\
\perp^{-1}(o) \subseteq T \ \wedge \ o &\in \perp^{-1}(o) & \forall o \in T_\perp \\
\perp(i) &= o & \forall o \in T_\perp,\ i \in \perp^{-1}(o) \\
\nexists i :\ i \in \perp^{-1}(o) \ \wedge \ i &\in \perp^{-1}(p) & \forall o \neq p
\end{aligned}
$$

Bezüglich der Eigenschaften von Wortstämmen werden hier folgende (in der Praxis gängige) Annahmen getroffen: Die thematische Bedeutung eines Wortes ist gleich der thematischen Bedeutung seines Wortstamms, was im TVSM einem Winkel von Null Grad zwischen den beiden Termvektoren entspricht. Des Weiteren wird angenommen, dass Wortstämme und Nicht-Wortstämme gleich gut geeignet sind, den Themenbezug eines Dokuments festzustellen. Somit wird angenommen, dass der Betrag der beiden Termvektoren gleich ist.

$$
\begin{aligned}
& \omega_{i,o} = \omega_{o,i} = 0^\circ \quad \wedge \quad |\vec{t_i}| = |\vec{t_o}| && \forall i \in T,\ o = \perp(i) \\
\Rightarrow \quad & \vec{t_i} = \vec{t_o}
\end{aligned}
$$

Aus diesen Annahmen kann gefolgert werden, dass der Termvektor eines beliebigen Terms zum Termvektor eines Wortstamms identisch ist, weil sowohl die Richtung als auch die Länge der beiden Vektoren gleich sind. Eine Anwendung dieser Folgerung auf die Berechnung

[19] Vgl. dazu die Abschnitte 2.3.2.2 und 3.1.

[20] Für das Stemming-Lemma ist es nicht relevant, ob ein Strong- oder Weak-Stemming (vgl. Abschnitt 2.3.2.2) durchgeführt wird. Daher wird hier der Begriff Wortstamm stellvertretend auch für den Begriff Wort in Grundform verwendet.

des Betrages der unnormierten Dokumentenvektoren nach Gleichung 4.8 führt zu folgendem Ergebnis:

$$
\begin{aligned}
|\vec{\delta}_k| &= \sqrt{\sum_{i\in T}\sum_{j\in T} a_{k,i}a_{k,j}\vec{t}_i\vec{t}_j} \\
&= \sqrt{\sum_{o\in T_\perp}\sum_{i\in\perp^{-1}(o)}\sum_{p\in T_\perp}\sum_{j\in\perp^{-1}(p)} a_{k,i}a_{k,j}\vec{t}_i\vec{t}_j} \\
&= \sqrt{\sum_{o\in T_\perp}\sum_{p\in T_\perp}\sum_{i\in\perp^{-1}(o)}\sum_{j\in\perp^{-1}(p)} a_{k,i}a_{k,j}\vec{t}_o\vec{t}_p} \\
&= \sqrt{\sum_{o\in T_\perp}\sum_{p\in T_\perp}\vec{t}_o\vec{t}_p\Big(\sum_{i\in\perp^{-1}(o)}\sum_{j\in\perp^{-1}(p)} a_{k,i}a_{k,j}\Big)} \\
&= \sqrt{\sum_{o\in T_\perp}\sum_{p\in T_\perp}\vec{t}_o\vec{t}_p\Big(\sum_{i\in\perp^{-1}(o)} a_{k,i}\Big)\Big(\sum_{j\in\perp^{-1}(p)} a_{k,j}\Big)} \\
&= \sqrt{\sum_{o\in T_\perp}\sum_{p\in T_\perp} a'_{k,o}a'_{k,p}\vec{t}_o\vec{t}_p} \qquad (4.13)\\
\text{mit}\quad a'_{k,o} &= \sum_{i\in\perp^{-1}(o)} a_{k,i} \quad\text{und}\quad a'_{k,p} = \sum_{j\in\perp^{-1}(p)} a_{k,j}
\end{aligned}
$$

Im Resultat heißt das, dass für die Berechnung des Dokumentenbetrages lediglich die Wortstämme und die aggregierte Anzahl der jeweiligen Vorkommen aller Wörter eines Wortstamms im Dokument notwendig sind.

Die aus den obigen Annahmen der eingangs gefolgerten Zusammenhänge lassen sich analog zur Gleichung 4.13 in die Gleichung 4.9 zur Berechnung der Dokumentenähnlichkeit mit ähnlichem Ergebnis einsetzen:

$$
\begin{aligned}
\vec{d}_k\vec{d}_l &= \frac{1}{|\vec{\delta}_k||\vec{\delta}_l|}\sum_{i\in T}\sum_{j\in T} a_{k,i}a_{l,j}\vec{t}_i\vec{t}_j \\
&\vdots \\
&= \frac{1}{|\vec{\delta}_k||\vec{\delta}_l|}\sum_{o\in T_\perp}\sum_{p\in T_\perp} a'_{k,o}a'_{l,p}\vec{t}_o\vec{t}_p \qquad (4.14)\\
\text{mit}\quad a'_{k,o} &= \sum_{i\in\perp^{-1}(o)} a_{k,i} \quad\text{und}\quad a'_{l,p} = \sum_{j\in\perp^{-1}(p)} a_{l,j}
\end{aligned}
$$

4.5 Synonym-Lemma

Das Synonym-Lemma fundiert das in der Praxis übliche Vorgehen, synonyme Wörter auf einen führenden Begriff vor der weiteren Verarbeitung durch das IR bzw. IF-System zurückzuführen, um die Zahl der insgesamt zu betrachtenden Wörter zu reduzieren.[21] Das Synonym-Lemma kann analog zum bereits vorgestellten Stemming-Lemma mit Hilfe des TVSM abgeleitet werden.

Dem Synonym-Lemma liegt die in Abschnitt 2.3.4.2 vorgestellte Definition für (Totale) Synonymie zu Grunde. Gemäß dieser Definition sind zwei Terme synonym, wenn sie trotz unterschiedlicher Benennung dieselbe Interpretation haben. Somit ist es für die Praxis ausreichend, für synonyme Terme einen führenden Term zu wählen und die anderen Terme auf den führenden Term zurückzuführen. Daher kann die Menge aller führenden Terme T_f als eine Teilmenge der Menge aller Terme T definiert werden. Des Weiteren kann eine Funktion $F : T \rightarrow T_f$ definiert werden, die jedem Term den dazugehörigen führenden Term zuordnet. Die Umkehrrelation $F^{-1} : T_f \rightarrow \wp(T)$ zur Funktion $F()$ liefert zu jedem führenden Term die Menge aller zu diesem Term gehörenden synonymen Terme, inklusive des führenden Terms selbst. Formal gilt somit Folgendes:

$$
\begin{aligned}
T_f &\subseteq T & \\
F(i) &\in T_f & \forall i \in T \\
F^{-1}(o) \subseteq T \ &\wedge \ o \in F^{-1}(o) & \forall o \in T_f \\
F(i) &= o & \forall o \in T_f,\ i \in F^{-1}(o) \\
\nexists i :\ i \in F^{-1}(o) \ &\wedge \ i \in F^{-1}(p) & \forall o \neq p
\end{aligned}
$$

In Bezug auf die Eigenschaften der führenden Terme können gemäß der Synonymie-Definition folgende Aussagen getroffen werden: Die thematische Bedeutung bzw. Aussagekraft eines Terms ist gleich der thematischen Bedeutung das zu dem Term passenden führenden Terms, was im TVSM einen Winkel von Null Grad zwischen den beiden Termvektoren entspricht. Des Weiteren wird angenommen, dass Terme und führende Terme gleich gut geeignet sind, den Themenbezug eines Dokuments festzustellen. Somit wird angenommen, dass der Betrag der beiden Termvektoren gleich ist.

$$
\begin{aligned}
& \omega_{i,o} = \omega_{o,i} = 0^\circ \quad \wedge \quad |\vec{t_i}| = |\vec{t_o}| && \forall i \in T,\ o = F(i) \\
\Rightarrow \quad & \vec{t_i} = \vec{t_o}
\end{aligned}
$$

Aus diesen Aussagen kann gefolgert werden, dass der Termvektor eines beliebigen Terms zum Termvektor des passenden führenden Terms identisch ist, weil sowohl die Richtung als auch die Länge der beiden Vektoren gleich sind. Eine Anwendung dieser Folgerung auf die Berechnung des Betrages der unnormierten Dokumentenvektoren nach Gleichung 4.8 führt zu folgendem Ergebnis, dass analog zu Gleichung 4.13 umgeformt wird:

[21] Vgl. Abschnitt 3.1.

$$
\begin{aligned}
|\vec{\delta}_k| &= \sqrt{\sum_{i \in T} \sum_{j \in T} a_{k,i} a_{k,j} \vec{t}_i \vec{t}_j} \\
&\vdots \\
&= \sqrt{\sum_{o \in T_f} \sum_{p \in T_f} a'_{k,o} a'_{k,p} \vec{t}_o \vec{t}_p} \qquad (4.15) \\
\text{mit} \quad a'_{k,o} &= \sum_{i \in F^{-1}(o)} a_{k,i} \quad \text{und} \quad a'_{k,p} = \sum_{j \in F^{-1}(p)} a_{k,j}
\end{aligned}
$$

Im Resultat heißt das, dass für die Berechnung des Dokumentenbetrages lediglich die führenden Terme und die aggregierte Anzahl das jeweiligen Vorkommen aller führenden Terme im Dokument notwendig sind. Die aus den obigen Annahmen gefolgerten Zusammenhänge lassen sich analog zur Gleichung 4.15 in die Gleichung 4.9 zur Berechnung der Dokumentenähnlichkeit einsetzen und umformen:

$$
\begin{aligned}
\vec{d}_k \vec{d}_l &= \frac{1}{|\vec{\delta}_k||\vec{\delta}_l|} \sum_{i \in T} \sum_{j \in T} a_{k,i} a_{l,j} \vec{t}_i \vec{t}_j \\
&\vdots \\
&= \frac{1}{|\vec{\delta}_k||\vec{\delta}_l|} \sum_{o \in T_f} \sum_{p \in T_f} a'_{k,o} a'_{l,p} \vec{t}_o \vec{t}_p \qquad (4.16) \\
\text{mit} \quad a'_{k,o} &= \sum_{i \in F^{-1}(o)} a_{k,i} \quad \text{und} \quad a'_{l,p} = \sum_{j \in F^{-1}(p)} a_{l,j}
\end{aligned}
$$

4.6 Vergleich mit anderen Modellen

Das TVSM ist ein algebraisches, Vektor-basiertes Modell mit transzendenten Terminterdependenzen. Es wird im Folgenden mit anderen Vektor-basierten Modellen (VSM und GVSM) sowie anderen Modellen mit transzendenten Terminterdependenzen (FSM, BNN und RbLI) ausführlich verglichen, weil diese Modelle eine hohe Anzahl an Gemeinsamkeiten zu dem TVSM aufweisen.[22]

Mit dem VSM hat das TVSM folgende Gemeinsamkeiten: Beide bilden Dokumente als Vektoren in einem Raum ab und beide verwenden das Kosinus-Maß als Maß für die Ähnlichkeit zwischen zwei Dokumenten bzw. einem Dokument und einer Anfrage. Folgender fundamentaler Unterschied lässt sich jedoch zwischen den beiden Modellen ausmachen: Beim

[22] Eine Übersicht über die linguistischen Eigenschaften aller in dieser Arbeit vorgestellten Modelle findet sich in Tabelle 3.2 auf Seite 82. Eine ausführliche Beschreibung der hier dem TVSM gegenübergestellten Modelle wurden bereits in Kapitel 3 gegeben.

VSM werden die Terme als Dimensionen des Raumes und somit als orthogonale Vektoren repräsentiert, wohingegen beim TVSM die Termvektoren einen beliebigen Winkel[23] zueinander haben können. Dieser Unterschied führt dazu, dass im VSM morphologische Phänomene sowie die Synonymie, Hyponymie und Meronymie nicht abgebildet werden können. Die beim VSM eingesetzten Gewichtungsverfahren (z. B tf-idf) können auch auf das TVSM angewendet werden. Da prinzipiell beim TVSM auch alle Termvektoren zueinander orthogonal gesetzt werden können, kann das TVSM das Ranking des VSM abbilden und ist somit ein Modell, dass von seiner Mächtigkeit her das VSM umfasst.

Wie beim TVSM werden beim GVSM Terme ebenfalls über Vektoren modelliert, die nicht zueinander orthogonal sein müssen. Allerdings modelliert das GVSM die Termvektoren als gewichtete Summe über Vektoren von sogenannten Mintermen. Diese Minterm-Vektoren sind Einheitsvektoren, sie sind daher zueinander orthogonal und spannen den Vektorraum des GVSM-Modells auf. Minterme repräsentieren eine mögliche Kombination von Termen in einem Dokument. Dadurch, dass in einen Termvektor nur diejenigen Minterm-Vektoren eingehen, bei denen der Term Bestandteil einer in den Dokumenten existierenden Kombination von Termen ist, haben Terme, die co-occurrent zueinander auftreten, einen geringen Winkel und somit eine hohe Ähnlichkeit. Somit handelt es sich beim GVSM um ein Modell mit immanenten Terminterdependenzen, weil die Termähnlichkeiten im Modell (über ein Co-Occurrenz-Maß) vorgegeben werden. Im Unterschied dazu legt das TVSM keine explizite Definition für Termähnlichkeiten fest. Es werden ausschließlich Rahmenbedingungen an die Termähnlichkeiten vorgegeben (wie z. B., dass synonyme Terme einen Winkel von 0° haben sollen). Somit ist das TVSM ein Modell mit transzendenten Terminterdependenzen. Die Co-Occurrenz-basierte Definition von Termähnlichkeiten des GVSM lässt sich auf das TVSM übertragen[24], wodurch das TVSM in der Lage ist, dasselbe Ranking durchzuführen wie das GVSM. Daraus folgt, dass das TVSM ein Modell ist, welches bezogen auf seine Mächtigkeit das GVSM umfasst.

Folgender fundamentaler Unterschied kann zwischen dem FSM und dem TVSM ausgemacht werden: das FSM ist ein mengentheoretisches Modell, während das TVSM ein algebraisches, Vektor-basiertes Modell ist. Daraus ergibt sich, dass das FSM in Bezug auf das IR gegenüber dem TVSM den Vorteil hat, dass es logische Verknüpfungsoperationen bei der Auswertung von Anfragen unterstützt. Umgekehrt hat das TVSM gegenüber dem FSM den Vorteil, dass es besser für den Vergleich von Dokumenten geeignet ist, weil der Ähnlichkeitsbegriff beim TVSM auf Dokumentenbasis definiert ist (als Winkel zwischen den Dokumentenvektoren). Des Weiteren unterstützt das TVSM Wortgewichte, wodurch eine Stoppwortliste in das Modell integriert werden kann. Das FSM bietet kein derartiges Konstrukt. Bei beiden Modellen werden die Terminterdependenzen über eine Termähnlichkeitsmatrix (beim FSM Keyword-Connection-Matrix genannt) von außen vorgegeben, weshalb beide Modelle in die Klasse der Modelle mit transzendenten Terminterdependenzen eingeordnet sind. Während

[23] Der Winkel ist aufgrund der Einschränkung der Achsen auf positive Werte in einem Bereich von 0° bis 90° eingeschränkt.

[24] Indem die Termvektoren nach demselben Prinzip wie bei dem GVSM definiert und berechnet werden. Unter Verwendung dieser Vektoren können alle möglichen Skalarproduktkombinationen berechnet und gespeichert werden wodurch sie in das TVSM integriert werden können.

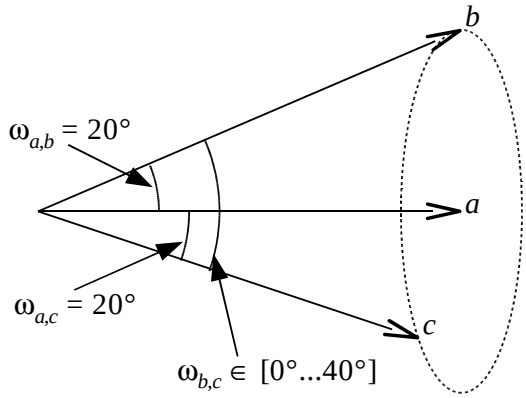


Abbildung 4.5: Ein Beispiel für die schwache Transitivität beim TVSM.

beim FSM die Termähnlichkeiten beliebig im Intervall $[0...1]$ und voneinander unabhängig festgelegt werden können, hat die Vektorbasierung des TVSM zur Folge, dass die Termähnlichkeiten bestimmte Konsistenzkriterien erfüllen müssen:

- *Symmetrie*: Die Ähnlichkeit des Terms i zum Term j ist genau so groß, wie die Ähnlichkeit des Terms j zum Term i. Dieses Konsistenzkriterium kann direkt aus der Ähnlichkeitsdefinition für Terme, die auf dem Winkel zwischen den Termen basiert, abgeleitet werden:

$$\omega_{i,j} = \omega_{j,i}$$

- *Schwache Transitivität*: Da Terme beim TVSM durch Termvektoren repräsentiert werden, müssen die Termähnlichkeiten, die mit den Termwinkeln korrespondieren im Sinne einer gültigen Anordnung von Vektoren in einem Raum zueinander konsistent sein, womit einige Kombinationen von Termähnlichkeiten ausgeschlossen werden. Abbildung 4.5 illustriert den Sachverhalt an einem Beispiel mit den drei Vektoren a, b und c in einem drei-dimensionalen Raum. Wird angenommen, dass der Winkel $\omega_{a,c}$ zwischen den Termen a und c genau 20° beträgt, dann muss die Spitze des Vektors c auf dem gestrichelt eingezeichneten Kreis liegen. Unter der Annahme, dass der Winkel $\omega_{a,b}$ zwischen den Vektoren a und b ebenfalls 20° beträgt und b bereits bekannt ist, folgt daraus, dass der Winkel $\omega_{b,c}$ zwischen b und c in einem Bereich von 0° (b und c überdecken sich) bis 40° (b und c liegen auf den gegenüberliegenden Punkten des Kreises) liegen muss. Die Transivitätsbedingung wird hier als *schwach* bezeichnet, weil für den Winkel von Vektor c mit Vektor b nicht exakt ein Wert, sondern lediglich ein ganzer Wertebereich festgelegt wird. Zudem kann leicht gezeigt werden, dass die Transivitätsbedingung für das gleiche Beispiel jedoch mit 45° statt 20° Winkeln dazu führt, dass

der Winkel zwischen b und c in einem Bereich von 0° bis 90° Grad liegen darf. Dieses bedeutet, dass keine Einschränkung vorliegt, weil die Winkel im TVSM aufgrund der positiven Achsen auf maximal 90° beschränkt sind. Allgemein kann zu der schwachen Transitivität beim TVSM folgende Ungleichung aufgestellt werden:

$$|\omega_{a,b} - \omega_{a,c}| \leq \omega_{b,c} \leq \min(\omega_{a,b} + \omega_{a,c}, 90°)$$

Die schwache Transitivität hat in Bezug auf linguistische Aspekte von Termen durchaus ihre Berechtigung. Wenn man über ein Objekt, das durch den Term a (z. B. Auto) repräsentiert wird, aussagt, dass es zu den Objekten, die durch b (z. B. Motorrad) und c (z. B. Panzer) repräsentiert werden, sehr ähnlich ist (Termwinkel = 20°), dann kann man im Allgemeinen unterstellen, dass auch b und c eine gewisse minimale Ähnlichkeit (Termwinkel $\geq$ 40°) zueinander haben. Wenn die Ähnlichkeit von a (z. B. Auto) zu b (z. B. Blume) und c (z. B. Stein) gering ist (Termwinkel = 45°), dann kann man es im intuitiven Verständnis durchaus akzeptieren, dass b eventuell keine Ähnlichkeit zu c aufweist (Termwinkel = 90°). Es ist aber auch möglich, dass b und c auch in diesem Fall eine hohe Ähnlichkeit haben. (Z. B. wenn c eine Rose repräsentiert.)

Der Unterschied zwischen dem TVSM und dem RbLI ist der, dass das RbLI ebenso wie das FSM keine Konsistenzbedingungen an Termähnlichkeiten stellt. Zusätzlich gehen aber beim RbLI in die Ähnlichkeitsberechnung zwischen Dokument und Anfrage nur diejenigen Terme aus dem Dokument ein, deren ähnlichste Terme im Dokument den Termen in der Anfrage entsprechen. Des Weiteren ist das RbLI ein probabilistisches Modell, während das TVSM ein algebraisches Modell ist.

Abschließend ist zu erwähnen, dass sich das TVSM vom BNN dadurch unterscheidet, dass beim TVSM die Termähnlichkeiten direkt vorgegeben werden während beim BNN die Termähnlichkeiten indirekt vorgegeben werden und somit erst aus vorgegebenen Trainingsdaten erlernt werden müssen. Dieses hat zwar den Vorteil, dass Termähnlichkeiten nicht explizit definiert werden müssen und dass das BNN theoretisch alle linguistischen Phänomene erfassen kann, aber auch den Nachteil, dass eine Vielzahl an möglichst ausgewogenen Trainingsdaten gebraucht werden, die vor dem Training zu erstellen sind. Zudem ist der Rechenaufwand für das Training des Netzes sehr hoch, wodurch eine Adaption des Netzes an neue Terme und Terminterdependenzen erschwert wird.

4.7 Kritik am TVSM

Im Unterschied zu den anderen, in Kapitel 3 vorgestellten Modellen mit transzendenten Terminterdependenzen, legt das TVSM seine den Termähnlichkeiten zu Grunde liegenden Annahmen explizit dar.[25] Allerdings sind die Angaben zu dem konkreten Maß von Termähnlichkeiten in Bezug auf einige linguistische Phänomene vage formuliert. Während für die Flexion und die Synonymie die Ähnlichkeit genau vorgegeben wird (nämlich ein Termwinkel

[25] Vgl. dazu Abschnitt 4.2.

von 0° zwischen Flexionsformen bzw. Synonymen zu einem Term), wird für die Komposition, die Derivation, Hyponymie und Meronymie nur die vage Angabe gemacht, dass eine gewisse Ähnlichkeit zwischen Termen, die über eines der genannten Phänomene miteinander verknüpft sind, bestehen muss. Diese vage Aussage ist für eine Operationalisierung der Termähnlichkeiten nicht hinreichend, zumal zusätzlich zu dem genannten Problem auch das Problem der in Abschnitt 4.6 vorgestellten und vektorbedingten Konsistenzbedingungen besteht. Das heißt also, dass das TVSM keine hinreichende Aussage darüber trifft, wie Termähnlichkeiten im Detail so festgelegt werden können, dass diese gemäß den Anforderungen des Vektorraumes zueinander konsistent sind und gleichzeitig die zu Grunde liegenden linguistischen Phänomene adäquat repräsentieren. Immerhin ist zu würdigen, dass das TVSM – im Unterschied zu anderen Modellen – zumindest einige Rahmenwerte für die Termähnlichkeiten vorgibt.

Ein weiterer Aspekt, der dem TVSM – und ebenso den übrigen in Kapitel 3 vorgestellten Modellen – angelastet werden kann, ist das Fehlen einer Repräsentation der Variabilität von Wortinterpretationen. Konkret: Homographie und Metonymie werden nicht abgebildet. Bei der Metonymie liegt eine nicht-wörtliche Verschiebung der Wortbedeutung vor (wie z. B. Berlin anstelle von Bundestag). Dieses kann die Suche bzw. den Vergleich von Dokumenten erschweren. Bei Homographen ist der Effekt noch extremer, weil Homographen zueinander keinen thematischen Bezug haben müssen. Daher ist es wünschenswert, dass ein IF/IR-Modell die Variabilität von Wortinterpretationen geeignet repräsentiert. Ebenso wie die Variabilität von Wortinterpretationen werden Wortgruppen vom TVSM (und den meisten anderen Modellen) nicht explizit berücksichtigt, weshalb dieser Aspekt negativ anzumerken ist.

Positiv ist zu erwähnen, dass das TVSM in der Lage ist, durch die Termgewichte Stoppwörter und durch die Termähnlichkeiten verschiedene Flexionsformen eines Terms abzubilden. Somit sind beim TVSM Stoppwortliste und Stemming in das Modell integriert. Zudem eignet sich das TVSM als Erklärungsmodell und zum theoretischen Nachweis, dass die in der Praxis extern eingesetzten Stoppwortlisten und Stemmingverfahren in ihrer Form auch aus theoretischer Sicht eine Daseinsberechtigung haben.[26] Bei Betrachtung der Gleichungen 4.8 und 4.9 zur Berechnung des Dokumentvektorbetrags bzw. der Dokumentenähnlichkeit fällt auf, dass beide in Bezug auf ihren Rechenaufwand stark von der Anzahl der in einem Dokument existierenden verschiedenen Terme abhängen. Es ist daher sinnvoll, die Zahl der verschiedenen Terme pro Dokument zu minimieren, um den Rechenbedarf gering zu halten. Aus diesem Grund ist es wünschenswert, das TVSM derart zu erweitern, dass Stoppwortliste und Stemming zwar im Modell integriert bleiben, dass diese aber unter Anwendung des Stoppwort- und Stemming-Lemma aus der Berechnung der Dokumentenähnlichkeit derart ausgelagert werden, dass Stoppwörter und Wortstämme nur noch einmal, z. B. beim Parsing, explizit berücksichtigt werden müssen.

[26] Vgl. dazu das Stoppwort-Lemma in Abschnitt 4.3 und das Stemming-Lemma in Abschnitt 4.4.

Kapitel 5

Enhanced TVSM (eTVSM)

In Abschnitt 4.7 wurden bereits die Mängel des TVSM vorgestellt. So wurde dort u. a. kritisiert, dass das TVSM nicht genügend Aussagen zu den paarweisen Ähnlichkeiten zwischen Termen trifft, um diese vollständig operationalisieren zu können. Des Weiteren berücksichtigt das TVSM keine Wortgruppen und es berücksichtigt nicht die Variabilität von Wortinterpretationen (Homographie und Metonymie). Diese Mängel werden nun aufgegriffen und mit der Einführung des eTVSM beseitigt. Das eTVSM unterscheidet sich vom TVSM dadurch, dass Dokumentenähnlichkeiten nicht mehr auf Basis von Termen, sondern auf Basis von Interpretationen berechnet werden. Die Berechnung an sich beruht jedoch auf demselben Vektorraumkonzept wie beim TVSM, das nun auf Interpretationen und nicht auf Terme angewandt wird. Zusätzlich werden die Erkenntnisse aus dem Stoppwort-, dem Stemming- und dem Synonym-Lemma[1] im eTVSM gewinnbringend (das heißt z. B. bezogen auf die Berechnung von Dokumentenähnlichkeiten: rechenaufwandreduzierend) umgesetzt. Um die genannten Verbesserungen gegenüber dem TVSM umzusetzen, ist es erforderlich, dass im eTVSM die linguistischen Fachbegriffe Wort, Wortstamm, Term, Interpretation und Thema explizit abgebildet werden. Da es sich bei diesen Erweiterungen um zusätzliche Entitäten handelt, wird die Diskussion von der Seite des Datenmodells und nicht von der Seite der mathematischen Repräsentation der Dokumente im Vektorraum aufgerollt. Konkret bedeutet das, dass der Schritt vom TVSM zum eTVSM zur Folge hat, dass das Datenmodell des eTVSM (vgl. Abbildung 5.1 auf Seite 111) zusätzliche Entitäten und Beziehungen im Vergleich zum Datenmodell des TVSM (vgl. Abbildung 4.4 auf Seite 95) aufweist. Aufbauend auf einigen von den genannten Erweiterungen kommt bei dem eTVSM eine Heuristik hinzu, welche die Berechnung von Themenvektoren aus extern vorgegebenen Themenstrukturen gestattet. Ausgehend von diesen Themenvektoren werden paarweise Ähnlichkeiten zwischen Interpretationen abgeleitet, wodurch diese operationalisiert werden.

Dieses Kapitel hat die folgende Gliederung: In Abschnitt 5.1 werden die theoretischen und datenmodellbezogenen Aspekte des eTVSM erläutert, anschließend wird in Abschnitt 5.2 eine

[1] Vgl. dazu Abschnitt 4.3 und folgende.

Ontologie-Sprache vorgestellt, die die wesentlichen und nicht-trivialen Aspekte der in einem eTVSM eingebetteten Ontologie grafisch anschaulich beschreiben kann. Unter Verwendung dieser Modellierungssprache wird eine Beispiel-Ontologie vorgestellt, die in Abschnitt 5.3 zur Erläuterung einer relationalen Implementierung des eTVSM verwendet wird. Zum Abschluss wird das eTVSM im Abschnitt 5.4 mit anderen Modellen verglichen.

5.1 Konzept

Eine Schwierigkeit bei dem Versuch das eTVSM zu vermitteln bzw. es nachzuvollziehen ist, dass die Entitäten und Beziehungen des Modells und ihre Interpretationen bzw. ihre Aufgaben hochgradig zueinander und zu dem zu Grunde liegenden mathematischen Modell interdependent sind. Somit ist eine Entität oder Beziehung zwischen zwei Entitäten in einigen Fällen für sich, ohne Verweis auf die anderen Strukturen nicht nachvollziehbar. Aus diesem Grunde kann das eTVSM nicht strikt sequentiell, d.h. ohne Verwendung von Vorgriffen auf noch zu tätigende Definitionen bzw. Erklärungen, vermittelt werden. Daher wird im Folgenden zunächst das Grobkonzept des eTVSM in Form einer Übersicht vorgestellt auf das in den Unterabschnitten dieses Abschnittes, in denen die datenmodellbezogenen und mathematischen Aspekte im Detail behandelt werden, zurückgegriffen wird. Dem eTVSM liegen die folgenden Gedanken zu Grunde, die die Datenstruktur des Modells maßgeblich bestimmen:

1. Speichere Dokumente (notfalls unter Verwendung von Redundanzen) derart ab, dass die Berechnung von Dokumentenähnlichkeiten mit einem geringen rechnerischen Aufwand durchgeführt werden kann.
2. Versuche möglichst viele linguistische Phänomene zu erfassen.
3. Verwende vorgegebene Themenstrukturen zur Ableitung von Ähnlichkeiten.

Aus den ersten beiden Gedanken folgt, dass das eTVSM – im Unterschied zu dem TVSM und den anderen in Kapitel 3 genannten Modellen – eine strikte Unterscheidung zwischen den Begriffen Wort, Wortstamm, Term und Interpretation vornimmt, um z. B. die Erkenntnisse des Stoppwort-, Stemming- und Synonym-Lemmas gewinnbringend umsetzen zu können. Diese Unterscheidung zwischen den Begriffen spiegelt sich im Datenmodell des eTVSM wieder (vgl. dazu Abbildung 5.1). Während bei den anderen Modellen ein Dokument aus einer gewichteten Menge von Termen besteht und Terme mit Worten bzw. Wortstämmen (falls vorab ein externes Stemming durchgeführt wurde) gleichgesetzt werden, haben diese Begriffe (die im Datenmodell durch jeweils eigene Entitytypen repräsentiert werden) beim eTVSM eine unterschiedliche und genau definierte Bedeutung:

- *Dokument*: Ein Dokument ist eine Liste von Worten, bei der jedem Wort eine eindeutige Position in dem Dokument zugewiesen wird.
- *Wort*: Worte sind die direkten Bestandteile eines Dokuments, die nach dem Entfernen von Formatierungen, Abbildungen sowie Satz- und Sonderzeichen übrig bleiben.

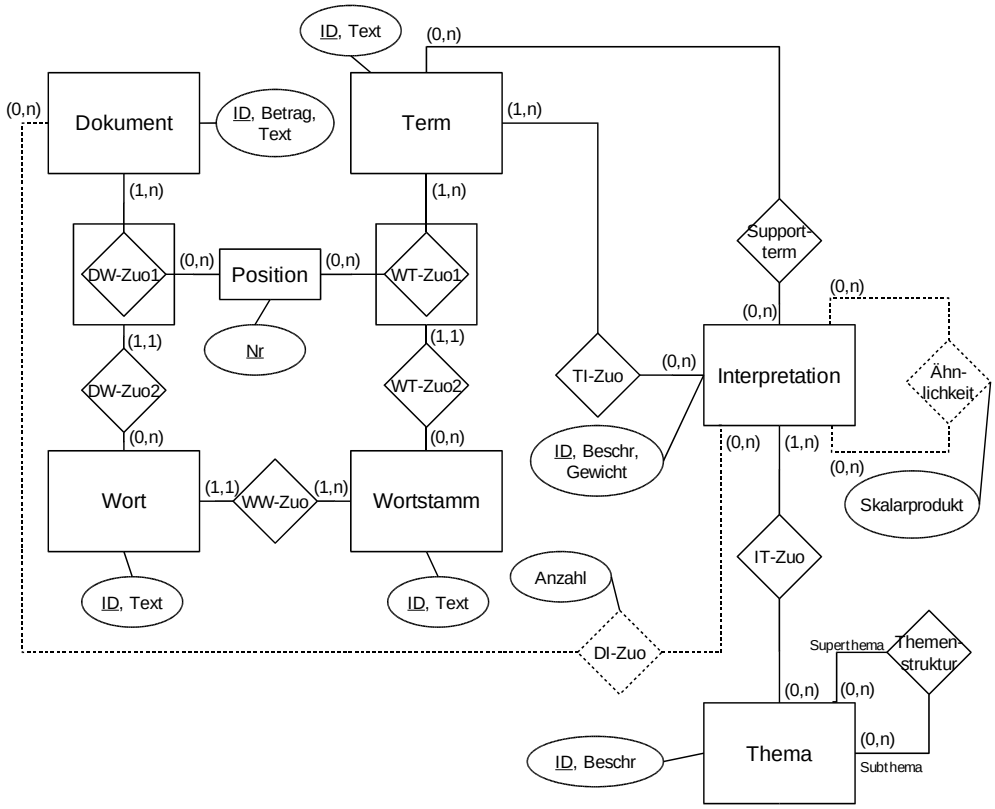


Abbildung 5.1: Relationales Datenmodell für das eTVSM.

- *Wortstamm*: Zu jedem Wort (z. B. geht) ist ein *Wortstamm* definiert, der je nach verwendetem Stemming-Verfahren entweder der Grundform des Wortes (z. B. gehen) beim Strong-Stemming oder dem Stamm eines Wortes (z. B. geh) beim Weak-Stemming entspricht.[2]

- *Term*: Ein Term besteht entweder aus einem einzelnen Wortstamm (z. B. Maus) oder aus einer Gruppe von mehreren Wortstämmen (z. B. New York).

- *Interpretation*: Eine Interpretation ist eine (mögliche) Bedeutung eines Terms. Jeder Term hat mindestens eine Interpretation, zwei Terme können sich aber auch eine Interpretation teilen. Beispielsweise haben Rechner und Computer beide dieselbe Interpretation, weshalb diese Worte als Synonyme bezeichnet werden. Andererseits kann

[2] Vgl. dazu Abschnitt 2.3.2.2.

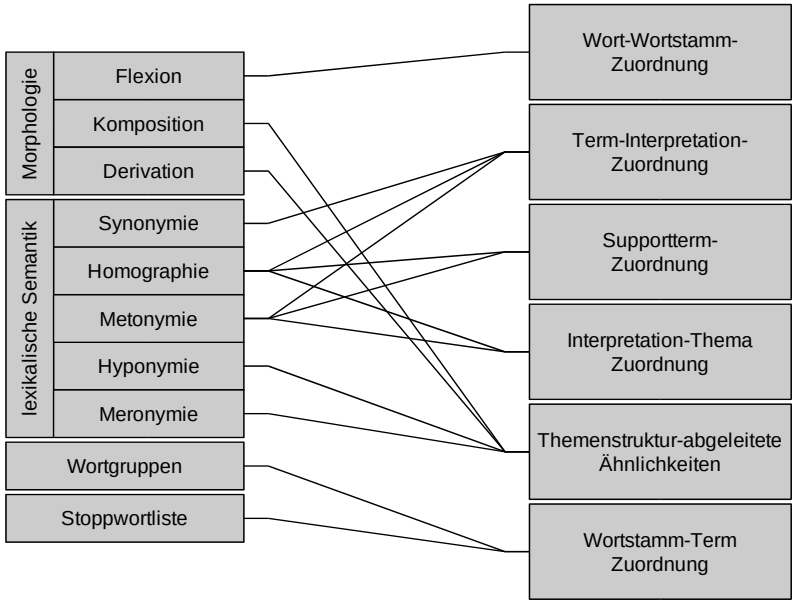


Abbildung 5.2: Konstrukte des eTVSM und ihre Bezug zu linguistischen Phänomenen.

ein Term wie z. B. Maus mehrere Interpretationen (Computereingabegerät oder kleines Nagetier) besitzen. In diesem Fall handelt es sich dann um einen Homographen.

- *Thema*: Themen werden beim eTVSM als höchste Abstraktionsstufe und als thematischer Bezug für Interpretationen verwendet. Themen sind strukturiert und die ihnen zu Grunde liegende Struktur wird beim eTVSM zur Ableitung von Ähnlichkeiten zwischen Interpretationen – im Sinne des dritten Gedankens – verwendet. Interpretationen müssen mindestens einem Thema zugeordnet werden. Wie in Abschnitt 5.1.2 gezeigt wird, ist es sinnvoll einigen speziellen Interpretationen (z. B. Homographen) mehrere Themen zuzuordnen.

Zur Verarbeitung von Dokumenten ist eine strikte Unterscheidung zwischen den Begriffen Wort, Wortstamm, Term, Interpretation und Thema alleine nicht ausreichend. Daher müssen zusätzlich zu den Begriffen, die im Datenmodell durch Entitäten repräsentiert werden, Zuordnungen zwischen den Begriffen definiert und mit Daten belegt werden. Diese Zuordnungen repräsentieren linguistische Phänomene und sind gleichzeitig eine Lösung für die verschiedenen linguistischen Problemstellungen. Die Abbildung 5.2 zeigt, welche Zuordnungen beim eTVSM gegenüber dem TVSM hinzugefügt werden (rechte Seite) und welche linguistischen Phänomene (linke Seite) durch welche von diesen Zuordnungen repräsentiert werden (dargestellt durch eine Verbindung).

Man kann der Abbildung 5.2 u. a. entnehmen, dass einige linguistische Phänomene zur Abbildung mehrere Zuordnungen benötigen und dass einige Zuordnungen zur Repräsentation mehrerer linguistischer Phänomene verwendet werden. Insbesondere die Homographie und die Metonymie erweisen sich als kompliziert, was mit ein Grund dafür sein dürfte, dass fast alle anderen gängigen Verfahren diese linguistischen Phänomene nicht berücksichtigt haben.[3]

Abbildung 5.3 zeigt die zeitlich-sachlogischen Zusammenhänge der verschiedenen Zuordnungen und Entitäten. Wie aus der Abbildung ersichtlich, werden zunächst ontologiebezogene linguistische Daten (insbesondere die Themenstruktur) durch eine Vorbereitungstransaktion zu Interpretations-Skalarprodukten verarbeitet. Die Vorbereitungstransaktion entspricht dabei dem dritten, bereits zum Anfang dieses Abschnittes genannten Gedanken des eTVSM. Sie dient der Ableitung von Ähnlichkeiten aus vorgegebenen Themenstrukturen. Zu diesem Zweck wird in Abschnitt 5.1.1 eine Heuristik vorgestellt, welche in der Lage ist, eine Struktur von Themen adäquat durch Vektoren in einem Raum zu repräsentieren, so dass aus diesen Vektoren die Ähnlichkeiten von Themen abgeleitet werden können. Aufbauend auf diesen Ähnlichkeiten können anschließend die Skalarprodukte für die verschiedenen Interpretationen abgeleitet werden. Die Skalarprodukte bilden – analog zum TVSM – neben den Dokument-Interpretation-Zuordnungen die Basis zur Berechnung von Dokumentenähnlichkeiten. Die Skalarprodukte sind somit streng genommen redundant, weshalb diese im Datenmodell (Abbildung 5.1) gestrichelt eingezeichnet sind. Diese Redundanz ist jedoch für eine hohe Performanz des Systems notwendig. Die Vorbereitungstransaktion wird idealer Weise nur einmal, zur Initialisierung des Modells ausgeführt.

Dem ersten und zweiten Gedanken des eTVSM entsprechend werden beim eTVSM möglichst viele linguistische Phänomene beim Einlesen von neuen Dokumenten über die Dokument-Einstellungstransaktionen erfasst (vgl. Abbildung 5.3). Diese Transaktionen verwenden linguistisches „Wissen“, dass in den verschiedenen Zuordnungen (wie z. B. der Wort-Wortstamm- und der Supportterm-Zuordnung) erfasst ist, um die Dokumente derart aufzubereiten, dass diese in Form einer Dokument-Interpretation-Zuordnung repräsentiert werden. Da diese Zuordnung streng genommen redundant ist, ist sie in Abbildung 5.1 gestrichelt eingezeichnet. Zusätzlich berechnen die Dokument-Einstellungstransaktionen die Dokumentenbeträge analog zum TVSM.

Wie bereits erwähnt, verläuft die Berechnung der Dokumentenähnlichkeiten beim eTVSM analog zum TVSM mit folgendem kleinen Unterschied: Beim eTVSM wird die Ähnlichkeit zwischen den Dokumenten basierend auf den (redundanten) Dokument-Interpretation-Zuordnungen, den Dokumentenbeträgen und den (redundanten) Interpretations-Skalarprodukten berechnet. Der Grund dafür, warum beim eTVSM Interpretationen und ihre Skalarprodukte als Basis für die Berechnung dienen, ist die gewählte Umsetzung des Synonym-Lemmas und die Erweiterung des eTVSM um die Phänomene der Homographie und Metonymie. Das Synonym-Lemma besagt, dass zwei Terme mit derselben Interpretation zu einem führenden Term zusammengefasst werden können, ohne dass dieses die Ähnlichkeiten zwischen zwei Dokumenten beeinflusst.[4] Somit ist es möglich, die Zahl der Terme zu reduzieren, indem man

[3] Vgl. dazu die Tabelle 3.2 auf Seite 82.
[4] Vgl. dazu Abschnitt 4.5.

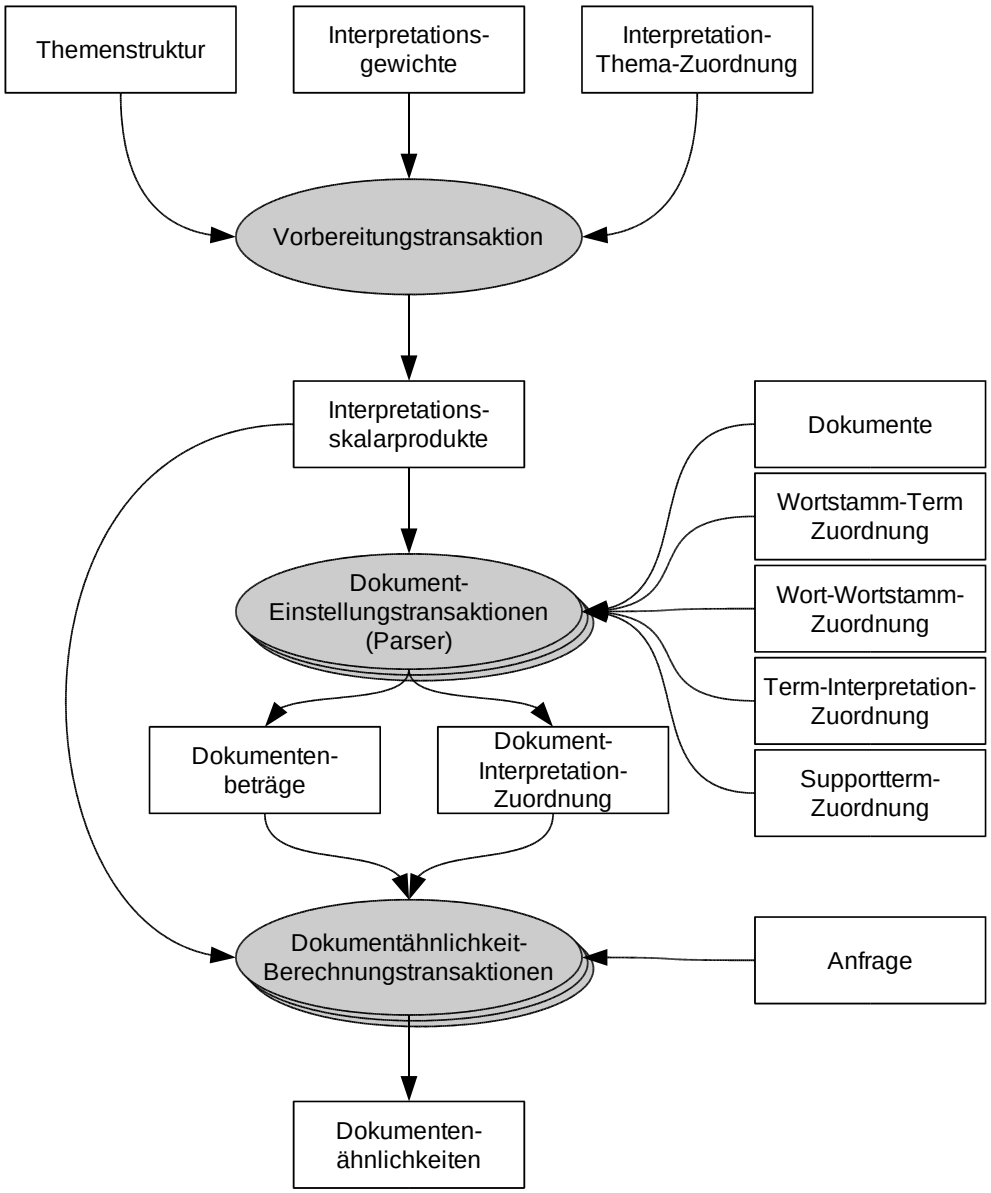


Abbildung 5.3: Transaktionen und ihre Ein-/Ausgabedaten.

die Berechnung auf sogenannte führende Terme reduziert. Aus konzeptioneller Sicht ist jedoch dieses Vorgehen „unschön“, weil die Wahl des führenden Terms aus einer Menge von synonymen Termen willkürlich ist und weil diese Art der Umsetzung nicht den realen Umständen entspricht. Synonyme Terme zeichnen sich dadurch aus, dass sie dieselbe Interpretation haben, somit ist es aus konzeptioneller Sicht sinnvoll, das Konstrukt der Interpretation einzuführen und synonymen Termen dieselbe Interpretation zuzuordnen. Somit übernimmt die Interpretation die Aufgabe eines führenden Begriffes. Zusätzlich ermöglicht eine derartige Umsetzung das Berücksichtigen von Homographie und Metonymie, indem einem Term mehrere Interpretationen zugeordnet werden. Die genauen Details zur Umsetzung von Homographie, Metonymie und Synonymie mit Hilfe von Interpretationen werden in Abschnitt 5.1.2 beschrieben. Zum Abschluss werden in den Abschnitten 5.1.3 und 5.1.4 die Repräsentationen von Stoppwortlisten, Stemming und Wortgruppen behandelt.

5.1.1 Paarweise Themen-Ähnlichkeiten

Einer der größten Kritikpunkte am TVSM ist, dass es die Existenz von Termähnlichkeiten als Kosinus des Winkels zwischen zwei Termvektoren postuliert, aber es bis auf zu Stoppwörter, Stemming und Synonymen keine genaueren Angaben zu der Beschaffenheit der Ähnlichkeiten gibt. Zwar legt das TVSM fest, dass jede Dimension einem elementaren Themengebiet entspricht und dass die Termvektoren passend zu ihrer thematischen Zugehörigkeit in Richtung der entsprechenden Dimensionen zeigen, allerdings ist diese Aussage für eine praktische Umsetzung zu vage. Erschwerend kommt hinzu, dass die elementaren Themengebiete wegen ihrer Dimensionseigenschaft zueinander orthogonal sein müssen. Insofern lassen sich intuitive Zusammengänge zwischen Themen nicht durch elementare Themengebiete abbilden, wodurch eine praktische Anwendung erschwert wird. Aus diesem Grunde wird für das eTVSM nun eine Heuristik vorgestellt, die in der Lage ist, eine Themenstruktur, die aus ggf. mehreren zykelfreien, gerichteten Graphen besteht, derart in einen Vektorraum abzubilden, dass der Kosinus der Winkel zwischen den einzelnen Vektoren ein Maß für die Ähnlichkeit von Themengebieten ist. In Abschnitt 5.1.2 wird gezeigt, wie diese Themenvektoren und Themenähnlichkeiten zur Herleitung der Skalarprodukte zwischen jeweils zwei Interpretationen verwendet werden.

5.1.1.1 Problemstellung

Die Problemstellung lautet, jeder möglichen Kombination aus jeweils zwei Themen $\tau_a, \tau_b \in \Theta$ aus der Menge aller Themen Θ eine Ähnlichkeit $\text{sim}(\tau_a, \tau_b)$ zuzuweisen. Diese Ähnlichkeiten sollen dabei derart beschaffen sein, dass Themen, die eng verwandt sind (wie z. B. τ_{Computer} und τ_{Software}), eine höhere Ähnlichkeit haben als Themen, die miteinander nur wenig verwandt sind (wie z. B. τ_{Computer} und τ_{Weltraum}). Für eine (sehr) geringe Anzahl an Themen lässt sich diese Zuordnung von Ähnlichkeiten manuell handhaben, allerdings nimmt die manuelle Handhabbarkeit mit der Anzahl der Themen rapide ab. Insbesondere die Sicherung der Konsistenz innerhalb der Daten wird mit einer zunehmenden Anzahl an Themenkombinationen problematisch. So müssen neben der thematischen Konsistenz auch die vom Modell geforderten Konsistenzbedingungen erfüllt werden. Da das eTVSM, wie in Abschnitt 5.1.2.6

noch gezeigt wird, Dokumente als normierte Summe von Interpretationensvektoren, die Themen zugeordnet sind, darstellt, müssen sowohl die Ähnlichkeiten zwischen Themen als auch die Ähnlichkeiten zwischen Interpretationen die Konsistenzkriterien dieser Repräsentationsform erfüllen. Diese Konsistenzkriterien sind analog zu den Konsistenzkriterien des TVSM, bei dem Dokumentvektoren eine normierte Summe von Termvektoren sind.[5] Die leichte Verschiebung der Bedeutung von Termvektoren beim TVSM zu Interpretationsvektoren beim eTVSM ist deshalb möglich, weil diese Verschiebung der Anwendung der Erkenntnisse aus dem Synonym-Lemma[6] entspricht. Die einzuhaltenden Konsistenzkriterien für die Themen sind:[7]

1. Normierung: $\text{sim}(\tau_a, \tau_b) \in [0...1]$
2. Symmetrie: $\text{sim}(\tau_a, \tau_b) = \text{sim}(\tau_b, \tau_a)$
3. Maximalität: $1 = \text{sim}(\tau_a, \tau_a) \geq \text{sim}(\tau_a, \tau_b)$
4. Schwache Transitivität: Da die Ähnlichkeiten beim eTVSM als Kosinus der Winkel $\omega_{a,b} = \cos^{-1}(\text{sim}(\tau_a, \tau_b))$ (analog für alle anderen Kombinationen) zwischen Vektoren interpretiert werden, muss folgendes für die Winkel gelten:

$$|\omega_{a,b} - \omega_{a,c}| \leq \omega_{b,c} \leq \min(\omega_{a,b} + \omega_{a,c}, 90°)$$

Aufgrund der Konsistenzbedingungen ist es sinnvoll, die paarweisen Ähnlichkeiten zwischen den Themen nicht manuell, sondern automatisiert abzuleiten. Da aber, wie bereits in Abschnitt 3.3 gezeigt, einfache statistische Verfahren zur vollautomatisierten Herleitung von Ähnlichkeiten auf Basis von Dokumentenkorpora (wie z. B. Co-Occurrenz-Verfahren) wenig geeignet sind, ist eine teilautomatisierte Herleitung von Ähnlichkeiten notwendig. Das heißt, dass Themenstrukturen z. B. über Ontologien von außen, also vom Menschen, vorgegeben werden. Die Vorgabe dieser Themenstrukturen sollte daher möglichst in einer Form geschehen, die für den Menschen leicht zu handhaben ist und die gleichzeitig unter Verwendung eines wohldefinierten Ähnlichkeitsmaßes die Ableitung von paarweisen Ähnlichkeiten ermöglicht, wobei die oben genannten Konsistenzkriterien erfüllen werden.

5.1.1.2 Repräsentationsform für Themenstrukturen

Eine gängige Repräsentationsform für Themenstrukturen sind Graphen, wie z. B. der in der Abbildung 5.4 dargestellte Graph, der Themen in Super- bzw. Subthema-Beziehungen setzt. Die Kanten sind nicht typisiert, sie können aber dennoch verschiedene Arten von Beziehungen zwischen den Themen darstellen. Sinnvolle Beziehungen sind z. B besteht-aus, ist-ein, ist-verwandt-mit, etc.. Die Tatsache, dass die verschiedenen Relationshiptypen (hier) nicht unterschieden werden, bedeutet dass der Typ der Beziehung bei dieser Darstellungsform keine Auswirkungen auf das Ausmaß der Ähnlichkeiten zwischen zwei Themen hat.

[5] Vgl. Abschnitt 4.2.2.
[6] Vgl. Abschnitt 4.5.
[7] Vgl. dazu auch die Abschnitte 4.2.1 und 4.6.

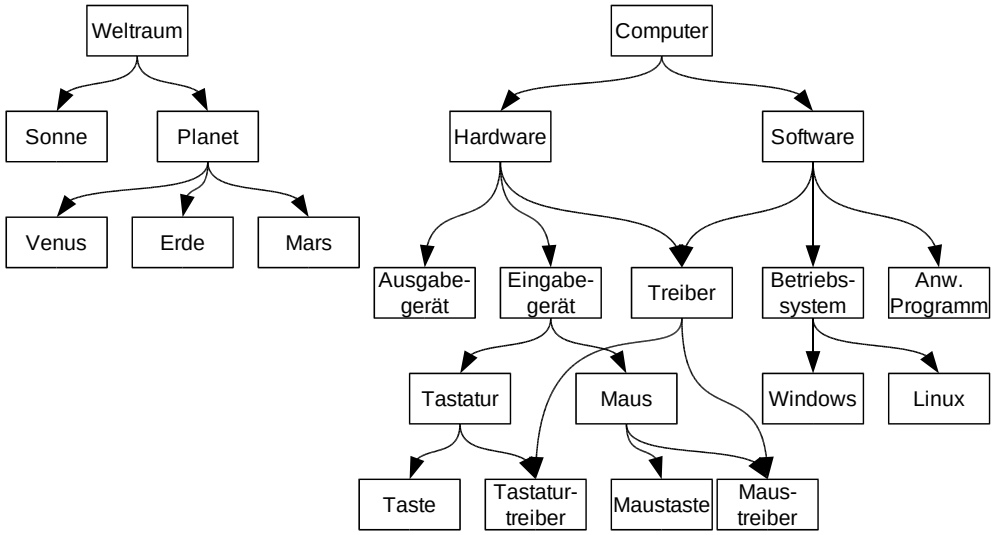


Abbildung 5.4: Ein Beispiel für eine Themenstruktur.

Jedes Thema kann beliebig viele Über- bzw. Unterthemen haben, das heißt, dass Themen keine strikte Hierarchie, sondern eine Struktur bilden. Aus diesem Grunde wird die Themenstruktur im Datenmodell durch einen rekursiven Relationshiptypen Themenstruktur, der den Entitytyp Thema mit sich selbst verbindet und der keine beschränkte Maximalkardinalität hat, dargestellt (vgl. dazu Abbildung 5.5). Diese Struktur muss jedoch folgendes Kriterium erfüllen, dass in einem ERM nicht explizit darstellbar ist: Sie muss frei von Zykeln sein.

Wie aus Abbildung 5.4 ersichtlich, kann die Themenstruktur einzelne Blöcke bilden, die miteinander nicht verbunden sind. Dieses ist als eine thematische Unabhängigkeit der verschiedenen Themenblöcke zu interpretieren. In der Abbildung wird demnach ausgedrückt, dass die Themen im Komplex τ_{Weltraum} thematisch vollkommen unabhängig sind von den The-

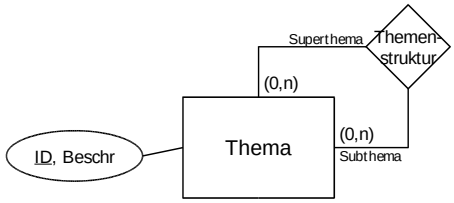


Abbildung 5.5: Themen und ihre rekursive Strukturbeziehung.

men in dem Themenkomplex τ_{Computer}. Thematische Unabhängigkeit bedeutet dabei, dass die Ähnlichkeit zwischen dem Thema eines Themenkomplexes zum Thema des anderen Themenkomplexes gleich Null ist. Allgemein ist es das Ziel, derartige Repräsentationen in Ähnlichkeiten zwischen Themen umzuwandeln, die den zuvor genannten Anforderungen (Normierung, Symmetrie, Maximalität und schwache Transitivität) einer Repräsentation im Vektorraum gerecht werden. Aus der Struktur in Abbildung 5.4 können bezüglich der paarweisen Ähnlichkeit von Themen folgende sinnvolle Aussagen abgeleitet werden: τ_{Venus} und τ_{Erde} haben eine hohe Ähnlichkeit zueinander. Des Weiteren sollte man erwarten, dass τ_{Venus} und τ_{Planet} eine noch höhere Ähnlichkeit zueinander haben als τ_{Venus} und τ_{Erde}, weil τ_{Venus} ein τ_{Planet} ist, während es gleichzeitig ein Objekt ist, dass zur τ_{Erde} verschieden ist. Da τ_{Planet} ein Subthema zu τ_{Weltraum} ist, ist zu erwarten, dass τ_{Mars} als Unterthema zu τ_{Planet} ebenfalls eine Ähnlichkeit zu τ_{Weltraum} aufweist, die allerdings geringer ist als die Ähnlichkeit zu τ_{Planet}, weil diese transitiv hergeleitet werden muss. Aus demselben Grund ist auch davon auszugehen, dass τ_{Mars} eine geringe Ähnlichkeit zur τ_{Sonne} hat, weil diese „über zwei Ecken“ (τ_{Planet} und τ_{Weltraum}) zueinander in Beziehung stehen. Wie man aus dieser Argumentation erkennt, hat die Richtung der Pfeile keine Auswirkung darauf, ob zwei Themen zueinander eine Ähnlichkeit aufweisen oder nicht. Vielmehr gilt folgendes:

> Wenn zwei Themen (mindestens) ein gemeinsames direktes oder indirektes *Superthema* haben, dann haben sie auch eine von Null verschiedene Ähnlichkeit.

Während in Abbildung 5.4 der Themenblock τ_{Weltraum} rein hierarchisch strukturiert ist, existieren beim Themenblock τ_{Computer} Themen mit mehreren Superthemen: z. B. τ_{Treiber} und $\tau_{\text{Maustreiber}}$. Im Fall von τ_{Treiber} bedeutet dies, dass das Thema τ_{Treiber} mit den beiden Themen τ_{Hardware} und τ_{Software} thematisch verwandt ist. Somit ist τ_{Treiber} zum Thema τ_{Software} ähnlicher als beispielsweise $\tau_{\text{Eingabegerät}}$ oder $\tau_{\text{Ausgabegerät}}$. Allerdings ist $\tau_{\text{Eingabegerät}}$ dem Thema τ_{Software} ebenfalls näher als $\tau_{\text{Ausgabegerät}}$, weil τ_{Software} über mehrere Stufen, indirekt über $\tau_{\text{Maustreiber}}$ und $\tau_{\text{Tastaturtreiber}}$, in das Thema $\tau_{\text{Eingabegerät}}$ eingeht. Generell kann folgende verallgemeinerte Aussage getroffen werden:

> Wenn zwei Themen (mindestens) ein gemeinsames direktes oder indirektes *Subthema* haben, dann haben sie auch eine von Null verschiedene Ähnlichkeit.

An dieser Stelle sei noch erwähnt, dass die Beispiele Maustreiber und Tastaturtreiber auch Beispiele zur Repräsentation von Kompositionen von Wörtern im eTVSM darstellen. Kompositionen können im eTVSM repräsentiert werden, indem die Komposita als eigene Themen repräsentiert werden, die ein Subthema zu den Themen sind, die die einzelnen Wörter, aus denen die Komposita bestehen, repräsentieren. Ähnliches gilt für Derivationen, die sich als Subthemen zu den Themen ihrer Worte, von denen sie abgeleitet werden, darstellen lassen. Zusätzlich sind bei den Derivationen noch weitere Themen als Superthemen zu definieren, falls bei der Derivation eine größere Interpretationsverschiebung entsteht. So sollte beispielsweise die Derivation zwergenhaft sowohl Zwerg als auch klein als Superthemen haben.

	Weltraum	Sonne	Planet	Venus	Erde	Mars	Computer	Hardware	Ausgabegerät	Eingabegerät	Tastatur
Weltraum	1,000	0,855	0,855	0,754	0,754	0,754	0,000	0,000	0,000	0,000	0,000
Sonne	0,855	1,000	0,463	0,408	0,408	0,408	0,000	0,000	0,000	0,000	0,000
Planet	0,855	0,463	1,000	0,882	0,882	0,882	0,000	0,000	0,000	0,000	0,000
Venus	0,754	0,408	0,882	1,000	0,667	0,667	0,000	0,000	0,000	0,000	0,000
Erde	0,754	0,408	0,882	0,667	1,000	0,667	0,000	0,000	0,000	0,000	0,000
Mars	0,754	0,408	0,882	0,667	0,667	1,000	0,000	0,000	0,000	0,000	0,000
Computer	0,000	0,000	0,000	0,000	0,000	0,000	1,000	0,931	0,659	0,885	0,815
Hardware	0,000	0,000	0,000	0,000	0,000	0,000	0,931	1,000	0,774	0,947	0,872
Ausgabegerät	0,000	0,000	0,000	0,000	0,000	0,000	0,659	0,774	1,000	0,565	0,520
Eingabegerät	0,000	0,000	0,000	0,000	0,000	0,000	0,885	0,947	0,565	1,000	0,920
Tastatur	0,000	0,000	0,000	0,000	0,000	0,000	0,815	0,872	0,520	0,920	1,000
Taste	0,000	0,000	0,000	0,000	0,000	0,000	0,652	0,753	0,516	0,826	0,915
Tastaturtreiber	0,000	0,000	0,000	0,000	0,000	0,000	0,839	0,844	0,436	0,860	0,915
Maus	0,000	0,000	0,000	0,000	0,000	0,000	0,815	0,872	0,520	0,920	0,695
Maustaste	0,000	0,000	0,000	0,000	0,000	0,000	0,652	0,753	0,516	0,826	0,605
Maustreiber	0,000	0,000	0,000	0,000	0,000	0,000	0,839	0,844	0,436	0,860	0,667
Treiber	0,000	0,000	0,000	0,000	0,000	0,000	0,906	0,912	0,471	0,928	0,855
Software	0,000	0,000	0,000	0,000	0,000	0,000	0,931	0,733	0,453	0,700	0,645
Betriebssystem	0,000	0,000	0,000	0,000	0,000	0,000	0,678	0,428	0,309	0,382	0,351
Windows	0,000	0,000	0,000	0,000	0,000	0,000	0,635	0,400	0,289	0,357	0,329
Linux	0,000	0,000	0,000	0,000	0,000	0,000	0,635	0,400	0,289	0,357	0,329
Anw. Programm	0,000	0,000	0,000	0,000	0,000	0,000	0,704	0,462	0,333	0,412	0,379

	Taste	Tastaturtreiber	Maus	Maustaste	Maustreiber	Treiber	Software	Betriebssystem	Windows	Linux	Anw. Programm
Weltraum	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
Sonne	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
Planet	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
Venus	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
Erde	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
Mars	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
Computer	0,652	0,839	0,815	0,652	0,839	0,906	0,931	0,678	0,635	0,635	0,704
Hardware	0,753	0,844	0,872	0,753	0,844	0,912	0,733	0,428	0,400	0,400	0,462
Ausgabegerät	0,516	0,436	0,520	0,516	0,436	0,471	0,453	0,309	0,289	0,289	0,333
Eingabegerät	0,826	0,860	0,920	0,826	0,860	0,928	0,700	0,382	0,357	0,357	0,412
Tastatur	0,915	0,915	0,695	0,605	0,667	0,855	0,645	0,351	0,329	0,329	0,379
Taste	1,000	0,676	0,605	0,600	0,507	0,639	0,462	0,239	0,224	0,224	0,258
Tastaturtreiber	0,676	1,000	0,667	0,507	0,714	0,926	0,718	0,404	0,378	0,378	0,436
Maus	0,605	0,667	1,000	0,915	0,915	0,855	0,645	0,351	0,329	0,329	0,379
Maustaste	0,600	0,507	0,915	1,000	0,676	0,639	0,462	0,239	0,224	0,224	0,258
Maustreiber	0,507	0,714	0,915	0,676	1,000	0,926	0,718	0,404	0,378	0,378	0,436
Treiber	0,639	0,926	0,855	0,639	0,926	1,000	0,776	0,436	0,408	0,408	0,471
Software	0,462	0,718	0,645	0,462	0,718	0,776	1,000	0,835	0,781	0,781	0,849
Betriebssystem	0,239	0,404	0,351	0,239	0,404	0,436	0,835	1,000	0,935	0,935	0,617
Windows	0,224	0,378	0,329	0,224	0,378	0,408	0,781	0,935	1,000	0,750	0,577
Linux	0,224	0,378	0,329	0,224	0,378	0,408	0,781	0,935	0,750	1,000	0,577
Anw. Programm	0,258	0,436	0,379	0,258	0,436	0,471	0,849	0,617	0,577	0,577	1,000

Tabelle 5.1: Ähnlichkeitsmatrix zur Themenstruktur aus Abbildung 5.4.

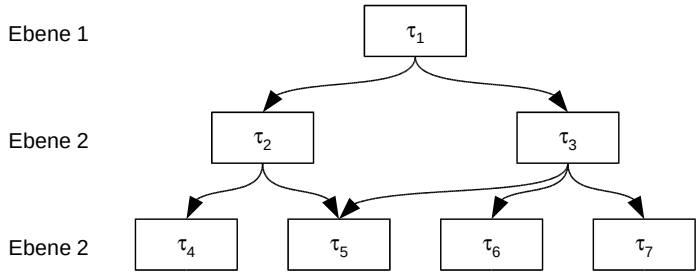


Abbildung 5.6: Ein abstraktes Beispiel für eine Themenstruktur.

5.1.1.3 Herleitung der Themen-Ähnlichkeiten

Im Folgenden wird eine Heuristik vorgestellt, mit deren Hilfe es möglich ist, paarweise Ähnlichkeiten zwischen Themen aus einer Themenstruktur abzuleiten. Die Heuristik ist derart konzipiert, so dass die in Abschnitt 5.1.1.1 genannten Konsistenzbedingungen erfüllt werden. Tabelle 5.1 zeigt die mit dem gleich vorgestellten Verfahren hergeleitete Ähnlichkeitsmatrix zu der Themenstruktur aus der Abbildung 5.4. Zur besseren Illustration wird die Heuristik an der relativ einfachen und beispielhaften Themenstruktur aus Abbildung 5.6 erläutert. Um eine übersichtlichere Schreibweise zu ermöglichen, ist das Beispiel abstrakt gehalten. Die einzelnen Themen τ_i sind von τ_1 bis τ_7 durchnummeriert.

Gegeben sei $\Theta = \{\tau_1, \tau_2, ..., \tau_{\#\Theta}\}$, die Menge aller Themen (im Beispiel gilt $\Theta = \{\tau_1, ..., \tau_7\}$) und die Superthemenrelation $S(\tau_i) \subseteq (\Theta \setminus \tau_i)$, die für alle $\tau_i \in \Theta$ definiert ist. Diese Relation legt zu jedem Thema τ_i die unmittelbar dazugehörigen übergeordneten Themen fest (also diejenigen Themen, die das Thema zum Subthema haben und genau eine Ebene über dem Thema liegen). Im Beispiel gilt u. a. $S(\tau_1) = \{\}$, $S(\tau_4) = \{\tau_2\}$ und $S(\tau_5) = \{\tau_2, \tau_3\}$. Die Superthemenrelation ist mit dem Relationshiptypen Themenstruktur aus dem Datenmodell identisch.

Aus der Superthemenrelation $S(\tau_i)$ lässt sich die transitive, p-Ebene Superthemenrelation $S^p(\tau_i)$ ableiten. Diese gibt die übergeordneten Themen zu einem Thema τ_i die genau p Ebenen über der Ebene des Themas τ_i liegen. Für die weitere Betrachtung ist allerdings die transitive, unbeschränkte Superthemenrelation $S^*(\tau_i)$ von Bedeutung, die sich aus den $S^p(\tau_i)$ wie folgt ableiten lässt:

$$
\begin{aligned}
S^p(\tau_i) &= S(\tau_i) && \text{für } p = 1 \\
S^p(\tau_i) &= \bigcup_{\tau_k \in S^{p-1}(\tau_i)} S(\tau_k) && \text{für } p > 1 \\
S^*(\tau_i) &= S^1(\tau_i) \cup S^2(\tau_i) \cup S^3(\tau_i) \cup ...
\end{aligned}
$$

Im Beispiel zeigt sich der Sachverhalt wie folgt:

$$S^*(\tau_1) = \{\}$$
$$S^*(\tau_4) = \{\tau_1, \tau_2\}$$
$$S^*(\tau_5) = \{\tau_1, \tau_2, \tau_3\}$$

Neben der Menge aller Themen Θ sei auch die Menge aller *Themenblätter* Θ_B definiert. Themenblätter zeichnen sich dadurch aus, dass sie keine Subthemen haben bzw. dass kein Element existiert, welches ein Themenblatt aus Θ_B zum Superthema hat:

$$\Theta_B = \{\tau_i \in \Theta : \nexists \tau_k \in \Theta \text{ mit } \tau_i \in S(\tau_k)\}$$

Im Beispiel sind folgende Elemente Themenblätter:

$$\Theta_B = \{\tau_4, \tau_5, \tau_6, \tau_7\}$$

Neben der Menge der Themenblätter lässt sich auch die Menge der *Themenknoten* Θ_K, also der Themen, zu denen mindestens ein Subthema existiert, definieren:

$$\Theta_K = \complement\Theta_B = \Theta \setminus \Theta_B$$

Jedem Thema $\tau_i \in \Theta$ ist genau ein Themenvektor $\vec{\tau_i} = (\tau_{i,1}, \tau_{i,2}, ..., \tau_{i,\#\Theta}) \in \mathbb{R}^{\#\Theta}$ zugeordnet. Die Herleitung dieses Themenvektors für ein konkretes Thema hängt davon ab, ob das Thema ein Blatt ist oder nicht. Jedem Themenblatt wird folgender Wert für den Themenvektor zugewiesen:

$$\forall \tau_i \in \Theta_B : \vec{\tau_i} = |(\tau^*_{i,1}, \tau^*_{i,2}, ..., \tau^*_{i,\#\Theta})| \tag{5.1}$$

mit:

$$\tau^*_{i,d} = \begin{cases} 1 & \text{falls } \tau_d \in S^*(\tau_i) \ \vee \ i = d \\ 0 & \text{sonst} \end{cases}$$

In Worten heißt das, dass die Themenvektoren der Themenblätter normiert sind. Des Weiteren sind einzelne Dimensionseinträge eines Themenvektors entweder Null oder sie haben einen definierten, für alle Einträge geltenden positiven Wert größer Null. Ein Dimensionseintrag repräsentiert genau ein Thema. Der Dimensionseintrag eines Themenblattes ist immer genau dann ungleich Null, wenn der Dimensionseintrag das Themenblatt selbst oder wenn der Dimensionseintrag ein (möglicherweise transitives) Superthema des Themenblattes repräsentiert. Für das Beispiel aus Abbildung 5.6 lassen sich für die Themenblätter folgende Themenvektoren herleiten:

$$\tau_4 = |(1;1;0;1;0;0;0)| = (\frac{1}{\sqrt{3}};\frac{1}{\sqrt{3}};0;\frac{1}{\sqrt{3}};0;0;0)$$
$$\tau_5 = |(1;1;1;0;1;0;0)| = (\frac{1}{2};\frac{1}{2};\frac{1}{2};0;\frac{1}{2};0;0)$$
$$\tau_6 = |(1;0;1;0;0;1;0)| = (\frac{1}{\sqrt{3}};0;\frac{1}{\sqrt{3}};0;0;\frac{1}{\sqrt{3}};0)$$
$$\tau_7 = |(1;0;1;0;0;0;1)| = (\frac{1}{\sqrt{3}};0;\frac{1}{\sqrt{3}};0;0;0;\frac{1}{\sqrt{3}})$$

Am Beispiel wird deutlich, dass die Ebenenanordnung der Themen und die Strukturbeziehungen zwischen den Themen im Vektor nicht verloren gehen, sondern dass sich vielmehr die einzelnen Ebenen der Themen in den Vektoren zu den Themenblätter sowie ihre Zugehörigkeit zu Superthemen widerspiegeln:

$$\tau_6 = |(\overbrace{1}^{\text{Ebene 1}};\overbrace{0;1}^{\text{Ebene 2}};\overbrace{0;0;1;0}^{\text{Ebene 3}})|$$

Für Themenknoten ist der Themenvektor als normierte Summe aller direkten Subthemen definiert:

$$\forall \tau_i \in \Theta_K : \vec{\tau_i} = |\sum_{\tau_s \in \Theta:\ \tau_i \in S(\tau_s)} \vec{\tau_s}| \tag{5.2}$$

Die Summe ist deshalb berechenbar, weil die Themenstruktur keine Zykel enthält. Die Idee hinter der Aufsummierung der Subthemenvektoren bei den Themenknoten ist die folgende Annahme: Interpretationen werden beim eTVSM einem (oder mehreren) Thema (Themen) zugeordnet. Ist das Vorkommen von Interpretationen in einem Dokument so, dass in etwa dem Dokument die Themen τ_5, τ_6 und τ_7 gleichermaßen viel zugeordnet werden können, dann kann man verallgemeinert sagen, dass das Dokument das Thema τ_3 behandelt. Am konkreten Beispiel erläutert: Ein Dokument mit einem *gleichmäßigen* und *hohen* Vorkommen von Interpretationen, die den Themen τ_{Windows}, τ_{Linux}, τ_{Mac} zugeordnet sind, behandelt mit hoher Wahrscheinlichkeit das Thema $\tau_{\text{Betriebssysteme}}$ im Allgemeinen und keines der genannten Themen im Speziellen. Im Beispiel aus Abbildung 5.6 haben die Vektoren der Themenknoten folgende Werte:

$$\tau_1 \approx (0,669;\ 0,429;\ 0,495;\ 0,174;\ 0,255;\ 0,120;\ 0,120)$$
$$\tau_2 \approx (0,607;\ 0,607;\ 0,282;\ 0,325;\ 0,282;\ 0;\ 0)$$
$$\tau_3 \approx (0,642;\ 0,194;\ 0,642;\ 0;\ 0,194;\ 0,224;\ 0,224)$$

Die Ähnlichkeit $\text{sim}(\tau_a, \tau_b)$ zwischen zwei Themen τ_a und τ_b wird nun als das Skalarprodukt zwischen den jeweiligen Themenvektoren definiert. Da die Themenvektoren normiert

	1	2	4	5	3	6	7
1	1,000	0,933	0,734	0,924	0,933	0,741	0,741
2	0,933	1,000	0,888	0,888	0,742	0,513	0,513
4	0,734	0,888	1,000	0,577	0,483	0,333	0,333
5	0,924	0,888	0,577	1,000	0,836	0,577	0,577
3	0,933	0,742	0,483	0,836	1,000	0,871	0,871
6	0,741	0,513	0,333	0,577	0,871	1,000	0,667
7	0,741	0,513	0,333	0,577	0,871	0,667	1,000

Tabelle 5.2: Ähnlichkeitsmatrix zur Themenstruktur aus Abbildung 5.6.

sind, entspricht das Skalarprodukt dem Kosinus des Winkels $\omega_{a,b}$ zwischen den Themenvektoren:

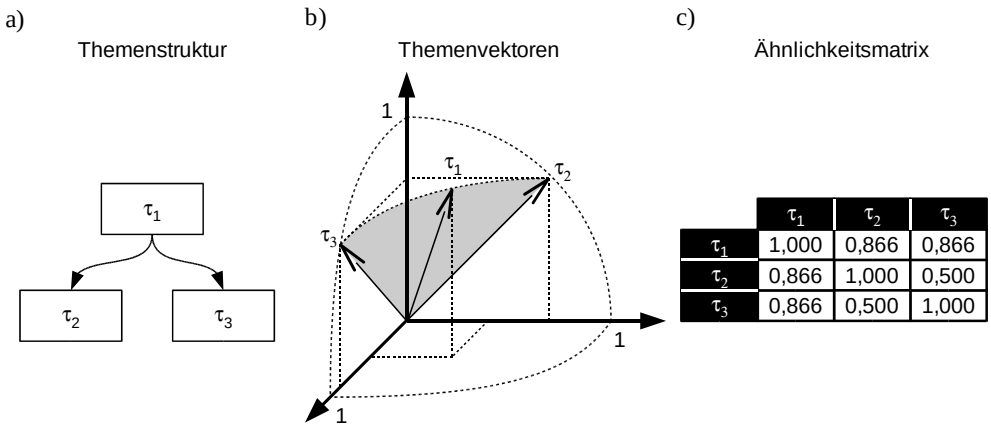
$$\begin{aligned} \text{sim}(\tau_a, \tau_b) &= \vec{\tau_a}\vec{\tau_b} \\ &= \sum_{i=1}^{\#\Theta} \tau_{a,i}\tau_{b,i} \\ &= \cos\omega_{a,b} \end{aligned} \tag{5.3}$$

Tabelle 5.2 zeigt alle paarweisen Kombinationen von Ähnlichkeiten zwischen den Themen der Themenstruktur aus dem Beispiel aus Abbildung 5.6. Da es im Beispiel nur eine zusammenhängende Struktur gibt, gibt es keine Themen, die zueinander unabhängig sind (im Gegensatz zum Beispiel aus Abbildung 5.4 bzw. Tabelle 5.1).

5.1.1.4 Eigenschaften der Vektoren und Ähnlichkeiten

Im Folgenden wird u. a. untersucht, in wie weit die gewonnenen Ähnlichkeiten den Anforderungen aus Abschnitt 5.1.1.1 genügen. Das Kriterium der Normierung wird von den mit der Heuristik gewonnenen Ähnlichkeiten erfüllt, weil die Vektoren der Themenblätter, gemäß Gleichung 5.1, ausschließlich positive Dimensionseinträge haben und somit Elemente eines Vektorraumes mit ausschließlich positiven Achsenabschnitten sind. Somit können die Winkel zwischen jeweils zwei solchen Vektoren nur im Bereich von 0° bis 90° liegen. Gemäß Gleichung 5.2 definieren sich die Vektoren der übrigen Themen aus der Summe der Vektoren der Blätter. Somit gilt auch für diese Vektoren, dass sie Bestandteil eines Raumes mit ausschließlich positiven Achsenabschnitten sind. Da gemäß Gleichung 5.3 die Ähnlichkeit zwischen den Themen dem Kosinus des Winkels der Themenvektoren entspricht, ist sicher gestellt, dass das Normierungkriterium $\text{sim}(\tau_a, \tau_b) \in [0...1]$ erfüllt ist.

Durch die Repräsentation der Themen mit Hilfe von Vektoren und durch die Definition der paarweisen Themenähnlichkeit als Winkel zwischen diesen Vektoren, sind automatisch auch die drei anderen Konsistenzkriterien (Symmetrie, Maximalität und schwache Transitivität) sichergestellt, weil diese die fundamentalen Eigenschaften von Winkeln in einem Vektorraum darstellen.



	τ_1	τ_2	τ_3
τ_1	1,000	0,866	0,866
τ_2	0,866	1,000	0,500
τ_3	0,866	0,500	1,000

Abbildung 5.7: Themenstruktur, Themenvektoren und Themenähnlichkeiten.

Der Gleichung 5.1 kann entnommen werden, dass zwei Themenblätter immer genau dann orthogonal zueinander sind (und somit eine Ähnlichkeit von Null haben), wenn diese kein gemeinsames Superthema haben. Sollten die Themen ein gemeinsames Superthema haben, dann haben die beiden Themen in dem Dimensionseintrag, der dieses Thema repräsentiert, einen von Null verschiedenen Wert, wodurch diese Themen nicht mehr zueinander orthogonal sein können, weil der Vektorraum auf positive Achsenabschnitte eingeschränkt ist. Für Themenknoten gilt die obige Aussage ebenfalls, jedoch mit folgender Einschränkung: Zwei Themenknoten sind zueinander orthogonal, wenn diese kein gemeinsames Superthema haben und wenn diese kein gemeinsames Subthema haben. Der Grund, warum die Einschränkung vorgenommen werden muss, ist der, dass Themenknoten gemäß Gleichung 5.2 aus der Summe ihrer Subthemen bestehen. Haben zwei Themen kein gemeinsames Superthema, aber dafür ein gemeinsames Subthema, dann folgt daraus, dass der Vektor des gemeinsamen Subthemas Bestandteil der Themenvektoren der beiden Themen ist. Somit können diese Themenvektoren zueinander nicht orthogonal sein, weil der Vektorraum auf positive Achsenabschnitte beschränkt ist. Somit werden die in Abschnitt 5.1.1.2 gemachten Aussagen bzw. Anforderungen zur Unabhängigkeit von Themen erfüllt.

Die genannte Eigenschaft zur Unabhängigkeit von Themen kann sinnvoll genutzt werden, um die Berechnung der Ähnlichkeiten bzw. Themenvektoren zu beschleunigen. Da Themen nur dann zueinander eine Ähnlichkeit größer als Null haben können, wenn sie gemeinsam Bestandteil eines zusammenhängenden Themenkomplexes sind, ist es ausreichend, bei der Berechnung von Ähnlichkeiten immer nur einen zusammenhängenden Themenkomplex zu betrachten. Themen, die aus zwei verschiedenen Themenkomplexen stammen, haben immer eine paarweise Ähnlichkeit von Null.

Eine weitere Eigenschaft der vorgestellten Heuristik ist, dass diese die Anzahl der Dimensionen gegenüber dem VSM reduziert und dass diese gegenüber dem VSM zusätzliche

Einschränkungen bezüglich des Bereiches macht, in dem sich Dokumentenvektoren „aufhalten“ können. Wie aus Abbildung 5.7 ersichtlich, bilden die Vektoren der Themenblätter einen Subraum (vgl. die schraffierte Fläche im Abbildungsteil b), in dem die Vektoren der Themenknoten liegen, weil diese gemäß Gleichung 5.2 eine Linearkombination der Blattvektoren sind. Da Interpretationen und Dokumente, wie in Abschnitt 5.1.2 noch gezeigt wird, ebenfalls eine Linearkombination aus den Themenvektoren sind, liegen deren Vektoren auch in dem Subraum. Somit werden Themen, Interpretationen und Dokumente zwar als Vektoren in einem $\#\Theta$-dimensionalen Raum repräsentiert, aber die Zahl der Dimensionen des Subraums, in dem die Vektoren liegen können, ist höchstens so groß wie die Anzahl der Blätter ($\#\Theta_B$) in der Themenstruktur. Zusätzlich zu der Reduktion der Dimensionen ist der Bereich, in dem die Vektoren liegen können, eingeschränkt, weil alle Linearkombinationen[8] Vektoren nur additiv zusammensetzen und weil die Blattvektoren selbst nur positive Dimensionseinträge haben und zueinander nicht notwendigerweise orthogonal sind (vgl. z. B. die Ähnlichkeit zwischen τ_2 und τ_3 in Abbildung 5.7c: $\text{sim}(\tau_2, \tau_3) = 0,5$ bedeutet, dass der Winkel $\omega_{2,3} = \cos^{-1}(0,5) = 60°$ groß ist). Somit stehen für die Vektoren diejenigen Punkte im Raum *nicht* zur Verfügung, die nur zu erreichen sind, indem mindestens einer der Blattvektoren der Themen subtraktiv in die Linearkombination eingeht.

Die Modellierung von Themenstrukturen ist nicht immer trivial. Es gibt Fälle, in denen der genaue Aufbau der Struktur von dem Standpunkt des Betrachters abhängt. Diese Problematik und ihre Auswirkungen auf die Ähnlichkeitsmatrix wird im Folgenden an dem Beispiel aus Abbildung 5.8 mit den drei Themen Schnee, Eis und Wasser verdeutlicht. Eine Möglichkeit, die drei genannten Themen zu strukturieren, ist die Strukturierung der Themen gemäß ihrer *ist-ein* Beziehung (Klassifikation). Das Ergebnis dieser Strukturierung ist in Abbildung 5.8a zu sehen: Eis und Schnee sind Subthemen von Wasser. In der Ähnlichkeitsmatrix kann man erkennen, dass in diesem Fall die Subthemen Eis und Schnee eine deutlich höhere Ähnlichkeit zu ihrem Superthema Wasser aufweisen als zueinander (0,866 im Vergleich zu 0,500). Des Weiteren gilt für diesen Fall, dass die Vektoren der Subthemen Eis und Schnee sich zu dem Vektor des Superthemas Wasser aufaddieren, wodurch die Ähnlichkeit zwischen der Vektorsumme und dem Vektor des Superthemas bei dem maximalen Wert von Eins liegt.

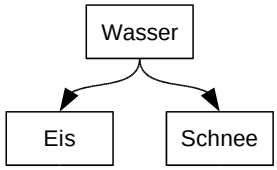
Es sind Fälle denkbar, in denen diese zuletzt genannte Eigenschaft nicht erwünscht ist, weil z. B. die Themenstruktur bewusst – aus Gründen der Komplexitätsreduktion – nicht vollständig modelliert werden soll. In diesem Fall kann ein Dummy-Subthema, wie in Abbildung 5.8b gezeigt, eingeführt werden. Aus der Matrix kann man erkennen, dass diese Modifikation keinen Einfluss auf die paarweise Ähnlichkeiten zwischen den Subthemen hat. Allerdings nimmt die Ähnlichkeit zwischen dem Superthema und den Subthemen ab.

Eine andere Sicht auf die Themenstruktur nehmen die drei Abbildungen 5.8c bis e ein: In diesen Abbildungen werden die drei Themen Wasser, Eis und Schnee gemäß ihrer besteht-aus Beziehung modelliert. Naturgemäß ergeben sich daraus gewisse Änderungen an den Ähnlichkeiten zwischen den Themen. Abbildung 5.8c modelliert den Sachverhalt in einer aus intuitiver Sicht korrekten, aber für die Anwendung der Heuristik nicht geeigneten Weise: Eis

[8] Gemeint sind sowohl die Vektoren der Themen, die keine Blätter sind, als auch die Vektoren von Interpretationen und Dokumenten.

Themenstrukturen | Ähnlichkeitsmatrizen

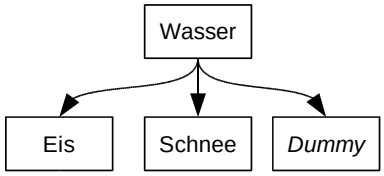
a)



	Wasser	Eis	Schnee
Wasser	1,000	0,866	0,866
Eis	0,866	1,000	0,500
Schnee	0,866	0,500	1,000

sim(Eis+Schnee, Wasser) = 1,000

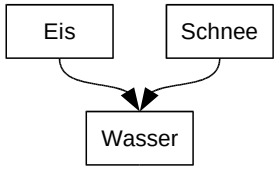
b)



	Wasser	Eis	Schnee
Wasser	1,000	0,816	0,816
Eis	0,816	1,000	0,500
Schnee	0,816	0,500	1,000

sim(Eis+Schnee, Wasser) = 0,943

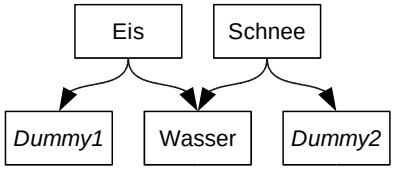
c)



	Wasser	Eis	Schnee
Wasser	1,000	1,000	1,000
Eis	1,000	1,000	1,000
Schnee	1,000	1,000	1,000

sim(Eis+Schnee, Wasser) = 1,000

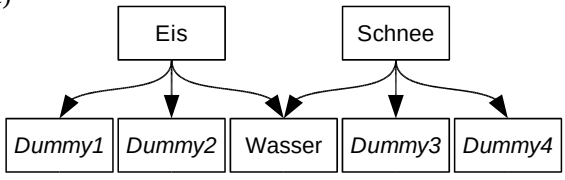
d)



	Wasser	Eis	Schnee
Wasser	1,000	0,839	0,839
Eis	0,839	1,000	0,644
Schnee	0,839	0,644	1,000

sim(Eis+Schnee, Wasser) = 0,925

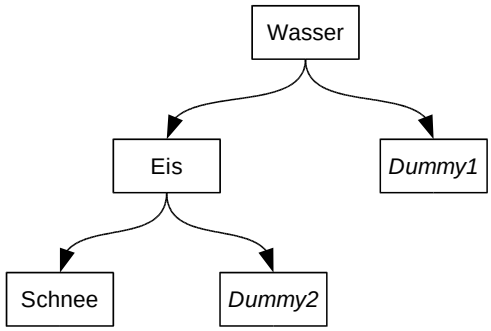
e)



	Wasser	Eis	Schnee
Wasser	1,000	0,765	0,765
Eis	0,765	1,000	0,467
Schnee	0,765	0,467	1,000

sim(Eis+Schnee, Wasser) = 0,894

Abbildung 5.8: Ähnlichkeitsmatrizen verschiedener Themenstrukturen.



	Wasser	Eis	Schnee
Wasser	1,000	0,851	0,777
Eis	0,851	1,000	0,913
Schnee	0,777	0,913	1,000

sim(Eis+Schnee, Wasser) = 0,832

Abbildung 5.9: Struktur und Ähnlichkeitsmatrix von Wasser, Eis und Schnee.

und Schnee werden als Superthemen zum Subthema Wasser definiert. Bedingt durch Gleichung 5.2 auf Seite 122 führt dieses jedoch dazu, dass alle drei Themen denselben Vektor zugewiesen bekommen und dass alle drei Themen somit zueinander die maximale paarweise Ähnlichkeit von Eins haben.

Der Grund, warum die Themenstruktur aus Abbildung 5.8c zur Modellierung des Sachverhaltes ungeeignet ist, ist der, dass der Sachverhalt in diesem Modell nicht vollständig abgebildet ist. Zwar bestehen Eis und Schnee aus Wasser, aber es sind noch andere Bedingungen für die Bildung von Eis und Schnee erforderlich (wie z. B. eine tiefe Temperatur). Zudem unterscheiden sich Eis und Schnee durch ihre kristalline Struktur voneinander, was in Abbildung 5.8c nicht zur Geltung kommt. Eine Möglichkeit diesen Sachverhalt im eTVSM abzubilden, ohne gleich alle Details umsetzten zu müssen, zeigt die Abbildung 5.8d. Bei dieser Abbildung wurden die beiden Superthemen in das Subthema Wasser und in jeweils ein Dummy-Subthema spezialisiert. Das Ergebnis ist eine verwendbare Ähnlichkeitsmatrix. Interessanter Weise ergibt die Strukturierung aus Abbildung 5.8d eine Ähnlichkeitsmatrix die Ähnlich zur Matrix aus Abbildung 5.8a bzw. b strukturiert ist. Auch bei dieser Matrix sind Eis und Schnee ähnlicher zu Wasser als zueinander.

Der Vergleich der beiden Abbildungen 5.8d und e verdeutlicht, dass die Ähnlichkeit zwischen zwei Superthemen davon abhängt, wie hoch der relative Anteil der gemeinsamen Subthemen ist. Da die beiden Superthemen in Abbildung 5.8e nur ein Subthema (Wasser) von insgesamt jeweils drei möglichen Subthemen gemeinsam haben, ist die paarweise Ähnlichkeit der beiden Superthemen Eis und Schnee geringer als in Abbildung 5.8d.

Bei genauerer Betrachtung stellt man fest, dass die Themenstrukturen für Wasser, Eis und Schnee in der Abbildung 5.8 den Sachverhalt immer noch nicht in geeigneter Weise wiedergeben, was sich auch darin widerspiegelt, dass die Ähnlichkeiten der Themen nicht dem intuitiven Empfinden entspricht. So ist intuitiv nicht nachvollziehbar, warum Eis und Schnee zueinander weniger ähnlich sein sollen als Eis und Wasser. Abbildung 5.9 zeigt eine Strukturierung der Themen, die den Sachverhalt besser widerspiegelt und deren Ähnlichkeitsmatrix eher dem intuitiven Empfinden entspricht. In der Abbildung ist Eis als Subthema zu Wasser

modelliert worden, weil Eis ein besonderer Aggregatzustand und somit ein Spezialfall von Wasser ist. *Dummy1* ist eingeführt worden, weil Wasser auch noch andere Aggregatzustände hat (wie z. B. Dampf), die aber aus Gründen der Komplexitätsreduktion an dieser Stelle nicht explizit abgebildet werden. Schnee ist in Abbildung 5.9 als Subthema zu Eis modelliert worden, weil Schnee Eis mit einer bestimmten kristallinen Struktur ist und somit einen Spezialfall von Eis darstellt. Auch hier wurde von anderen Spezialfällen von Eis[9] abstrahiert, die durch *Dummy2* repräsentiert werden.

Wie man der Ähnlichkeitsmatrix aus Abbildung 5.9 entnehmen kann, haben bei dieser Modellierung Eis und Schnee die größte Ähnlichkeit, gefolgt von der Kombination Wasser und Eis. Die geringste, aber immer noch eine relativ hohe Ähnlichkeit haben Wasser und Schnee.

5.1.2 Interpretationen und ihre Beziehungen

Unter einer Interpretation wird beim eTVSM die Bedeutung eines Terms im Sinne der lexikalischen Semantik[10] verstanden. Daher sind Interpretationen der zentrale Bestandteil des eTVSM, weshalb sie auch im Datenmodell eine zentrale und tragende Rolle spielen. Abbildung 5.10 zeigt den Entitytyp Interpretation und seine Beziehungen zu benachbarten Entitytypen. Wie aus der Abbildung ersichtlich geht der Entitytyp Interpretation in fünf verschiedene Relationshiptypen ein, die nun erläutert werden:

Der Relationshiptyp TI-Zuo dient zusammen mit der Zuordnung Supportterm der Repräsentation von Synonymen, Homographen und der Metonymie. Wie dem ERM in Abbildung 5.10 entnommen werden kann, ist jedem Term im eTVSM mindestens eine Interpretation zugeordnet. Andererseits können einer Interpretation keine oder beliebig viele Terme zugeordnet werden. Die Supportterm-Zuordnung dient dazu, dem System einen Hinweis zu geben, für welche Interpretation sich das System entscheiden soll, falls einem Term mehrere Interpretationen zugeordnet sind (Disambiguierung). Der Relationshiptyp IT-Zuo ordnet jeder Interpretation mindestens ein oder mehrere Themen zu. Umgekehrt können mehrere Themen auch keiner, einer oder mehreren Interpretationen zugeordnet sein. Zusätzlich werden anhand dieser Zuordnung die Winkel und Skalarprodukte zwischen den Interpretationen auf Basis der Themenvektoren bzw. der paarweisen Themenähnlichkeiten[11] berechnet. Auf das Zusammenspiel der drei Zuordnungen und die Berechnung der paarweisen Interpretationsähnlichkeiten wird in den folgenden Abschnitten im Detail eingegangen.

Die beiden letzten Zuordnungen sind in Abbildung 5.10 gestrichelt eingezeichnet, weil diese redundant sind. Sie werden aus Gründen einer höheren Berechnungsgeschwindigkeit und zur Gewährleistung der relationalen Berechenbarkeit[12] der paarweisen Dokumentenähnlichkeiten benötigt. Da der Aufwand für die Bestimmung der paarweisen Interpretationsähn-

[9] Z. B. Kunstschnee, der ja bekanntlich eine andere kristalline, mehr kompakte Struktur hat als Naturschnee, der tendenziell eher flockige Kristalle hat.

[10] Vgl. Abschnitt 2.3.4.2.

[11] Vgl. Abschnitt 5.1.1.3.

[12] Mit *relationaler Berechenbarkeit* ist gemeint, dass die Berechnung der Dokumentenähnlichkeiten mit einer relationalen Sprache, wie z. B. SQL (vgl. Abschnitt 2.2.2) durchgeführt werden kann.

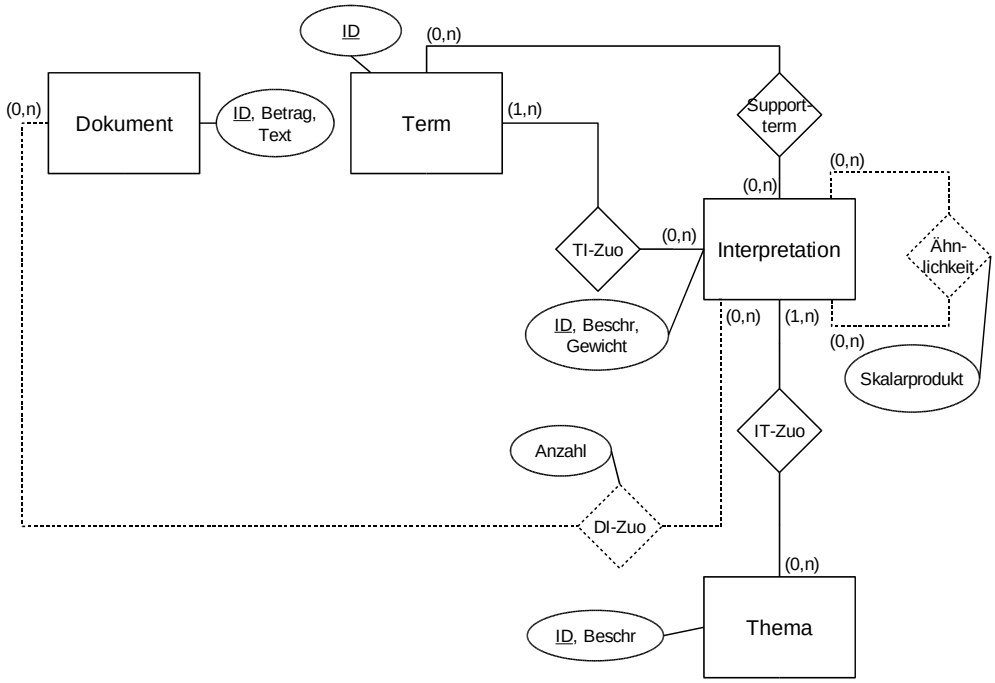


Abbildung 5.10: Die Term-Interpretation- und Supportterm-Zuordnung.

lichkeiten in Form von Skalarprodukten zwischen Interpretationen nicht zu unterschätzen ist, werden die Skalarprodukte in dem Relationshiptypen Ähnlichkeit gespeichert. Auch das Parsing (inkl. Stemming, Disambiguierung und dem Erkennen von Wortgruppen) von Dokumenten ist aufwändig, daher wird das Ergebnis des Parsing in dem Relationshiptyp DI-Zuo, inklusive der Anzahl der Vorkommen einer bestimmten Interpretation, in einem Dokument gespeichert. Abschnitt 5.1.2.6 zeigt, wie mit Hilfe dieser beiden Relationshiptypen die Ähnlichkeit zwischen Dokumenten berechnet wird.

5.1.2.1 Herleitung der Interpretations-Ähnlichkeiten

Für eine einfachere Formalisierung werden die für die Berechnung der paarweisen Ähnlichkeiten zwischen Interpretationen benötigten Entitytypen und Relationshiptypen des Datenmodells durch folgende Symbole dargestellt: Φ ist die Menge aller Interpretationen $\phi_i \in \Phi$, die dem Entitytyp Interpretation äquivalent ist. Die Funktion $g(\phi_i) \in [0...1]$ liefert zu jeder Interpretation ϕ_i den dazugehörigen Wert des Attributs Gewicht im Entitytyp Interpretation. Der Relationshiptyp IT-Zuo wird im mathematischen Modell durch die Relation $T(\phi_i) \in \wp(\Theta) \setminus \{\}$ repräsentiert. Der Vektor einer Interpretation $\vec{\phi_i} = (\phi_{i,1}, \phi_{i,2}, ..., \phi_{i,\#\Theta})$ berechnet

sich als die normierte und mit dem Gewicht der Interpretation multiplizierte Summe über alle Themen, die dieser Interpretation zugeordnet sind:

$$\vec{\phi}_i = \frac{g(\phi_i)}{|\sum_{\tau_k \in T(\phi_i)} \vec{\tau}_k|} \cdot \sum_{\tau_k \in T(\phi_i)} \vec{\tau}_k \tag{5.4}$$

Das Skalarprodukt $\vec{\phi}_i \vec{\phi}_k$ und der Winkel $\omega_{i,k}$ zwischen jeweils zwei Interpretationen lassen sich somit wie folgt berechnen:

$$\vec{\phi}_i \vec{\phi}_k = \sum_{n=1}^{\#\Theta} \phi_{i,n} \phi_{k,n} \tag{5.5}$$

$$\omega_{i,k} = \cos^{-1} \frac{\vec{\phi}_i \vec{\phi}_k}{g(\phi_i) g(\phi_k)}$$

Um eine performante Berechnung von Dokumentenähnlichkeiten sicherzustellen, werden die Skalarprodukte – wie beim TVSM – im Attribut Skalarprodukt des Relationshiptypen Ähnlichkeit für alle paarweisen Kombinationen aus Interpretationen, bei denen das Skalarprodukt größer als Null ist, hinterlegt.

Ein Nachteil des Speicherns der Skalarprodukte zu den Interpretationen ist, dass die Skalarprodukte bei einer Modifikation der Themenstruktur oder bei einer Modifikation von Gewichten neu berechnet werden müssen. Allerdings kann diese Neu-Berechnung von Skalarprodukten selektiv vorgenommen werden. Es brauchen nur diejenigen Skalarprodukte zu Interpretationskombinationen neu berechnet zu werden, bei denen eine der beiden Interpretationen mit dem modifizierten Thema bzw. den Themen der modifizierten Interpretation ein gemeinsames Sub- oder Superthema haben. Diese Selektionsmöglichkeit begründet sich in den in Abschnitt 5.1.1.4 beschriebenen Eigenschaften der Themenvektoren bzw. -ähnlichkeiten.

Einschränkend sollte jedoch erwähnt werden, dass Modifikationen von Interpretationen oder Themenstrukturen von einem größeren Umfang in einem eingespielten System eher seltener stattfinden sollten. Des Weiteren kann die Durchführung der Modifikationen bei einer Datenbank-basierten Implementierung, wie sie hier vorgeschlagen wird, in Form einer Transaktion im Hintergrund, parallel zum normalen Betrieb, durchgeführt werden.[13] Da auf der Datenbank nur sehr wenige Schreiboperationen (Einstellen von neuen Dokumenten und evtl. die Änderung der Themenstrukturen) stattfinden, ist die Wahrscheinlichkeit von *Deadlocks*[14] eher gering. Zur Sicherheit erscheint es sinnvoll, das Einstellen neuer Dokumente während der Anpassung von Themenstrukturen zu stoppen.

[13] Der Datenbank ist dafür natürlich ein hinreichend großer Transaktionspuffer bzw. ein hinreichend großer Bereich für die Log-Daten zur Verfügung zu stellen, damit die neu berechneten Skalarprodukte bis zum Abschluss der Transaktion zwischengespeichert werden können.

[14] Ein *Deadlock* ist eine Situation, bei der sich mehrere Transaktionen gegenseitig sperren, so dass keine von ihnen ausgeführt werden kann. Zu formalen Details von Deadlocks vgl. z. B. VOSSEN [155, S. 557f].

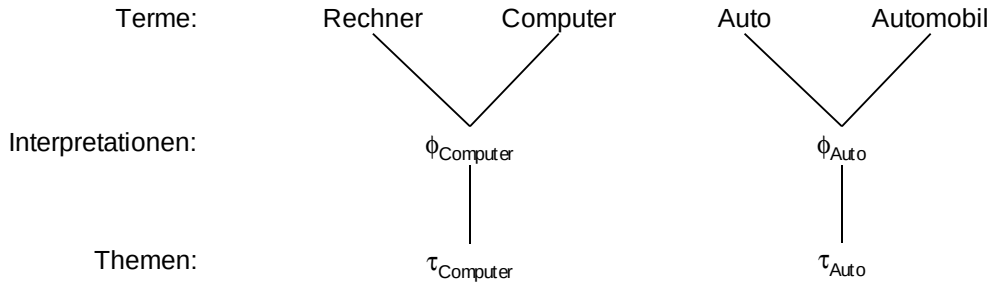


Abbildung 5.11: Beispiele für synonyme Terme und ihre Zuordnungen.

5.1.2.2 Repräsentation der (Totalen) Synonymie

Synonyme sind Terme (also Worte oder auch Wortgruppen), die dieselbe Interpretation und dasselbe Thema haben. Beispiele für derartige Terme sind:

1. Auto und Automobil
2. Rechner und Computer
3. VSM und Vector Space Model

Synonyme werden im eTVSM repräsentiert, indem sich mehrere Terme eine Interpretation teilen, die genau einem Thema zugeordnet ist, das heißt: Mehrere Terme wie z. B. Rechner und Computer werden einer Interpretation, z. B. der Interpretation $\phi_{Computer}$, zugeordnet. Die Interpretation $\phi_{Computer}$ ist dabei ihrerseits dem Thema $\tau_{Computer}$ zugeordnet (vgl. dazu auch Abbildung 5.11), wodurch die Synonyme denselben Vektor zugewiesen bekommen und somit zueinander eine maximale Ähnlichkeit von Eins (bzw. einen Winkel von $0°$) haben.

Eine Repräsentation von Synonymen ist für ein IF-/IR-System deshalb von Bedeutung, weil Synonyme eine häufige Ursache für Fehler beim Filtering bzw. Retrieval sind. Der Grund dafür ist, dass insbesondere in kürzeren Dokumenten mehrere synonyme Terme zu einer Interpretation nur relativ selten gleichzeitig vorkommen.[15] Somit ist die Wahrscheinlichkeit recht groß, dass ein IR-System, welches Synonymie nicht berücksichtigt, auf Anfragen, wie z. B. nach Auto, keine Dokumente liefert, die nur den Term Automobil enthalten. Synonyme stellen insbesondere für unerfahrene Anwender von IR-Systemen eine Hürde dar, weil diese sich dieser Problemstellung häufig nicht bewusst sind.

[15] Vgl. dazu auch die Diskussion zur Co-Occurrenz von Termen in Abschnitt 3.3.

5.1.2.3 Repräsentation der Homographie

An dem folgenden Beispiel wird nun das Problem der Homographen erläutert: Der Term Maus ist ein Homograph und kann ohne weiteren Kontextbezug sowohl als $\phi_{\text{Computermaus}}$ (mausförmiges Eingabegerät eines Computers) als auch als $\phi_{\text{(Tier-)Maus}}$ (kleines Nagetier mit der Bezeichnung Maus) interpretiert werden. Wird dieses Phänomen von einem IF-/IR-Modell nicht adäquat repräsentiert, dann ist eine Unterscheidung zwischen den beiden Interpretationen des Terms Maus nicht möglich. Somit wird eine Anfrage nach dem Term dazu führen, dass sowohl Computer- als auch Nagetier-affine Dokumente dem Benutzer präsentiert werden. Es wäre daher sinnvoll, wenn das System den Term Maus als Homographen erkennen und den Benutzer nach der gewünschten Interpretation des Begriffes, vor dem Durchführen einer Suche, fragen würde. Ein ähnliches Problem ergibt sich im übrigen beim IF. Beim Vergleich zweier Dokumente attestiert das System zwei Dokumenten eine hohe Ähnlichkeit, wenn in beiden Dokumenten der Term Maus vorkommt. Es kann aber sein, dass in dem einen Dokument die Maus als $\phi_{\text{Computermaus}}$ und in dem anderen als $\phi_{\text{(Tier-)Maus}}$ interpretiert wird. In diesem Fall ist die Ähnlichkeit, die das System den beiden Dokumenten zuordnet, zu hoch.

Insbesondere für Benutzer, die mit dem Konzept des IR unerfahren sind, stellen Homographen in der praktischen Anwendung von IR-Systemen ein Problem dar, weil diese im Gegensatz zu den erfahrenen Benutzern häufig nicht genau wissen, wie sie ihre Anfrage geeignet erweitern müssen, um die Suche hinreichend einzuengen. Aber auch erfahrenen Benutzern wird die Suche durch Homographen erschwert, wenn sie sich nicht in allen Themenbereichen der in der Dokumentendatenbank enthaltenen Dokumente auskennen. Es kommt vor, dass verschiedene Fachrichtungen denselben Begriff unterschiedlich interpretieren. In diesem Fall kann nicht ausgeschlossen werden, dass die Anwender in ihrer Suche Homographen verwenden, ohne dass sie sich dessen bewusst sind, das es sich um Homographen handelt, weil ihnen nicht bekannt ist, dass der gesuchte Term in anderen Fachgebieten mit einer anderen als der vom Benutzer intendierten Bedeutung verwendet wird. Das Ergebnis in solchen Situationen ist oft eine frustrierend hohe Anzahl von falschen Treffern.

Um dieses Problem zu lösen, können einem Term im eTVSM mehrere Interpretationen über den Relationshiptyp TI-Zuo zugeordnet werden. Homographen werden beim eTVSM mindestens zwei Interpretationen zugeordnet: jeweils eine Interpretation für jede aus linguistischer Sicht reale (fachgebietbezogene) Interpretation des Homographen. Zusätzlich zu den realen Interpretationen kann auch eine Standard-Interpretation zugewiesen werden, die für alle Interpretationen übergreifend ist und eine Art Notfalllösung für den Fall darstellt, dass die genaue Interpretation eines Terms nicht ermittelt werden kann. Zusätzlich dazu können für reale Interpretationen so genannte *Supportterme* über die Supportterm Zuordnung definiert werden, die im Falle ihres Vorkommens in einem Dokument oder einer Anfrage die Annahme, dass in dem Dokument eine bestimmte Interpretation gemeint ist, stützen.

Für das Beispiel des Homographen Maus bedeutet dies, dass zum Term Maus die folgenden drei Interpretationen zu definieren sind:

1. $\phi_{\text{Computermaus}}$ mit den Supporttermen Tastatur, Computer, Rechner, Taste, ...
2. $\phi_{\text{(Tier-)Maus}}$ mit den Supporttermen Tier, Feldmaus, Nagetier, ...

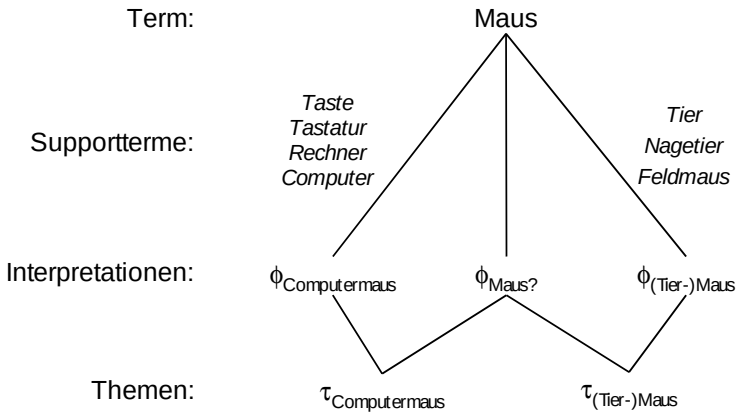


Abbildung 5.12: Der Homograph Maus und seine Zuordnungen.

3. $\phi_{\text{Maus?}}$ als Standard-Interpretation ohne Supportterme.

Die Idee hinter den Supporttermen ist die, dass der Parser bei der Analyse der Dokumente beim Auftreten von Homographen die Dokumente auch auf das Vorhandensein von Supporttermen prüft und mit ihrer Hilfe eine Disambiguierung von Homographen vornimmt. Wenn das Vorhandensein der Supportterme den Schluss nahelegt, dass ein Homograph in einem bestimmten Kontext (im Falle der Maus z. B. als $\phi_{\text{Computermaus}}$) zu interpretieren ist, dann wird der Term Maus in diesem Dokument durch die zu diesem Kontext passende Interpretation repräsentiert (Disambiguierung). Anderenfalls wird diejenige Interpretation gewählt, zu der keine Supportterme definiert wurden (im Beispiel also die Standard-Interpretation $\phi_{\text{Maus?}}$). Das Vorgehen des Parsers wird in Abschnitt 5.3.3 im Detail beschrieben. Die verschiedenen Interpretationen müssen ihrerseits den Themen zugeordnet werden. Dabei ist es sinnvoll, den Standard-Interpretationen eine Kombination aus allen Themen der realen Interpretationen zuzuordnen. Abbildung 5.12 zeigt u. a. die Zuordnungen des Terms Maus zu den Interpretationen und die Zuordnungen der Interpretationen zu den Themen.

Dadurch, dass die verschiedenen Interpretationen eines Homographen durch verschiedene Themen und somit auch verschiedene Vektoren repräsentiert werden, können die oben beschriebenen Probleme mit Homographen in vielen Situationen umgangen werden. Um auch Fälle abdecken zu können, in denen das System sich nicht für eine Interpretation entscheiden kann, weist die Standard-Interpretation in Richtung der Kombination der verschiedenen realen Interpretationsvektoren. Auf das Beispiel des Homographen Maus übertragen heißt das, dass die erste Interpretation den Vektor $\vec{\tau}_{\text{Computermaus}}$, die zweite den Vektor $\vec{\tau}_{\text{(Tier-)Maus}}$ und die dritte die normierte Summe aus den beiden Vektoren $|\vec{\tau}_{\text{Computermaus}} + \vec{\tau}_{\text{(Tier-)Maus}}|$ der beiden Themengebiete zugewiesen bekommt (vgl. Gleichung 5.4 auf Seite 130). Somit ist sichergestellt, dass die Standard-Interpretation in Richtung der beiden Themen $\tau_{\text{Computermaus}}$

und $\tau_{\text{(Tier-)Maus}}$ weist und somit zu beiden Themen eine Ähnlichkeit aufweist, die größer oder gleich $\cos 45° \approx 0,707$ ist.[16]

Ein Problem, welches bei der Repräsentation von Homographen bisher noch nicht hinreichend behandelt wurde, ist die Definition von Supportterme. Da für eine Vielzahl von Homographen Supportterme zu definieren sind, ist eine automatisierte oder zumindest teilautomatisierte Lösung für das Problem wünschenswert. Eine Möglichkeit zur Lösung des Problems ist die Nutzung von vorhandenen Themenstrukturen. Da Themen, die in einer Themenstruktur nahe beieinander liegen, miteinander thematisch verwandt sind, ist es logisch, diese Information zur Definition von Supportterme zu verwenden. Die Abbildung 5.13 zeigt den Sachverhalt für das Beispiel mit dem Homographen Maus und einer Themenstruktur, in der beide Interpretationen des Terms ($\tau_{\text{Computermaus}}$ und $\tau_{\text{(Tier-)Maus}}$) enthalten sind. Die Idee dabei ist die folgende: Themen, die mit einer der Interpretationen des Homographen ein Sub- oder Superthema gemeinsam haben, oder eines der Themen ein Sub- bzw. Superthema zu dem anderen ist, weisen genau dann über ihre Interpretationen auf gute Supportterme hin, wenn sie eine eindeutige Interpretation haben. Demnach weist im Beispiel das Thema τ_{Computer}, das $\tau_{\text{Computermaus}}$ zum Subthema hat, über die Interpretation ϕ_{Computer} auf die beiden synonymen Terme Computer und Rechner hin, die als Supportterme für die Disambiguierung des Homographen Maus auf die Interpretation $\phi_{\text{Computermaus}}$ geeignet sind.

Zu dieser Definition von Supportterme ist allerdings folgende Einschränkung zu treffen: Weist ein Thema (wie z. B. $\tau_{\text{materielles Objekt}}$) auf Supportterme hin, die zu mehreren konkurrierenden Interpretationen zugeordnet sind, dann sind diese Supportterme für beide Interpretationen während der Disambiguierung dieses Falls zu streichen bzw. zu ignorieren. Andere Verfahren zur Disambiguierung von Termen, die evtl. mit dem eTVSM kombiniert werden können, werden u. a. in den folgenden Publikationen beschrieben: SUSSNA [142], AGIRRE und RIGAU [4] sowie LI ET AL. [89].

5.1.2.4 Repräsentation der Partiellen Synonymie

Bei der Diskussion über Synonymie wird bei den gängigen IF- und IR-Modellen zur Repräsentation von natürlichsprachlichen Dokumenten üblicherweise unterstellt, dass Synonymie immer „total“ ist. Das heißt, dass zwei Synonyme in jedem Kontext gegeneinander ausgetauscht werden können und dass es somit ausreicht die beiden synonymen Begriffe durch einen führenden Begriff zu ersetzten. Diese Annahme entspricht jedoch nicht der realen Situation. In der Tat ist es so, dass die meisten synonymen Begriffe nur in bestimmten Kontexten zueinander synonym sind. [28, S. 673f] Solange ein IF- bzw. IR-System Dokumente aus nur einem oder aus einigen wenigen verwandten Kontexten verarbeitet, wird das Problem der Partiellen Synonyme kaum sichtbar. Sobald jedoch Dokumente aus verschiedensten Kontexten verarbeitet werden, wird das Problem evident. Beispielsweise sind die beiden synonymen Terme Computer und Rechner nur im Informatik-nahen Kontexten gegeneinander austauschbar.

[16] Da die Themenvektoren Elemente eines Vektorraumes mit ausschließlich positiven Achsenabschnitten sind (vgl. Abschnitt 5.1.1.3), beträgt der Winkel zwischen $\vec{\tau}_{\text{Computermaus}}$ und $\vec{\tau}_{\text{(Tier-)Maus}}$ maximal 90°. Wegen der Normierung der beiden Vektoren folgt somit, dass $|\vec{\tau}_{\text{Computermaus}} + \vec{\tau}_{\text{(Tier-)Maus}}|$ zu den Vektoren einen maximalen Winkel von 45° hat.

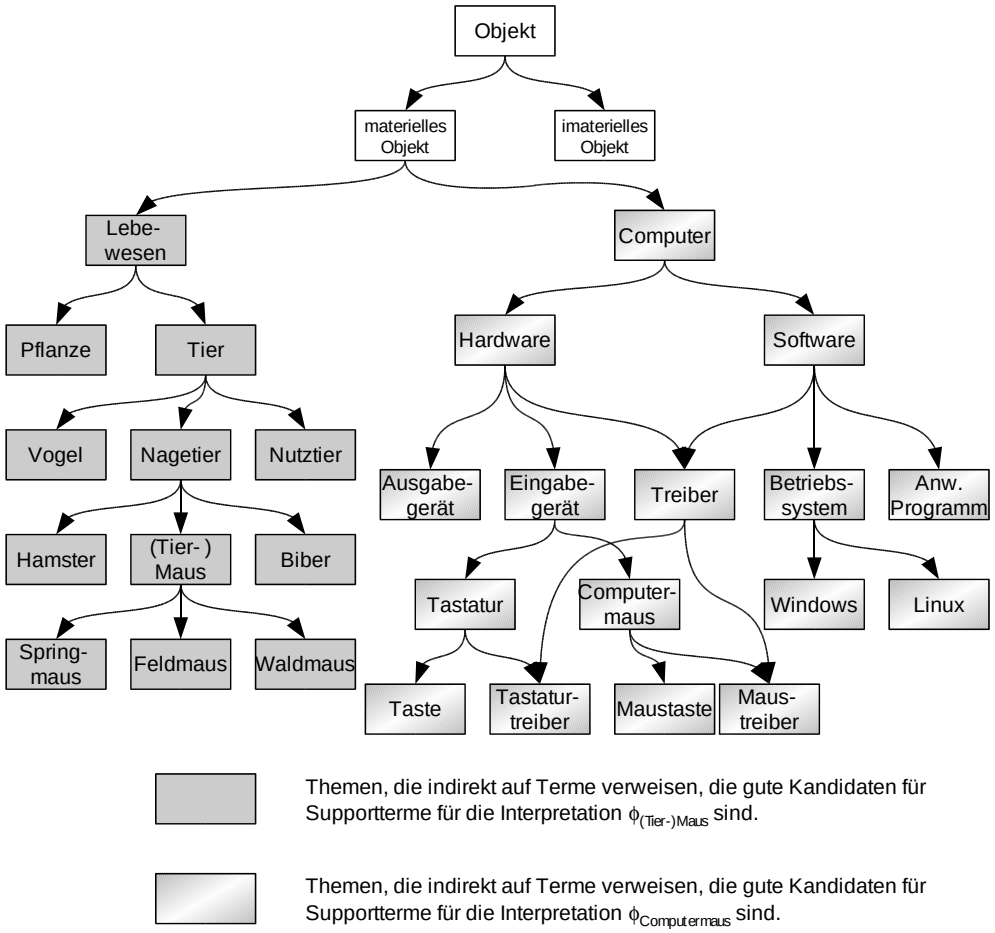


Abbildung 5.13: Ableitung von Supporttermen aus einer beispielhaften Themenstruktur zu dem Term Maus.

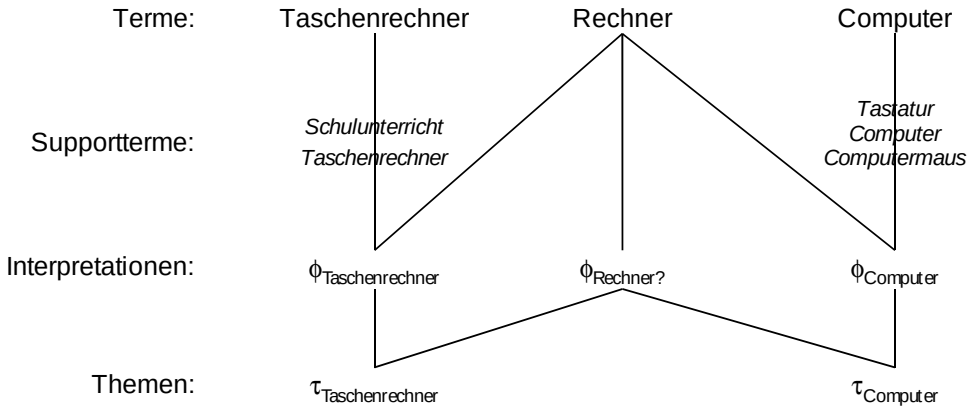


Abbildung 5.14: Beispiel für partiell synonyme Terme und ihre Zuordnungen.

Im Kontext der Schule und des Schulunterrichts ist Rechner hingegen eher mit Taschenrechner synonym als mit Computer. Durch die Trennung von Termen, Interpretationen und Themen lassen sich auch Partielle Synonymien im eTVSM korrekt abbilden, indem die in den beiden Abschnitten 5.1.2.2 und 5.1.2.3 genannten Vorgehen zur Repräsentation von Totaler Synonymie und Homographie miteinander kombiniert werden. Die Abbildung 5.14 zeigt die Repräsentation der Partiellen Synonymie am Beispiel der drei Terme Computer, Rechner und Taschenrechner.

5.1.2.5 Repräsentation der Metonymie

Eine Metonymie liegt dann vor, wenn eine nicht-wörtliche Verschiebung der Interpretation eines Terms vorgenommen wird.[17] Ein Beispiel für eine Metonymie ist folgender Satz:

> Berlin entschied sich gegen einen Einsatz im Irak.

In diesem Beispiel wird der Term Berlin nicht im Sinne seiner eigentlichen Interpretation als die Stadt Berlin verwendet, sondern der Term wird mit dem Parlament, welches sich in Berlin befindet, gleichgesetzt. Zur Erkennung und Auflösung von Metonymien wird hier eine ähnliche Vorgehensweise wie bei der Homographie vorgeschlagen.

Abbildung 5.15 zeigt die für das Beispiel notwendigen Zuordnungen zwischen Termen, Supporttermen, Interpretationen und Themen. Dem Term Berlin werden dabei zwei Interpretationen $\phi_{\text{Berlin(Stadt)}}$ und $\phi_{\text{Berlin(Parlament)}}$ zugewiesen. Wobei die Interpretation $\phi_{\text{Berlin(Stadt)}}$ als die übliche, die Standard-Interpretation verstanden wird. Aus diesem Grunde werden dieser

[17] Vgl. Abschnitt 2.3.4.2.

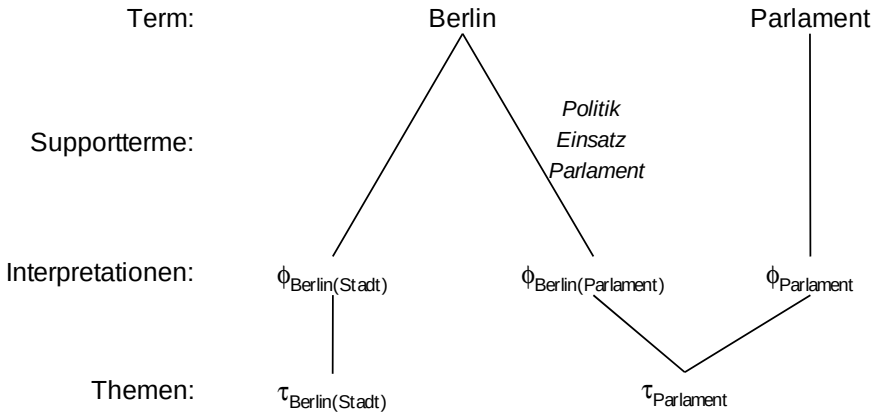


Abbildung 5.15: Auflösung von Metonymie für den Term Berlin.

Interpretation keine Supportterme zugewiesen. Die Interpretation $\phi_{\text{Berlin(Parlament)}}$ ist die Interpretation des Terms Berlin im übertragenen Sinne als Parlament. Diese Interpretation wird dem Term Berlin jedoch nur dann zugewiesen, wenn einer der zu dieser Interpretation definierten Supportterme in dem Dokument (ggf. mit einer zusätzlichen Einschränkung auf die Nähe zum Term Berlin) enthalten ist. Der Abbildung 5.15 kann zudem entnommen werden, dass die Interpretationen $\phi_{\text{Berlin(Parlament)}}$ und $\phi_{\text{Parlament}}$ demselben Thema zugeordnet werden. Somit erhalten die beiden Interpretationen einen identischen Interpretationsvektor und sind somit thematisch identisch.

Die Bestimmung der Supportterme für die übertragene Interpretation eines Terms kann auf dieselbe Weise geschehen, wie es Abschnitt 5.1.2.3 bereits bei den Supporttermen für Homographen beschrieben wurde.

5.1.2.6 Definition der Dokumenten-Ähnlichkeiten

Wie es schon zum Anfang dieses Kapitels erwähnt wurde, wird die Ähnlichkeit zwischen zwei Dokumenten im eTVSM analog zum TVSM berechnet, jedoch mit folgendem Unterschied: Beim TVSM ist die Basis für die Berechnung die Zuordnung von Termen zu einem Dokument. Beim eTVSM hingegen ist die Zuordnung von Interpretationen zu einem Dokument die Basis für die Berechnung der Dokumentenähnlichkeiten. Das Verschieben der Berechnungsbasis von Termen zu Interpretationen ist notwendig, um die linguistischen Phänome der Homographen und der Metonymie zu erfassen. Wie in den Abschnitten 5.1.2.3 und 5.1.2.5 bereits gezeigt wurde, werden zur Repräsentation von Homographen bzw. der Metonymie zwingend die Konzepte bzw. Entitytypen Interpretation und Thema benötigt.

Wie aus dem Datenmodell in Abbildung 5.10 auf Seite 129 ersichtlich, liegt dem eTVSM

die Annahme zu Grunde, dass Dokumente aus Interpretationen bestehen. Dieses wird durch den Relationshiptypen DI-Zuo repräsentiert, der seinerseits vom Parser erst unter Anwendung von anderen Relationshiptypen hergeleitet wird, was im weiteren Verlauf dieses Kapitels gezeigt wird. Interpretationen werden ihrerseits durch Vektoren repräsentiert. Zu diesen Vektoren werden in dem Relationshiptypen Ähnlichkeit die Skalarprodukte gespeichert, die wie in Abschnitt 5.1.2.1 gezeigt wurde, aus Themenstrukturen abgeleitet werden.

Zur mathematischen Herleitung der Dokumentenähnlichkeiten werden folgende Symbole benötigt: Die Menge aller Interpretationen bzw. der Entitytyp Interpretation wird durch das Symbol Φ repräsentiert. Zu jeder Kombination von zwei Interpretationen $i, j \in \Phi$ ist ein Vektor $\vec{\phi}_i$ und $\vec{\phi}_j$ gemäß Gleichung 5.4 auf Seite 130 definiert, sowie das dazugehörige Skalarprodukt $\vec{\phi}_i\vec{\phi}_j$ gemäß Gleichung 5.5 auf Seite 130.

Zu jedem Dokument k aus der Menge aller Dokumente D sind Einträge $a_{k,i}$ hinterlegt, die die Vorkommensanzahl der Interpretation i in dem Dokument k festlegen. Diese Einträge entsprechen dem Relationshiptyp DI-Zuo. Der Dokumentenvektor $\vec{d}_k$ zum Dokument k definiert sich somit wie folgt:

$$\forall k \in D: \quad \vec{d}_k = \frac{1}{|\vec{\delta}_k|}\vec{\delta}_k \quad \Rightarrow \quad |\vec{d}_k| = 1 \tag{5.6}$$

mit

$$\vec{\delta}_k = \sum_{i \in T} a_{k,i}\vec{\phi}_i \tag{5.7}$$

Analog zum TVSM kann der Betrag eines Dokumentenvektors $\vec{d}_k$ allein unter Verwendung der Skalarprodukte zwischen den Interpretationsvektoren der im Dokument enthaltenen Interpretationen berechnet werden:

$$\begin{aligned} |\vec{\delta}_k| &= |\sum_{i \in \Phi} a_{k,i}\vec{\phi}_i| \\ &= \sqrt{|\sum_{i \in \Phi} a_{k,i}\vec{\phi}_i|^2} \\ &= \sqrt{(\sum_{i \in \Phi} a_{k,i}\vec{\phi}_i)^2} \\ &= \sqrt{\sum_{i \in \Phi}\sum_{j \in \Phi} a_{k,i}a_{k,j}\vec{\phi}_i\vec{\phi}_j} \end{aligned} \tag{5.8}$$

Die Ähnlichkeit $\text{sim}(k, l)$ zwischen zwei Dokumenten $k, l \in D$ wird analog zum TVSM als das Skalarprodukt der beiden Dokumentenvektoren definiert, was durch die Normierung der Dokumentenvektoren dem Kosinus des Winkels zwischen den beiden Vektoren entspricht:

$$
\begin{aligned}
\mathrm{sim}(k,l) &= \vec{d_k}\vec{d_l} \\
&= \frac{1}{|\vec{\delta_k}|}\vec{\delta_k}\frac{1}{|\vec{\delta_l}|}\vec{\delta_l} \\
&= \frac{1}{|\vec{\delta_k}||\vec{\delta_l}|}\vec{\delta_k}\vec{\delta_l} && \text{Assoz.Ges.} \\
&= \frac{1}{|\vec{\delta_k}||\vec{\delta_l}|}\sum_{i\in\Phi} a_{k,i}\vec{\phi_i}\sum_{j\in\Phi} a_{l,j}\vec{\phi_j} \\
&= \frac{1}{|\vec{\delta_k}||\vec{\delta_l}|}\sum_{i\in\Phi}\sum_{j\in\Phi} a_{k,i}a_{l,j}\vec{\phi_i}\vec{\phi_j} && \text{Distrib.Ges.} \qquad (5.9)
\end{aligned}
$$

5.1.3 Wortstamm-Term-Zuordnung

Die Wortstamm-Term-Zuordnung dient sowohl zur Repräsentation von Wortgruppen als auch zur Repräsentation einer Stoppwortliste. Der Grund für die, auf den ersten Blick ungewöhnliche Kombination der beiden Phänomene ist, dass eine herkömmliche Stoppwortliste nicht sinnvoll angewandt werden kann, wenn Wortgruppen berücksichtigt werden sollen. Die Ursache dafür ist, dass es fest stehende Wortgruppen geben kann, die sowohl aus einem Nicht-Stoppwort als auch aus einem Stoppwort bestehen. So ist z. B. das englische Wort me ein klassisches Beispiel für ein Stoppwort, dass in einer Produktbezeichung verwendet wird: Windows ME[18].

Unter einem Term wird beim eTVSM entweder ein einzelnes Wort oder eine Wortgruppe (wie z. B. New York, Vector Space Model, Katze aus dem Sack, raining cats and dogs[19]) mit einer eigenen Interpretation verstanden. Um die Reihenfolge der einzelnen Wörter abbilden zu können, steht ein Term über die Zuordnung WT-Zuo1 mit einem oder mehreren Entitäten vom Typ Position in Beziehung. Der Entitytyp Position enthält dabei lediglich die Menge der natürlichen Zahlen und dient dazu, einzelne Wortstämme in eine Reihenfolge zu bringen. Aus diesem Grund steht WT-Zuo1 über den Relationshiptypen WT-Zuo2 mit dem Entitytyp Term in Beziehung. (Vgl. Abbildung 5.16 und Abbildung 5.17.) Der Entitytyp Term steht deshalb mit Wortstamm und nicht mit Wort in indirekter Beziehung, weil so automatisch auch ein Stemming von Wortgruppen basierend auf dem Stemming der Wörter der Wortgruppe sichergestellt ist. Zur Identifikation von Wortgruppen ist die Reihenfolge der Wörter von Bedeutung, daher reicht es nicht aus, zu den Dokumenten – wie bei anderen IF-/IR-Modellen üblich – nur die Anzahl der jeweils in einem Dokument enthaltenen Wörter festzuhalten. Aus diesem Grunde wird die Zuordnung zwischen Dokument und Wort analog zu der Zuordnung zwischen Wortstamm und Term gelöst: Ein Dokument geht über den Relationshiptypen DW-Zuo1 eine Beziehung mit dem Entitytyp Position ein, um Worten eine

[18] Windows ME ist ein eingetragenes Warenzeichen und der Name eines Betriebssystems der amerikanischen Firma Microsoft.

[19] Raining cats and dogs ist eine englische Redewendung für einen heftigen Regenschauer.

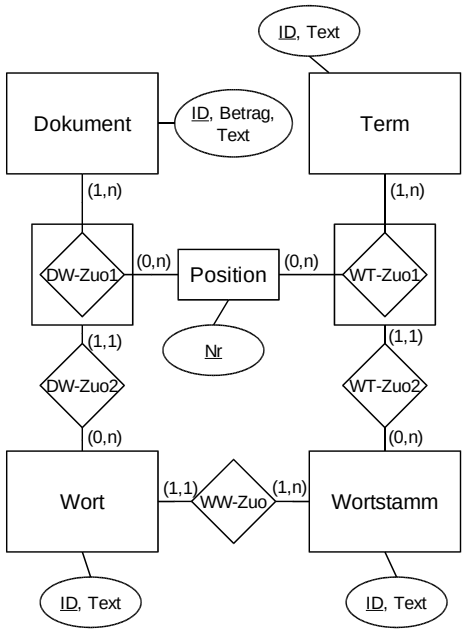


Abbildung 5.16: Die Wortstamm-Term-Zuordnung und weitere, von der Repräsentation von Wortgruppen indirekt betroffene Entitäten/Beziehungen.

eindeutige Position im Dokument zuweisen zu können. Die Zuordnung der Dokumentpositionen zu den einzelnen Wörtern geschieht dann über den Relationshiptypen DW-Zuo2. (Vgl. Abbildung 5.16.)

Die Stoppwortliste wird dadurch abgebildet, dass Wortstämme von Stoppwörtern keinem Term zugeordnet werden, der nur aus einem Wort besteht. Für das englische Beispielwort me heißt das z. B., dass me im Term Windows Me vorkommt[20] und *nicht* allein einem Term, bestehend nur aus dem Wortstamm me, zugeordnet ist. Alle Worte, die *keine* Stoppwörter sind, sind immer (zumindest) Bestandteil eines Terms, welcher nur aus dem einen Wort besteht.

5.1.4 Wort-Wortstamm-Zuordnung

Bei dem Konstrukt der Wort-Wortstamm-Zuordnung beim eTVSM handelt es sich um eine Anwendung des Stemming-Lemmas des TVSM aus Abschnitt 4.3. Ein in Abschnitt 4.7 als positiv genannter Aspekt des TVSM ist, dass es das Stemming über die Termähnlichkeiten im Modell integriert. Negativ an der Umsetzung beim TVSM ist allerdings, dass durch diese Realisierung ein erhöhter Rechenaufwand bei der Berechnung von Dokumentenähnlichkeiten in

[20] An dieser Stelle wird die Groß-/Kleinschreibung, wie bei den meisten IR-/IF-Systemen üblich, ignoriert.

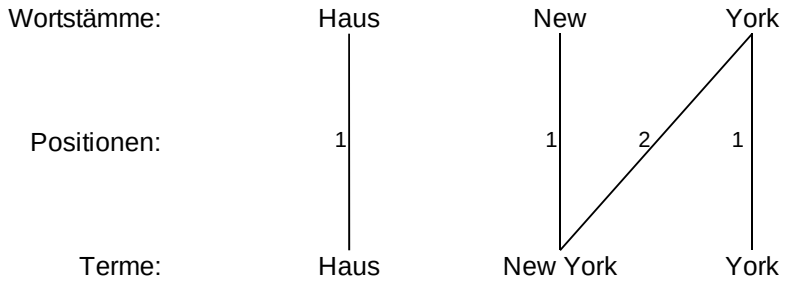


Abbildung 5.17: Beispiel für Beziehungen zwischen Wortstämmen und Termen.

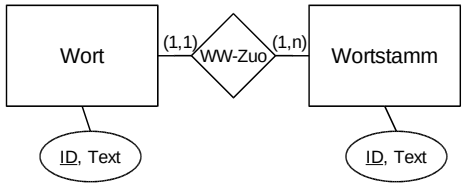


Abbildung 5.18: Die Wort-Wortstamm-Zuordnung für das Stemming.

Kauf genommen wird. Dadurch, dass das Stemming derart realisiert wird, dass Worte mit gleichem Wortstamm eine maximale Ähnlichkeit (Winkel 0° zwischen den beiden Wortvektoren und somit eine Ähnlichkeit von $\cos 0^\circ = 1$) haben, folgt daraus, dass das Stemming implizit jedesmal beim Vergleich von zwei Dokumenten durchgeführt wird. Das Stemming-Lemma hat gezeigt, dass das bei anderen Verfahren in der Praxis gängige Vorgehen Dokumente vorab zu stemmen, mit dem TVSM konsistent ist. Durch Anwendung des Stemming-Lemmas kann erreicht werden, dass das Stemming nur einmalig, beim Einstellen eines Dokuments, vom Parser durchgeführt wird. Zudem hat die Reduktion der verschiedenen Wortformen auf den Wortstamm oder die führende Wortform den Vorteil, dass die Anzahl der verschiedenen Wörter in einem Dokument abnimmt, wodurch die Berechnung der paarweisen Dokumentenähnlichkeiten beschleunigt wird.[21] Damit das Stemming aber ein Bestandteil des Modells ist, wird es hier in Form der Wort-Wortstamm-Zuordnung in das Modell des eTVSM aufgenommen.

Die Realisierung des Stemming als Wort-Wortstamm-Zuordnung beim eTVSM zeigt der Ausschnitt aus dem ERM-Gesamtmodell in Abbildung 5.18. Jedem Wort (wie z. B. Häuser) wird genau ein Wortstamm (wie z. B. Haus) zugeordnet und jedem Wortstamm werden mindestens eine und maximal beliebig viele Entitäten vom Typ Wort über den Relationship-

[21] Vgl. dazu die Struktur der Gleichung 4.9 auf Seite 94 zur Bestimmung von Dokumentenähnlichkeiten.

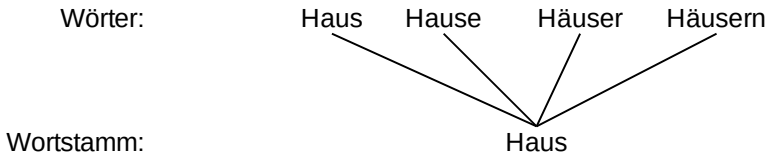


Abbildung 5.19: Beispiel für Zuordnungen zwischen Worten und Wortstämmen.

typ WW-Zuo zugeordnet (vgl. Abbildung 5.19). Somit wird durch das eTVSM ein Lexikon-basiertes Stemming[22] realisiert. Ob dabei ein Weak-Stemming oder ein Strong-Stemming durchgeführt wird, wird nicht festgelegt, weil diese Entscheidung für das eTVSM keine Rolle spielt. Allerdings sollte das Stemming konsistent durchgeführt werden, das heißt: entweder durchgehend ein Weak- oder Strong-Stemming.

Ein weiterer Aspekt, der bei der Implementierung der Wort-Wortstamm-Zuordnung betrachtet werden sollte, ist die Frage, ob Worte in ihrer Schreibweise aus dem Dokument oder grundsätzlich in Kleinbuchstaben (oder Großbuchstaben) geschrieben werden. Eine Translation der Worte in Kleinbuchstaben hat den Vorteil, dass z. B. die beiden Schreibweisen von Der und der nicht explizit unterschieden werden müssen. Zusätzlich ist das System robuster gegenüber Tippfehlern wie z. B. dEr. Andererseits können durch die Translation Informationen verloren gehen, bspw. werden Abkürzungen oder Teile von technischen Fachbegriffen häufig in Großbuchstaben geschrieben. Durch den Verzicht auf eine Translation kann vermieden werden, dass z. B. die englische Abkürzung IS[23] nach einer Translation von dem englischen Stoppwort is nicht unterschieden werden kann und somit falsch als Stoppwort eingeordnet wird. Eine Möglichkeit, die Nachteile beider Verfahren zu minimieren, ist es, die WW-Zuo nicht nur für das Stemming, sondern auch für eine Groß-/Kleinbuchstaben-Translation zu nutzen. Die Idee ist, dass z. B. dem Wortstamm gehen nicht nur die Worte geh, gehen, geht, ... sondern auch die passenden Worte in Großschreibweise, also Geh, Gehen, Geht, ... zugeordnet werden. Sollte es z. B. eine bekannte Abkürzung GEH geben, dann kann diese einem eigenen Wortstamm GEH zugewiesen werden, der eine andere Interpretation hat als der Wortstamm gehen. Zur Erhöhung der Fehlertoleranz gegenüber Rechtschreibfehlern wie z. B. gEhen kann der Parser derart programmiert werden, dass er ein Wort zunächst in der originalen Schreibweise aufzulösen versucht. Sollte dieses misslingen, dann wandelt der Parser das Wort in Kleinbuchstaben um und versucht es dann erneut. Somit würde gEhen nach dem zweiten Schritt korrekt dem Wortstamm gehen zugeordnet werden.

[22] Vgl. Abschnitt 2.3.2.2.
[23] Information Systems

5.2 Das eTVSM und der Ontologie-Begriff

Die eigentliche Aufgabe des eTVSM ist die Herleitung von paarweisen Ähnlichkeiten zwischen Dokumenten, allerdings impliziert das eTVSM zur Lösung dieser Aufgabe auch eine Modellierungssprache zur Repräsentation einer Ontologie. Bei der Modellierungssprache handelt es sich um eine relativ einfache, formale Sprache, bei der verschiedene typisierte Entitäten über typisierte Beziehungen miteinander verknüpft werden. Konkret werden gemäß dem Datenmodell des eTVSM aus Abbildung 5.1 auf Seite 111 u. a. Wörter, Wortstämme, Terme, Interpretationen und Themen zueinander in Beziehung gesetzt. Beispiele für konkrete Ausprägungen derartiger Beziehungen wurden bereits in den Abbildungen 5.4, 5.11, 5.12, 5.17 und 5.19 auf den Seiten 117, 131, 133, 141 und 142 gegeben. Bei genauerer Betrachtung der verschiedenen, durch das Modell zueinander in Bezug gesetzten Entitäten, kann man feststellen, dass es sich bei den verknüpften Entitäten um sprachliche Ausdrucksmittel handelt, die zur Kommunikation zwischen Menschen genutzt werden und auf die sich Menschen einer Sprachgemeinschaft geeinigt haben. Somit ist das eTVSM-Datenmodell u. a. eine Ontologie-Modellierungssprache, weil eine konkrete Instanz des Datenmodells alle Kriterien der Ontologiedefinition aus Abschnitt 2.4 erfüllt. Bezugnehmend auf die Klassifikation von Ontologie-Modellierungssprachen fällt diese Sprache unter die Klasse der Thesauren und Wortnetze, weil zum Einen keine strikt hierarchische Klassifikation vorgenommen werden muss und zum Anderen die Ausdrucksmächtigkeit im Unterschied zu den logisch-mathematischen Repräsentationen und den semiotischen Thesauren u. a. wegen des Kriteriums der Zykelfreiheit bei den Themenstrukturen eingeschränkt ist. Die Tatsache, dass das eTVSM in sich eine Ontologie einbettet, ist aus Sicht der Aufgabenstellung, die das eTVSM bearbeitet, sinnvoll: Für eine adäquate Lösung des IF-/IR-Problems ist es aus sprachlicher und logischer Sicht unerlässlich, dass die in Dokumenten verwendeten sprachlichen Ausdrucksmittel und ihre Beziehungen zueinander in irgendeiner Form berücksichtigt werden.

5.2.1 Eine grafische Ontologie-Repräsentation für das eTVSM

Da das eTVSM primär zur Lösung von IF-/IR-Aufgaben durch einen Computer und nicht zu Zwecken einer anschaulichen Repräsentation von Ontologien entwickelt worden ist, ist die Repräsentation einer Ontologie im eTVSM auf die „Bedürfnisse“ des Computers zugeschnitten und für den Menschen schwer lesbar. Dieses begründet sich insbesondere darin, dass im eTVSM explizit zwischen Wörtern, Wortstämmen, Termen, Interpretationen und Themen auch dann unterschieden wird, wenn diese gleich benannt sind. Somit neigt diese Darstellungsform – an menschlichen Ansprüchen gemessen – zur Unübersichtlichkeit. Aus diesem Grund wird an dieser Stelle eine grafische Sprache vorgestellt, die – von der Wort-Wortstamm- und der Wortstamm-Term-Zuordnung abgesehen – dieselbe Mächtigkeit aufweist wie die in dem eTVSM eingebettete Sprache zur Modellierung von Ontologien. Allerdings ist diese Sprache grafisch besser darstellbar und für den Menschen weniger unübersichtlich als die Sprache des eTVSM. Auf die Darstellung der Zuordnungen zwischen Wörtern und Wortstämmen und zwischen Wortstämmen und Termen wird bewusst verzichtet, weil diese vom Menschen aufgrund seiner allgemeinen Erfahrung mit natürlichen Sprachen intuitiv nachvollzogen werden kann,

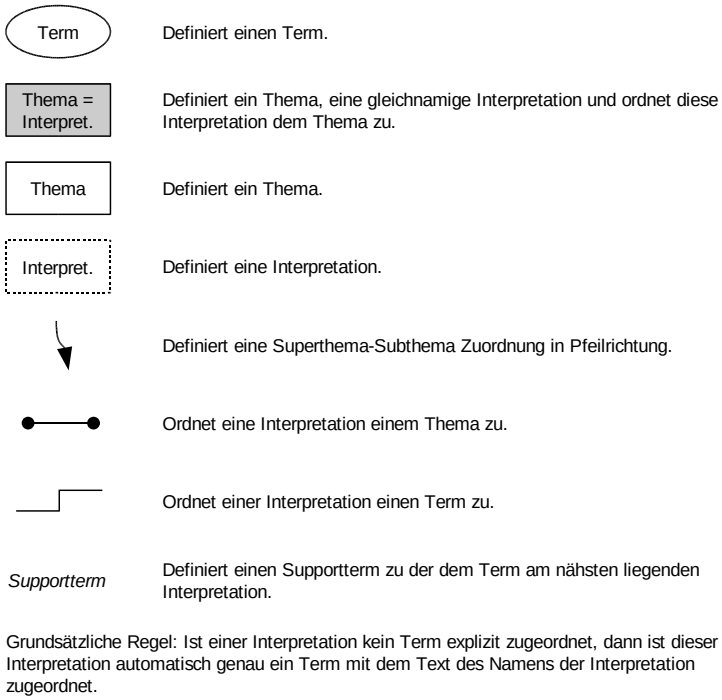


Abbildung 5.20: Die Elemente der Sprache.

ohne dass sie explizit dargestellt werden müssen.

Die Abbildung 5.20 zeigt die Elemente dieser Sprache. Da die meisten Interpretationen genau einem Thema zugeordnet sind, wurde aus Gründen der Übersichtlichkeit und der Platzersparnis ein eigenes Symbol (das grau unterlegte Kästchen) für diesen Fall konzipiert. Aus demselben Grund wurde eine Regel geschaffen, die einer Interpretation implizit genau einen gleichnamigen Term zuordnet, wenn zu einer Interpretation kein Term explizit definiert wurde. Wird jedoch mindestens ein Term explizit definiert, dann wird dieser Interpretation kein gleichnamiger Term implizit zugeordnet. Die Darstellung der verschiedenen linguistischen Phänomene in der hier vorgestellten Modellierungsprache, wie z. B. Synonymie, Homographen und Metonymie werden exemplarisch an den Abbildungen 5.21, 5.22, 5.23 und 5.24 illustriert.

5.2.2 Eine Beispiel-Ontologie

In diesem Abschnitt wird eine relativ kleine und beispielhafte Ontologie vorgestellt, die im eTVSM abgebildet werden kann. Diese Ontologie wird im Abschnitt 5.3 dazu verwendet,

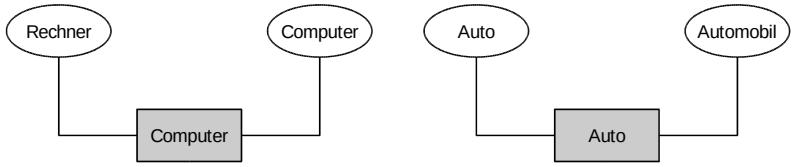


Abbildung 5.21: Darstellung der (Totalen) Synonyme aus Abbildung 5.11.

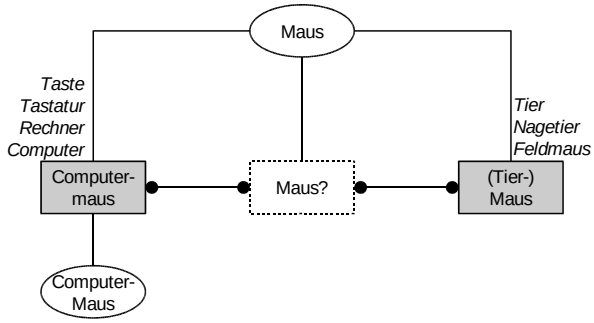


Abbildung 5.22: Darstellung der Homographen aus Abbildung 5.12.

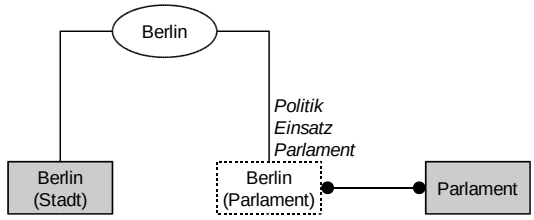


Abbildung 5.23: Darstellung der Metonyme aus Abbildung 5.15.

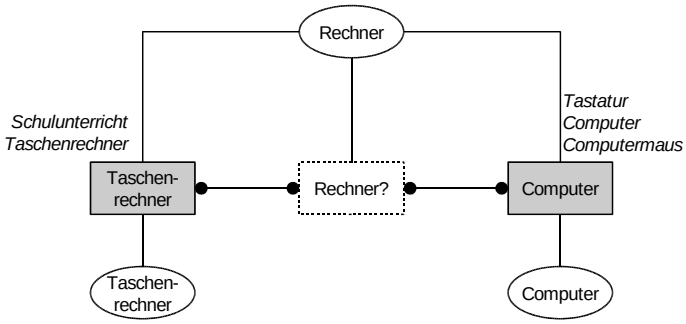


Abbildung 5.24: Darstellung der Partiellen Synonyme aus Abbildung 5.14.

die Umsetzung des eTVSM mit Hilfe einer relationalen Datenbank möglichst plastisch zu illustrieren. Aus diesem Grunde ist die Ontologie relativ klein gehalten, um dem Leser das Nachvollziehen zu erleichtern.

Abbildung 5.25 zeigt die Ontologie in der in Abschnitt 5.2.1 eingeführten Notationsform. Wie man der Abbildung entnehmen kann, beschreibt die Ontologie u. a. einige Zusammenhänge zwischen den beiden Firmen Microsoft und SCO, sowie zwischen einigen Betriebssystemen. So kann man der Ontologie z. B. entnehmen, dass das Thema Microsoft ein Subthema von Firma ist und dass Bill Gates und Steve Ballmer ihrerseits Subthemen von Microsoft sind. Dieser Zusammenhang wird dadurch motiviert, dass die Personen Mitarbeiter der Firma Microsoft sind. Zusätzlich dazu ist auch Windows ein Subthema von Microsoft, weil Windows ein Produkt der Firma ist. Gleichzeitig ist Windows jedoch ein Betriebssystem, weshalb es als Subthema zu Betriebssystem modelliert wurde. Die Themen Preisvorteil, Gemeinde und Sicherheitslücke sind neben einigen anderen Themen als unabhängig von allen anderen Themen modelliert worden. Das heißt, dass sie in keine Superthema-Subthema-Beziehung mit anderen Themen eingehen.

Neben der Themenstruktur sind in der Ontologie auch einige Synonyme definiert worden. Z. B. existieren zu dem Thema Sicherheitslücke die beiden synonymen Terme Sicherheitslücke und Bug. Bei den Namen der verschiedenen Personen sind ebenfalls Synonyme definiert worden: So sind z. B. Linus Torvalds und Torvalds als Synonym definiert, weil diese in dem hier betrachteten Kontext (Computer) synonym verwendet werden. Neben den Synonymen definiert die Ontologie den Homographen Maus, der in Abhängigkeit von den Supportwörtern als Maus (Nagetier) oder als Computermaus oder einfach als Maus, bei der die genaue Interpretation unbekannt ist, spezifiziert wird.

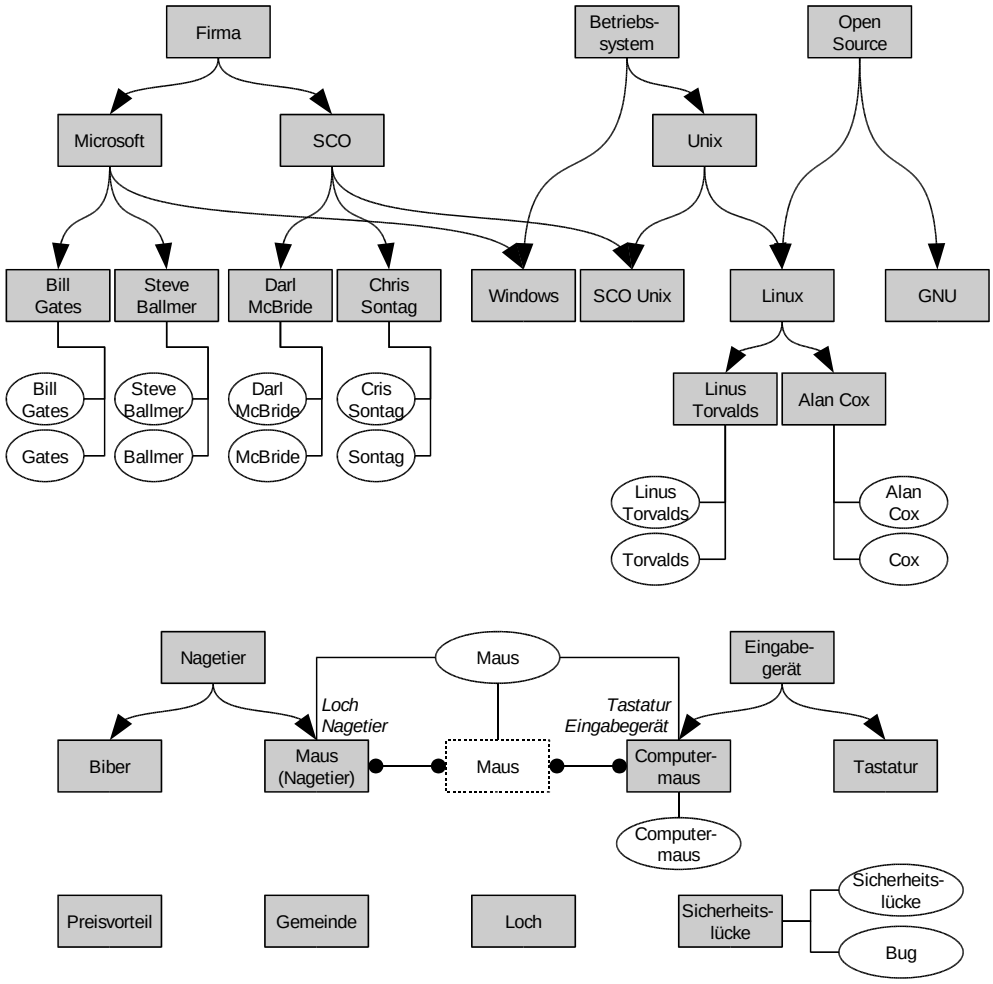


Abbildung 5.25: Beispiel für eine Ontologie.

5.3 Umsetzung mit einer relationalen Datenbank

Dieser Abschnitt widmet sich einer beispielhaften Umsetzung des eTVSM unter Verwendung einer relationalen Datenbank. Die Vorteile einer derartigen Umsetzung sind der relativ geringe Aufwand und die gleichzeitige Nutzung der datenbankimmanenten Eigenschaften wie z. B. Datensicherheit, Konsistenz und Anfrageoptimierung.[24]

5.3.1 Datenmodell

Zur Erstellung der Tabellen der Entitytypen aus dem eTVSM-Datenmodell (Abbildung 5.1 auf Seite 111) in einer relationalen Datenbank sind folgende SQL-Befehle notwendig:

```
CREATE TABLE Dokument (
       Id        INTEGER UNIQUE NOT NULL,
       Betrag    DOUBLE PRECISION DEFAULT NULL,
       Text      TEXT NOT NULL,
       PRIMARY KEY(Id));

CREATE TABLE Wortstamm (
       Id        INTEGER UNIQUE NOT NULL,
       Text      TEXT NOT NULL,
       PRIMARY KEY(Id));
CREATE INDEX Wortstamm_Text ON Wortstamm(Text);

CREATE TABLE Wort (
       Id        INTEGER UNIQUE NOT NULL,
       Text      TEXT NOT NULL,
       Wortstamm INTEGER NOT NULL REFERENCES Wortstamm(Id),
       PRIMARY KEY(Id));
CREATE INDEX Wort_Text ON Wort(Text);

CREATE TABLE Term (
       Id        INTEGER UNIQUE NOT NULL,
       Text      TEXT,
       PRIMARY KEY(Id));

CREATE TABLE Interpretation (
       Id        INTEGER UNIQUE NOT NULL,
       Beschr    TEXT,
       Gewicht   DOUBLE PRECISION DEFAULT NULL,
       PRIMARY KEY(Id));

CREATE TABLE Thema (
       Id        INTEGER UNIQUE NOT NULL,
```

[24] Vgl. dazu Abschnitt 2.2.2.

```
       Beschr    TEXT,
       PRIMARY KEY(Id));
```

Zu jedem Entitytyp (mit Ausnahme des Entitytypen Position) wird durch diese Befehlsfolge eine eigene Tabelle angelegt, wobei das Attribut `Id` als identifizierendes Attribut, auch Primärschlüssel genannt, definiert wird. Bei diesen Befehlen wird davon ausgegangen, dass Primärschlüssel von der Datenbank – wie bei den meisten relationalen Datenbanken üblich – für einen schnellen Zugriff automatisch mit Indices versehen werden. Des Weiteren sind die Primärschlüssel eindeutig und sie müssen mit einem Wert belegt werden (sie dürfen nicht `NULL` sein). Der Entitytyp Position braucht nicht in einer eigenen Tabelle repräsentiert zu werden, weil dieser der Menge der natürlichen Zahlen entspricht. Somit reicht es aus, diesen Entitytypen an den Stellen, an denen er in eine Beziehung eingeht, einfach durch ein Attribut vom Typ `INTEGER` abzubilden.

Des Weiteren werden zu den Indices der Primärschlüssel für die beiden Tabellen `Wort` und `Wortstamm` jeweils zusätzliche Indices auf das jeweilige Attribut `Text` angelegt. Diese beiden Indices sind hilfreich beim Einfügen von neuen Wörtern bzw. Wortstämmen und sie ermöglichen eine performante Abfrage nach Wortstämmen zu in den Dokumenten vorgegebenen Wörtern. Da die WW-Zuo aus dem Datenmodell ein Relationshiptyp mit den Kardinalitäten (1,1)-(1,n) ist, wird diese aus Effizienzgründen in die Tabelle `Wort` einbezogen. Deshalb enthält diese Tabelle das Fremdschlüsselattribut `Wortstamm`, welches auf das Primärschlüsselattribut `Id` der Tabelle `Wortstamm` verweist.

Zur Umsetzung der verschiedenen (nicht redundanten bzw. nicht gestrichtelten) Relationshiptypen aus dem eTVSM-Datenmodell sind folgende Befehle zum Erstellen von Tabellen und Indices auszuführen:

```
CREATE TABLE DW_Zuo (
       Dokument  INTEGER NOT NULL REFERENCES Dokument(Id),
       Wort      INTEGER NOT NULL REFERENCES Wort(Id),
       Position  INTEGER NOT NULL,
       PRIMARY KEY(Dokument, Position));

CREATE TABLE WT_Zuo (
       Wortstamm INTEGER NOT NULL REFERENCES Wortstamm(Id),
       Term      INTEGER NOT NULL REFERENCES Term(Id),
       Position  INTEGER NOT NULL,
       PRIMARY KEY(Term, Position));

CREATE TABLE TI_Zuo (
       Term      INTEGER NOT NULL REFERENCES Term(Id),
       Interpretation INTEGER NOT NULL
                      REFERENCES Interpretation(Id),
       PRIMARY KEY(Term, Interpretation));

CREATE TABLE Supportterm (
       Term      INTEGER NOT NULL REFERENCES Term(Id),
```

```
       Interpretation INTEGER NOT NULL
                      REFERENCES Interpretation(Id),
       PRIMARY KEY(Term, Interpretation));

CREATE TABLE IT_Zuo (
       Interpretation INTEGER NOT NULL REFERENCES
                      Interpretation(Id),
       Thema     INTEGER NOT NULL REFERENCES Thema(Id),
       PRIMARY KEY(Interpretation, Thema));
CREATE INDEX IT_Zuo_pkey2 ON IT_Zuo(Thema,Interpretation);

CREATE TABLE Themenstruktur(
       Superthema INTEGER NOT NULL REFERENCES Thema(Id),
       Subthema   INTEGER NOT NULL REFERENCES Thema(Id),
       PRIMARY KEY(Superthema, Subthema));
CREATE INDEX Themenstruktur_pkey2 ON
             Themenstruktur(Subthema, Superthema);
```

Der Relationshiptyp WT-Zuo2 geht in die beiden Entitytypen WT-Zuo1 und Wortstamm mit den Kardinalitäten (1,1)-(0,n) ein. Daher können die beiden Relationshiptypen WT-Zuo1 und WT-Zuo2 zu einer Tabelle WT_Zuo zusammengefasst werden. Dieses gilt analog für die beiden Relationshiptypen DW-Zuo1 und DW-Zuo2, die zu einer einzigen Tabelle DW_Zuo zusammengefasst werden.

Dadurch, dass die relationale Datenbank z. B. für die Tabelle Themenstruktur automatisch einen zusammengesetzten Index aus den Primärschlüsselattributen Superthema und Subthema in genau dieser Reihenfolge erstellt, kann sehr effizient abgefragt werden, welche Subthemen zu einem Superthema existieren. Eine effiziente Abfrage dieser Tabelle in die umgekehrte Richtung (Welche Superthemen existieren zu einen Subthema?) ist nur dann möglich, wenn ein zusätzlicher Index erstellt wird. Dieser Index Themenstruktur_pkey2 umfaßt die beiden Primärschlüsselattribute in umgekehrter Reihenfolge. Aus demselben Grund ist auch für die Tabelle IT_Zuo ein zusätzlicher Index angelegt worden.

Zusätzlich zu den obigen Tabellen müssen noch die beiden Tabellen für die redundanten Relationshiptypen Ähnlichkeit und DI-Zuo erstellt werden:

```
CREATE TABLE Aehnlichkeit (
       Int1       INTEGER NOT NULL
                  REFERENCES Interpretation(Id),
       Int2       INTEGER NOT NULL REFERENCES
                  Interpretation(Id),
       Skalarprodukt DOUBLE PRECISION NOT NULL,
       PRIMARY KEY(Int1, Int2));

CREATE TABLE DI_Zuo (
       Dokument   INTEGER NOT NULL REFERENCES Dokument(Id),
       Interpretation INTEGER NOT NULL
```

```
                    REFERENCES Interpretation(Id),
    Anzahl     INTEGER NOT NULL,
    PRIMARY KEY(Dokument, Interpretation));
```

Anhang A.1 zeigt, wie die Beispiel-Ontologie aus Abbildung 5.25 auf Seite 147 in den Tabellen abgebildet werden kann.

5.3.2 Initialisierung

Die Aufgabe der Initialisierungstransaktion ist es, basierend auf den vorgegebenen Themenstrukturen, den Gewichten der Interpretationen und den Zuordnungen zwischen Interpretationen und Themen die Skalarprodukte der Interpretationen zu berechnen und in der Tabelle `Aehnlichkeit` zu hinterlegen (vgl. Abbildung 5.3 auf Seite 114). Das hier vorgestellte Verfahren zur Berechnung der Skalarprodukte ist stark Datenbank zentriert, weil der mächtige Befehlssatz einer relationalen Datenbank die Implementierung des Verfahrens vereinfacht, indem der Implementierer seinen Blick auf das Wesentliche konzentrieren kann, weil viele kleine Details (wie z. B. das Verarbeiten von Mengen bzw. Tabellen und einzelnen Datensätzen) von der Datenbank übernommen werden. Somit wird das Verfahren für den Leser leichter nachvollziehbar, was jedoch an einigen Stellen zu Lasten der Bearbeitungsgeschwindigkeit geht und eine zusätzliche redundante Tabellen erfordert. Dieser Nachteil ist jedoch von einer geringeren Bedeutung, weil die Initialisierung nur selten – idealerweise nur einmal – ausgeführt werden muss. Wie oben erwähnt, benötigt das hier vorgestellte Verfahren zur Initialisierung eine zusätzliche Tabelle, die der Speicherung der Themenvektoren dient:

```
CREATE TABLE Themavektor (
       Thema      INTEGER NOT NULL REFERENCES Thema(Id),
       Dimension  INTEGER NOT NULL REFERENCES Thema(Id),
       Wert       DOUBLE PRECISION NOT NULL,
       PRIMARY KEY(Thema, Dimension));
```

Da jedes Thema durch eine Dimension in einem Themenvektor repräsentiert wird[25], ordnet die Tabelle `Themavektor` jedem Thema über das Attribut `Dimension` ein Thema zu. Im Attribut `Wert` wird der dazugehörige Wert des Eintrags zu dem Vektor gespeichert. Aus Gründen der Platzersparnis werden nur diejenigen Dimensionseinträge eines Vektors gespeichert, bei denen der Wert größer als Null ist.

Das Verfahren wird im Folgenden unter Verwendung der Beispiel-Ontologie illustriert. Dabei wird davon ausgegangen, dass die Ontologie bereits in Tabellenform mit den im Anhang A.1 spezifizierten Daten vorliegt. Für alle anderen Tabellen wird angenommen, dass diese keine Datensätze enthalten.

5.3.2.1 Initialisierung der Themenblätter

Die Initialisierung der Themenblätter kann mit dem hier verwendeten SQL92-Dialekt nicht vollständig in SQL umgesetzt werden, weil für eine Implementierung eine rekursive Bearbei-

[25] Vgl. dazu Abschnitt 5.1.1.3.

tung der Tabelle `Themenstruktur` erforderlich ist.[26] Aus diesem Grunde ist das Verfahren derart konzipiert, dass es mehrere u. a. parametrisierte Anfragen an die Datenbank unter Verwendung der redundanten Hilfstabelle `Themavektor` stellt. Somit kann der rekursive und im Folgenden natürlich-sprachlich beschriebene Teil des Verfahrens von einer prozeduralen oder objektorientierten Programmiersprache (wie z. B. *Java*) implementiert werden.[27]

Das Verfahren zur Initialisierung der Themenblätter beginnt zunächst damit, dass alle diejenigen Themenblätter gesucht werden, zu denen kein Vektoreintrag existiert. Diese Aufgabe löst die folgende Anfrage:

```
SELECT Id, Beschr
FROM   Thema
WHERE  NOT EXISTS(SELECT Superthema
                  FROM   Themenstruktur
                  WHERE  Superthema = Thema.Id)
   AND NOT EXISTS(SELECT Thema
                  FROM   Themavektor
                  WHERE  Thema = Thema.Id);
```

Die Anfrage selektiert alle Themen aus der Tabelle `Thema`, zu denen kein Eintrag in der Tabelle `Themenstruktur` existiert, bei dem die selektierten Themen als Superthema auftreten. Zusätzlich wird die Selektion auf alle Themen beschränkt, zu denen kein Eintrag in der Tabelle `Themavektor` existiert. Bezogen auf die Beispiel-Ontologie liefert die Anfrage das folgende Ergebnis:

```
 Id |      Beschr
----+------------------
  7 | Bill Gates
  8 | Steve Ballmer
  9 | Darl McBride
 10 | Chris Sontag
 11 | Windows
 12 | SCO Unix
 14 | GNU
 15 | Linus Torvalds
 16 | Alan Cox
 18 | Biber
 19 | Maus (Nagetier)
 21 | Computermaus
 22 | Tastatur
 23 | Preisvorteil
```

[26] SQL92 basiert auf der Relationenalgebra, die nachweislich aufgrund ihrer niedrigen polynomiellen Komplexität, nicht in der Lage ist, die transitive Hülle zu einer zweistelligen Relation/Tabelle zu berechnen, wenn die maximale Zahl der Transitionen nicht vorab bekannt ist bzw. wenn der relationale Ausdruck nicht im Hinblick auf diese maximale Anzahl der Transitionen konzipiert wird. Eine graphentheoretische Begründung für diese Aussage findet sich bspw. in VOSSEN [155, S. 162].

[27] Alternativ zur Verwendung von externen Programmiersprachen ist die Anwendung einer relationalen Datenbank (wie z. B. *DB2* von IBM) denkbar, die rekursive Datenbankanfragen unterstützt.

```
24 | Gemeinde
25 | Loch
26 | Sicherheitslücke
```

Die folgend genannten Schritte sind für jedes, durch die obige Anfrage gefundene Themenblatt auszuführen. Der Wert des Attributs `Id` des Themenblattes, das gerade bearbeitet wird, wird im Folgenden durch die Variable `<ThemaId>` repräsentiert.

1. Um den Vektor eines Themenblattes gemäß Abschnitt 5.1.1.3 zu berechnen, sind folgende Schritte erforderlich: Als erstes wird im Vektor des Themenblattes der Dimensionseintrag, der das Themenblatt selbst repräsentiert, auf Eins gesetzt. Dieses wird durch den folgenden Datenbankbefehl ausgeführt:

   ```
   INSERT INTO Themavektor(Thema, Dimension, Wert)
   VALUES (<ThemaId>, <ThemaId>, 1.0)
   ```

   Am Beispiel des Themas Windows gezeigt, enthält die Tabelle `Themavektor` nun folgenden Eintrag:

   ```
       Thema     |  Dimension   | Wert
   --------------+--------------+------
    11 (Windows) | 11 (Windows) |    1
   ```

   Die Angaben in Klammern hinter den Zahlenwerten sind nachträglich zu einer besseren Übersicht eingefügt worden, um dem Leser das Lesen und Interpretieren der Tabellen zu vereinfachen. Sie enthalten den Text oder die Beschreibung zu einem referenzierten Tabelleneintrag.

2. Als nächstes beginnt eine Schleife, die solange wiederholt wird, bis die im Folgenden vorgestellte Anfrage eine leere Tabelle zurückliefert.

   (a) Die folgende Anfrage sucht zu dem durch `<ThemaId>` repräsentierten Thema alle diejenigen Superthemen, zu denen ein direkt untergeordneter Eintrag im Themenvektor zu `<ThemaId>` existiert, die jedoch selbst im Themenvektor noch nicht eingetragen sind.[28]

   ```
   SELECT DISTINCT ts.Superthema
   FROM   Themenstruktur ts, Themavektor tv
   WHERE  tv.Thema = <ThemaId>
      AND tv.Dimension = ts.Subthema
      AND NOT EXISTS(SELECT tv2.Thema
                     FROM   Themavektor tv2
                     WHERE  tv2.Thema = tv.Thema
                        AND tv2.Dimension = ts.Superthema);
   ```

[28] Eine direkte Suche nach allen Superthemen zu `<ThemaId>` ist, wegen der bereits erwähnten Unfähigkeit rekursive Anfragen in SQL92 zu formulieren, nicht möglich.

(b) Für jedes Superthema (im Folgenden repräsentiert durch `<SuperId>`), das im Ergebnis der obigen Anfrage vorkommt ist ein Eintrag im Vektor des Themas `<ThemaId>` einzufügen:

```
INSERT INTO Themavektor(Thema, Dimension, Wert)
VALUES (<ThemaId>, <SuperId>, 1.0)
```

Im Beispiel des Themas Windows liefert die obige Anfrage beim ersten Schleifendurchlauf folgendes Ergebnis:

```
     Superthema
------------------
 2 (Betriebssystem)
 4 (Microsoft)
```

Nach dem Einfügen der Werte in die Tabelle `Themavektor` hat diese nach dem ersten Schleifendurchlauf folgende Einträge:

```
    Thema     |      Dimension      | Wert
--------------+---------------------+------
 11 (Windows) |  2 (Betriebssystem) |    1
 11 (Windows) |  4 (Microsoft)      |    1
 11 (Windows) | 11 (Windows)        |    1
```

Nach dem zweiten Schleifendurchlauf liefert die obige Abfrage folgendes Ergebnis:

```
 Superthema
------------
 1 (Firma)
```

Die Tabelle `Themavektor` hat nach dem Einfügen des passenden Wertes folgenden Inhalt:

```
    Thema     |      Dimension      | Wert
--------------+---------------------+------
 11 (Windows) |  1 (Firma)          |    1
 11 (Windows) |  2 (Betriebssystem) |    1
 11 (Windows) |  4 (Microsoft)      |    1
 11 (Windows) | 11 (Windows)        |    1
```

Diese Einträge entsprechen dem folgenden Vektor in mathematischer Notation:

$$\vec{\tau}_{\text{Windows}} = (1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, ..., 0)$$

Für das Beispiel mit dem Thema Windows gibt es keinen weiteren Schleifendurchlauf, weil das Ergebnis der obigen Abfrage beim dritten Schleifendurchlauf leer ist.

3. Zum Abschluss muss der Vektor des gerade bearbeiteten Themas normiert werden, indem alle Vektoreinträge durch den Vektorbetrag dividiert werden. Der Vektorbetrag `<Betrag>` kann dem Ergebnis der folgenden Abfrage entnommen werden, die die Wurzel aus der Summe über alle Quadrate der Dimensionseinträge zum Themenvektor des Themas `ThemaId` berechnet.

```
SELECT SQRT(SUM(Wert*Wert))
FROM   Themavektor
WHERE  Thema = <ThemaId>
```

Zur Normierung sind alle Dimensionseinträge des Vektors durch den Betrag zu dividieren:

```
UPDATE Themavektor
SET    Wert = Wert / <Betrag>
WHERE  Thema = <ThemaId>
```

Die Resultate dieses Vorgehens sind im hier betrachteten Beispiel die folgenden Einträge in der Tabelle `Themavektor`:

```
    Thema     |      Dimension      | Wert
--------------+---------------------+------
 11 (Windows) |  1 (Firma)          |  0.5
 11 (Windows) |  2 (Betriebssystem) |  0.5
 11 (Windows) |  4 (Microsoft)      |  0.5
 11 (Windows) | 11 (Windows)        |  0.5
```

Diese Einträge sind mit dem folgenden Vektor in mathematischer Schreibweise äquivalent, womit dieses Verfahren die Gleichung 5.1 auf Seite 121 erfüllt.

$$\vec{\tau}_{\text{Windows}} = |(1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, ..., 0)|$$

Die von diesem und den nachfolgend präsentierten Verfahren verwendeten parametrisierten Anfragen werden von den meisten relationalen Datenbanken derart unterstützt, dass die Anfragen in vorbereiteter und compilierter Form nach der Definition in der Datenbank hinterlegt werden. Somit lässt sich dieses Verfahren auch in Kombination aus SQL und „normalen“ (prozeduralen oder objektorientierten) Programmiersprachen performant umsetzen.

5.3.2.2 Initialisierung der Themenknoten

Nachdem die Themenblätter initialisiert sind, kann die Initialisierung der Themenknoten unter Verwendung einer Schleife vorgenommen werden. Über die folgende Abfrage werden alle diejenigen Themenknoten gesucht, bei denen ein Themenvektor zu allen ihren Subthemen bereits berechnet wurde und dieser in der Tabelle `Themavektor` gespeichert ist. Zusätzlich darf zu den gefundenen Themenknoten noch kein Eintrag in der Tabelle `Themavektor` existieren.

```
SELECT Id, Beschr
FROM   Thema
WHERE  NOT EXISTS(SELECT Thema
                  FROM  Themavektor
                  WHERE Thema = Thema.Id)
   AND (SELECT COUNT(*)
        FROM   Themenstruktur
        WHERE  Thema.Id = Superthema)
       = (SELECT COUNT(*)
          FROM   Themenstruktur
          WHERE  Thema.Id = Superthema
            AND  EXISTS(SELECT * FROM Themavektor
                        WHERE Thema = Subthema));
```

Sollte die obige Anfrage eine leere Tabelle zum Ergebnis haben, dann wird die Schleife verlassen und die Initialisierung der Themenknoten ist beendet. Anderenfalls sind für jedes der gefundenen Themen, die im Folgenden durch die Variable `<ThemaId>` repräsentiert werden, die folgenden Schritte auszuführen und die Schleife ist anschließend von vorne, beginnend mit der obigen Abfrage, abzuarbeiten. Im Beispiel liefert die Anfrage im ersten Schleifendurchlauf folgendes Ergebnis:

```
 Id |    Beschr
----+--------------
  4 | Microsoft
  5 | SCO
 13 | Linux
 17 | Nagetier
 20 | Eingabegerät
```

Somit sind nun folgende Schritte für diese Themen durchzuführen:

1. Für jedes durch die Variable `<ThemaId>` repräsentierte Thema ist der Themenvektor als die Summe über die Vektoren aller Subthemen zu dem Thema zu berechnen und in die Tabelle `Themavektor` einzufügen. Diese Aufgabe wird von folgendem SQL-Befehl erledigt:

   ```
   INSERT INTO Themavektor
   SELECT ts.Superthema, tv.Dimension, SUM(tv.Wert)
   FROM   Themenstruktur ts, Themavektor tv
   WHERE  ts.Superthema = <ThemaId>
      AND ts.Subthema = tv.thema
   GROUP BY ts.Superthema, tv.Dimension;
   ```

   Für das Thema Microsoft stehen nach dem Ausführen des Befehls folgende Daten in der Tabelle `Themavektor`:

```
     Thema     |      Dimension      |       Wert
---------------+---------------------+-------------------
 4 (Microsoft) |  1 (Firma)          | 1.65470053837925
 4 (Microsoft) |  2 (Betriebssystem) | 0.5
 4 (Microsoft) |  4 (Microsoft)      | 1.65470053837925
 4 (Microsoft) |  7 (Bill Gates)     | 0.577350269189626
 4 (Microsoft) |  8 (Steve Ballmer)  | 0.577350269189626
 4 (Microsoft) | 11 (Windows)        | 0.5
```

2. Im nächsten Schritt muss der Themenvektor normiert werden, um die Anforderungen der Gleichung 5.2 auf Seite 5.2 zu erfüllen. Dieses geschieht analog zur Normierung der Themenblätter in Abschnitt 5.3.2.1, indem zunächst der Betrag des Vektors mit folgender Anfrage berechnet wird und in die Variable `<Betrag>` eingestellt wird.

   ```
   SELECT SQRT(SUM(Wert * Wert))
   FROM   Themavektor
   WHERE  Thema = <ThemaId>
   ```

   Anschließend wird die Normierung mit folgendem Befehl auf alle Dimensionseinträge des Vektors übertragen:

   ```
   UPDATE Themavektor
   SET    Wert = Wert / <Betrag>
   WHERE  Thema = <ThemaId>
   ```

   Bezogen auf das Beispiel mit dem Thema Microsoft ergeben sich folgende Einträge in der Tabelle `Themavektor`:

   ```
     Thema     |      Dimension      |       Wert
---------------+---------------------+-------------------
 4 (Microsoft) |  1 (Firma)          | 0.642016166044643
 4 (Microsoft) |  2 (Betriebssystem) | 0.193997690565051
 4 (Microsoft) |  4 (Microsoft)      | 0.642016166044643
 4 (Microsoft) |  7 (Bill Gates)     | 0.224009237739796
 4 (Microsoft) |  8 (Steve Ballmer)  | 0.224009237739796
 4 (Microsoft) | 11 (Windows)        | 0.193997690565051
   ```

Eine Auflistung aller Einträge in der Tabelle `Themavektor`, die die Gesamtresultate des Verfahrens zur Initialisierung der Themenvektoren sind, findet sich im Anhang A.2.1.

5.3.2.3 Ähnlichkeiten zwischen Themen

Die paarweise Ähnlichkeit zwischen Themen kann mit der folgenden View angezeigt werden, die das Skalarprodukt zwischen den Themenvektoren auf Basis der Tabelle `Themavektor` berechnet (vgl. Gleichung 5.3 auf Seite 123):

```
CREATE VIEW ThemenAehnlichkeit AS
SELECT tv1.Thema AS Thema1, tv2.Thema AS Thema2,
       SUM(tv1.Wert * tv2.Wert) AS Wert
FROM   Themavektor tv1, Themavektor tv2
WHERE  tv1.Dimension = tv2.Dimension
GROUP BY tv1.thema, tv2.thema;
```

Der Vollständigkeit halber ist darauf hinzuweisen, dass diese View lediglich diejenigen Einträge enthält, bei denen die Ähnlichkeit einen Wert größer als Null hat. Eine vollständige Auflistung der Ähnlichkeiten für die Beispiel-Ontologie befindet sich im Anhang A.2.2.

5.3.2.4 Skalarprodukte zwischen Interpretationen

Die Berechnung der Skalarprodukte zwischen Interpretationen (gemäß den Gleichungen 5.4 und 5.5 auf Seite 130) kann mit Hilfe von vier aufeinander aufbauenden Views vollständig Datenbank-integriert realisiert werden.

Die erste View dient der Berechnung der unnormierten Interpretationsvektoren und entspricht somit einer einfachen mathematischen Addition aller Termvektoren der Terme, die der Interpretation zugeordnet sind.

```
CREATE VIEW IntVektor_unnorm AS
SELECT it.Interpretation, tv.Dimension, SUM(tv.Wert) AS Wert
FROM   Themavektor tv, IT_Zuo it
WHERE  (tv.Thema = it.Thema)
GROUP BY it.Interpretation, tv.Dimension;
```

Die zweite View baut auf der ersten auf und berechnet den Betrag eines Interpretationsvektors.

```
CREATE VIEW IntVektor_betrag AS
SELECT interpretation, SQRT(SUM(Wert*Wert)) AS Wert
FROM   IntVektor_unnorm
GROUP BY Interpretation;
```

Die dritte View verwendet die beiden ersten Views dazu, um einen normierten Interpretationsvektor zu berechnen und diesen Vektor mit dem jeweiligen Gewicht der Interpretation zu multiplizieren.

```
CREATE VIEW IntVektor AS
SELECT iu.Interpretation, iu.Dimension,
       iu.Wert / ib.Wert * interpretation.Gewicht AS Wert
FROM   IntVektor_unnorm iu, IntVektor_betrag ib, Interpretation
WHERE  iu.Interpretation = ib.Interpretation
   AND Interpretation.id = iu.Interpretation;
```

Die vierte und letzte View baut auf der dritten auf und berechnet die Skalarprodukte durch Multiplikation und Addition der einzelnen Dimensionseinträge der verschiedenen Interpretationen.

```
CREATE VIEW IntAehnlichkeit AS
SELECT iv1.Interpretation AS Int1, iv2.Interpretation AS Int2,
       SUM(iv1.Wert * iv2.Wert) AS Wert
FROM   IntVektor iv1, IntVektor iv2
WHERE  iv1.Dimension = iv2.Dimension
GROUP BY iv1.Interpretation, iv2.Interpretation;
```

Auch an dieser Stelle ist darauf hinzuweisen, dass die View `IntAehnlichkeit` nur für die Kombinationen aus Interpretationen einen Eintrag liefert, bei denen die Vektoren der Interpretationen zueinander nicht orthogonal sind.

Um die Berechnung der Dokumentenähnlichkeiten zu beschleunigen, wird das Skalarprodukt zwischen den Interpretationen in der Tabelle `Aehnlichkeit` mit folgendem Befehl dauerhaft gespeichert:

```
INSERT INTO Aehnlichkeit
SELECT * FROM IntAehnlichkeit;
```

Eine Auflistung aller Einträge in der Tabelle `Aehnlichkeit` für die Beispiel-Ontologie findet sich im Anhang A.2.3.

5.3.3 Einstellen neuer Dokumente

Das Einstellen neuer Dokumente in ein vorhandenes IF-/IR-System, das das eTVSM umsetzt, baut auf den zuvor durch die Initialisierungstransaktion (vgl. Abschnitt 5.3.2) erstellten Daten auf, um die Dokumente in einer für die Abfrage effizienten Form in der Datenbank zu hinterlegen. Zur Illustration des Parsingprozesses und der Dokumenteinstellungstransaktion werden folgende vereinfachte Beispieldokumente in die Datenbank eingestellt:[29]

1. Torvalds schreibt an SCO.
2. McBride warnt die Open-Source-Gemeinde.
3. Windows hat Preisvorteile gegenüber Linux.
4. Microsoft schließt Sicherheitslücken.
5. Neue Bugs in Windows.
6. Mit Maus und Tastatur geht es leichter.
7. Mäuse leben gerne in Löchern.

[29] Die Inhalte der ersten fünf Beispieldokumente sind den Überschriften einiger Artikel entnommen worden, die im September 2003 im *Heise-Newsticker* erschienen sind. Die Web-Adresse des *Heise-Newstickers* lautet: `http://www.heise.de/newsticker`

Die Dokumente sind mit der Beispiel-Ontologie aus Abschnitt 5.2.2 kompatibel, wenn die folgende Annahme getroffen wird: Alle in der Ontologie nicht enthaltenen Wörter bzw. Wortstämme werden als Stoppwörter angesehen.

Das Einstellen von neuen Dokumenten geschieht in drei Schritten: Als erstes werden die Dokumente durch das Parsing (Abschnitt 5.3.3.1) in eine Liste, die aus den einzelnen Wörtern der Dokumente besteht, zerlegt und gespeichert. Im zweiten Schritt wird diese Liste aufgegriffen und es werden aus dieser Liste die Zuordnungen des Dokuments zu einzelnen Interpretationen gewonnen (Abschnitt 5.3.3.2). Dieser Schritt impliziert u. a. das Stemming der einzelnen Worte. Zum Abschluss wird zu jedem neu eingefügten Dokument der Dokumentenbetrag berechnet und in der Datenbank hinterlegt.

5.3.3.1 Parsing

Bevor mit dem Parsing des Dokuments begonnen wird, wird dem Dokument zunächst eine eindeutige Nummer `<Dokument_Id>` zugewiesen und das Dokument wird in der Tabelle `Dokument` zusammen mit dem Originaltext `<Dokument_Text>` gespeichert. Das Speichern kann von dem folgenden SQL-Statement vorgenommen werden:

```
INSERT INTO Dokument(Id, Text)
VALUES(<Dokument_Id>, <Dokument_Text>);
```

Der Inhalt der Tabelle `Dokument` nach dem Einfügen der sieben Beispieldokumente findet sich im Anhang A.3.1.

Im nächsten Schritt werden die Dokumente geparst. In diesem Fall bedeutet das konkret, dass zunächst alle Sonderzeichen, also alle Zeichen, die keine Buchstaben und die keine Ziffern darstellen, entfernt werden. Sollte ein Dokument Formatierungen enthalten, dann ist an dieser Stelle festzulegen, wie diese behandelt werden sollen. Üblicherweise werden Formatierungen vollständig entfernt und somit ignoriert.[30] Als nächstes wird aus dem von den Dokumenten Übriggebliebenem eine Liste von einzelnen Wörtern extrahiert. Diese Liste hat z. B. bei dem zweiten Dokument das folgende Aussehen:

(McBride, warnt, die, Open, Source, Gemeinde)

Im letzten Schritt werden die einzelnen Listen in der Datenbank durch das Einfügen von neuen Einträgen in den Tabelle `DW_Zuo` gespeichert. Hierbei ist zu entscheiden, wie mit Wörtern umgegangen werden soll, die in der Tabelle `Wort` nicht enthalten sind. Die folgenden zwei Alternativen stehen zur Auswahl:

1. In Dokumenten vorkommende Wörter, die in der Tabelle `Wort` nicht vorkommen, werden beim Parsing automatisch in der Tabelle angelegt. Der Administrator des IF-/IR-Systems wird über diesen Sachverhalt informiert, so dass er diese Wörter vor der wei-

[30] Alternativ ist es denkbar, formatierte Wörter (z. B. Fettdruck, Überschriften etc.) schwerer zu gewichten, indem die formatierten Passagen z. B. verdoppelt werden. Dadurch werden diese Passagen bzw. Wörter bei der späteren Aggregation der Wortliste zu Interpretationshäufigkeiten in Abschnitt 5.3.3.2 doppelt gewichtet.

teren Verarbeitung prüfen kann und ggf. Modifikationen an Zuordnungen (z. B. die Zuordnung eines Wortes zu einem bestehenden Wortstamm, für den einfachen Fall, dass die Flexionsform in der Ontologie vergessen wurde) oder an der Ontologie (z. B. Erstellen eines neuen Themas oder einer neuen Interpretation) vornehmen kann.[31]

2. Wörter, die in der Tabelle `Wort` nicht vorkommen, werden als Stoppwörter behandelt und somit ignoriert.[32]

Aus didaktischen Gründen wird zur Illustration des Beispiels die zweite Alternative gewählt.[33] Einträge in die Tabelle `DW_Zuo` können z. B. über das folgende Statement in der Datenbank gespeichert werden. Dieses Statement ist für jeden Listeneintrag genau einmal auszuführen. Dabei enthalten die Variablen `<Wort>` und `<Position>` die Zeichenfolge und die Position des Wortes in dem jeweils gerade betrachteten Dokument `<Dokument_Id>`.

```
INSERT INTO DW_Zuo
SELECT <Dokument_Id>, Wort.Id, <Position>
FROM   Wort
WHERE  Text = <Wort>;
```

Das obige Statement fügt nur dann eine neue Zeile in die Tabelle `DW_Zuo` ein, wenn das `<Wort>` in der Tabelle `Wort` vorhanden ist. Somit enthält die Tabelle `DW_Zuo` beispielsweise nach dem Abarbeiten der Liste des zweiten Dokuments folgende Einträge:

```
 Dokument |       Wort      | Position
----------+-----------------+----------
        2 |   30 (McBride)  |        1
        2 |    3 (Open)     |        4
        2 |   43 (Source)   |        5
        2 |   24 (Gemeinde) |        6
```

Eine Auflistung der Einträge der Tabelle `DW_Zuo` für alle sieben Dokumente findet sich im Anhang A.3.2.

---

[31] Die erste Alternative verstößt jedoch gegen die Kardinalität (1,1) des Relationshiptyps WW-Zuo in Bezug auf den Entitytyp Wort im eTVSM-Datenmodell in Abbildung 5.1 auf Seite 111. Zum Zwecke einer besseren Administration des IF-/IR-Systems ist es jedoch sinnvoll, diese Kardinalität zu modifizieren.

[32] Genau genommen verstößt die zweite Alternative gegen die in Abschnitt 5.1.3 gemachten Aussagen zur Repräsentation von Stoppwörtern. Da jedoch in einem realen Anwendungsszenario niemals alle jemals in zukünftigen Dokumenten vorkommenden Wörter vorab erfasst werden können, ist es unter Umständen sinnvoll, die zweite Alternative zuzulassen.

[33] Die Beispiel-Ontologie aus Abschnitt 5.2.2 enthält außer Substantiven keine Präpositionen, Verben, etc.. Somit werden alle Nicht-Substantive in diesem Beispiel automatisch zu Stoppwörtern. Dieses ist in sofern möglich/erlaubt, weil der thematische Bezug eines Dokuments im Allgemeinen am besten über Substantive festgemacht werden kann und weil die Nicht-Substantive bezüglich einer thematischen Einordnung nur einen gering Beitrag leisten können. Zudem kann die Beispiel-Ontologie problemlos um weitere Themen, Interpretationen, Terme etc. erweitert werden. Die starke Einschränkung der Themen in der Ontologie zu einem „toy example“ ist nur deshalb gemacht worden, um die Beispiel-Ontologie nicht unnötig ausufern zu lassen, was weder der Verständlichkeit noch der Nachvollziehbarkeit an dieser Stelle dienlich wäre.

5.3.3.2 Zuordnung zu Interpretationen

Ein Ziel des Einfügens von Dokumenten in die Datenbank ist, die Dokumente derart zu speichern, dass ein Vergleich von Dokumenten auf Ähnlichkeiten möglichst effizient durchgeführt werden kann. Zu diesem Zweck existiert die Tabelle `DI_Zuo`, die den Dokumenten einzelne, in den Dokumenten vorkommende Interpretationen zuordnet. Die Einträge in der Tabelle `DI_Zuo` lassen sich aus den Einträgen in `DW_Zuo` in zwei Schritten herleiten.

Im ersten Schritt werden die Wörter, die einem Dokument zugeordnet sind, gestemmt und zu Termen umgewandelt, was unter Umständen bedeutet, dass mehrere Wörter bzw. Wortstämme zu einem Term zusammengefasst werden müssen. An dieser Stelle wird eine gemischte Lösung für das Problem am Beispiel des ersten und zweiten Dokuments präsentiert, die sowohl auf prozedurale als auch auf relationale Konzepte zurückgreift. Diese Lösung zeichnet sich durch einen relativ einfach nachvollziehbaren Aufbau aus.

Das Vorgehen sieht dabei wie folgt aus: Für jedes Dokument (hier repräsentiert durch die Variable `<Dokument_Id>`) und für jede Position in dem Dokument (repräsentiert durch die Variable `<Position>`) ist zu überprüfen, wie viele und welche Terme aus dem Wort in dem Dokument `<Dokument_Id>` an der Position `<Position>` abgeleitet werden können[34] und aus wievielen Wörtern jeder dieser ableitbaren Terme besteht. Der soeben definierte Informationsbedarf wird durch folgende Anfrage erfüllt:

```
SELECT wt.Term AS Term, (SELECT COUNT(*)
                         FROM   wt_zuo
                         WHERE  wt_zuo.Term = wt.Term) AS wz
FROM   DW_Zuo dw, Wort w, WT_Zuo wt
WHERE  dw.Dokument = <Dokument_Id>
   AND dw.Position = <Position>
   AND w.Id = dw.Wort
   AND wt.Wortstamm = w.Wortstamm
   AND wt.Position = 1
ORDER BY wz;
```

Diese Anfrage liefert beispielsweise für das zweite Dokument und die erste Position das folgende Ergebnis:

```
      Term      | wz
----------------+----
   30 (McBride) |  1
```

In einem solchen Fall ist klar, dass dem zweiten Dokument an der ersten Position der Term McBride zuzuordnen ist, weil dieser Term aus nur einem Wort (`wz = 1`) besteht und der einzige mögliche Term ist.

Anders ist die Situation für die vierte Position im zweiten Dokument. Hier liefert die obige Anfrage das folgende Ergebnis:

[34] Ausschließlich unter Betrachtung dieser einen Position.

```
        Term        | wz
--------------------+----
    3 (Open Source) |  2
```

In diesem Fall ist zwar der Term Open Source die einzige Alternative, allerdings ist noch nicht sicher, ob dieser Term in dem Dokument tatsächlich vorkommt, weil bisher nur eine Position betrachtet wurde. Der Term aber besteht aus zwei Wörtern und nimmt somit zwei Positionen ein. Somit muss eine zweite Anfrage an die Datenbank gestellt werden, die prüft, welche Terme aus zwei Wörtern bestehen, die zu den Wörtern im Dokument an zwei zusammenhängenden Positionen passen:

```
SELECT wt1.Term AS Term, (SELECT COUNT(*)
                          FROM   wt_zuo
                          WHERE  wt_zuo.Term = wt1.Term) AS wz
FROM   DW_Zuo dw1, Wort w1, WT_Zuo wt1,
       DW_Zuo dw2, Wort w2, WT_Zuo wt2
WHERE  dw1.Dokument = <Dokument_Id>
   AND dw1.Position = <Position>
   AND w1.Id = dw1.Wort
   AND wt1.Wortstamm = w1.Wortstamm
   AND wt1.Position = 1
   AND dw2.Dokument = dw1.Dokument
   AND dw2.Position = dw1.Position + 1
   AND w2.id = dw2.Wort
   AND wt2.Wortstamm = w2.Wortstamm
   AND wt2.Position = wt1.Position + 1
   AND wt1.Term = wt2.Term
ORDER BY wz;
```

Das Ergebnis dieser zweiten Anfrage ist für das zweite Dokument und die vierte Position identisch mit dem Ergebnis der ersten Anfrage:

```
        Term        | wz
--------------------+----
    3 (Open Source) |  2
```

Somit ist es klar, dass dem zweiten Dokument an der vierten Position der Term Open Source zugeordnet werden muss, weil es nur einen aus zwei Wörtern zusammengesetzten Term gibt, der zu den beiden Positionen im Dokument passt. An der fünften Position wird dem Dokument nichts zugeordnet.

Alternativ, am Beispiel des ersten Dokuments zeigt sich eine andere Situation beim Betrachten der vierten Position. Die erste Anfrage liefert das folgende Ergebnis:

```
       Term      | wz
-----------------+----
    5 (SCO)      |  1
   12 (SCO Unix) |  2
```

Somit muss auch die zweite Anfrage ausgeführt werden, weil es mehrere mögliche Terme gibt, die zu dem Wort an der vierten Position (bei alleiniger Betrachtung dieser Position) passen. In diesem Fall führt jedoch das Ausführen der zweiten Anfrage zu einer leeren Tabelle. Somit ist klar, dass dem ersten Dokument an der vierten Position der Term SCO zugewiesen werden muss, weil der Term SCO Unix beim Betrachten der vierten und fünften Position herausfällt, was dadurch bedingt wird, dass es im Dokument keine fünfte Position gibt.

Ignoriert man die nicht belegten Positionen, dann ist das Ergebnis des ersten Schrittes eine Menge von Listen (eine pro Dokument), die aus Termen bestehen. Für die sieben Beispieldokumente werden durch das Verfahren folgende Listen hergeleitet:

1. (Torvalds, SCO)
2. (McBride, Open Source, Gemeinde)
3. (Windows, Preisvorteil, Linux)
4. (Microsoft, Sicherheitslücke)
5. (Bug, Windows)
6. (Maus, Tastatur)
7. (Maus, Loch)

Anzumerken ist, dass die oben vorgestellten Anfragen das Stemming der Wörter durch das Auflösen des Fremdschlüsselattributes `Wortstamm` in der Tabelle `Wort` implizieren. Des Weiteren können mit dem Verfahren auch aus mehr als zwei Wörtern zusammengesetzte Terme berücksichtigt werden, wenn bei einem nicht eindeutigen Ergebnis der zweiten Anfrage eine dritte Anfrage gestellt wird, die drei aufeinanderfolgende Positionen berücksichtigt. Dieses Vorgehensmodell kann somit bei Bedarf sukzessive erweitert werden, bis die maximal gewünschte Anzahl an Wörtern pro zusammengesetzten Term berücksichtigt wird. Alternativ können auch andere, rekursive Verfahren implementiert werden, die keine feste Begrenzung der maximalen Anzahl der Wörter pro Term haben. Diese Verfahren sind allerdings aufwändiger, weil diese u. a. nicht in dem Maße auf die Auswertungsmöglichkeiten einer relationalen Datenbank zurückgreifen können, oder weil diese die Datenbankanfragen dynamisch zur Laufzeit aus Bausteinen zusammensetzen müssen.

Im zweiten Schritt werden aus den zu den Dokumenten gefundenen Termen die Interpretationen abgeleitet, die den Dokumenten zugeordnet werden. Dazu wird für jeden Term in der Termliste eines Dokuments geprüft, welche Interpretationen zu diesem Term passen. Ist einem Term über die `TI_Zuo` lediglich nur eine einzige Interpretation zugeordnet, dann wird dem Dokument diese Interpretation zugeordnet. Anderenfalls, wenn mehrere Interpretationen in Frage kommen, ist anhand der Supportterme zu prüfen, welche der verschiedenen Interpretationen die geeignete ist. Dabei wird für jede Interpretation ϕ, zu der Supportterme in der Tabelle `Supportterm` definiert wurden, geprüft, ob die Supportterme in der näheren

Umgebung[35] des betrachteten Terms vorkommen. Ist dieses der Fall und gibt es keine andere Interpretation, deren Supportterme in der näheren Umgebung vorkommen, dann wird die Interpretation ϕ dem Dokument zugewiesen. Anderenfalls wird diejenige Interpretation dem Dokument zugewiesen, zu der keine Supportterme definiert wurden.

Im Beispiel mit dem sechsten Dokument wird der Term Maus durch die Interpretation $\phi_{\text{Computermaus}}$ ersetzt, weil der Supportterm Tastatur im Dokument enthalten ist. Im siebten Dokument hingegen wird der Term Maus durch die Interpretation $\phi_{\text{Maus(Nagetier)}}$ ersetzt, weil der Term Loch ein Supportterm dieser Interpretation ist. Ein Zwischenergebnis des zweiten Schrittes sind somit Listen von Interpretationen zu den einzelnen Dokumenten. Im Beispiel der sieben Dokumente ergeben sich folgende Listen:

1. $(\phi_{\text{Linux Torvalds}}, \phi_{\text{SCO}})$
2. $(\phi_{\text{Darl McBride}}, \phi_{\text{Open Source}}, \phi_{\text{Gemeinde}})$
3. $(\phi_{\text{Windows}}, \phi_{\text{Preisvorteil}}, \phi_{\text{Linux}})$
4. $(\phi_{\text{Microsoft}}, \phi_{\text{Sicherheitslücke}})$
5. $(\phi_{\text{Sicherheitslücke}}, \phi_{\text{Windows}})$
6. $(\phi_{\text{Computermaus}}, \phi_{\text{Tastatur}})$
7. $(\phi_{\text{Maus(Nagetier)}}, \phi_{\text{Loch}})$

Jede dieser Listen von Interpretationen wird zum Abschluss nach den Interpretationen gruppiert und aggregiert, so dass zu jeder Interpretation eine Vorkommenshäufigkeit pro Dokument eingetragen werden kann, sofern diese Interpretation im Dokument vorkommt. Das Ergebnis dieser Eintragung sind z. B. für das sechste und siebte Dokument folgende Zeilen in der Tabelle `DI_Zuo`.

```
 Dokument |   Interpretation    | Anzahl
----------+---------------------+--------
        6 | 21 (Computermaus)   |      1
        6 | 22 (Tastatur)       |      1
        7 | 19 (Maus(Nagetier)) |      1
        7 | 25 (Loch)           |      1
```

Ein vollständiger Auszug dieser Tabelle für alle Dokumente des Beispiels findet sich im Anhang A.3.3.

[35] Der Begriff der *näheren Umgebung* ist für eine Implementierung zu operationalisieren. Werden von dem IF-/IR-System ausschließlich kurze, Nachrichtenticker-ähnliche Dokumente verarbeitet, dann ist es sinnvoll, als nähere Umgebung das gesamte Dokument zu definieren, weil derartige Dokumente im Allgemeinen monothematisch sind. Anderenfalls kann z. B. die Distanz zwischen den Termen in der Liste als ein Maß für die Nähe dienen. Alternativ können Formatierungen des Dokuments (wie z. B. Abschnitte oder Absätze) zur Definition der Nähe verwendet werden.

5.3.3.3 Berechnung der Dokumentenbeträge

Die Berechnung der, für die Ähnlichkeitsberechnung notwendigen Beträge der Dokumentenvektoren gemäß Gleichung 5.8 auf Seite 138, lässt sich vollständig innerhalb einer relationalen Datenbank mit Hilfe der folgenden View berechnen:

```
CREATE VIEW DokBetrag AS
SELECT di1.Dokument,
       SQRT(SUM(di1.Anzahl * di2.Anzahl
                * aehn.Skalarprodukt)) AS Wert
FROM   DI_Zuo di1, DI_Zuo di2, Aehnlichkeit aehn
WHERE  di1.Dokument = di2.Dokument
   AND di1.Interpretation = aehn.Int1
   AND di2.Interpretation = aehn.Int2
GROUP BY di1.Dokument;
```

Somit lassen sich die Beträge mit dem folgenden, relativ einfachen SQL-Statement für alle Dokumente, zu denen diese Beträge noch nicht berechnet wurden, berechnen und in dem Attribut `Betrag` der Tabelle `Dokument` speichern:

```
UPDATE Dokument
SET Betrag = (SELECT Wert
              FROM   DokBetrag
              WHERE  Dokument.Id = DokBetrag.Dokument)
WHERE Betrag IS NULL;
```

Die daraus resultierenden Einträge in der Tabelle `Dokument` für das Beispiel mit den sieben Dokumenten findet sich im Anhang A.3.4.

5.3.4 Anfrageausführung

Die Berechnung der Ähnlichkeiten zwischen zwei Dokumenten kann – analog zum TVSM – vollständig innerhalb der relationalen Datenbank ausgeführt werden. Zur Umsetzung der Gleichung 5.9 auf Seite 139, die die Ähnlichkeit zwischen zwei Dokumenten definiert, sind zwei Views auf Datenbankebene erforderlich: Die erste View berechnet die paarweise Ähnlichkeit zwischen Dokumenten, ohne jedoch diese Ähnlichkeit mit dem Betrag der Dokumentenvektoren zu normieren.

```
CREATE VIEW DokAehn_unnorm AS
SELECT di1.Dokument AS Dok1, di2.Dokument AS Dok2,
       SUM(di1.Anzahl * di2.Anzahl * aehn.Skalarprodukt) AS Wert
FROM   DI_Zuo di1, DI_Zuo di2, Aehnlichkeit aehn
WHERE  di1.Interpretation = aehn.Int1
   AND di2.Interpretation = aehn.Int2
GROUP BY di1.Dokument, di2.Dokument;
```

Die Normierung der Ähnlichkeit wird von der zweiten View, die auf dem Ergebnis der ersten View aufbaut, berechnet.

```
CREATE VIEW DokAehn AS
SELECT da.Dok1, da.Dok2, da.Wert / d1.Betrag / d2.Betrag AS Wert
FROM   DokAehn_unnorm da, Dokument d1, Dokument d2
WHERE  da.Dok1 = d1.Id
   AND da.Dok2 = d2.Id;
```

Die Ähnlichkeit zwischen dem vierten Dokument aus dem Beispiel und allen anderen Dokumenten kann somit relativ einfach über die folgende Anfrage ermittelt werden (hier mit absteigender Sortierung bezüglich der errechneten Ähnlichkeit):

```
SELECT *
FROM   DokAehn
WHERE  Dok1 = 4
ORDER BY Wert DESC;
```

Das Ergebnis dieser Anfrage sieht wie folgt aus:

```
 Dok1 | Dok2 |       Wert
------+------+-------------------
    4 |    4 | 1.000000000000000
    4 |    5 | 0.918006928304847
    4 |    3 | 0.351989763853554
    4 |    1 | 0.245140411270901
    4 |    2 | 0.174187508636031
```

Man kann feststellen, dass das vierte Dokument erwartungsgemäß zu sich selbst die maximale Ähnlichkeit von Eins hat. Interessant ist, dass das vierte Dokument (Microsoft schließt Sicherheitslücken) mit einem Wert von ca. 0,918 eine relativ hohe Ähnlichkeit zum fünften Dokument (Neue Bugs in Windows) hat, obwohl beide Dokumente *keine gemeinsamen* Terme haben. Die hohe Ähnlichkeit begründet sich durch die ontologischen Zusammenhänge der einzelnen Terme: In der Beispiel-Ontologie sind Sicherheitslücke und Bug als Synonyme definiert worden und Windows ist ein Subthema von Microsoft.

Des Weiteren fällt auf, dass die Dokumente eins, zwei und drei als entfernt ähnlich zu dem vierten Dokument eingestuft werden. Dieses liegt daran, dass diese Dokumente einige wenige, teilweise auch über „mehrere Ecken“ verwandte Interpretationen mit dem vierten Dokument gemeinsam haben. Zusätzlich kann man feststellen, dass die Dokumente sechs und sieben in dem Ergebnis nicht auftauchen. Dieses bedeutet, dass die Ähnlichkeit zwischen dem vierten Dokument und den Dokumenten sechs und sieben genau Null beträgt. Die View zur Berechnung der Dokumentenähnlichkeit liefert immer nur dann einen Tabelleneintrag, wenn die Ähnlichkeit zwischen zwei Dokumenten größer als Null ist. Dieses ist für die praktische Anwendung des hier vorgestellten Konzeptes eher nützlich denn schädlich, weil Dokumente mit einer Null-Ähnlichkeit beim IF bzw. IR für den Benutzer (unter der Annahme, dass die Einschätzung des Systems richtig ist) nicht relevant sind. Ein vollständiger Auszug der Ergebnisse der View DokAehn findet sich im Anhang A.4.1.

5.4 Vergleich mit anderen Modellen / Kritik

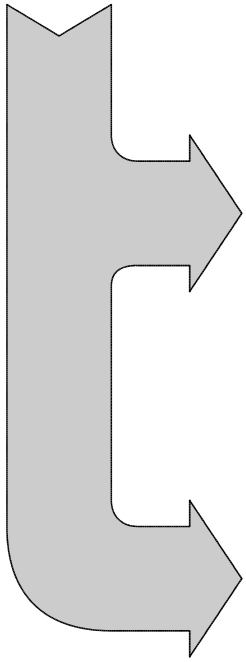
Das eTVSM baut deutlich sichtbar auf dem TVSM auf und beseitigt die in dem Abschnitt 4.7 am TVSM negativ angemerkten Kritikpunkte. Dieses geschieht insbesondere dadurch, dass das eTVSM die Termähnlichkeiten durch die Einführung von Themen und Themenstrukturen operationalisiert und dadurch, dass Stoppwortliste und Stemming explizit in einem eigenen Konzept im Datenmodell berücksichtigt werden. Zusätzlich führt das eTVSM Konzepte ein, um Wortgruppen bzw. zusammengesetzte Terme, Homographen und Metonymie zu berücksichtigen.

Dadurch, dass das eTVSM auf dem TVSM aufbaut, gehört es zur Kategorie der Modelle mit transzendenten Terminterdependenzen mit direkt in Form einer Ontologie vorgegebenen Terminterdependenzen. Die sich somit ergebenden qualitativen Unterschiede zu Modellen anderer Kategorien sind bereits in Abschnitt 3.5 diskutiert worden und gelten auch in Bezug auf das eTVSM. Innerhalb der Kategorie der Modelle mit transzendenten Terminterdependenzen mit direkt vorgegebenen Interdependenzen unterscheidet sich das eTVSM zu den anderen Modellen durch dieselben Merkmale, durch die sich das eTVSM zum TVSM unterscheidet (siehe oben). Gegenüber dem BNN hat das eTVSM den Nachteil, dass es bezüglich dem Maß der Ähnlichkeiten zwischen einzelnen Termen durch die in Abschnitt 5.1.1.1 genannten Konsistenzbedingungen stärker eingeschränkt ist. Zum Vorteil gereicht es dem eTVSM hingegen, dass der Aufwand für das Aufstellen einer expliziten Ontologie immer noch geringer sein dürfte, als der Aufwand Millionen von unterschiedlichen Einzelbeispielen zu erstellen, die die Ontologie indirekt beschreiben.

In der Arbeit sind die Unterschiede zwischen den einzelnen Modellen bisher ausschließlich anhand von qualitativen Merkmalen diskutiert worden. Im Folgenden sollen die Unterschiede zwischen dem eTVSM und dem VSM, welches an dieser Stelle exemplarisch für die Modelle ohne Termähnlichkeiten steht, an den Beispieldokumenten aus Abschnitt 5.3.3 illustriert werden. Zu diesem Zwecke zeigt die Abbildung 5.26 alle paarweisen Dokumentenähnlichkeiten zwischen den Dokumenten, einmal mit dem eTVSM und einmal mit dem VSM berechnet. Die Berechnung der Ähnlichkeiten mit dem VSM beruht dabei auf denselben Daten, wie die Berechnung der Ähnlichkeiten mit dem eTVSM. Das dabei gewählte Vorgehen wird im Detail im Anhang B beschrieben.

Beim Vergleich der beiden Ähnlichkeitsmatrizen fällt auf, dass die Matrix des eTVSM dichter belegt ist (das heißt: mehr Einträge ungleich Null hat), als die Matrix des VSM. Dieses begründet sich darin, dass das eTVSM durch die Berücksichtigung von Synonymen und thematischen Beziehungen über die Themenstruktur auch unterschiedlichen Termen Ähnlichkeiten zuweist. Allerdings wird dieser globale Effekt teilweise durch das Berücksichtigen von Homographen kompensiert, indem dieselben Worte unterschiedliche Interpretationen zugewiesen bekommen. Konkret kann man z. B. erkennen, dass die thematisch ähnlichen Dokumente vier und fünf vom eTVSM korrekterweise eine hohe Ähnlichkeit ($0,918$) zugewiesen bekommen. Das VSM ist nicht in der Lage, eine Ähnlichkeit zwischen den beiden Dokumenten auszumachen (die Ähnlichkeit beträgt Null), weil diese Dokumente keine gemeinsamen Worte haben. Bei den Dokumenten drei und fünf ist der Unterschied nicht so gravierend, weil

1. Torvalds schreibt an SCO.
2. McBride warnt die Open-Source-Gemeinde.
3. Windows hat Preisvorteile gegenüber Linux.
4. Microsoft schließt Sicherheitslücken.
5. Neue Bugs in Windows.
6. Mit Maus und Tastatur geht es leichter.
7. Mäuse leben gerne in Löchern.



Dokumentenähnlichkeiten: eTVSM

	1	2	3	4	5	6	7
1	1,000	0,660	0,615	0,245	0,292	0,000	0,000
2	0,660	1,000	0,387	0,174	0,177	0,000	0,000
3	0,615	0,387	1,000	0,352	0,469	0,000	0,000
4	0,245	0,174	0,352	1,000	0,918	0,000	0,000
5	0,292	0,177	0,469	0,918	1,000	0,000	0,000
6	0,000	0,000	0,000	0,000	0,000	1,000	0,000
7	0,000	0,000	0,000	0,000	0,000	0,000	1,000

Dokumentenähnlichkeiten: VSM

	1	2	3	4	5	6	7
1	1,000	0,000	0,000	0,000	0,000	0,000	0,000
2	0,000	1,000	0,000	0,000	0,000	0,000	0,000
3	0,000	0,000	1,000	0,000	0,224	0,000	0,000
4	0,000	0,000	0,000	1,000	0,000	0,000	0,000
5	0,000	0,000	0,224	0,000	1,000	0,000	0,000
6	0,000	0,000	0,000	0,000	0,000	1,000	0,293
7	0,000	0,000	0,000	0,000	0,000	0,293	1,000

Abbildung 5.26: Vergleich der Dokumentenähnlichkeiten zwischen eTVSM und VSM.

beide Dokumente das gemeinsame Wort Windows haben. Allerdings weist das eTVSM dem Dokumentenpaar eine höhere Ähnlichkeit zu als das VSM ($0,469$ gegenüber $0,224$). Umgekehrt sieht die Situation bei den beiden Dokumenten sechs und sieben aus: das eTVSM weist dem Dokumentenpaar eine Ähnlichkeit von Null zu, weil der Term Maus als Homograph mit jeweils unterschiedlicher Interpretation in den beiden Dokumenten erkannt wird. Das VSM hingegen weist dem Dokumentenpaar die Ähnlichkeit von $0,293$ zu, weil die Dokumente den Term Maus gemeinsam haben.

Als Fazit kann man zusammenfassen, dass das eTVSM ein neues und vielversprechendes Konzept zur Lösung der IF- und IR-Problematik darstellt, dessen Praktikabilität und Vorteile in einem kleinen Maßstab nachgewiesen sind. Jedoch steht ein Nachweis der Praktikabilität und der Vorteile des eTVSM in einem großen Maßstab noch aus und sollte daher in der Zukunft in Angriff genommen werden.

Kapitel 6

Anwendung des eTVSM in der Praxis

6.1 Ontologien für das eTVSM

Wie in Kapitel 5 gezeigt wurde, stellt eine extern vorgegebene Ontologie beim eTVSM das Fundament zur Bestimmung von paarweisen Dokumentenähnlichkeiten dar. Daher ist es notwendig, dass man sich vor der Anwendung des eTVSM für praktische Zwecke des IF und IR Gedanken macht, wie die benötigte Ontologie erstellt werden soll. Das Erstellen von Ontologien von Grund auf ist aufwändig, weshalb die Wiederverwendung von bereits vorhandenen Ontologien in Betracht gezogen werden sollte. Die beiden folgenden Unterabschnitte 6.1.1 und 6.1.2 befassen sich mit diesen beiden Möglichkeiten zur Gewinnung von Ontologien für das eTVSM.

6.1.1 Erstellung einer Ontologie

Beim Erstellen einer Ontologie von Grund auf muss sich eine Gruppe von Benutzern zusammensetzen und die Ontologie im Konsens gemeinsam modellieren. Hierbei lassen sich zwei Vorgehensweisen unterscheiden: Zum Einen kann die Ontologie auf einmal aufgestellt werden und dann als relativ statisch akzeptiert werden (*Big-Bang-Vorgehen*) oder die Ontologie wird im laufenden Betrieb des IF/IR-Systems sukzessive erweitert (*Sukzessives-Vorgehen*). Als Ausgangsbasis beim Sukzessiven-Vorgehen dient dabei eine automatisch erstellbare Trivial-Ontologie, bei der alle Terme, die in dem Dokumentenbestand vorkommen, als zueinander unabhängig angesehen werden.

Der Vorteil des Big-Bang-Vorgehen ist, dass durch eine längere Diskussion der beteiligten Modellierer vor dem Einsatz des Systems, Fehler in der Modellierung vermieden werden können. Der Nachteil ist jedoch, dass der Modellierungsprozess an sich sehr lang und aufwän-

dig ist und dass das IF/IR-System während dieser Zeit nicht einsetzbar ist. Dieser Sachverhalt sieht bei dem Sukzessiven-Vorgehen genau umgekehrt aus. Das System ist sofort nach dem Erstellen der Trivial-Ontologie einsetzbar, die Qualität des Systems entspricht jedoch lediglich der Qualität eines gewöhnlichen IF-/IR-Systems auf VSM-Basis. Bei Bedarf lässt sich die Qualität in abgegrenzten Gebieten erhöhen, indem diese Gebiete in der Ontologie modelliert werden. Der Nachteil dieses Vorgehens ist klar: Es besteht die Gefahr, dass viele lokale, fachgebietsspezifische Modelle entstehen, die zusammengenommen zueinander nicht konsistent bzw. kompatibel sind.

Grundsätzlich ist anzumerken, dass das manuelle Erstellen einer vollständigen Ontologie von Grund auf, wegen des extremen Arbeitsaufwandes und der vielen Terme in natürlichsprachlichen Dokumenten eher unpraktikabel ist. Jedoch kann der Erstellungsprozess einer Ontologie stark vereinfacht werden, wenn es sich bei den verwalteten Dokumenten um fachspezifische Dokumente mit fachspezifischen und kontrolliertem (eng eingegrenztem) Vokabular handelt und zudem die Zusammenhänge zwischen den Termen bereits z. B. durch Fachbegriffsmodelle oder Gesetzes- bzw. Regeltexte vorgegeben sind. Aufgrund der zunehmenden Bedeutung von Ontologien in verschiedenen Anwendungsbereichen der Informatik (wie z. B. dem der Software-Agenten insbesondere unter dem Stichwort Semantic Web oder des IF und IR) wird zur Zeit verstärkt an Werkzeugen und Methoden zur Erstellung von Ontologien geforscht.[1] Neben der reinen Unterstützung der manuellen Tätigkeit der Ontologieentwicklung werden auch zunehmend Verfahren und Werkzeuge zur teilautomatisierten Herleitung von Ontologien aus Datenbankschemata, natürlichsprachlichen Lexika und größeren natürlichsprachlichen Textkorpora vorgestellt. Ein Beispiel für ein derartiges System ist das *Text-To-Onto* Werkzeug von MAEDCHE [92].

6.1.2 Nutzung vorhandener Ontologien

Da das manuelle Erstellen von Ontologien einen hohen Aufwand bedeutet, kann es durchaus ökonomischer sein, bereits vorhandene, z. B. von Linguisten erstellte, Ontologien zu nutzen und ggf. bei Bedarf um fachspezifisches Vokabular und fachspezifische Zusammenhänge zu erweitern. Im Folgenden werden drei bekannte Ontologien (zwei für die deutsche und eine für die englische Sprache) vorgestellt.

6.1.2.1 Wortschatz-Lexikon

Das *Wortschatz-Lexikon*[2] entstand Anfang der 90er Jahre am Institut für Informatik der Universität Leipzig und ist mittlerweile mit seinen ca. 2,5 Mio Wortformen eine der umfangreichsten frei zugänglichen Datensammlungen der deutschen Sprache, die zudem seit 1998 über das Internet frei zugänglich ist (vgl. Abbildung 6.1). Bei dem Lexikon handelt es sich um eine

[1] Bezüglich der Methoden zum Engineering von Ontologien sei hier auf die Arbeiten von USCHOLD und KING [148], USCHOLD und GRUNINGER [149], GUARINO und WELTY [62] sowie BATEMAN [8, 9] verwiesen. Prototypische Implementierungen von Modellierungswerkzeugen für die Ontologieentwicklung sind z. B. *Protégé* [57] und *OilEd* [72].

[2] Wortschatz-Lexikon: `http://wortschatz.uni-leipzig.de`

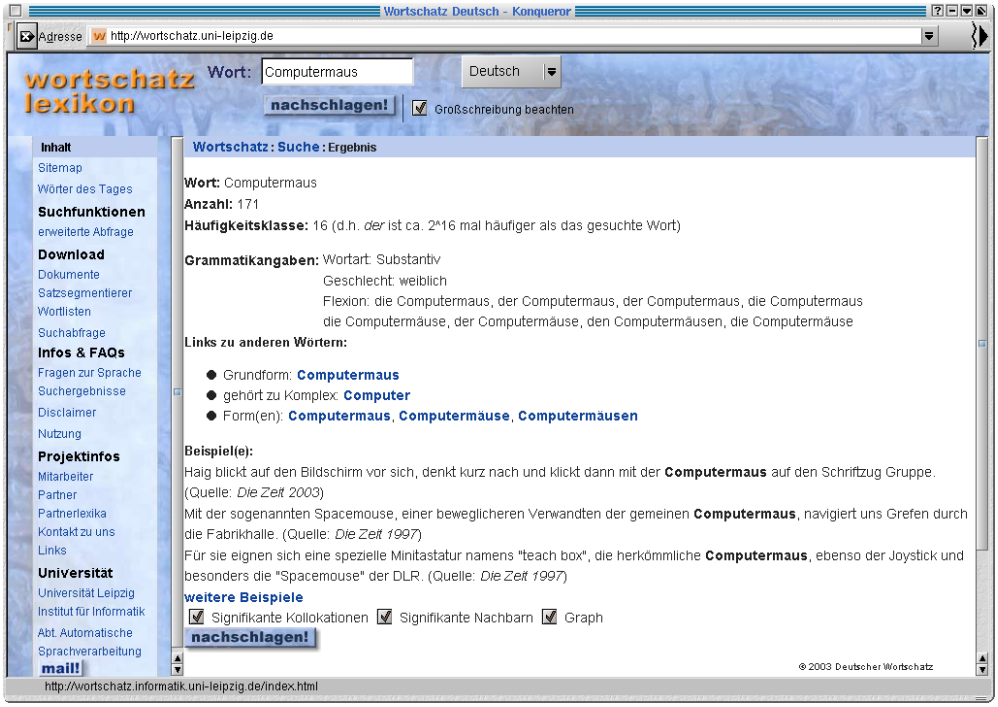


Abbildung 6.1: Die Webseite des Wortschatz-Lexikons.

Sammlung von Vollformen. Das bedeutet, dass zu jedem Begriff die Grundform und alle bekannten Flexionsformen explizit gespeichert werden. Zusätzlich dazu enthält das Wortschatz-Lexikon eine Reihe von Beziehungen (Antonymie, Hyponymie, Meronymie und andere) zwischen einzelnen Termen sowie zusätzliche termspezifische Informationen, wie z. B. die Häufigkeitsklasse oder Beispiele, in denen ein Term verwendet wird. [119, 118]

Dadurch, dass das Wortschatz-Lexikon ein Vollformlexikon mit einer relativ großen Zahl an Wortformen ist, ist es ein sehr attraktiver Kandidat für die Verwendung im Bereich des Stemming von Wörtern. Da das Lexikon zu jedem Wort die Grundform als eigene Beziehung parat hält, dürfte das Auslesen und Konvertieren dieser Daten in eine für das eTVSM verarbeitbare Form relativ problemlos sein. Dazu muss lediglich die Beziehung Grundform (vgl. das Beispiel in Abbildung 6.1) für jeden benötigten Term ausgewertet werden und eins-zu-eins in Form der Beziehung WW-Zuo (vgl. relationales Datenbankmodell des eTVSM in Abbildung 4.4 auf Seite 95) abgebildet werden.

Anders sieht die Situation aus, wenn man die Anwendung des Wortschatz-Lexikons für das Aufbauen von Themenstrukturen betrachtet. Hier fällt auf, dass das Wortstamm-Lexikon lediglich Terme und keine Interpretationen oder Themen miteinander in Beziehung setzt. Die-

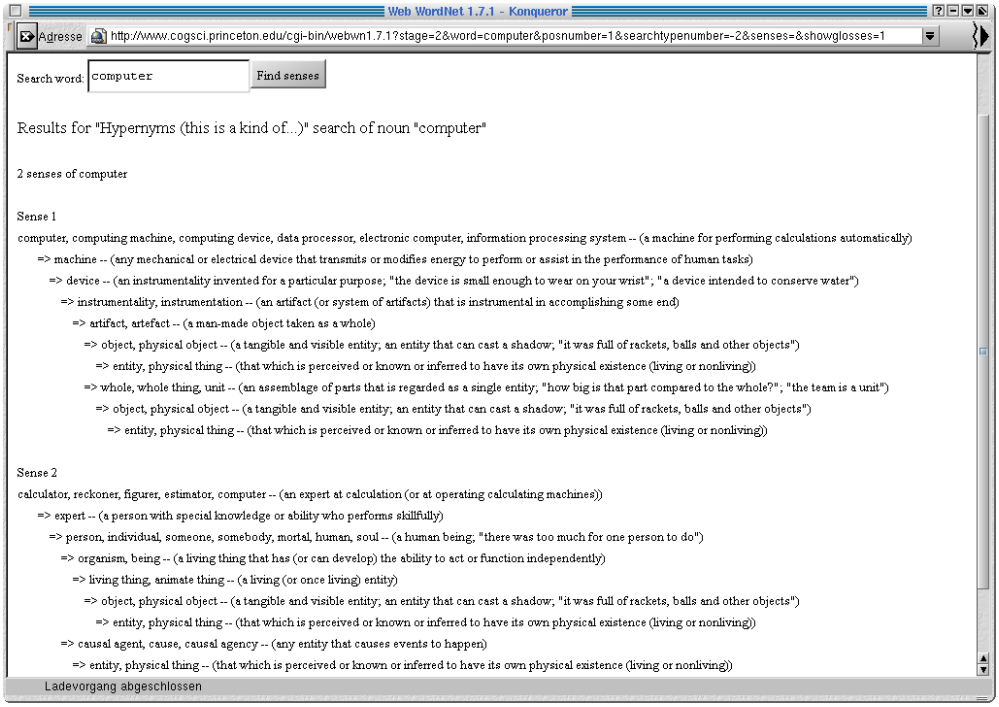


Abbildung 6.2: Online-Zugriff auf das WordNet.

ses führt dazu, dass sich unterschiedliche Interpretationen eines Begriffes nicht eindeutig voneinander unterscheiden lassen. Somit ist das Wortschatz-Lexikon zur Herleitung von Themenstrukturen, wie sie vom eTVSM benötigt werden, nicht anwendbar.

6.1.2.2 WordNet und GermaNet

Bei *WordNet*[3] und *GermaNet*[4] handelt es sich um zwei Wortnetze, mit demselben grundlegenden Konzept. Das WordNet ist im Jahre 1985 von Psychologen und Linguisten an der Princeton Universität konzipiert und entwickelt worden. Das Ziel war die Schaffung eines englischen Wörterbuches, welches nicht alphabetisch sondern konzeptionell aufgebaut ist. Mittlerweile enthält das WordNet ca. 138.000 Wortformen und ist frei über das Internet zugänglich (vgl. Abbildung 6.2). Alle Wortformen sind in der Grundform und gehören zu einer der folgenden vier Wortklassen: Nomen, Verben, Adjektive oder Adverbien. Das WordNet umfasst folgende Relationshiptypen zwischen den Wortformen: Synonymie, Homographie bzw. Poly-

[3] WordNet: `http://www.cogsci.princeton.edu/~wn`
[4] GermaNet: `http://www.sfs.nphil.uni-tuebingen.de/lsd`

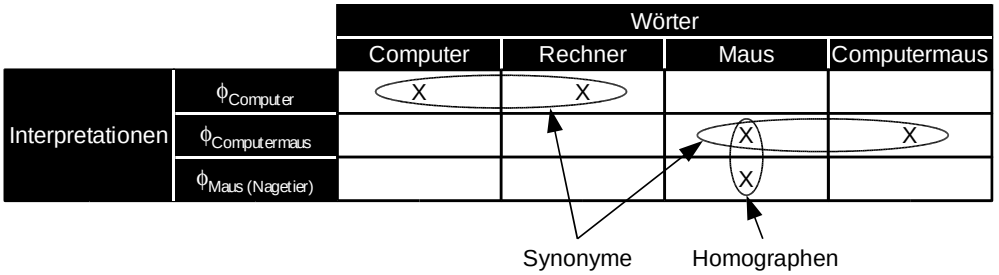


Tabelle 6.1: Zuordnungsmatrix zwischen Wörtern und Interpretationen.

semie, Antonymie, Hyponymie und Meronymie. Eine ausführliche Dokumentation des WordNet sowohl aus linguistischer als auch aus implementierungstechnischer Sicht findet sich in [99, 98, 49, 48, 15].

Das GermaNet ist ein deutsches Wörterbuch, dass auf demselben Konzept und derselben (teilweise leicht modifizierten) Software basiert wie das WordNet. Es ist am Seminar für Sprachwissenschaften der Universität Tübingen entwickelt worden und umfasst mittlerweile ca. 52.000 Wordformen (nur Grundformen) der deutschen Sprache. Zum jetzigen Zeitpunkt ist das GermaNet nicht frei erhältlich; Dokumentationen zum GermaNet finden sich z. B. in [2, 66]. Das dem WordNet und dem GermaNet zu Grunde liegende Konzept unterscheidet sich deutlich von dem Konzept des Wortschatz-Lexikon. Wie im Folgenden gezeigt wird, lassen sich die beiden Wortnetze WordNet und GermaNet für die Herleitung der vom eTVSM benötigten Themenstrukturen anwenden, daher wird im Folgenden ausführlicher auf das Konzept der beiden Wortnetze eingegangen.

Konzept: Die grundlegende Erkenntnis, die in das Konzept der beiden Wortnetze WordNet und GermaNet einfließt, ist die bereits in Abschnitt 2.3.4.2 vorgestellte Unterscheidung zwischen *Wörtern* und ihren *Interpretationen* sowie die zwischen diesen beiden Entitäten bestehende n-zu-n Beziehung (vgl. das Datenmodell in Abbildung 2.9 auf Seite 30).[5] Eine Möglichkeit, die Beziehung zwischen Wörtern und Interpretatationen formal darzustellen, ist die Verwendung einer Matrix. Eine solche Matrix zeigt die Tabelle 6.1 bei der Wörter durch die Spalten und Interpretationen durch die Zeilen der Matrix repräsentiert werden. Gültige Kombinationen aus einer Interpretation und einem Wort werden durch ein X gekennzeichnet. Bei Verwendung einer derartigen Matrix fällt auf, dass synonyme Wörter sich dadurch auszeichnen, dass sie in derselben Zeile ein X haben. Homographen haben hingegen in derselben Spalte jeweils ein X.

Aus diesem Zusammenhang heraus wurde für das WordNet die Idee geboren, Interpreta-

[5] Im englischen Originaltext [99] zur Beschreibung des Konzeptes von WordNet verwenden die Autoren die englischen Begriffe *word form* für den hier verwendeten Begriff Wort und *word meaning* für den hier verwendeten Begriff Interpretation.

tionen nicht durch künstliche, atomare Identifikatoren zu repräsentieren, sondern Interpretationen durch eine Menge von synonymen Wörtern, die sogenannten *Synsets*, darzustellen. Aus der Tabelle 6.1 ergeben sich beispielsweise folgende Synsets für die drei, in der Abbildung vorkommenden, Interpretationen:

$$\begin{aligned}\phi_{\text{Computer}} &= \{\text{Computer}, \text{Rechner}\}\\ \phi_{\text{Computermaus}} &= \{\text{Maus}, \text{Computermaus}\}\\ \phi_{\text{Maus (Nagetier)}} &= \{\text{Maus}\}\end{aligned}$$

Der Vorteil einer derartigen Repräsentation der Interpretationen ist der, dass Interpretationen eine innere Struktur, basierend auf den ihnen zugeordneten Wörtern, erhalten und dass diese Struktur für den Menschen im Allgemeinen intuitiv verständlich ist. Der Nachteil in der praktischen Umsetzung ist allerdings, dass nicht alle Interpretationen – insbesondere wenn sie nur aus einem einzigen Element bestehen – eindeutig unterscheidbar sind. Dieses Problem wurde beim WordNet und GermaNet dadurch gelöst, dass in solchen Fällen ein zusätzlicher natürlichsprachlicher Kommentar zur eindeutigen Unterscheidung derartiger Synsets eingeführt wurde. Aufbauend auf den Synsets werden in den beiden Wortnetzen u. a. folgende hier relevante Relationshiptypen zwischen jeweils zwei Synsets abgebildet:

- Hyponymie: ϕ_x ist ein (engl.: is-a) ϕ_y
- Meronymie: es werden die folgenden drei Beziehungssubtypen bei der Meronymie unterschieden.
  - ϕ_x hat Element (engl.: has-member) ϕ_y
  - ϕ_x hat Teil (engl.: has-part) ϕ_y
  - ϕ_x besteht aus der Substanz (engl.: has-substance) ϕ_y

Für den Anwender ermöglichen die beiden Wortnetze einen Zugriff auf ihre Daten über eine kommandozeilenorientierte Benutzerschnittstelle, zu der auch eine Fenster-basierte grafische Schnittstelle existiert, die allerdings lediglich die Eingabe der verschiedenen Parameter erleichtert und bedauerlicherweise keine neue, grafische Sicht auf die Netze bietet. Im Folgenden wird beispielhaft anhand einiger Synsets aus dem GermaNet illustriert, welche Daten in den Wortnetzen enthalten sind und wie diese Daten sinnvoll im Zusammenhang mit dem eTVSM genutzt werden können, d. h. wie das Wortnetz in eine für das eTVSM anwendbare Ontologie umgewandelt werden kann.

Ausgehend von dem Wort Platte werden mit dem folgenden Befehl alle Synsets ausgegeben, die das Wort Platte enthalten:

```
:~> gwn Platte -synsn
Synonyms/Hypernyms of nomen platte

4 senses of platte
```

```
Sense 1
Platte
       => ?nicht definite Raumeinheit

Sense 2
Platte
       => Speicher

Sense 3
Schallplatte, LP, Platte
       => Tonträger

Sense 4
Platte
       => Geschirr
```

Das Ergebnis dieser Anfrage ist wie folgt zu interpretieren: Es existieren insgesamt vier Interpretationen bzw. Synsets für das Wort Platte. Die erste Bedeutung des Begriffes ist ein Spezialfall einer „?nicht definierten Raumeinheit“[6] (Hyponymie-Beziehung). Die zweite Bedeutung meint einen speziellen Speicher und die vierte eine spezielle Art von Geschirr. Die dritte Bedeutung meint einen speziellen Tonträger, zu dem auch die Synonyme Schallplatte und LP existieren.

Die folgende Anfrage gibt zu allen Synsets von Platte alle im GermaNet definierten Meronyme aus:

```
:~> gwn Platte -meron
Meronyms of nomen platte

1 of 4 senses of platte

Sense 3
Schallplatte, LP, Platte
          MERONYM:: Plattenhülle, Plattencover, Cover
```

Das Ergebnis sagt aus, dass lediglich zur dritten Interpretation von Platte ein Meronym existiert: Eine Platte (im Sinne eines Tonträgers, der auch als Schallplatte oder LP bezeichnet wird) besteht aus etwas, was als Plattenhülle, Plattencover oder einfach nur als Cover bezeichnet wird.

Die folgende Anfrage gibt wieder, was die Subklassen/Subsynsets (im Sinne einer ist-ein Klassifikation bzw. Hyponymie) zu den einzelnen Synsets des Begriffs Platte sind.

```
:~> gwn Platte -hypon
Hyponyms of nomen platte
```

[6] Das Fragezeichen vor dem Begriff bedeutet, dass es sich hier um einen künstlich geschaffenen Begriff handelt, der lediglich zu Einordnungszwecken im GermaNet geschaffen wurde.

```
3 of 4 senses of platte

Sense 2
Platte
       => Diskette, Floppy Disk
       => Festplatte

Sense 3
Schallplatte, LP, Platte
       => Single, Singleauskopplung

Sense 4
Platte
       => Kuchenplatte
```

Demnach existieren für die Synsets zwei, drei und vier folgende Hyponyme: Eine Diskette bzw. eine Floppy Disk ist eine Platte (im Sinne eines Speichers). Dieses gilt ebenso für das Synset, dass durch das Wort Festplatte beschrieben wird. Eine Single bzw. Singleauskopplung ist eine Platte im Sinne eines Tonträgers. Eine Kuchenplatte ist eine Platte im Sinne von Geschirr.

Zur Abrundung des Beispiels soll im Folgenden geprüft werden, welche unterschiedlichen Bedeutungen die beiden Worte Diskette, und Festplatte sowie das künstliche Konstrukt „?nicht definite Raumeinheit“ haben können.

```
:~> gwn Diskette -synsn
Synonyms/Hypernyms of nomen diskette

1 sense of diskette

Sense 1
Diskette, Floppy Disk
       => Medium, Speichermedium
       => Platte
```

Das Ergebnis dieser Anfrage sagt aus, dass es nur eine Interpretation bzw. nur ein Synset zu Diskette existiert und dass eine Diskette nicht nur eine Platte, sondern auch ein Medium bzw. Speichermedium ist.

```
:~> gwn Festplatte -synsn
Synonyms/Hypernyms of nomen festplatte

1 sense of festplatte

Sense 1
Festplatte
       => Medium, Speichermedium
       => Platte
```

Zur Festplatte erfahren wir, dass es ebenfalls nur eine Interpretation zu diesem Wort existiert und dass die Festplatte (wie die Diskette) nicht nur eine Platte, sondern auch ein Medium bzw. Speichermedium ist.

```
:~> gwn "?nicht definite Raumeinheit" -synsn
Synonyms/Hypernyms of nomen ?nicht_definite_raumeinheit

1 sense of ?nicht definite raumeinheit

Sense 1
?nicht definite Raumeinheit
       => Raumeinheit, Raummaß, Kubikmaß, Hohlmaß
```

Die letzte Anfrage teilt uns mit, dass der künstliche Begriffe der „?nicht definiten Raumeinheit“ ein Spezialfall (ist-ein Beziehung) des Synsets ist, dass durch folgende Worte definiert wird: Raumeinheit, Raummaß, Kubikmaß, Hohlmaß.

Integration mit dem eTVSM: Die Transformation einer WordNet bzw. GermaNet Ontologie (im Folgendem übergreifend als Wortnetz bezeichnet) in eine zu dem Modell des eTVSM kompatible Form, kann mit den folgenden Schritten durchgeführt werden:

1. Alle Wörter des Wortnetzes werden eins-zu-eins in Terme im TVSM umgesetzt.

2. Jede Synset des Wortnetzes wird eins-zu-eins durch ein eigenes Thema und eine eigene Interpretation im eTVSM repräsentiert. Das Thema und die Interpretation, die dasselbe Synset repräsentieren, werden einander über die Beziehung IT-Zuo zugeordnet.

3. Alle Terme, die Wörter eines bestimmten Synset repräsentieren, werden der Interpretation, die diese Synset darstellt, über die Beziehung TI-Zuo zugeordnet. Dieser Vorgang wird für alle Synsets bzw. Interpretationen durchgeführt.

4. Die Beziehungen (Hyponymie und/oder Meronymie) zwischen Synsets werden in Form der Themenstruktur äquivalent abgebildet.

5. Für alle Terme, zu denen es mehrere Interpretationen gibt (Homographen), werden die Supportterme zu den Interpretationen gemäß des in Abschnitt 5.1.2.3 vorgestellten Verfahren definiert. Zusätzlich wird jedem dieser Terme eine zusätzliche (Standard-) Interpretation zugeordnet, die allen Themen der Interpretationen des jeweils betrachteten Terms zugeordnet werden.

6. Zum Abschluss wird geprüft, ob es ein Thema gibt, zu dem nur ein einziges Subthema existiert. Für jedes dieser Themen wird ein Dummy-Thema definiert, dass dem Superthema zugeordnet wird.[7]

[7] Zur Begründung vgl. Abschnitt 5.1.1.4.

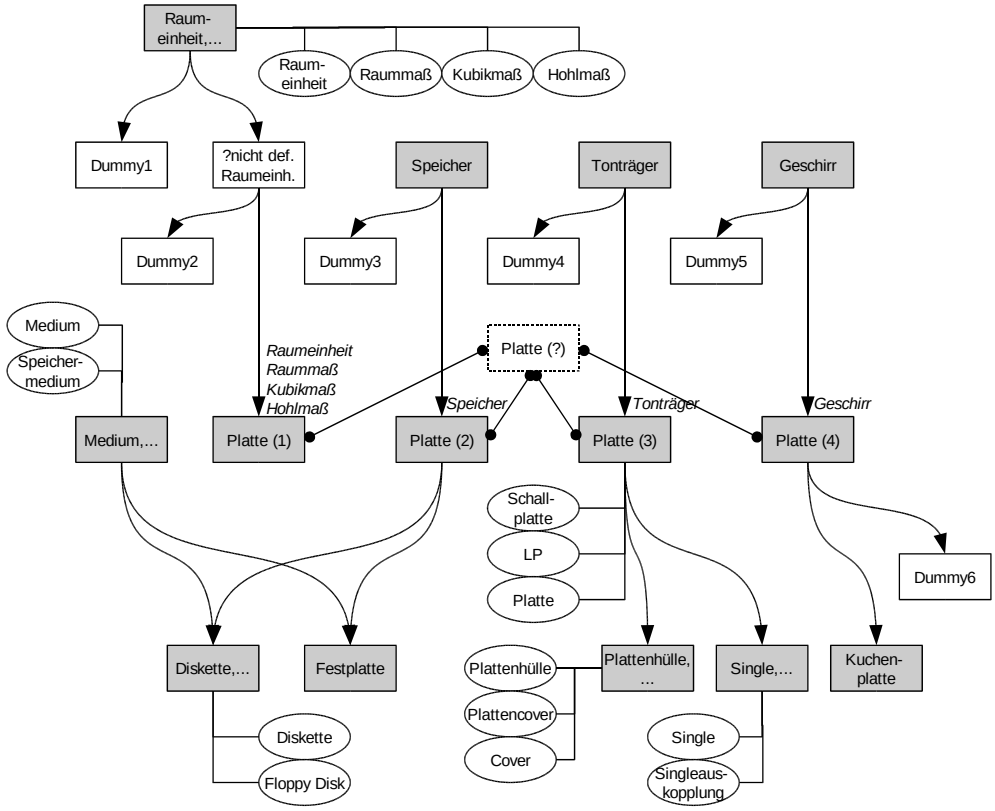


Abbildung 6.3: eTVSM-Ontologie zu den GermaNet-Beispielen.

Wendet man diese Schritte auf die Synsets und Wörter aus dem vorherigen Beispiel zur Illustration von GermaNet an, dann entsteht die in Abbildung 6.3 gezeigte Ontologie[8]. Die daraus resultierende, mit dem eTVSM berechnete Ähnlichkeitsmatrix für die Themen (ohne Dummy-Themen) zeigt die Tabelle 6.2.

Vergleich mit einem anderen Ansatz: VORHEES beschreibt in ihrem Artikel [154] einen Ansatz, der das WordNet für das IR nutzbar machen soll, der auf den ersten Blick zu dem hier vorgestellten Ansatz ähnlich erscheint. Ihr Ansatz basiert jedoch im Unterschied zu dem hier vorgestellten Ansatz auf dem VSM[9]. VORHEES schlägt vor das VSM derart zu modifizieren, dass die grundlegenden Bausteine zum Vergleich von Dokumenten nicht mehr einzelne Terme, sondern einzelne Synsets sind. Das heißt, dass ein Dokument in eine Menge von Syn-

[8] Die Ontologie ist in der in Abschnitt 5.2.1 vorgestellten Sprache notiert.
[9] Vgl. Abschnitt 3.2.2.

	Raumeinheit	Platte (1)	Speicher	Platte (2)	Diskette	Festplatte	Medium	Tonträger	Platte (3)	Plattenhülle	Single	Geschirr	Platte (4)	Kuchenplatte
Raumeinheit	1,000	0,777	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
Platte (1)	0,777	1,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
Speicher	0,000	0,000	1,000	0,830	0,776	0,776	0,830	0,000	0,000	0,000	0,000	0,000	0,000	0,000
Platte (2)	0,000	0,000	0,830	1,000	0,935	0,935	1,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
Diskette	0,000	0,000	0,776	0,935	1,000	0,750	0,935	0,000	0,000	0,000	0,000	0,000	0,000	0,000
Festplatte	0,000	0,000	0,776	0,935	0,750	1,000	0,935	0,000	0,000	0,000	0,000	0,000	0,000	0,000
Medium	0,000	0,000	0,830	1,000	0,935	0,935	1,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
Tonträger	0,000	0,000	0,000	0,000	0,000	0,000	0,000	1,000	0,851	0,777	0,777	0,000	0,000	0,000
Platte (3)	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,851	1,000	0,913	0,913	0,000	0,000	0,000
Plattenhülle	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,777	0,913	1,000	0,667	0,000	0,000	0,000
Single	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,777	0,913	0,667	1,000	0,000	0,000	0,000
Geschirr	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	1,000	0,851	0,777
Platte (4)	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,851	1,000	0,913
Kuchenplatte	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,777	0,913	1,000

Tabelle 6.2: Ähnlichkeitsmatrix zu der Ontologie aus Abbildung 6.3.

sets zerlegt wird und dass die Ähnlichkeit zwischen zwei Dokumenten anhand der Anzahl der gemeinsamen Synsets bestimmt wird. Bei Experimenten mit Anfragen auf verschiedene Dokumentenkollektionen stellt VORHEES fest, dass ihr Ansatz im Vergleich zum klassischen VSM deutlich schlechtere Ergebnisse liefert. Eine genauere Untersuchung der Ursache kommt zu folgendem Ergebnis: Bei der Zerlegung von Dokumenten in Mengen von Synsets gibt es Probleme mit Homographen, da bei diesen zunächst unklar ist, welches Synset zu dem Dokument gewählt werden muss. VORHEES löst das Problem in der Weise, dass sie die umgebenden Worte des Dokuments betrachtet und diejenige Synset auswählt, die am besten zu diesen Worten passt. Ist jedoch eine eindeutige Entscheidung nicht möglich, dann wird für den betroffenen Homographen keine Synset gewählt. Dieses führt jedoch dazu, dass Informationen über Dokumente verloren gehen, die zur korrekten Auswertung einer Benutzeranfrage erforderlich wären. Besteht beispielsweise eine Anfrage aus zwei Synsets, dann reduziert das nicht vorhanden sein eines Synsets in einem Dokument die Relevanz des Dokuments bezüglich der Anfrage in einem erheblichen Maße. Da die Anzahl der Dokumente mit nicht eindeutig auflösbaren Homographen durchaus einen nennenswerten Umfang erreicht, ist es in sofern nicht verwunderlich, dass das Verfahren von VORHEES schlechtere Ergebnisse liefert als das klassische VSM. Allerdings konnte VORHEES feststellen, dass das manuelle Zuweisen von korrekten Synsets zu den einzelnen Dokumenten, die Ergebnisqualität ihres Verfahrens gegenüber dem VSM erhöhen konnte.

Bei dem hier bereits vorgestellten eTVSM besteht ebenfalls das Problem, Homographen ihren zu einem Dokument passenden Interpretationen zuzuordnen. Allerdings müssen beim

eTVSM nicht alle Interpretationen (bzw. im WordNet-Jargon: Synsets) orthogonal zueinander sein, wodurch die Verwendung von Standard-Interpretationen[10] möglich ist. Diese Standard-Interpretationen haben zu allen möglichen Interpretationen eines Homographen eine hohe Ähnlichkeit und werden immer dann eingesetzt, wenn eine eindeutige Entscheidung für eine Interpretation eines Homographen nicht getroffen werden kann. Somit entspricht die Verwendung einer Standard-Interpretation zwar einem Nicht-Auflösen eines Homographen, aber die Information des Homographen geht im Unterschied zu dem Ansatz von VORHEES nicht verloren. Daher ist zu erwarten, dass das eTVSM im Zusammenhang mit einem Wortnetz die Ergebnisqualität von IF- und IR-Aufgaben gegenüber herkömmlichen Verfahren steigern kann.

6.2 Anwendung für das Information-Retrieval

Ziel des IR ist die Deckung des akuten Informationsbedarfs eines konkreten Anwenders. Das Vorgehen zum Erreichen dieses Ziels sieht dabei wie folgt aus: Es werden vom Benutzer Anfragen formuliert, die der Suche geeigneter Dokumente aus einem bestehenden Dokumentenbestand dienen.[11] Gemäß GUHA, MCCOOL und MILLER [63] lassen sich zwei Arten von Informationsbedürfnissen bzw. Suchen[12] unterscheiden:

- *Navigational Searches:* Bei dieser Art von Suche kennt der Anwender das zu suchende Dokument bereits, er weiß aber nicht mehr, wo sich dieses Dokument im Datenbestand befindet.[13] Der Benutzer bemüht sich daher, eine Kombination von Suchwörtern zu finden, die möglichst nur in dem einen gesuchten Dokument vorkommen. Diese Wörter stehen dabei nicht notwendigerweise in irgendeinem thematischen oder konzeptuellen Zusammenhang zueinander. In diesen Such-Fällen verwendet der Anwender das IR-System lediglich als ein Navigationswerkzeug. Beispiele für Suchwörter derartiger Suchen sind: ULB Münster Fernleihe DigiBib Anmeldung[14] oder Semantic Search Guha McCool Miller[15].

- *Research Searches:* Im Unterschied zu der zuvor beschriebenen Art von Suche kennt der Anwender bei den Research Searches die zu suchenden Dokumente nicht. Der Anwender formuliert seine Anfrage daher so, dass die eingegebenen Suchbegriffe das Themengebiet, zu dem der Anwender einen Informationsbedarf hat, möglichst gut umschreiben. Der Benutzer erwartet somit nicht idealerweise nur ein Dokument vom System, sondern alle Dokumente, die zu dem von ihm vorgegebenen Thema passen. Beispiele für derartige Anfragen sind: Modell Handelsinformationssystem oder Vector Space Model Information Retrieval.

[10] Die Standard-Interpretation zu dem Homographen Platte in Abbildung 6.3 ist beispielsweise „Platte (?)“.
[11] Vgl. Abschnitt 2.1.1.
[12] Hier ist der Plural von Suche gemeint.
[13] Das heißt z. B. im Falle des Internet, dass er die URL des Dokuments vergessen hat.
[14] Sucht die Webseiten der Universitäts- und Landesbibliothek Münster zur Online-Bestellung von Büchern.
[15] Sucht das digitale Dokument zu [63].

Während derzeit gängige IR-Systeme die Problemestellung der Navigational Searches gut meistern, sind diese Systeme zur Lösung von Research Searches eher weniger geeignet. Der Grund dafür ist, dass gängige IR-Systeme linguistische Phänomene wie z. B. Synonymie und Homographie etc. nicht berücksichtigen. Dieses führt auf der einen Seite, durch Homographen bedingt, häufig zu irrelevanten Ergebnissen, weil auch Dokumente vom System gefunden werden, die den Homographen in einer anderen Interpretation als ursprünglich vom Benutzer intendiert enthalten. Auf der anderen Seite führt das Ignorieren von Synonymen dazu, dass relevante Dokumente nicht gefunden werden, weil der Benutzer beispielsweise ein Synonym zu dem im gesuchten Dokument enthaltenen Wort in der Anfrage vorgegeben hat und dieses in der Anfrage genannte Synonym nicht im Dokument vorkommt. Daher hat das eTVSM gerade im Aufgabenbereich der Research Searches ein hohes Anwendungs- und Nutzenpotential, weil es linguistische Phänomene und Themenstrukturen explizit berücksichtigt.

Neben der direkten eins-zu-eins Umsetzung herkömmlicher Suchstrategien (Erstens: Eingabe eine Anfrage. Zweitens: Präsentation von Dokumenten.) sind auch interaktivere Suchstrategien mit Hilfe des eTVSM denkbar. So wäre es z. B. sinnvoll, Benutzer vor der Ausführung einer Anfrage darauf aufmerksam zu machen, dass er einen Homographen (z. B. Maus) benutzt und ihn um eine genauere Spezifikation des Homographen (Computermaus oder Nagetier-Maus?) zu bitten. Ebenso ist es sinnvoll, den Benutzer nach dem Ausführen von Anfragen über thematische Zusammenhänge und Super- oder Sub-Begriffe zu informieren. Damit kann insbesondere unerfahrenen oder fachfremden Benutzern die thematische Einschränkung oder thematische Erweiterung einer Suchanfrage erleichtert werden.

6.3 Anwendung für das Information-Filtering

Die Aufgabe des IF ist es, den langfristigen Informationsbedarf eines Anwenders durch die Selektion von Dokumenten – aus zeitlicher Sicht betrachtet – aus einem Strom von Dokumenten zu decken.[16] Der Unterschied zum IR besteht beim IF in der Langfristigkeit des Informationsbedarfs und dem überwiegend passiven Verhalten des Benutzers gegenüber dem System. Ein weiterer Unterschied besteht darin, dass ein IF-System neue, relevante Dokumente dem Benutzer möglichst zeitnahe präsentieren sollte. Nur dann ist ein hoher Nutzen des Systems sicher gestellt, weil es den Benutzer so von der lästigen Aufgabe des wiederholten Suchens nach neuen, relevanten Dokumente entlastet.

Ein einfacher Ansatz zur Lösung des IF-Problems ist es, den Benutzer seinen langfristigen Informationsbedarf – analog zum IR – in Form einer Anfrage bzw. eines Auswahlkriteriums, das üblicherweise als *Benutzerprofil* bezeichnet wird, explizieren zu lassen. Neue Dokumente werden anhand dieses Benutzerprofils bezüglich ihrer Relevanz für den Benutzer evaluiert und ggf. dem Benutzer präsentiert oder verworfen. Die Praxis zeigt allerdings, dass die Anwendung dieses Ansatzes das Problem aufwirft, dass die Benutzer im Allgemeinen nicht in der Lage sind ein „gutes“ Benutzerprofil aufzustellen, weil die Explikation eines Benutzerprofils

[16] Vgl. Abschnitt 2.1.2.

ein nicht-triviales Unterfangen ist. Die Schwierigkeit der formalen Explikation des langfristigen Informationsbedarfs hat folgende Gründe:

1. *Komplexität und Größe des Benutzerprofils:* Im Vergleich zum IR ist das Benutzerprofil eines Benutzers, der ein IF-System anwenden möchte, eine statische Daueranfrage an das IF-System, weil der Benutzer sich dem System gegenüber passiv verhält und somit seine Anfrage nicht interaktiv – über mehrere Versuche – an seinen Informationsbedarf anpassen kann.[17] Somit reicht es nicht, wenn das Benutzerprofil einen akuten Informationsbedarf abdeckt, sondern es muss nach Möglichkeit alle dauerhaften und idealerweise zukünftigen Informationsbedürfnisse eines Benutzers abdecken. Zusätzlich muss das Benutzerprofil derart formuliert werden, dass nicht nur aktuelle Dokumente korrekt klassifiziert werden, sondern dass auch zukünftige Dokumente, deren genauer Inhalt noch nicht bekannt ist, korrekt eingeordnet werden. Aus den beiden genannten Aspekten (mehrere Informationsbedürfnisse und korrekte Klassifikation zukünftiger Dokumente) folgt im Allgemeinen, dass Benutzerprofile die Tendenz dazu haben, groß und komplex zu werden.

2. *Aufwand-Nutzen-Frage:* Der Gesamtnutzen eines IF-Systems hängt für den Benutzer von zwei einander widerstrebenden Faktoren ab: Erstens, ein IF-System ist von Nutzen, wenn es einen hohen Anteil neuer Dokumente korrekt und zeitnahe klassifiziert. Zweitens, der Aufwand in der Erstellung und Pflege des Benutzerprofils sollte möglichst klein sein. Das heißt, dass die Sprache zur Definition des Benutzerprofils einfach zu verstehen sein sollte und somit eine geringe Komplexität haben sollte und gleichzeitig eine kurze und mächtige Definition des Benutzerprofils ermöglichen sollte. Hierbei tritt das Problem auf, dass je einfacher die Sprache zur Definition eines Benutzerprofils ist und je kürze das Benutzerprofil ist, desto wahrscheinlicher ist es, dass der langfristige Informationsbedarf des Benutzer nicht vollständig erfasst werden kann und dass der Anteil der korrekt klassifizierten Dokumente gering ist.

3. *Verinnerlichte Regeln:* Die Regeln, nach denen ein Benutzer für sich selbst entscheidet, ob ein Dokument für ihn relevant ist oder nicht, hat der Benutzer im Allgemeinen verinnerlicht. Das heißt, dass er sich häufig dieser Regeln und teilweise auch des Entscheidungsprozesses an sich nicht bewusst ist. Somit fällt es einem Benutzer häufig schwer, seinen langfristigen Informationsbedarf in allen Details vorab zu explizieren. Erschwerend kommt hinzu, dass sich der langfristige Informationsbedarf mit der Zeit ändert und dass die Entscheidungen des Benutzers über die Relevanz von spezifischen Dokumenten zu unterschiedlichen Zeitpunkten in Abhängigkeit von z. B. der Tagesform oder von Ereignissen variieren kann.

4. *Natürlichsprachliche Phänomene:* Zusätzlich zu den oben genannten Aufgabenstellung-bezogenen Problemen kommen noch die Phänomene der natürlichen Sprache (Ho-

[17] Natürlich ist es dem Benutzer möglich seine Anfrage mit der Zeit zu ändern, allerdings sind die Zeiträume zwischen zwei Veränderungen am Benutzerprofil üblicherweise um mehrere Zehnerpotenzen größer als die interaktiv durchgeführten Veränderungen von Anfragen beim IR. Daher kann das Benutzerprofil eines IF-Systems gegenüber von Benutzeranfragen zur Deckung eines akuten Informationsbedarfs bei IR-Systemen als statisch angesehen werden.

mographen, Synonyme, etc.) hinzu, die das Erstellen von Benutzerprofilen erschweren. Wenn das IF-System linguistische Phänomene – wie in der Praxis zur Zeit üblich – nicht explizit berücksichtigt, dann muss der Benutzer diese in seinem Benutzerprofil manuell berücksichtigen, indem er z. B. im Fall von Synonymen alle synonymen Begriffe aufnimmt und im Fall von Homographen diese entweder meidet oder sich Strategien zur Disambiguierung von Homographen überlegt.

Aus diesen Gründen wird der einfache Ansatz zur Lösung des IF-Problems nur in wenigen Bereichen, wie z. B. dem Filtern von Werbe-E-Mails[18] angewandt. Eine teilweise Lösung bzw. Reduktion der oben genannten Probleme beim IR ermöglichen die im Folgenden vorgestellten adaptiven Ansätze.

Adaptive Ansätze versuchen das Benutzerprofil zu einem Benutzer über Kommunikation oder Beobachten des Benutzers zu erlernen. Üblicherweise sieht das z. B. so aus, dass der Benutzer Dokumente, die ihm vom System als relevant präsentiert wurden, bewerten kann. Diese Bewertung wird vom IF-System zur Modifikation des Benutzerprofils herangezogen. Der Vorteil von adaptiven Ansätzen ist somit, dass diese den Benutzer von der aufwändigen Erstellung und Wartung seines Benutzerprofils entlasten und somit zumindest eine Teillösung für die oben genannten Probleme darstellen. Generell lassen sich drei Klassen von adaptiven Ansätzen unterscheiden:

1. *Ansätze zur adaptiven Anfragemodifikation:* Diese Ansätze haben die höchste Ähnlichkeit zu dem oben vorgestellten einfachen Ansatz. Bei diesen Ansätzen entspricht das Benutzerprofil in etwa einer Anfrage in einem IR-System. Diese Daueranfrage wird in Abhängigkeit von den vom Benutzer zurückgegebenen Dokumentenbewertungen um neue Terme und ggf. Verknüpfungsoperatoren automatisch vom System erweitert bzw. es werden ggf. Terme modifiziert oder entfernt. Dieser Ansatz wird z. B. von dem *NewsSIEVE* [67] IF-System im Zusammenhang mit genetischen Algorithmen angewandt. Der Vorteil dieses Ansatzes ist, dass das Benutzerprofil für den einzelnen Benutzer ggf. nach einigem Aufwand in das Erlernen der Anfragesystemetik nachvollziehbar und auch modifizierbar ist. Des Weiteren kann dieser Ansatz prinzipiell auf allen Modellen zur Repräsentation von Dokumenten aufbauen, die den Vergleich von Dokumenten mit Anfragen ermöglichen.

2. *Fall-basierte Ansätze:* Bei den Fall-basierten Ansätzen werden von einem Benutzer bewertete Dokumente eins-zu-eins in dem Benutzerprofil (inklusive ihrer Bewertung)

[18] Bei den Werbe-E-Mails wird das Filtern dadurch vereinfacht, dass diese im Allgemeinen einen hohen Anteil an bekanntermaßen „verdächtigen“ Begriffen oder Absenderinformationen enthalten, über die diese E-Mails relativ leicht identifiziert werden können. Mit der Zeit ist jedoch eine Art Kampf zwischen den Werbe-Mailern und den Anti-Werbe-Mail Werkzeugherstellern entstanden, da die Werbe-Mailer sich zunehmend Mühe geben, ihre Werbebotschaften so „unauffällig“ zu gestalten, dass diese möglichst nicht einfach durch automatische IR-Systeme gefiltert werden können. Dadurch, dass aber eine große Anzahl an Benutzern dasselbe Interesse daran haben, von Werbe-E-Mails verschont zu werden, ist es ökonomisch sinnvoll, dass einige wenige Anwender bzw. Werkzeughersteller sich die Mühe machen aufwändige Profile zum Filtern von Werbe-E-Mails zu erstellen, die mit den anderen Benutzern geteilt werden.

gespeichert. Neue Dokumente werden mit allen Dokumenten im Benutzerprofil verglichen und es wird die Bewertung des oder der ähnlichsten Dokumente für das neue Dokument übernommen. Das bekannteste Verfahren dieser Art ist das k-nearest neighbour Verfahren [40], dessen Einsatz im Bereich des IF u. a. in folgenden Publikationen beschrieben und evaluiert wurde: [162, 96]. Der Vorteil von Fall-basierten Ansätzen ist der, dass das Benutzerprofil intuitiv für den Menschen verständlich ist und somit einer manuellen Anpassung leicht zugänglich ist. Ein Nachteil ist, dass die Benutzerprofile im Allgemeinen durch das Speichern von vielen Dokumenten recht aufwendig sind. Des Weiteren lassen sich diese Ansätze mit allen Verfahren anwenden, die den paarweisen Ähnlichkeitsvergleich von Dokumenten unterstützten.

3. *Modell-penetrierende Ansätze:* Die letzte Klasse der adaptiven Verfahren ist wesentlich heterogener und zeichnet sich dadurch aus, dass diese Verfahren direkt in die Modelle zur Repräsentation von Dokumenten eingreifen bzw. Modifikationen an diesen Modellen vornehmen. Bekannte Vertreter dieser Verfahren sind z. B. die Support Vector Machines [162, 153], Neuronale Netze [162] (die in Unterschied zu Abschnitt 3.4.3 explizit auf die Klassifikation und nicht auf den paarweisen Vergleich von Dokumenten zugeschnitten sind) und Naiver Bayes Ansätze [162], die auf die Bestimmung von Wahrscheinlichkeiten für explizite Klassen modifiziert wurden. Üblicherweise haben derartige Verfahren den Nachteil, dass ihre Benutzerprofile für den normalen Anwender, der sich nicht tiefergehend mit dem Modell zur Repräsentation der Dokumente beschäftigt hat, nicht nachvollziehbar sind und sich somit jeglicher Modifikation oder Prüfung durch den Anwender entziehen. Zusätzlich sind diese Ansätze im Allgemeinen nicht oder nur mit Aufwand auf andere Modelle zur Repräsentation von Dokumenten übertragbar.

Während es relativ klar ist, warum adaptive Verfahren zur (Teil-)Lösung der ersten drei oben genannten Probleme des IF geeignet sind, bedarf das vierte Problem einer Erklärung: Dadurch, dass ein adaptives IF-System das Benutzerprofil erlernt, wird auch das „Wissen“ um linguistische Phänomene in das Benutzerprofil eines Benutzers eingebettet (sofern es im Benutzerprofil darstellbar ist). Dieses wird an dem folgenden, zur besseren Illustration idealisierten und verkürzten Beispiel erläutert: Angenommen, ein Benutzer interessiert sich für Nachrichten (Dokumente) aus dem Themenbereich Computer.

- In diesem Falle wird das IF-System spätestens nach der ersten Fehlklassifikation eines Dokuments mit dem Wort Computer lernen, dass Computer ein Begriff ist, der relevante Dokumente auszeichnet.
- Das Problem ist nun, dass das System erst wieder einen Fehler machen muss, um z. B. zu erlernen, dass auch das Wort Rechner ein Begriff ist, der relevante Dokumente auszeichnet, weil es ein Synonym zu Computer ist.
- Bei der nächsten Fehlklassifikation eines Dokuments mit dem Wort Maus, dass weder Rechner noch Computer enthält, wird das System lernen, dass auch Maus ein Begriff ist, der relevante Dokumente auszeichnet.

- Zum Abschluss nehmen wir einmal an, dass das IF-System ein Dokument klassifizieren muss, dass sowohl die Wörter Maus als auch Nagetier enthält. Da das System gelernt hat, dass Maus ein Wort ist, dass relevante Dokumente kennzeichnet, wird das System das Dokument als relevant kennzeichnen. Dieses ist eine Fehlklassifikation, die dem System vom Benutzer mitgeteilt wird. Aus dieser Information wird das System – ein hinreichend mächtiges adaptives Verfahren vorausgesetzt – lernen, dass Maus in Zusammenhang mit Nagetier kein Begriff ist, der relevante Dokumente klassifiziert. Ist das adaptive Verfahren nicht mächtig genug, dann kann es passieren, dass das IF-System den falschen Schluss zieht und Maus generell als Begriff definiert, der nicht relevante Dokumente kennzeichnet.

Wie man aus dem Beispiel erkennen kann, ist ein IF-System unter idealen Umständen theoretisch in der Lage linguistische Phänomene vom Benutzer zu erlernen und im Benutzerprofil zu hinterlegen. Das Problem dabei ist jedoch, dass der Lernprozess durch die linguistischen Phänomene unnötig verlängert wird und somit der Anteil der korrekt klassifizierten Dokumente unnötig nach unten gedrückt wird.

Daher ist zu erwarten, dass der Einsatz des eTVSM im Zusammenhang mit einer geeigneten Ontologie (wie z. B. GermaNet) die Klassifikation von Dokumenten verbessern und den Lernprozess verkürzen kann. Im obigen Beispiel wäre es im Zusammenhang mit dem eTVSM ausreichend, wenn sich das System nur die Interpretation von Computer als Kennzeichen für relevante Dokumente merkt, weil die linguistischen Phänomene vom eTVSM behandelt werden. Somit bietet das eTVSM zusätzlich noch die Chance, dass Benutzerprofil zu verkleinern. Da das eTVSM sowohl den Vergleich von Dokumenten mit Dokumenten als auch mit Anfragen unterstützt, sollte eine Anwendung des eTVSM mit Ansätzen zur adaptiven Anfragemodifikation als auch mit Fall-basierten Ansätzen integrierbar sein. In Bezug auf die Integration mit Modell-penetrierenden Ansätzen sind weitere detailliertere Nachforschungen zur Beantwortung dieser Frage notwendig. Jedoch erscheint die Support Vector Machine viel versprechend, weil dieser Ansatz, ebenso wie das eTVSM, auf einem Vektorraum basiert.

6.4 Quantitative Evaluierung

Die quantitative Evaluierung von IF- und IR-Systemen ist eine anspruchsvolle Aufgabenstellung, zu der es keine einzelne, alle Aspekte umfassende Lösung gibt. Das Vorgehen zur Evaluierung vorhandener Systeme hängt von verschiedensten Faktoren, wie z. B. den Versuchsbedingungen, den Evaluationsmaßen und den Testdaten ab. Zusätzlich kommen bei der Evaluierung verschiedene Methoden der Statistik (von der einfachen Mittelwertbildung bis zu Signifikanztests beim Vergleich zweier Systeme) sowie unterschiedliche grafische Darstellungsmethoden zur Illustration der Ergebnisse zum Einsatz. Aufgrund dieser Vielfalt kann dieses Thema nicht in einem einzelnen Abschnitt abschließend und vollständig behandelt werden, weshalb die Problemstellung der quantitativen Evaluierung von IF- und IR-Systemen im Folgenden lediglich skizziert wird. Für einen detaillierten Einblick in die Problemstellung und Problemlösung wird daher auf die einschlägige Literatur, insbesondere die Artikelsammlung

[76] von JONES und auf das Buch [129, S. 167ff] von SALTON und MCGILL verwiesen.

6.4.1 Evaluationsmaße

IF- und IR-Systeme können nach verschiedenen Maßen evaluiert werden. Grundsätzlich lassen sich diese Maße in zwei Klassen, in die Effizienz- und die Effektivitätsmaße aufteilen. *Effizienzmaße* dienen dazu, zu beurteilen, welchen Aufwand der Betrieb und die Anwendung von IF- und IR-Systemen erfordert. Gängige Maße in diesem Zusammenhang sind z. B.:

- *Kosten* der Anschaffung, des Betriebs und der Wartung oder pro Anfrage.
- *Performanz:* Die Frage nach der Bearbeitungsdauer von Benutzeranfragen im Falle von IR oder die Frage, mit welcher Verzögerung neue, relevante Nachrichten dem Benutzer im Falle von IF gemeldet werden.
- *Bedienbarkeit:* Aufwand der aus Benutzersicht zum Einsatz des Systems notwendig ist. Angemessenheit der Ergebnispräsentation. Etc.

Da die konkrete Auswahl und Definition der Effizienzmaße stark von konkreten, bereits existierenden und vermarkteten Implementierungen von Systemen sowie dem genauen Einsatzgebieten der Systeme z. B. in Unternehmen abhängen, werden Effizienzmaße in der wissenschaftlichen IR- und IF-Gemeinde, die an neuen Verfahren und weniger an der Entwicklung von reifen und vermarktbaren Systemen arbeitet, häufig nur am Rande behandelt.

Effektivitätsmaße dienen im Unterschied zu den Effizienzmaßen dazu, ein System danach zu bewerten, in wie weit es seine IF- bzw. IR-Aufgabe erfüllt. Ausgangspunkt für die Bestimmung der Effektivitätsmaße sind dabei die Testdaten und die vom System zurückgelieferten Ergebnisse. Die Testdaten und Ergebnisse haben dabei üblicherweise die folgende Struktur:

1. Dem Testsystem wird eine vorab-definierte, endliche Menge von Dokumenten D zur Bewertung gegeben. Für statistisch aussagekräftige Ergebnisse sollte die Anzahl der Dokumente $\#D$ möglichst hoch sein.
2. Üblicherweise werden mehrere Tests über eine Menge von verschiedenen Kriterien K durchgeführt, bei denen die Dokumente in Abhängigkeit eines Kriteriums $k \in K$ vom System klassifiziert werden. Im Falle von IR ist k eine Anfrage, im Falle von IF ist k ein Benutzerprofil. Auch hier gilt: Für statistisch abgesicherte Ergebnisse sollte die Zahl der Tests und damit die Anzahl der verschiedenen Kriterien $\#K$ möglichst hoch sein.
3. Für jedes Kriterium $k \in K$ auf der Menge aller Dokumente D gibt das getestete System zwei Mengen zurück: $R_k^S \subseteq D$, die Menge aller Dokumente, die laut System relevant sind und $\complement R_k^S = D \setminus R_k^S$, die Menge aller Dokumente, die laut System nicht relevant sind.
4. Zur Evaluierung des Systemergebnisses werden Testmengen, d. h. die korrekten, vom Tester bzw. Benutzer definierten Idealergebnisse, herangezogen. $R_k^T \subseteq D$, die Menge

aller Dokumente, die laut Tester/Benutzer für das Kriterium k relevant sind und $\complement R_k^T = D \setminus R_k^T$, die Menge aller Dokumente, die laut Tester/Benutzer für das Kriterium k nicht relevant sind.

Anzumerken ist, dass der in diesem Zusammenhang verwendete Begriff der Relevanz von dem Versuchsaufbau und von der Art des getesteten Systems (IF oder IR) abhängt. Die Problematik der Begriffsdefinition wird in den Abschnitten 6.4.2 und 6.4.3 IR- bzw. IF-spezifisch diskutiert. Basierend auf den Testdaten lassen sich z. B. folgende, häufig verwendete Effektivitätsmaße aufstellen:

- *Precision* ist ein Maß für die Genauigkeit des Ergebnisses und ist definiert als der Anteil der tatsächlich relevanten Dokumente, die das System als relevant eingestuft hat, an den vom System als relevant eingestuften Dokumenten.

$$\text{Prec}_k = \frac{\#(R_k^S \cap R_k^T)}{\#R_k^S} \in [0...1]$$

- *Recall* definiert sich als der Anteil der tatsächlich relevanten Dokumente, die vom System als relevant eingestuft wurden, an der Gesamtzahl der tatsächlich relevanten Dokumenten und ist ein Maß für die Vollständigkeit des vom System gelieferten Ergebnisses.

$$\text{Rec}_k = \frac{\#(R_k^S \cap R_k^T)}{\#R_k^T} \in [0...1]$$

- *Fehlerrate* ist der Anteil der vom System falsch eingestuften Dokumente an der Gesamtzahl der Dokumente.

$$\text{Err}_k = \frac{\#(\complement R_k^S \cap R_k^T) + \#(R_k^S \cap \complement R_k^T)}{\#D} \in [0...1]$$

Während man bei der Konzeption eines Verfahrens bemüht ist, die Fehlerrate möglichst zu minimieren, versucht man die Precision und den Recall möglichst zu maximieren. Da die beiden Maße Precision und Recall in Kombination eine genauere Aussage über die Effektivität eines IF-/IR-Systems aus Benutzersicht liefern als die Fehlerrate alleine, werden diese Maße in der IF- und IR-Literatur bevorzugt verwendet. Eine allgemeine Feststellung, die man regelmäßig bei Verwendung der beiden Maße macht, ist die, dass die Verbesserung eines Verfahrens in Bezug auf die Precision zu Lasten des Recall geht und umgekehrt.[19]

Da man im Allgemeinen Systeme nicht an allen $\#K$ Kriterien im einzelnen bewerten möchte, werden die Messgrößen üblicherweise gemittelt, um aggregierte und besser vergleichbare Ergebnisse zu bekommen. Dabei stehen für Precision und Recall grundsätzlich zwei Alternativen der Mittelwertbildung zur Auswahl: Bei der *Makrobewertung*, die gelegentlich auch

[19] Vgl. dazu z. B. SALTON und MCGILL [129, S. 180], RIJSBERGEN [152, S. 33] und FERBER [50, S. 52].

als nutzerorientierter Ansatz bezeichnet wird, werden die arithmetischen Mittel zu Precision und Recall wie gewohnt gebildet:

$$\text{Prec}^{\text{Makro}} = \frac{1}{\#K} \sum_{k \in K} \text{Prec}_k$$

$$\text{Rec}^{\text{Makro}} = \frac{1}{\#K} \sum_{k \in K} \text{Rec}_k$$

Bei dieser Art der Mittelwertbildung gehen alle Versuche bzw. Kriterien k im gleichen Maße in das Ergebnis ein, unabhängig davon, ob es zu einem Kriterium viele oder wenige relevante Dokumente existieren bzw. vom System als solche deklariert werden. Allerdings besteht bei diesem Ansatz ein nummerisches Problem (Division durch Null), wenn es zu einem Kriterium keine relevanten Dokumente gibt, bzw. wenn das Testsystem keine findet.

Eine alternative Methode zur Mittelwertbildung ist die *Mikrobewertung*, die gelegentlich auch als systemorientierter Ansatz bezeichnet wird. Bei der Mikrobewertung setzen sich die Mittelwerte aus den Summen der Dokumentenzahlen in den Formeln zu Precision und Recall zusammen:

$$\text{Prec}^{\text{Mikro}} = \frac{\sum_{k \in K} \#(R_k^S \cap R_k^T)}{\sum_{k \in K} \#R_k^S}$$

$$\text{Rec}^{\text{Mikro}} = \frac{\sum_{k \in K} \#(R_k^S \cap R_k^T)}{\sum_{k \in K} \#R_k^T}$$

Dieser Ansatz weist im Allgemeinen keine numerischen Probleme auf. Die Messungen der einzelnen Kriterien gehen in Abhängigkeit von der Anzahl der relevanten Dokumente ein. Beide Arten der Mittelwertbildung finden in der Literatur Verwendung. Ein Vergleich beider Arten von Mittelwerten kann Einblicke in die Effektivität eines Systems in Bezug auf z. B. sehr spezielle Anfragen (mit nur wenigen relevanten Dokumenten) und eher allgemeine Anfragen (mit vielen relevanten Dokumenten) liefern. Ist bei einem System der Makro-Recall z. B. deutlich höher als der Mikro-Recall, dann heißt das (unter der Voraussetzung, dass eine hinreichende Anzahl an Dokumenten und Anfragen beider Kategorien im Test vorhanden ist), dass das System zu speziellen Anfragen tendenziell anteilsmäßig mehr relevante Dokumente findet als zu allgemeinen Anfragen.[20]

6.4.2 Evaluation von IR-Systemen

Eines der größten Probleme bei dem Versuch ein IR-System bezüglich der Effektivität objektiv zu evaluieren, ist die Bereitstellung der Testdaten. Insbesondere werden Dokumente benötigt,

[20] Da $\#D$ über alle Kriterien $k \in K$ konstant ist, gilt für die Fehlerrate, dass die Makrobewertung gleich der Mikrobewertung ist. Somit berechnet sich der Makro- wie Mikromittelwert der Fehlerrate wie folgt:

$$\text{Err} = \frac{1}{\#K} \sum_{k \in K} \text{Err}_k = \frac{\sum_{k \in K} (\#(\complement R_k^S \cap R_k^T) + \#(R_k^S \cap \complement R_k^T))}{\sum_{k \in K} \#D}$$

die vom Testsystem bezüglich ihrer Relevanz zu bestimmten Anfragen bewertet werden sollen. Die Dokumentenmenge sollte aus theoretischer Sicht möglichst heterogen und möglichst groß sein. Zusätzlich sollten zu dieser Dokumentenmenge möglichst viele realistische Anfragen und die Zuordnung von relevanten Dokumenten zu diesen Anfragen vorliegen. Hierbei bestehen grundsätzlich zwei Probleme: Erstens, ist es sehr aufwändig zu einer hohen Anzahl von Dokumenten und einer hohen Anzahl von Anfragen die Relevanz eines jeden Dokuments in Bezug auf jede Anfrage manuell zu bestimmen. Zweitens, sollte die Relevanz der Dokumente zu den Anfragen objektiv sein. Dieses ist jedoch faktisch nicht realisierbar, weil es immer bestimmte Kombinationen von Anfragen und Dokumenten geben wird, zu denen verschiedene Personen unterschiedliche Meinungen haben. Um diesen Effekt zu mildern, ist es somit idealerweise erforderlich, die Relevanz von Dokumenten zu spezifischen Anfragen nicht nur von einer, sondern möglichst von mehreren (vielen) Personen zu bewerten. Aufgrund des hohen Aufwands dieser Art von Testdatenerhebung ist es nicht verwunderlich, dass bei der Evaluation von vielen System auf bereits existierende Testdaten zurückgegriffen wird, die zudem häufig relativ klein (1.000 bis 10.000 Dokumente umfassend) und zudem themenspezifisch sind. Eine Liste einiger bekannter (leider ausschließlich englischsprachiger) Testkollektionen findet sich in [50, S. 56]. Neben diesen Testkollektionen gibt es fast jährlich sogenannte TREC[21] Experimente des National Institute of Standards and Technology der USA. Bei diesen Großversuchen können verschiedene Forschergruppen ihre Testsysteme gegen einen großen Testkorpus in eine Art Wettbewerb gegeneinander antreten lassen.

Stehen die Testdaten erst einmal zur Verfügung, dann müssen im ersten Schritt die Dokumente in das Testsystem eingestellt werden. Anschließend können die Testanfragen an das System gestellt werden und die Systemergebnisse müssen mit den Relevanzinformationen zu den einzelnen Dokumenten verglichen werden. Bei IR-Verfahren, die Dokumente nicht explizit in relevante und nicht relevante Dokumente trennen, sondern die Dokumente lediglich in eine relevanz-spezifische Reihenfolge bringen (wie z. B. beim VSM), wird üblicherweise ein Schwellenwert festgelegt, ab dem Dokumente als relevant bzw. nicht relevant gelten. Dieser Schwellenwert kann in einer Reihe von mehreren Versuchen variiert werden, so dass eine Tabelle oder Grafik von verschiedenen Precision-Recall-Kombinationen für das Verfahren aufgestellt werden kann.

6.4.3 Evaluation von IF-Systemen

Im Unterschied zu den IR-Systemen wird die Relevanz von Dokumenten bei den IF-Systemen unabhängig von einer konkreten Anfrage subjektiv, d. h. benutzerspezifisch definiert. Somit entfällt hier die Problematik der Objektivität der Relevanz. Für eine möglichst objektive Evaluierung eines IF-Systems sind allerdings neben einer möglichst großen Zahl von Dokumenten auch eine möglichst große Anzahl von Benutzern mit verschiedenen langfristigen Informationsbedarfen erforderlich. Zudem kann ein solches IF-System nur dann unter realistischen Bedingungen getestet werden, wenn es in Form eines Feldexperimentes über eine längere Zeit durchgeführt wird. Nur so kann auch untersucht werden, wie sich bei adaptiven IF-Systemen

[21] Text Retrieval Conference: `http://trec.nist.gov`

die Benutzerprofile an veränderte Benutzerinteressen und an neue Dokumente anpassen. Problematisch dabei ist allerdings die Erfassung der benutzerspezifischen Relevanzen von Dokumenten, die gegen die Systembewertungen getestet werden. Um die Relevanzen lückenlos zu erfassen, ist es erforderlich, dass die Systembenutzer alle Dokumente manuell bewerten. Dieses widerspricht jedoch dem ursprünglichen Gedanken des IR-Systems, dass den Benutzer gerade von dieser Aufgabe entlasten soll, wodurch die Akzeptanz des Systems leidet. Eine Alternative zur vollständigen und expliziten Erfassung aller benutzerspezifischen Relevanzen ist die genaue Beobachtung des Benutzerverhaltens[22], um implizit auf Relevanzen zu schließen. Allerdings werden so nicht alle Relevanzen erfasst, insbesondere werden diejenigen Dokumentrelevanzen (meistens) nicht erfasst, deren Dokumente vom System als nicht relevant eingestuft wurden, weil diese Dokumente seltener von den Benutzern gelesen werden.

[22] Z. B.: Welche Dokumente werden wie lange gelesen?

Kapitel 7

Zusammenfassung

Ausgehend von der Problemstellung der Informationsüberflutung wurden in Kapitel 1 IF- und IR-Systeme zur Lösung der Problemstellung motiviert. In Kapitel 2 wurden die grundlegenden Begriffe und Methoden aus den Fachgebieten der Datenmodellierung und der Computerlinguistik, die für diese Arbeit von Bedeutung sind, vorgestellt. Darauf aufbauend wurde in Kapitel 3 eine Vielzahl von bereits bekannten und teilweise in der Literatur oft diskutierten Modellen zur Repräsentation von natürlichsprachlichen Dokumenten vorgestellt. Im Unterschied zu den, in der Fachliteratur üblichen quantitativen Vergleichen (über Precision, Recall etc.), sind die Modelle in dieser Arbeit nach qualitativen, insbesondere linguistischen Aspekten, verglichen worden. Dabei wurde festgestellt, dass viele der gängigen Modelle eine Vielzahl von linguistischen Phänomenen nicht oder nicht angemessen repräsentieren bzw. dass sehr viele Trainingsdaten für adaptive Modelle benötigt werden, um linguistische Zusammenhänge zu erlernen. Zudem wurde deutlich, dass keines der vorgestellten gängigen Modelle als Erklärungsmodell zur Erklärung der, in der Praxis üblichen Vorgehensweisen der Stoppwort-Elimination, des Stemming und der Synonymersetzung geeignet ist.

Daher wurde in Kapitel 4 das TVSM vorgestellt, welches in der Lage ist, eine Vielzahl an linguistischen Phänomenen zu repräsentieren, und das zudem als Erklärungsmodell, wie anhand der Stoppwort-, Stemming- und Synonym-Lemmata gezeigt wurde, geeignet ist. Da das TVSM zum einen in Bezug auf die konkrete Ausprägung einiger Termähnlichkeiten nicht hinreichend operationalisiert ist und zum anderen nicht in der Lage ist, die linguistischen Phänomene der Homographie, Metonymie und Wortgruppen zu repräsentieren, wurde in Kapitel 5 eine Erweiterung des Modells (das eTVSM) vorgestellt. Das eTVSM beseitigt die Unzulänglichkeiten des TVSM und ist, zumindest aus theoretischer Sicht und auf linguistische Aspekte bezogen, den vorgestellten gängigen Modellen (bis auf dem BNN) überlegen. Gegenüber dem BNN hat das eTVSM den Vorzug, dass linguistische Zusammenhänge dem Modell direkt über eine Ontologie vermittelt werden können und es somit nicht die Zusammenhänge indirekt über eine, aufgrund der hohen Anzahl an Wortinterdependenzen nicht realisierbar hohe Anzahl von Dokumentenähnlichkeitsbeispielen, erlernen muss.

Die prinzipielle Realisierbarkeit und Überlegenheit des eTVSM wurde in Kapitel 5 anhand einer relationalen Beispielimplementierung und anhand einiger Spielzeugbeispiele, sogenannter *Toy-Examples*, demonstriert. Für die Zukunft heißt das, dass sich das eTVSM zunächst in aufwändigen praxisnahen Tests und später in Praxisanwendungen beweisen muss, damit es auch anhand von quantitativen und realitätsnahen Messdaten mit anderen Modellen verglichen werden kann. Wie das eTVSM in der Praxis eingesetzt werden kann und wie praxisnahe Tests im Groben aussehen können, wurde in Kapitel 6 erläutert.

Fazit: Das in dieser Arbeit vorgestellte eTVSM zeichnet sich durch die folgenden Eigenschaften aus:

- Das eTVSM birgt das Potential IF- und IR-Aufgaben besser zu lösen als die bisherigen Ansätze, weil es Wortzusammenhänge unter Verwendung von Ontologien berücksichtigt.
- Das Modell kann entweder schrittweise (auch während der Einsatzphase) um Wortzusammenhänge erweitert werden oder an bereits vorhandene Ontologien (wie z. B. *WordNet* oder *GermaNet*) angebunden werden.
- Das eTVSM kann (weitestgehend) unter Verwendung von relationalen Datenbanken implementiert werden, wodurch eine Massendaten-taugliche Implementierung des Verfahrens mit einem relativ geringen Programmieraufwand möglich wird. Zusätzlich kann es über die SQL-Schnittstelle der Datenbank relativ leicht an andere Anwendungen angeschlossen werden.
- Eine ausführliche Evaluation des eTVSM mit aufwändigen und praxisnahen Tests steht noch aus. Versuche mit einfachen Beispielen unter idealisierten Bedingungen sind jedoch vielversprechend.

Anhang A

Datenbankeinträge der Beispiel-Ontologie

Die folgenden Ausdrucke von Datenbanktabellen enthalten die vollständigen Datensätze zu Tabellen und Views, die bei der Umsetzung der Beispiel-Ontologie aus Abbildung 5.25 auf Seite 147 in einer relationalen Implementierung des eTVSM anfallen. Die Definition der Tabellen und Views sind in Abschnitt 5.3 dokumentiert.

A.1 Vor der Initialisierung bekannt

Die folgenden Datensätze lassen sich (vom Stemming abgesehen) direkt aus der Beispiel-Ontologie ableiten. Sie müssen vor der Initialisierung des eTVSM entweder manuell oder unter Verwendung eines grafischen Modellierungswerkzeugs aus der grafischen Repräsentation der Ontologie abgeleitet werden. Alle anderen Tabellen sind zu diesem Zeitpunkt leer, d. h. sie enthalten keine Datensätze.

A.1.1 Tabelle `Thema`

Id	Beschr
1	Firma
2	Betriebssystem
3	Open Source
4	Microsoft
5	SCO
6	Unix
7	Bill Gates
8	Steve Ballmer

```
  9 | Darl McBride
 10 | Chris Sontag
 11 | Windows
 12 | SCO Unix
 13 | Linux
 14 | GNU
 15 | Linus Torvalds
 16 | Alan Cox
 17 | Nagetier
 18 | Biber
 19 | Maus (Nagetier)
 20 | Eingabegerät
 21 | Computermaus
 22 | Tastatur
 23 | Preisvorteil
 24 | Gemeinde
 25 | Loch
 26 | Sicherheitslücke
```

A.1.2 Tabelle `Themenstruktur`

```
 Superthema | Subthema
------------+----------
          1 |        4
          1 |        5
          2 |        6
          2 |       11
          3 |       13
          3 |       14
          4 |        7
          4 |        8
          4 |       11
          5 |        9
          5 |       10
          5 |       12
          6 |       12
          6 |       13
         13 |       15
         13 |       16
         17 |       18
         17 |       19
         20 |       21
         20 |       22
```

A.1.3 Tabelle `Interpretation`

Id	Beschr	Gewicht
1	Firma	1
2	Betriebssystem	1
3	Open Source	1
4	Microsoft	1
5	SCO	1
6	Unix	1
7	Bill Gates	1
8	Steve Ballmer	1
9	Darl McBride	1
10	Chris Sontag	1
11	Windows	1
12	SCO Unix	1
13	Linux	1
14	GNU	1
15	Linus Torvalds	1
16	Alan Cox	1
17	Nagetier	1
18	Biber	1
19	Maus (Nagetier)	1
20	Eingabegerät	1
21	Computermaus	1
22	Tastatur	1
23	Preisvorteil	1
24	Gemeinde	1
25	Loch	1
26	Sicherheitslücke	1
27	Maus	1

A.1.4 Tabelle `IT_Zuo`

Interpretation	Thema
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10
11	11

```
            12 |    12
            13 |    13
            14 |    14
            15 |    15
            16 |    16
            17 |    17
            18 |    18
            19 |    19
            20 |    20
            21 |    21
            22 |    22
            23 |    23
            24 |    24
            25 |    25
            26 |    26
            27 |    19
            27 |    21
```

A.1.5 Tabelle `Term`

```
 Id |       Text
----+------------------
  1 | Firma
  2 | Betriebssystem
  3 | Open Source
  4 | Microsoft
  5 | SCO
  6 | Unix
  7 | Bill Gates
  8 | Steve Ballmer
  9 | Darl McBride
 10 | Chris Sontag
 11 | Windows
 12 | SCO Unix
 13 | Linux
 14 | GNU
 15 | Linus Torvalds
 16 | Alan Cox
 17 | Nagetier
 18 | Biber
 20 | Eingabegerät
 21 | Computermaus
 22 | Tastatur
 23 | Preisvorteil
 24 | Gemeinde
 25 | Loch
```

```
 26 | Sicherheitslücke
 27 | Maus
 28 | Gates
 29 | Ballmer
 30 | McBride
 31 | Sontag
 32 | Torvalds
 33 | Cox
 34 | Bug
```

A.1.6 Tabelle TI_Zuo

Term	Interpretation
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10
11	11
12	12
13	13
14	14
15	15
16	16
17	17
18	18
20	20
21	21
22	22
23	23
24	24
25	25
26	26
27	19
27	21
27	27
28	7
29	8
30	9
31	10

```
   32 |             15
   33 |             16
   34 |             26
```

A.1.7 Tabelle `Supportterm`

```
 Term | Interpretation
------+----------------
   17 |             19
   25 |             19
   20 |             21
   22 |             21
```

A.1.8 Tabelle `Wortstamm`

```
 Id |       Text
----+-----------------
  1 | Firma
  2 | Betriebssystem
  3 | Open
  4 | Microsoft
  5 | SCO
  6 | Unix
  7 | Bill
  8 | Steve
  9 | Darl
 10 | Chris
 11 | Windows
 13 | Linux
 14 | GNU
 15 | Linus
 16 | Alan
 17 | Nagetier
 18 | Biber
 20 | Eingabegerät
 21 | Computermaus
 22 | Tastatur
 23 | Preisvorteil
 24 | Gemeinde
 25 | Loch
 26 | Sicherheitslücke
 27 | Maus
 28 | Gates
 29 | Ballmer
 30 | McBride
 31 | Sontag
```

```
 32 | Torvalds
 33 | Cox
 34 | Bug
 35 | Source
```

A.1.9 Tabelle WT_Zuo

Wortstamm	Term	Position
1	1	1
2	2	1
3	3	1
35	3	2
4	4	1
5	5	1
6	6	1
7	7	1
28	7	2
8	8	1
29	8	2
9	9	1
30	9	2
10	10	1
31	10	2
11	11	1
5	12	1
6	12	2
13	13	1
14	14	1
15	15	1
32	15	2
16	16	1
33	16	2
17	17	1
18	18	1
20	20	1
21	21	1
22	22	1
23	23	1
24	24	1
25	25	1
26	26	1
27	27	1
28	28	1
29	29	1
30	30	1

```
  31 |   31 |        1
  32 |   32 |        1
  33 |   33 |        1
  34 |   34 |        1
```

A.1.10 Tabelle `Wort`

Id	Text	Wortstamm
1	Firma	1
2	Betriebssystem	2
4	Microsoft	4
5	SCO	5
6	Unix	6
11	Windows	11
13	Linux	13
14	GNU	14
17	Nagetier	17
18	Biber	18
20	Eingabegerät	20
21	Computermaus	21
22	Tastatur	22
23	Preisvorteil	23
24	Gemeinde	24
25	Loch	25
26	Sicherheitslücke	26
27	Maus	27
28	Gates	28
29	Ballmer	29
30	McBride	30
31	Sontag	31
32	Torvalds	32
33	Cox	33
34	Bug	34
7	Bill	7
8	Steve	8
9	Darl	9
10	Chris	10
15	Linus	15
16	Alan	16
35	Preisvorteile	23
36	Sicherheitslücken	26
37	Bugs	34
38	Mäuse	27
39	Mäusen	27
40	Löcher	25

```
 41 | Löchern           |        25
 42 | Lochs             |        25
  3 | Open              |         3
 43 | Source            |        35
```

A.2 Nach der Initialisierung

A.2.1 Tabelle `Themavektor`

```
 Thema | Dimension |       Wert
-------+-----------+-------------------
     1 |         1 | 0.751241577797594
     1 |         2 | 0.218424156832324
     1 |         4 | 0.378198242924232
     1 |         5 | 0.373043334873361
     1 |         6 | 0.104144190713902
     1 |         7 | 0.131959138402905
     1 |         8 | 0.131959138402905
     1 |         9 |  0.13444957207973
     1 |        10 |  0.13444957207973
     1 |        11 | 0.114279966118422
     1 |        12 | 0.104144190713902
     2 |         1 | 0.456491323742935
     2 |         2 | 0.623266273168069
     2 |         3 | 0.166774949425134
     2 |         4 | 0.298274714690091
     2 |         5 | 0.158216609052844
     2 |         6 | 0.324991558477978
     2 |        11 | 0.298274714690091
     2 |        12 | 0.158216609052844
     2 |        13 | 0.166774949425134
     2 |        15 | 0.083387474712567
     2 |        16 | 0.083387474712567
     3 |         2 | 0.288675134594813
     3 |         3 | 0.721687836487032
     3 |         6 | 0.288675134594813
     3 |        13 | 0.288675134594813
     3 |        14 | 0.433012701892219
     3 |        15 | 0.144337567297406
     3 |        16 | 0.144337567297406
     4 |         1 | 0.642016166044643
     4 |         2 | 0.193997690565051
     4 |         4 | 0.642016166044643
     4 |         7 | 0.224009237739796
     4 |         8 | 0.224009237739796
```

```
 4 |        11 | 0.193997690565051
 5 |         1 | 0.633265373662469
 5 |         2 |  0.17679155122716
 5 |         5 | 0.633265373662469
 5 |         6 |  0.17679155122716
 5 |         9 | 0.228236911217655
 5 |        10 | 0.228236911217655
 5 |        12 |  0.17679155122716
 6 |         1 | 0.265219613431248
 6 |         2 | 0.544785632961958
 6 |         3 |  0.27956601953071
 6 |         5 | 0.265219613431248
 6 |         6 | 0.544785632961958
 6 |        12 | 0.265219613431248
 6 |        13 |  0.27956601953071
 6 |        15 | 0.139783009765355
 6 |        16 | 0.139783009765355
 7 |         1 | 0.577350269189626
 7 |         4 | 0.577350269189626
 7 |         7 | 0.577350269189626
 8 |         1 | 0.577350269189626
 8 |         4 | 0.577350269189626
 8 |         8 | 0.577350269189626
 9 |         1 | 0.577350269189626
 9 |         5 | 0.577350269189626
 9 |         9 | 0.577350269189626
10 |         1 | 0.577350269189626
10 |         5 | 0.577350269189626
10 |        10 | 0.577350269189626
11 |         1 |               0.5
11 |         2 |               0.5
11 |         4 |               0.5
11 |        11 |               0.5
12 |         1 | 0.447213595499958
12 |         2 | 0.447213595499958
12 |         5 | 0.447213595499958
12 |         6 | 0.447213595499958
12 |        12 | 0.447213595499958
13 |         2 | 0.471404520791032
13 |         3 | 0.471404520791032
13 |         6 | 0.471404520791032
13 |        13 | 0.471404520791032
13 |        15 | 0.235702260395516
13 |        16 | 0.235702260395516
14 |         3 | 0.707106781186547
14 |        14 | 0.707106781186547
15 |         2 | 0.447213595499958
```

```
    15 |          3 | 0.447213595499958
    15 |          6 | 0.447213595499958
    15 |         13 | 0.447213595499958
    15 |         15 | 0.447213595499958
    16 |          2 | 0.447213595499958
    16 |          3 | 0.447213595499958
    16 |          6 | 0.447213595499958
    16 |         13 | 0.447213595499958
    16 |         16 | 0.447213595499958
    17 |         17 | 0.816496580927726
    17 |         18 | 0.408248290463863
    17 |         19 | 0.408248290463863
    18 |         17 | 0.707106781186547
    18 |         18 | 0.707106781186547
    19 |         17 | 0.707106781186547
    19 |         19 | 0.707106781186547
    20 |         20 | 0.816496580927726
    20 |         21 | 0.408248290463863
    20 |         22 | 0.408248290463863
    21 |         20 | 0.707106781186547
    21 |         21 | 0.707106781186547
    22 |         20 | 0.707106781186547
    22 |         22 | 0.707106781186547
    23 |         23 |                 1
    24 |         24 |                 1
    25 |         25 |                 1
    26 |         26 |                 1
```

A.2.2 View `ThemenAehnlichkeit`

```
 Thema1 | Thema2 |        Wert
--------+--------+--------------------
      1 |      1 |  0.999999999999999
      1 |      2 |  0.735310444818873
      1 |      3 |  0.093117461143933
      1 |      4 |  0.848782586984761
      1 |      5 |  0.848782586984761
      1 |      6 |   0.50153409331889
      1 |      7 |  0.728269028606177
      1 |      8 |  0.728269028606177
      1 |      9 |  0.726730693609073
      1 |     10 |  0.726730693609073
      1 |     11 |  0.731071971836286
      1 |     12 |  0.693627146642159
      1 |     13 |  0.152060177297383
      1 |     15 |  0.144256950500628
```

```
1 |     16 |  0.144256950500628
2 |      1 |  0.735310444818873
2 |      2 |  0.999999999999999
2 |      3 |  0.466313581070383
2 |      4 |  0.663348821660994
2 |      5 |  0.584888581519564
2 |      6 |  0.838153513145593
2 |      7 |  0.435764375464495
2 |      8 |  0.435764375464495
2 |      9 |  0.354901790472641
2 |     10 |  0.354901790472641
2 |     11 |  0.838153513145593
2 |     12 |  0.769736157761703
2 |     13 |  0.643559191497707
2 |     14 |  0.117927697670556
2 |     15 |  0.610533856280767
2 |     16 |  0.610533856280767
3 |      1 |  0.093117461143933
3 |      2 |  0.466313581070383
3 |      3 |  0.999999999999999
3 |      4 | 0.0560023094349489
3 |      5 |  0.102070649691452
3 |      6 |  0.637347165105882
3 |     11 |  0.144337567297406
3 |     12 |  0.258198889747161
3 |     13 |  0.816496580927725
3 |     14 |  0.816496580927725
3 |     15 |  0.774596669241483
3 |     16 |  0.774596669241483
4 |      1 |  0.848782586984761
4 |      2 |  0.663348821660994
4 |      3 | 0.0560023094349489
4 |      4 |                  1
4 |      5 |  0.440863759937088
4 |      6 |  0.275962434022612
4 |      7 |  0.870668206289966
4 |      8 |  0.870668206289966
4 |      9 |  0.370668206289966
4 |     10 |  0.370668206289966
4 |     11 |  0.836013856609694
4 |     12 |  0.373876762702207
4 |     13 | 0.0914513883553847
4 |     15 | 0.0867584047162847
4 |     16 | 0.0867584047162847
5 |      1 |  0.848782586984761
5 |      2 |  0.584888581519564
5 |      3 |  0.102070649691452
```

```
5 |      4 |  0.440863759937088
5 |      5 |                  1
5 |      6 |  0.575424376353918
5 |      7 |  0.365615933952496
5 |      8 |  0.365615933952496
5 |      9 |  0.863004510035513
5 |     10 |  0.863004510035513
5 |     11 |  0.405028462444814
5 |     12 |  0.803600525157373
5 |     13 |  0.166680672972285
5 |     15 |  0.158127170556626
5 |     16 |  0.158127170556626
6 |      1 |   0.50153409331889
6 |      2 |  0.838153513145593
6 |      3 |  0.637347165105882
6 |      4 |  0.275962434022612
6 |      5 |  0.575424376353918
6 |      6 |  0.999999999999999
6 |      7 |    0.1531246152089
6 |      8 |    0.1531246152089
6 |      9 |    0.3062492304178
6 |     10 |    0.3062492304178
6 |     11 |  0.405002623196603
6 |     12 |  0.843100534146368
6 |     13 |  0.843100534146368
6 |     14 |  0.197683028199496
6 |     15 |  0.799835395322126
6 |     16 |  0.799835395322126
7 |      1 |  0.728269028606177
7 |      2 |  0.435764375464495
7 |      4 |  0.870668206289966
7 |      5 |  0.365615933952496
7 |      6 |    0.1531246152089
7 |      7 |                  1
7 |      8 |  0.666666666666667
7 |      9 |  0.333333333333333
7 |     10 |  0.333333333333333
7 |     11 |  0.577350269189626
7 |     12 |  0.258198889747161
8 |      1 |  0.728269028606177
8 |      2 |  0.435764375464495
8 |      4 |  0.870668206289966
8 |      5 |  0.365615933952496
8 |      6 |    0.1531246152089
8 |      7 |  0.666666666666667
8 |      8 |                  1
8 |      9 |  0.333333333333333
```

```
 8 |     10 |  0.333333333333333
 8 |     11 |  0.577350269189626
 8 |     12 |  0.258198889747161
 9 |      1 |  0.726730693609073
 9 |      2 |  0.354901790472641
 9 |      4 |  0.370668206289966
 9 |      5 |  0.863004510035513
 9 |      6 |    0.3062492304178
 9 |      7 |  0.333333333333333
 9 |      8 |  0.333333333333333
 9 |      9 |                   1
 9 |     10 |  0.666666666666667
 9 |     11 |  0.288675134594813
 9 |     12 |  0.516397779494322
10 |      1 |  0.726730693609073
10 |      2 |  0.354901790472641
10 |      4 |  0.370668206289966
10 |      5 |  0.863004510035513
10 |      6 |    0.3062492304178
10 |      7 |  0.333333333333333
10 |      8 |  0.333333333333333
10 |      9 |  0.666666666666667
10 |     10 |                   1
10 |     11 |  0.288675134594813
10 |     12 |  0.516397779494322
11 |      1 |  0.731071971836286
11 |      2 |  0.838153513145594
11 |      3 |  0.144337567297406
11 |      4 |  0.836013856609694
11 |      5 |  0.405028462444814
11 |      6 |  0.405002623196603
11 |      7 |  0.577350269189626
11 |      8 |  0.577350269189626
11 |      9 |  0.288675134594813
11 |     10 |  0.288675134594813
11 |     11 |                   1
11 |     12 |  0.447213595499958
11 |     13 |  0.235702260395516
11 |     15 |  0.223606797749979
11 |     16 |  0.223606797749979
12 |      1 |  0.693627146642159
12 |      2 |  0.769736157761703
12 |      3 |  0.258198889747161
12 |      4 |  0.373876762702207
12 |      5 |  0.803600525157373
12 |      6 |  0.843100534146368
12 |      7 |  0.258198889747161
```

```
12 |      8 |  0.258198889747161
12 |      9 |  0.516397779494322
12 |     10 |  0.516397779494322
12 |     11 |  0.447213595499958
12 |     12 |                  1
12 |     13 |  0.421637021355784
12 |     15 |                0.4
12 |     16 |                0.4
13 |      1 |  0.152060177297383
13 |      2 |  0.643559191497707
13 |      3 |  0.816496580927725
13 |      4 | 0.0914513883553847
13 |      5 |  0.166680672972285
13 |      6 |  0.843100534146368
13 |     11 |  0.235702260395516
13 |     12 |  0.421637021355784
13 |     13 |                  1
13 |     14 |  0.333333333333333
13 |     15 |  0.948683298050514
13 |     16 |  0.948683298050514
14 |      2 |  0.117927697670556
14 |      3 |  0.816496580927725
14 |      6 |  0.197683028199496
14 |     13 |  0.333333333333333
14 |     14 |                  1
14 |     15 |  0.316227766016838
14 |     16 |  0.316227766016838
15 |      1 |  0.144256950500628
15 |      2 |  0.610533856280767
15 |      3 |  0.774596669241483
15 |      4 | 0.0867584047162847
15 |      5 |  0.158127170556626
15 |      6 |  0.799835395322127
15 |     11 |  0.223606797749979
15 |     12 |                0.4
15 |     13 |  0.948683298050514
15 |     14 |  0.316227766016838
15 |     15 |                  1
15 |     16 |                0.8
16 |      1 |  0.144256950500628
16 |      2 |  0.610533856280767
16 |      3 |  0.774596669241483
16 |      4 | 0.0867584047162847
16 |      5 |  0.158127170556626
16 |      6 |  0.799835395322127
16 |     11 |  0.223606797749979
16 |     12 |                0.4
```

```
    16 |     13 |  0.948683298050514
    16 |     14 |  0.316227766016838
    16 |     15 |                0.8
    16 |     16 |                  1
    17 |     17 |                  1
    17 |     18 |  0.866025403784439
    17 |     19 |  0.866025403784439
    18 |     17 |  0.866025403784439
    18 |     18 |                  1
    18 |     19 |                0.5
    19 |     17 |  0.866025403784439
    19 |     18 |                0.5
    19 |     19 |                  1
    20 |     20 |                  1
    20 |     21 |  0.866025403784439
    20 |     22 |  0.866025403784439
    21 |     20 |  0.866025403784439
    21 |     21 |                  1
    21 |     22 |                0.5
    22 |     20 |  0.866025403784439
    22 |     21 |                0.5
    22 |     22 |                  1
    23 |     23 |                  1
    24 |     24 |                  1
    25 |     25 |                  1
    26 |     26 |                  1
```

A.2.3 Tabelle `Aehnlichkeit`

```
 Int1 | Int2 |   Skalarprodukt
------+------+--------------------
    1 |    1 |                  1
    1 |    2 |  0.735310444818874
    1 |    3 | 0.0931174611439331
    1 |    4 |  0.848782586984761
    1 |    5 |  0.848782586984762
    1 |    6 |  0.501534093318891
    1 |    7 |  0.728269028606178
    1 |    8 |  0.728269028606178
    1 |    9 |  0.726730693609073
    1 |   10 |  0.726730693609073
    1 |   11 |  0.731071971836287
    1 |   12 |  0.693627146642159
    1 |   13 |  0.152060177297383
    1 |   15 |  0.144256950500628
    1 |   16 |  0.144256950500628
```

```
2 |    1 |  0.735310444818874
2 |    2 |                  1
2 |    3 |  0.466313581070383
2 |    4 |  0.663348821660995
2 |    5 |  0.584888581519564
2 |    6 |  0.838153513145594
2 |    7 |  0.435764375464496
2 |    8 |  0.435764375464496
2 |    9 |  0.354901790472642
2 |   10 |  0.354901790472642
2 |   11 |  0.838153513145594
2 |   12 |  0.769736157761703
2 |   13 |  0.643559191497708
2 |   14 |  0.117927697670556
2 |   15 |  0.610533856280768
2 |   16 |  0.610533856280768
3 |    1 | 0.0931174611439331
3 |    2 |  0.466313581070383
3 |    3 |                  1
3 |    4 |  0.056002309434949
3 |    5 |  0.102070649691452
3 |    6 |  0.637347165105883
3 |   11 |  0.144337567297406
3 |   12 |  0.258198889747161
3 |   13 |  0.816496580927726
3 |   14 |  0.816496580927726
3 |   15 |  0.774596669241483
3 |   16 |  0.774596669241483
4 |    1 |  0.848782586984761
4 |    2 |  0.663348821660995
4 |    3 |  0.056002309434949
4 |    4 |                  1
4 |    5 |  0.440863759937088
4 |    6 |  0.275962434022612
4 |    7 |  0.870668206289966
4 |    8 |  0.870668206289966
4 |    9 |  0.370668206289966
4 |   10 |  0.370668206289966
4 |   11 |  0.836013856609694
4 |   12 |  0.373876762702207
4 |   13 | 0.0914513883553847
4 |   15 | 0.0867584047162847
4 |   16 | 0.0867584047162847
5 |    1 |  0.848782586984762
5 |    2 |  0.584888581519564
5 |    3 |  0.102070649691452
5 |    4 |  0.440863759937088
```

```
5 |    5 |                  1
5 |    6 |  0.575424376353917
5 |    7 |  0.365615933952495
5 |    8 |  0.365615933952495
5 |    9 |  0.863004510035512
5 |   10 |  0.863004510035512
5 |   11 |  0.405028462444814
5 |   12 |  0.803600525157373
5 |   13 |  0.166680672972284
5 |   15 |  0.158127170556626
5 |   16 |  0.158127170556626
6 |    1 |  0.501534093318891
6 |    2 |  0.838153513145594
6 |    3 |  0.637347165105883
6 |    4 |  0.275962434022612
6 |    5 |  0.575424376353917
6 |    6 |                  1
6 |    7 |    0.1531246152089
6 |    8 |    0.1531246152089
6 |    9 |    0.3062492304178
6 |   10 |    0.3062492304178
6 |   11 |  0.405002623196604
6 |   12 |  0.843100534146369
6 |   13 |  0.843100534146369
6 |   14 |  0.197683028199496
6 |   15 |  0.799835395322127
6 |   16 |  0.799835395322127
7 |    1 |  0.728269028606178
7 |    2 |  0.435764375464496
7 |    4 |  0.870668206289966
7 |    5 |  0.365615933952495
7 |    6 |    0.1531246152089
7 |    7 |                  1
7 |    8 |  0.666666666666667
7 |    9 |  0.333333333333333
7 |   10 |  0.333333333333333
7 |   11 |  0.577350269189626
7 |   12 |  0.258198889747161
8 |    1 |  0.728269028606178
8 |    2 |  0.435764375464496
8 |    4 |  0.870668206289966
8 |    5 |  0.365615933952495
8 |    6 |    0.1531246152089
8 |    7 |  0.666666666666667
8 |    8 |                  1
8 |    9 |  0.333333333333333
8 |   10 |  0.333333333333333
```

```
 8 |   11 |  0.577350269189626
 8 |   12 |  0.258198889747161
 9 |    1 |  0.726730693609073
 9 |    2 |  0.354901790472642
 9 |    4 |  0.370668206289966
 9 |    5 |  0.863004510035512
 9 |    6 |    0.3062492304178
 9 |    7 |  0.333333333333333
 9 |    8 |  0.333333333333333
 9 |    9 |                   1
 9 |   10 |  0.666666666666667
 9 |   11 |  0.288675134594813
 9 |   12 |  0.516397779494322
10 |    1 |  0.726730693609073
10 |    2 |  0.354901790472642
10 |    4 |  0.370668206289966
10 |    5 |  0.863004510035512
10 |    6 |    0.3062492304178
10 |    7 |  0.333333333333333
10 |    8 |  0.333333333333333
10 |    9 |  0.666666666666667
10 |   10 |                   1
10 |   11 |  0.288675134594813
10 |   12 |  0.516397779494322
11 |    1 |  0.731071971836287
11 |    2 |  0.838153513145594
11 |    3 |  0.144337567297406
11 |    4 |  0.836013856609694
11 |    5 |  0.405028462444814
11 |    6 |  0.405002623196604
11 |    7 |  0.577350269189626
11 |    8 |  0.577350269189626
11 |    9 |  0.288675134594813
11 |   10 |  0.288675134594813
11 |   11 |                   1
11 |   12 |  0.447213595499958
11 |   13 |  0.235702260395516
11 |   15 |  0.223606797749979
11 |   16 |  0.223606797749979
12 |    1 |  0.693627146642159
12 |    2 |  0.769736157761703
12 |    3 |  0.258198889747161
12 |    4 |  0.373876762702207
12 |    5 |  0.803600525157373
12 |    6 |  0.843100534146369
12 |    7 |  0.258198889747161
12 |    8 |  0.258198889747161
```

```
12 |    9 |  0.516397779494322
12 |   10 |  0.516397779494322
12 |   11 |  0.447213595499958
12 |   12 |                  1
12 |   13 |  0.421637021355784
12 |   15 |                0.4
12 |   16 |                0.4
13 |    1 |  0.152060177297383
13 |    2 |  0.643559191497708
13 |    3 |  0.816496580927726
13 |    4 | 0.0914513883553847
13 |    5 |  0.166680672972284
13 |    6 |  0.843100534146369
13 |   11 |  0.235702260395516
13 |   12 |  0.421637021355784
13 |   13 |                  1
13 |   14 |  0.333333333333333
13 |   15 |  0.948683298050514
13 |   16 |  0.948683298050514
14 |    2 |  0.117927697670556
14 |    3 |  0.816496580927726
14 |    6 |  0.197683028199496
14 |   13 |  0.333333333333333
14 |   14 |                  1
14 |   15 |  0.316227766016838
14 |   16 |  0.316227766016838
15 |    1 |  0.144256950500628
15 |    2 |  0.610533856280768
15 |    3 |  0.774596669241483
15 |    4 | 0.0867584047162847
15 |    5 |  0.158127170556626
15 |    6 |  0.799835395322127
15 |   11 |  0.223606797749979
15 |   12 |                0.4
15 |   13 |  0.948683298050514
15 |   14 |  0.316227766016838
15 |   15 |                  1
15 |   16 |                0.8
16 |    1 |  0.144256950500628
16 |    2 |  0.610533856280768
16 |    3 |  0.774596669241483
16 |    4 | 0.0867584047162847
16 |    5 |  0.158127170556626
16 |    6 |  0.799835395322127
16 |   11 |  0.223606797749979
16 |   12 |                0.4
16 |   13 |  0.948683298050514
```

```
  16 |   14 |  0.316227766016838
  16 |   15 |                0.8
  16 |   16 |                  1
  17 |   17 |                  1
  17 |   18 |  0.866025403784439
  17 |   19 |  0.866025403784439
  17 |   27 |  0.612372435695794
  18 |   17 |  0.866025403784439
  18 |   18 |                  1
  18 |   19 |                0.5
  18 |   27 |  0.353553390593274
  19 |   17 |  0.866025403784439
  19 |   18 |                0.5
  19 |   19 |                  1
  19 |   27 |  0.707106781186548
  20 |   20 |                  1
  20 |   21 |  0.866025403784439
  20 |   22 |  0.866025403784439
  20 |   27 |  0.612372435695794
  21 |   20 |  0.866025403784439
  21 |   21 |                  1
  21 |   22 |                0.5
  21 |   27 |  0.707106781186548
  22 |   20 |  0.866025403784439
  22 |   21 |                0.5
  22 |   22 |                  1
  22 |   27 |  0.353553390593274
  23 |   23 |                  1
  24 |   24 |                  1
  25 |   25 |                  1
  26 |   26 |                  1
  27 |   17 |  0.612372435695794
  27 |   18 |  0.353553390593274
  27 |   19 |  0.707106781186548
  27 |   20 |  0.612372435695794
  27 |   21 |  0.707106781186548
  27 |   22 |  0.353553390593274
  27 |   27 |                  1
```

A.3 Dokumente einfügen

A.3.1 Tabelle `Dokument` vor der Betragsberechnung

```
 Id | Betrag |                    Text
----+--------+-------------------------------------------
```

```
  1 |        | Torvalds schreibt an SCO.
  2 |        | McBride warnt die Open-Source-Gemeinde.
  3 |        | Windows hat Preisvorteile gegenüber Linux.
  4 |        | Microsoft schließt Sicherheitslücken.
  5 |        | Neue Bugs in Windows.
  6 |        | Mit Maus und Tastatur geht es leichter.
  7 |        | Mäuse leben gerne in Löchern.
```

A.3.2 Tabelle DW_Zuo

Dokument	Wort	Position
1	32	1
1	5	4
2	30	1
2	3	4
2	43	5
2	24	6
3	11	1
3	35	3
3	13	5
4	4	1
4	36	3
5	37	2
5	11	4
6	27	2
6	22	4
7	38	1
7	41	5

A.3.3 Tabelle DI_Zuo

Dokument	Interpretation	Anzahl
1	5	1
1	15	1
2	3	1
2	9	1
2	24	1
3	11	1
3	13	1
3	23	1
4	4	1
4	26	1
5	11	1
5	26	1

```
     6 |             21 |      1
     6 |             22 |      1
     7 |             19 |      1
     7 |             25 |      1
```

A.3.4 Tabelle `Dokument` nach der Betragsberechnung

```
 Id |      Betrag      |                    Text
----+------------------+-------------------------------------------
  1 | 1.52192455171511 | Torvalds schreibt an SCO.
  2 | 1.73205080756888 | McBride warnt die Open-Source-Gemeinde.
  3 | 1.86317055601226 | Windows hat Preisvorteile gegenüber Linux.
  4 | 1.41421356237310 | Microsoft schließt Sicherheitslücken.
  5 | 1.41421356237310 | Neue Bugs in Windows.
  6 | 1.73205080756888 | Mit Maus und Tastatur geht es leichter.
  7 | 1.41421356237310 | Mäuse leben gerne in Löchern.
```

A.4 Dokumentenähnlichkeit

A.4.1 Ergebnisse der View `DokAehn`

```
 Dok1 | Dok2 |       Wert
------+------+------------------
    1 |    1 |                1
    1 |    2 | 0.659953870659781
    1 |    3 | 0.615036023381974
    1 |    4 | 0.245140411270901
    1 |    5 | 0.292072465008709
    2 |    1 | 0.659953870659781
    2 |    2 |                1
    2 |    3 | 0.387191885607645
    2 |    4 | 0.174187508636031
    2 |    5 | 0.176776695296637
    3 |    1 | 0.615036023381974
    3 |    2 | 0.387191885607645
    3 |    3 |                1
    3 |    4 | 0.351989763853554
    3 |    5 | 0.468971262471725
    4 |    1 | 0.245140411270901
    4 |    2 | 0.174187508636031
    4 |    3 | 0.351989763853554
    4 |    4 |                1
    4 |    5 | 0.918006928304847
    5 |    1 | 0.292072465008709
    5 |    2 | 0.176776695296637
```

```
5 |    3 | 0.468971262471725
5 |    4 | 0.918006928304847
5 |    5 |                 1
6 |    6 |                 1
7 |    7 |                 1
```

Anhang B

VSM: simuliert mit den eTVSM-Tabellen

Dieser Abschnitt zeigt, wie das VSM aus Abschnitt 3.2.2 mit dem in Abschnitt 5.3.1 vorgestellten Datenmodell des eTVSM simuliert werden kann.

B.1 View-Definitionen

Die Simulation des VSM basiert auf einer Reihe von Views, die die Datenstrukturen des eTVSM an die Bedürfnisse des VSM virtuell anpassen. Da diese Simulation lediglich einen experimentellen Charakter hat und dem Vergleich der Ergebnisse zwischen VSM und eTVSM dient, werden jegliche Ausführungsgeschwindigkeit-betreffende Aspekte ignoriert. Schwerpunkt ist vielmehr eine nachvollziehbare Darstellung der Views.

B.1.1 Anzahl der Terme pro Dokument

Die Ausgangsbasis für die Berechnung der Dokumentenähnlichkeiten gemäß des VSM ist die Anzahl eines Terms $t_i \in T$ in einem Dokument $d \in D$: a_{d,t_i}. Diese Anzahl von Termen in einem Dokument berechnet die folgende View:

```
CREATE VIEW VSM_a AS
SELECT dw.Dokument, w.Wortstamm, COUNT(*) AS Anzahl
FROM   DW_Zuo dw, Wort w
WHERE  dw.Wort = w.Id
GROUP BY dw.Dokument, w.Wortstamm;
```

Die View basiert auf der Tabelle `DW_Zuo` und verwendet das Attribut `Wortstamm` der Tabelle `Wort` um die Worte zu stemmen. Des Weiteren unterliegt die View der Annahme, dass

alle Worte, die nicht in der Tabelle `Wort` eingetragen sind, Stoppworte sind.[1] Das Ergebnis der View basierend auf den Daten aus Anhang A.1 findet sich im Anhang B.2.1.

B.1.2 Dokumentabhängige Termgewichte

Zur Bestimmung der Dokumentenabhängigen Termgewichte w_{d,t_i} gemäß der *tf-idf* Gleichung 3.1 auf Seite 51 sind insgesamt vier Views erforderlich. Die erste View dient der Berechnung des Teilausdrucks $\max_{t \in T} a_{d,t}$:

```
CREATE VIEW VSM_max_a AS
SELECT VSM_a.Dokument, MAX(VSM_a.anzahl) AS Anzahl
FROM   VSM_a
GROUP BY VSM_a.Dokument;
```

Die Anzahl der Dokumente $\#D$ wird über die zweite View bestimmt:

```
CREATE VIEW VSM_num_D AS
SELECT COUNT(*) AS Anzahl
FROM   Dokument;
```

Mit Hilfe der dritten View wird der Teilausdruck $\#\{e \in D : \ a_{e,t_i} > 0\}$ berechnet:

```
CREATE VIEW VSM_num_a_positiv AS
SELECT VSM_a.Wortstamm, COUNT(*) AS Anzahl
FROM   VSM_a
WHERE  VSM_a.Anzahl > 0
GROUP BY VSM_a.Wortstamm;
```

Die vierte und letzte View fasst die Ergebnisse der drei vorangehenden Views zusammen und berechnet die Termgewichte w_{d,t_i}:

```
CREATE VIEW VSM_w AS
SELECT VSM_a.Dokument, VSM_a.Wortstamm,
       (VSM_a.Anzahl / VSM_max_a.Anzahl) *
       LOG(VSM_num_D.Anzahl / VSM_num_a_positiv.Anzahl) AS Gewicht
FROM   VSM_a, VSM_max_a, VSM_num_D, VSM_num_a_positiv
WHERE  VSM_a.Dokument = VSM_max_a.Dokument
   AND VSM_a.Wortstamm = VSM_num_a_positiv.Wortstamm;
```

Eine vollständige Auflistung der Termgewichte für die Daten aus Anhang A.1 findet sich in Anhang B.2.2.

[1] Beim VSM entspricht der Begriff Term dem Begriff Wortstamm im eTVSM, weil das VSM keine aus mehreren Worten zusammenhängenden Terme berücksichtigt.

B.1.3 Berechnung der Dokumentenähnlichkeit

Da die Berechnung der Dokumentenähnlichkeiten beim VSM auf dem normierten Skalarprodukt basiert, sind zur Berechnung drei Views erforderlich: Die erste View dient der Berechnung des Betrages eines Dokumentenvektors $|\vec{d}|$:

```
CREATE VIEW VSM_dokbetrag AS
SELECT Dokument, SQRT(SUM(Gewicht * Gewicht)) AS Betrag
FROM   VSM_w
GROUP BY Dokument;
```

Mit der zweiten View wird das unnormierte Skalarprodukt $\vec{d_i}\vec{d_j}$ zwischen den Dokumentenvektoren berechnet:

```
CREATE VIEW VSM_dokaehn_unnorm AS
SELECT w1.Dokument AS Dok1, w2.Dokument AS Dok2,
       SUM(w1.Gewicht * w2.Gewicht) AS Skalarprodukt
FROM   VSM_w w1, VSM_w w2
WHERE  w1.Wortstamm = w2.Wortstamm
GROUP BY Dok1, Dok2;
```

Abschließend fasst die letzte View die Ergebnisse zu der Ähnlichkeit $\text{sim}(d_i, d_j)$, dem normierten Skalarprodukt, zusammen:

```
CREATE VIEW VSM_dokaehn AS
SELECT aehn.Dok1, aehn.Dok2,
       aehn.Skalarprodukt / betrag1.Betrag / betrag2.Betrag AS Wert
FROM   VSM_dokaehn_unnorm aehn,
       VSM_dokbetrag betrag1, VSM_dokbetrag betrag2
WHERE  aehn.Dok1 = betrag1.Dokument
   AND aehn.Dok2 = betrag2.Dokument;
```

Wie beim TVSM und eTVSM liefert diese View nur dann einen Tabelleneintrag, wenn die Ähnlichkeit zwischen zwei Dokumenten größer als Null ist. Eine vollständige Auflistung der Dokumentenähnlichkeiten für die Daten aus Anhang A findet sich in Anhang B.2.3.

B.2 View-Ergebnisse

B.2.1 View VSM_a

```
 Dokument | Wortstamm | Anzahl
----------+-----------+--------
        1 |         5 |      1
        1 |        32 |      1
        2 |         3 |      1
        2 |        24 |      1
```

```
        2 |         30 |      1
        2 |         35 |      1
        3 |         11 |      1
        3 |         13 |      1
        3 |         23 |      1
        4 |          4 |      1
        4 |         26 |      1
        5 |         11 |      1
        5 |         34 |      1
        6 |         22 |      1
        6 |         27 |      1
        7 |         25 |      1
        7 |         27 |      1
```

B.2.2 View VSM_w

```
 Dokument | Wortstamm |      Gewicht
----------+-----------+-------------------
        1 |          5 | 0.845098040014257
        1 |         32 | 0.845098040014257
        2 |          3 | 0.845098040014257
        2 |         24 | 0.845098040014257
        2 |         30 | 0.845098040014257
        2 |         35 | 0.845098040014257
        3 |         11 | 0.544068044350276
        3 |         13 | 0.845098040014257
        3 |         23 | 0.845098040014257
        4 |          4 | 0.845098040014257
        4 |         26 | 0.845098040014257
        5 |         11 | 0.544068044350276
        5 |         34 | 0.845098040014257
        6 |         22 | 0.845098040014257
        6 |         27 | 0.544068044350276
        7 |         25 | 0.845098040014257
        7 |         27 | 0.544068044350276
```

B.2.3 View VSM_dokaehn

```
 Dok1 | Dok2 |       Wert
------+------+------------------
    1 |    1 |                1
    2 |    2 |                1
    3 |    3 |                1
    3 |    5 | 0.224276964299624
    4 |    4 |                1
    5 |    3 | 0.224276964299624
```

```
5 |    5 |                1
6 |    6 |                1
6 |    7 | 0.293021007494381
7 |    6 | 0.293021007494381
7 |    7 |                1
```

Literaturverzeichnis

[1] *German stemming algorithm*.
http://snowball.tartarus.org/german/stemmer.html

[2] *GermaNet – Homepage*. Geladen am 01.10.2003,
http://www.sfs.nphil.uni-tuebingen.de/lsd/

[3] ADAMSON, G.; BOREHAM, J.: *The Use of an Association Measure Based on Character Structure to Identify Semantically Related Pairs of Words and Document Titles*. In *Information Storage and Retrieval* (10), 1974: S. 253–260

[4] AGIRRE, E.; RIGAU, G.: *Word Sense Disambiguation using Conceptual Density*. In *Proceedings of COLING'96*. Kopenhagen, 1996, S. 16–22.
http://www.lsi.upc.es/dept/techreps/ps/R96-8.ps.gz

[5] ARPÍREZ, J. C.; GÓMEZ-PÉREZ, A.; LOZANO, A.; PINTO, H. S.: *(ONTO)^2 Agent: An ontology-based WWW-broker to select ontologies*. In *Workshop on Applications on Ontologies and Problem-Solving Methods at the European Conference on Artificial Intelligence*. Brighton, 1998, S. 16–24

[6] BACH, E.: *An extension of classical transformational grammar*. In *Problems if Linguistic Metatheory — Proceedings of the 1976 Conference*. Michigan State University, 1976

[7] BAEZA-YATES, R.; RIBEIRO-NETO, B.: *Modern Information Retrieval*. Addision Wesley Publishing Company, 1999

[8] BATEMAN, J. A.: *Ontology construction and natural language*. In *Proceedings of the International Workshop on Formal Ontology in Conceptual Analysis and Knowledge Representation*. Padova, 1993, S. 83–93

[9] BATEMAN, J. A.: *On the relationship between ontology construction and natural language: a socio-semiotic view*. In *International Journal on Human-Computer Studies* 43, 1995: S. 929–944

[10] BATES, M.: *Subject Access in Online Catalogs: A Design Model*. In *Journal of the American Society for Information Science* (11), 1986: S. 357–376

[11] BECKER, J.: *Integrationsorientierte Wirtschaftsinformatik*.
http://www.wi.uni-muenster.de/is/aws60.de/becker/modell.htm

[12] BECKER, J.; KUROPKA, D.: *Topic-based Vector Space Model*. In *Proceedings of the 6th International Conference on Business Information Systems*. 2003, S. 7–12

[13] BECKER, J.; SCHÜTTE, R.: *Handelsinformationssysteme*. Verlag Moderne Industrie, Landsberg/Lech, 1996

[14] BECKER, T.; KÖNIG, E.: *Lexikonfreie Lemmatisierung für Substantive im Deutschen*. In *Elektronischer Tagungsband der 6. Konferenz zur Verarbeitung natürlicher Sprache*. 2002.
http://konvens2002.dfki.de/cd/pdf/17V-Becker.pdf

[15] BECKWITH, R.; MILLER, G. A.; TENGI, R.: *Design and Implementation of the WordNet Lexical Database and Searching Software*. Fachbericht, Princeton University, 1993.
ftp://clarity.princeton.edu/pub/wordnet/5papers.ps

[16] BELEW, R.: *Adaptive Information Retrieval: Using a Connectionist Representation to Retrieve and Learn about Documents*. In BELKIN, N.; VAN RIJSBERGEN, C. (Hrsg.): *Proceedings of the 12th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, 1989, S. 11–20

[17] BELKIN, N.; CROFT, W.: *Information filtering and information retrieval: two sides of the same coin?*. In *Communications of the ACM* 35(12), 1992: S. 29–38

[18] BIERWISCH, M.: *Semantische und konzeptuelle Repräsentation lexikalischer Einheiten*. In RUZICKA, R.; MOTSCH, W. (Hrsg.): *Untersuchungen zur Semantik*, Akademie Verlag, Berlin. 1983, S. 61–99

[19] BIGUS, J.: *Data Mining With Neural Networks: Solving Business Problems from Application Development to Decision Support*. McGraw Hill Text, New York, 1996

[20] BILLSUS, D.: *Improving User Model Acquisition from Labeled Text Documents*. In *Proceedings of the 7th International Conference on User Modeling*. 1999

[21] BODE, J.: *Betriebliche Produktion von Informationen*. Wiesbaden, 1993

[22] BOGER, Z.; KUFLIK, T.; SHAPIRA, B.; SHOVAL, P.: *Information Filtering and Automatic Keyword Indentification by Artifical Neuronal Networks*. In HANSEN, H. R.; BICHLER, M.; MAHRER, H. (Hrsg.): *Proceedings of the 8th European Conference on Information Systems*. Vienna, 2000, S. 379–385

[23] BORGO, S.; GUARINO, N.; MASOLO, C.: *Stratified Ontologies: the case of physical objects*. In *Proceedings of the Workshop on Ontological Engineering*. Budapest, 1996, S. 5–15.
http://www.ladseb.pd.cnr.it/infor/Ontology/Papers/StratOntologies.pdf

[24] BRÉAL: *Essai de śemantique*. In *Science des significations*, Paris. 1897

[25] BRESNAN, J. (Hrsg.): *The Mental Representation of Grammatical Relations*. MIT Press, Cambridge (MA), 1982

[26] BRONSTEIN, I. N.; SEMANDJAJEW, K. A.: *Taschenbuch der Mathematik*. Verlag Harri Deutsch, Thun, Frankfurt/Main, 24. Auflage, 1989

[27] BUCHER, W.; MAUERER, H.: *Theoretische Grundlagen der Programmiersprachen*. Bibliographisches Institut, Mannheim, 1984

[28] BUSSMANN, H. (Hrsg.): *Lexikon der Sprachwissenschaft*, Band 3. Alfred Kröner Verlag, Stuttgart, 2002

[29] BUSCH, E.: *Managing Infoglut: Search and Retrieval*. In *BYTE* 17(6), 1992

[30] CARSTENSEN, K.-U.; EBERT, C.; ENDRISS, C.; JEKAT, S.; KLABUNDE, R.; LANGER, H. (Hrsg.): *Computerlinguistik und Sprachtechnologie: eine Einführung*. Spektrum Akademischer Verlag, 2001

[31] CHARNIAK, E.: *Statistical Language Mearning*. The MIT Press, Cambridge, MA, 1993

[32] CHEN, P. P.-S.: *The entity-relationship model — toward a unified view of data*. In *ACM Transactions on Database Systems* 1(1), 1976: S. 9 – 36

[33] CICHOCKI, A.; UNBEHAUEN, R.: *Neuronal Networks for Optimization and Signal Processing*. B. G. Teubner GmbH, Stuttgart, 1993

[34] CORCHO, O.; FERNÁNDEZ-LÓPEZ, M.; HÓMEZ-PÉREZ, A.: *OntoWeb Technical Roadmap v. 1.0*.
`http://www.ontoweb.org/download/deliverables/D11_v1_0.pdf`

[35] CRESTANI, F.: *Logical Imaging and Probabilistic Information Retrieval*. In CRESTANI, F.; LALMAS, M.; VAN RIJSBERGEN, C. (Hrsg.): *Information Retrieval, Uncertainty and Logics*, Kluwer Academic Publisher, Norwell, MA, USA. 1998, S. 247–280

[36] CRESTANI, F.; SANDERSON, M.; VAN RIJSBERGEN, C. J.: *Sense resolution properties of logical imaging*. In *The New Review of Document and Text Management* (1), 1996: S. 277–298

[37] CRESTANI, F.; SEBASTIANI, F.; VAN RIJSBERGEN, C. J.: *Imaging and information retrieval: variation on a theme*. In *Proceedings of the 2nd International Workshop on Information Retrieval, Unvertainty and Logic*. Glasgow, UK, 1996, S. 27–31

[38] CRESTANI, F.; VAN RIJSBERGEN, C. J.: *Information Retrieval by Logical Imaging*. In *Journal of Documentation* 51(1), 1995: S. 1–15

[39] CRESTANI, F.; VAN RIJSBERGEN, C. J.: *Probability kinematics in information retrieval*. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*. Seattle, WA, USA, 1995, S. 291–299

[40] DASARATHY, B. (Hrsg.): *Nearest Neigbor (NN) Norms: NN pattern classification techniques*. IEEE Computer Society Press, Washington, Brüssel, Tokyo, 1991

[41] DAWSON, J.: *Suffix Removal and Word Conflation*. In *ALLC Bulletin* 1974: S. 33–46

[42] DEGEN, W.; HELLER, B.; HERRE, H.; SMITH, B.: *GOL: A General Ontological Language*. In *Proceedings of Formal Ontology in Information Systems*. 2001

[43] DITTMANN, L.; PENZEL, J.: *Platons Gütekriterium für Ontologien*. In *Proceedings der Tagung Wissenschaftstheorie in Ökonomie und Wirtschaftsinformatik*. 2003, S. 412–431

[44] DÖLLING, J.: *Sortale Selektionsbeschränkungen und systematische Bedeutungsvariationen*. In SCHWARZ, N. (Hrsg.): *Kognitive Semantik / Cognitive Semantics*, Narr Verlag, Tübingen. 1994, S. 41–59

[45] DORFFNER, G.: *Konnektionismus. Von neuronalen Netzwerken zu einer netürlichen KI*. Teubner, Stuttgart, 1991

[46] DROTT, M.: *A Big Stop List*.
`http://drott.cis.drexel.edu/retrieval.html`

[47] EHLERS, L.: *Content Management Anwendungen*. Logos, 2003

[48] FELLBAUM, C.: *English Verbs as a Semantic Net*. In *International Journal of Lexocography* 3(4), 1990: S. 270–301.
`ftp://clarity.princeton.edu/pub/wordnet/5papers.ps`

[49] FELLBAUM, C.; GROSS, D.; MILLER, K.: *Adjectives in WordNet*. In *International Journal of Lexicography* 1990.
`ftp://clarity.princeton.edu/pub/wordnet/5papers.ps`

[50] FERBER, R.: *Data Mining und Information Retrieval*, 2000.
`http://information-retrieval.de/dm-ir`

[51] FORSYTHE, G. E.; MALCOM, M. A.; MOLER, C. B.: *Computer Methods for Mathematical Computations*. Prentice Hall, Englewood Cliffs (NJ), 1977

[52] Foundation for Intelligent Physical Agents: *FIPA Agent Management Specification*, 2002.
`http://www.fipa.org/specs/fipa00023`

[53] FURNAS, G. W.; DEERWESTER, S.; DUMAIS, S. T.; LANDAUER, T. K.; HARSHMAN, R. A.; STREETER, L. A.; LOCHBAUM, K. E.: *Information Retrieval using a Singular Value Decomposition Model of Latent Semantic Structure*. In *Proceedings of the 11th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1988, S. 465..480

[54] GAZDAR, G.; KLEIN, E.; PULLUM, G.; SAG, I.: *Generalized Phrase Structure Grammar*. Basil Blackwell, Oxford, 1985

[55] GENCSERETH, M.; FIKES, R.: *Knowledge Interchange Format Version 3.0*. Computer Science Department, Stanford University, 1994.
`http://meta2.stanford.edu/kif/Hypertext/kif-manual.html`

[56] GROSSER, B.: *Ein paralleler und hochgenauer $O(n^2)$ Algorithmus für die bidiagonale Singulärwertzerlegung*. Dissertation, Bergische Universität Gesamthochschule Wuppertal, 2001.
`http://www.bib.uni-wuppertal.de/elpub/fb07/diss2001/grosser`

[57] GROSSO, E.; ERIKSSON, H.; FERGERSON, R. W.; TU, S. W.; MUSEN, M. M.: *Knowledge modeling at the millenium — the design and evolution of Protégé-2000*. In *Proceedings of the 12th International Workshop on Knowledge Acquisition, Modeling and Management (KAW'99)*. Banff, 1999

[58] GROTE, A.: *Immer mehr Spam – und vielleicht auch Stress*. In *Telepolis – Magazin der Netzkultur* 2002.
`http://www.heise.de/tp/deutsch/inhalt/te/13834/1.html`

[59] GRUBER, T.: *A Translation Approach to Portable Ontology Specifications*. In *Knowledge Acquisition* (5), 1993: S. 199–220

[60] GRUBER, T.: *Toward Principles for the Design of Ontologies Used for Knowledge Sharing*. In *International Journal on Human-Computer Studies* (43), 1995: S. 907–928

[61] GUARINO, N.: *Formal Ontology and Information Systems*. In *Formal Ontology in Information Systems*, IOS Press. 1998

[62] GUARINO, N.; WELTY, C.: *Identity, unity, and individuality: Towards a formal toolkit for ontological analysis*. In *Proceedings of ECAI-2000*. 2000

[63] GUHA, R.; MCCOOL, R.; MILLER, E.: *Semantic Search*.
`http://tap.stanford.edu/ess.pdf`

[64] HAFER, M.; WEISS, S.: *Word Segmentation by Letter Successor Varieties*. In *Information Storage and Retrieval* (10), 1974: S. 371–385

[65] HAGENGRUBER, R.: *Philosophische Kategorien in der Informatik: Ihre Möglichkeiten für den Entwurf und die Evaluation von Ontologien in der Wirtschaftsinformatik*. In *Proceedings der Tagung Wissenschaftstheorie in Ökonomie und Wirtschaftsinformatik*. 2003, S. 393–404

[66] HAMP, B.; FELDWEG, H.: *GermaNet — a Lexical-Semantic Net for German*. In *Proceedings of ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*. Madrid, 1997

[67] HANEKE, E.: *NewsSIEVE – Ein selbstadaptiver Filter für textuelle Informationen*, 2001.
ftp://ftp3.informatik.uni-bonn.de/pub/paper/tr/IAI-TR-2001-1.pdf.gz

[68] HARS, A.: *Referenzdatenmodelle – Grundlagen effizienter Datenmodellierung*. Gabler, 1994

[69] HIEMSTRA, D.: *Term-Specific Smoothing for the Language Modeling Approach to Information Retrieval: The Importance of a Query Term*. In *Proceedings of the 25st ACM SIGIR Conference on Research and Development in Information Retrieval*. 2002, S. 35–41

[70] HOPCROFT, J.; ULLMAN, J.: *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, Reading, Massachusetts, 1979

[71] HOPCROFT, J.; ULLMAN, J.: *Einführung in die Automatentheorie, formale Sprachen und Komplexitätstheorie*. Addison-Wesley, Bonn, 1994. Deutsche Übersetzung von [70]

[72] HORROCKS, I.: *Using an expressive description logic: Fact or fiction?*. In *Proceedings of the 6th International Conference on Principles of Knowledge Represenation and Reasoning (KR'98)*. Morgan Kaufmann, Trento, 1998, S. 636–649

[73] International Standards Organization: *ISO/IEC 9075*, 1992

[74] Internet Society: *RFC 822 — Standard for the Format of ARPA Internet Text Messages*, 1982.
http://www.faqs.org/rfcs/rfc822.html

[75] JIN, R.; HAUPTMANN, A. G.; ZHAI, C. X.: *Title Language Model for Information Retrieval*. In *Proceedings of the 25st ACM SIGIR Conference on Research and Development in Information Retrieval*. 2002, S. 42–48

[76] JONES, K. S. (Hrsg.): *Information Retrieval Experiment*. Butterworths, 1981

[77] JUNG, Y.; PARK, H.; DU, D.: *An Effective Term-Weighting Scheme for Information Retrieval*. Fachbericht TR008, Department of Computer Science, University of Minnesota, Minneapolis, Minnesota, 2000.
http://citeseer.nj.nec.com/jung00effective.html

[78] KAMP, H.: *A theory of truth and semantic interpretation*. In *Formal Methods in the Study of Language*, Mathematical Centre, Foris, Dordrecht. 1981

[79] KAMP, H.; REYLE, U.: *From Discourse to Logic: Introduction to Modelltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*. Kluwer, Dordrecht, 1993

[80] KAY, J.; MCCREATH, E.: *Automatic Induction of Rules for e-mail Classification*. In *Proceedings of the 9th International Conference on User Modeling*. 2001

[81] KRAFT, D. H.; BUELL, D. A.: *Fuzzy sets and generalized boolean retrieval systems*. In *International Journal on Man-Machine Studies* (19), 1983: S. 45–56

[82] KRIPKE, S. A.: *Semantical considerations on modal logic*. In LINSKY, L. (Hrsg.): *Reference and modality*, Oxford University Press, Oxford, UK. 1971, S. 63–73

[83] KUGELER, M.: *Informationsmodellbasierte Organisationsgestaltung – Modellierungskonventionen und Referenzvorgehensmodell zur prozessorientierten Reorganisation*. Logos-Verlag, Berlin, 2000

[84] KUHN, J.; ROHRER, C.: *Approaching ambiguity in real-life sentences — the application of an Optimality Theory-inspired constraint ranking in a large-scale LFG grammar*. In *6. Fachtagung der Sektion Computerlinguistik DGfS-CL* 1997.
`http://elib.uni-stuttgart.de/opus/volltexte/1999/518/`

[85] KWOK, K. L.: *A neural network for probabilistic information retrieval*. In BELKIN, N.; VAN RIJSBERGEN, C. (Hrsg.): *Proceedings of the 12th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, 1989, S. 21–30

[86] LANGER, H.: *Parsing-Experimente: Praxisorientierte Untersuchungen zur automatischen Analyse des Deutschen*. Peter Lang, Frankfurt (Main), 2001

[87] LAVRENKO, V.; CROFT, W. B.: *Relevance-Based Language Models*. In *Proceedings of the 24st ACM SIGIR Conference on Research and Development in Information Retrieval*. 2001, S. 120–127

[88] LEWIS, D.: *Probability of conditionals and conditonals probabilities*. In HARPER, W.; STALNAKER, R.; PEARCE, G. (Hrsg.): *The University of Western Ontario Series in Philosophy of Science*, D. Riedel Publishing Company, Dordrecht, Holland. 1981, S. 129–147

[89] LI, X.; SZPAKOWICZ, S.; MATWIN, S.: *A WordNet-based Algorithm for Word Sense Disambiguation*. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*. Montreal, 1995, S. 1368–1374.
`http://www.csi.uottawa.ca/tanka/uploadable/IJCAI95.WSD.ps`

[90] LOVINS, J.: *Development of a Stemming Algorithm*. In *Mechanical Translation and Computational Linguistics* 11(1-2), 1968: S. 22–31

[91] LYMANN, P.; VARIAN, H. R.; DUNN, J.; STRYGIN, A.; SWEARINGEN, K.: *How much Information?*. University of California, 2000.
http://www.sims.berkeley.edu/how-much-info

[92] MAEDCHE, A.: *Ontology Learning for the Semantic Web*. Kluwer Academic Publishers, 2002

[93] MAES, P.: *Agents that Reduce Work and Information Overload*. In *Communications of the ACM* 37(7), 1994

[94] MANDL, T.: *Tolerantes Information Retrieval: neuronale Netze zur Erhöhung der Adaptivität und Flexibilität bei der Informationssuche*. UVK-Verlagsgesellschaft, Konstanz, 2001

[95] MANNING, C. D.; SCHÜTZE, H.: *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, 1999

[96] MASAND, B.; LINOFF, G.; WALTZ, D.: *Classifying News Stories using Memory Based Reasoning*. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 1992, S. 59–65

[97] MEIBAUER, J.: *Pragmatik: Eine Einführung*. Stauffenburg, Tübingen, 1999

[98] MILLER, G. A.: *Nouns in WordNet: a Lexical Inheritance System*. In *International Journal of Lexicography* 3(4), 1994: S. 245–264.
ftp://clarity.princeton.edu/pub/wordnet/5papers.ps

[99] MILLER, G. A.; BECKWITH, R.; FELLBAUM, C.; GROSS, D.; MILLER, K.: *Introduction to WordNet: An Online Lexical Database*, 1993.
ftp://clarity.princeton.edu/pub/wordnet/5papers.ps

[100] MIYAMOTO, S.: *Information retrieval based on fuzzy associations*. In *Fuzzy Sets and Systems* (38), 1990: S. 191–205

[101] MIYAMOTO, S.; NAKAYAMA, K.: *Fuzzy information retrieval based on a fuzzy pseudothesaurus*. In *IEEE Transaction Systems Man Cybernet* 16(2), 1986: S. 278–282

[102] MONTAGUE, R.: *English as a Formal Language*. Edizioni di Communità, Milan, 1970, S. 189–224

[103] MONTAGUE, R.: *Universal Grammar*. In *Theoria* (36), 1970: S. 373–398

[104] MONTAGUE, R.: *The Proper Treatment of Quantification in Ordinary English*. In HINTIKKA, J.; MORAVCSIK, J.; SUPPES, P. (Hrsg.): *Approaches to Natural Language*, Reidel, Dordrecht. 1973, S. 247–270

[105] MOTHE, J.: *Search mechanisms using a neural network-Comparison with the vector space model*. In *Proceedings of the 4th RIAO Intelligent Multimedia Information Retrieval Systems and Management*. New York, 1994, Band 1, S. 275–294

[106] MURAI, T.; MIYAKOSHI, M.; SHIMBO, M.: *A fuzzy document retrieval method based on two-valued indexing*. In *Fuzzy Sets and Systems* (30), 1989: S. 103–120

[107] OGAWA, Y.; MORITA, T.; KOBAYASHI, K.: *A fuzzy document retrieval system using the keyword connection matrix and a lerning method*. In *Fuzzy Sets and Systems* (39), 1991: S. 163–179

[108] Oracle Corporation: *Oracle9i Materialized Views*, 2001

[109] PAICE, C.: *Another Stemmer*. In *ACM SIGIR Forum* 24(3), 1990: S. 56–61

[110] PEARL, J.: *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., 1988

[111] POLLARD, C.; SAG, I.: *Head Driven Phrase Structure Grammar*. In *CSLI Stanford and University of Chicago Press* 1994

[112] PONTE, J.; CROFT, W.: *A Language Modeling Approach to Information Retrieval*. In *Proceedings of the 21st ACM SIGIR Conference on Research and Development in Information Retrieval*. 1998, S. 275–281

[113] PONTE, J. M.: *A Language Modeling Approach to Information Retrieval*. Dissertation, University of Massachusetts at Amherst, 1998

[114] PORTER, M.: *An algorithm for suffix stripping*. In *Program* 14(3), 1980: S. 130–137

[115] The PostgreSQL Global Development Group: *PostgreSQL 7.2 Reference Manual*

[116] The PostgreSQL Global Development Group: *PostgreSQL 7.2 User's Guide*

[117] PRINCIPE, J. C.; EULIANO, N. R.; LEFEBVRE, W. C.: *Neuronal and Adaptive Systems: Fundemantals through Simulation*. John Wiley & Sons, 1999

[118] QUASTHOFF, U.: *Deutscher Wortschatz im Internet*. In *LDV-Forum* (2), 1998. `http://wortschatz.uni-leipzig.de/Papers/DeutscherWortschatzimInternet.doc`

[119] QUASTHOFF, U.: *Projekt Deutscher Wortschatz*. In *Linguistik und neue Medien* 1998. `http://wortschatz.uni-leipzig.de/Papers/Projekt_Wortschatz_97.ps.gz`

[120] RADECKI, T.: *Outline of a fuzzy logic approach to information retrieval*. In *International Journal on Man-Machine Studies* (14), 1981: S. 169–178

[121] RADECKI, T.: *Generalized Boolean methods of information retrieval*. In *International Journal on Man-Machine Studies* (18), 1983: S. 407–439

[122] RIBEIRO-NETO, B.; MUNTZ, R.: *A belief network model for IR*. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Zürich, Schweiz, 1996, S. 253–260

[123] ROBERTSON, S.: *The methodology of information retrieval experiment*. In JONES, S. (Hrsg.): *Information Retrieval Experiment*. Butterworths, 1981, S. 9–31

[124] ROBERTSON, S. E.; JONES, K. S.: *Relevance weighting of search terms*. In *Journal of the American Society for Information Sciences* 27(3), 1976: S. 129–146

[125] ROSEMANN, M.: *Vorbereitung der Prozessmodellierung*. In BECKER, J.; KUGELER, M.; ROSEMANN, M. (Hrsg.): *Prozessmanagement*, Springer, Berlin. 2000, S. 45–90

[126] SALTON, G.: *Automatic Information Organization and Retrieval*. McGraw-Hill, New York, 1968

[127] SALTON, G.: *The SMART Retrieval System — Experiments in Automatic Document Processing*. Prentice Hall Inc., Englewood Cliffs (NJ), 1971

[128] SALTON, G.; FOX, E.; WU, H.: *Extended Boolean Information Retrieval*. In *Communications of the ACM* 26(11), 1983: S. 1022–1036

[129] SALTON, G.; MCGILL, M. J.: *Information Retrieval – Grundlegendes für Informationswissenschaftler*. McGraw-Hill, Hamburg, New York, 1987

[130] SCHEER, A.-W.: *Wirtschaftsinformatik: Referenzmodelle für industrielle Geschäftsprozesse*. Springer, 2. Auflage, 1998

[131] SCHLAGETER, G.; STUCKY, W.: *Datenbanksysteme: Konzepte und Modelle*. Teubner, 2. Auflage, 1983

[132] SCHÖNING, U.: *Theoretische Informatik – kurzgefaßt*. Spektrum Akademischer Verlag, Heidelberg, 1999

[133] SCHÜTTE, R.: *Wissen und Information: Antonymie oder Integration zweier Grundbegriffe der Wirtschaftsinformatik*. In SCHEER, A.-W.; ROSEMANN, M.; SCHÜTTE, R. (Hrsg.): *Arbeitsberichte des Institutes für Wirtschaftsinformatik*, 65, Jörg Becker and Hans-Lothar Grob and Stefan Klein and Herbert Kuchen and Ulrich Müller-Funk and Gottfried Vossen, Münster. 1999, S. 144–161.
`http://www.wi.uni-muenster.de/inst/arbber/ab65.pdf`

[134] SEBASTIANI, F.: *Information retrieval, imaging and probablilistic logic*. In *Deliverable D3: A Theory of Uncertainty for Information Retrieval*, ESPRIT. 1996, S. 56–65. Nummer 1/96 im FERMI Technical Report 7, ESPRIT Basic Research Action, Project Number 8134 – FERMI

[135] SHIEBER, S.: *An Introduction to Unification-Based Approaches to Grammar*. In *CSLI Lecture Notes* 1986

[136] SINZ, E. J.: *Objektorientierte Modellierung von Anwendungssystemen – Methodenvergleich*. In *Handout zum Tutorial im Rahmen der Tagung Wirtschaftsinformatik '95*. Frankfurt am Main, 1995

[137] SMITH, B.: *Ontology and Information Systems*, 2002.
http://wings.buffalo.edu/philosophy/faculty/smith/articles/ontologies.htm

[138] SONG, F.; CROFT, W. B.: *A General Language Model for Information Retrieval*. In *Proceedings on the 8th International Conference on Information and Knowledge Management (CIKM'99)*. 1999, S. 316–321

[139] SPECK, M.: *Geschäftsprozessorientierte Datenmodellierung*. Logos Verlag, Berlin, 2001

[140] STALNAKER, R.: *Probability and conditionals*. In HARPER, W.; STALNAKER, R.; PEARCE, G. (Hrsg.): *The University of Western Ontario Series in Philosophy of Science*, D. Riedel Publishing Company, Dordrecht, Holland. 1981, S. 107–128

[141] STEINMÜLLER, W.: *Eine sozialwissenschaftliche Konzeption der Informationswissenschaft*. In *Informationstechnologie und Nachrichtenrecht I* (32), 1981: S. 69–77

[142] SUSSNA, M.: *Word Sense Disambiguation for Free-text Indexing Using a Massive Semantic Network*. In *Proceedings of the Second International Conference on Information and Knowledge Management*. Arlington, Virginia, 1993

[143] TopicMaps.org: *XML Topic Maps (XTM) 1.0*, 2001.
http://www.topicmaps.org/xtm/1.0

[144] TURING, A. M.: *On computable numbers, with an Application to the Entscheidungsproblem*. In *Proceedings of London Mathematical Society* 2(42), 1936: S. 230–236

[145] TURTLE, H.; CROFT, W. B.: *Inference networks for document retrieval*. In *Proceedings of the 13th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Brüssel, Belgien, 1990, S. 1–24

[146] TURTLE, H.; CROFT, W. B.: *Evaluation of an inference network-based retrieval model*. In *ACM Transactions onf Information Systems* 9(3), 1991: S. 187–222

[147] UMSTÄTTER, W.: *Digitales Lehr- und Handbuch der Bibliothekswissenschaft*.
http://www.ib.hu-berlin.de/~wumsta/infopub/bookindex.html

[148] USCHOLD, M.; KING, M.: *Towards a Methodology for Building Ontologies*. In *Proceedings of the IJCAI'95 Workshop on Basis Ontological Issues in Knowledge Sharing*. 1995

[149] USHOLD, M.; GRUNINGER, M.: *Ontologies: Principles, methods and applications*. In *Knowledge Sharing and Review* 11(2), 1996: S. 93–155

[150] USZKOREIT, H.; BACKOFEN, R.; CALDER, J.; CAPSTICK, J.; DINI, L.; DÖRRE, J.; ERBACH, G.; ESTIVAL, D.; MANANDHAR, S.; MINEUR, A.-M.; OEPEN, S.: *The EAGLES Formalisms Working Group - Final Report Expert Advisory Group on Language Engineering Standards*. Fachbericht LRE 61-100, 1996.
http://www.dfki.de/dfkibib/publications/docs/eagles-fwg-report.ps.gz

[151] VAN RIJSBERGEN, C. J.: *A theoretical basis for the use of co-occurence data in Information Retrieval*. In *Journal of Documentation* 33(2), 1977: S. 106–119

[152] VAN RIJSBERGEN, C. J.: *Retrieval effectivness*. In JONES, K. S. (Hrsg.): *Information Retrieval Experiment*, Butterworths. 1981

[153] VAPNIK, V. N.: *The Nature of Statistical Learning Theory — Statistics for Engineering and Information Science*. Springer Verlag, 2. Auflage, 1999

[154] VORHEES, E. M.: *Using WordNet for Text Retrieval*. In FELLBAUM, C. (Hrsg.): *WordNet: an electronic lexical database*, MIT Press, Cambridge, London. 1998, S. 285–303

[155] VOSSEN, G.: *Datenmodelle, Datenbanksprachen und Datenbank-Management-Systeme*. R. Oldenbourg Verlag, München, Wien, 3. Auflage, 1999

[156] WAHLSTER, W.: *Verbmobil: Translation of face-to-face dialogues*. In *Proceedings of the 3rd European Conference on Speech Communication and Technology*. Berlin, 1993

[157] WAUSCHKUHN, O.: *Ein Werkzeug zur partiellen syntaktischen Analyse deutscher Textkorpora*. In GIBBON, D. (Hrsg.): *Natural Language Processing and Speech Technology — Results of the 3rd KONVENS Conference (Bielefeld)*. Mouton de Gryter, Berlin, 1996, S. 356–368

[158] WEBRE, N. W.: *An Extended Entity-Relationship Model and Its Use on a Defense Project*. In CHEN, P. P. (Hrsg.): *Entity-Relationship Approach to Information Modelling and Analysis, Proceedings of the 2nd International Conference on Entity-Relationship Approach*. Amsterdam, New York, Oxford, 1983, S. 173–193

[159] WILKINSON, R.; HINGSTON, P.: *Using the cosine measure in neuronal network for document retrieval*. In *Proceedings of the 14th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Chicaco, USA, 1991, S. 202–210

[160] WONG, S. K. M.; ZIARKO, W.; RAGHAVAN, V. V.; WONG, R. C. N.: *On Modeling of Information Retrieval Concepts in Vector Spaces*. In *ACM Transactions on Database Systems* 12(2), 1987: S. 299–321

[161] YANG, Y.; CHUTE, C.: *An exmaple-based mapping method for text categorization and retrieval*. In *ACM Transaction on Information Systems* 12(3), 1994: S. 252–277

[162] YANG, Y.; LIU, X.: *A re-examination of text categorization methodes*. In HEARST, F. G.; TONG, R. (Hrsg.): *Proceedings of the 22nd Annual ACM SIGIR Conference on Research and Development in Information Retrieval*. Berkley, 1999, S. 42–49

[163] ZADEH, L. A.: *Fuzzy sets*. In *Information and Control* (8), 1965: S. 338–353

[164] ZADEH, L. A.: *Fuzzy sets*. In DUBOIS, D.; PRADE, H.; YAGER, R. R. (Hrsg.): *Readings in Fuzzy Sets for Intelligent Systems*, Mogran Kaufmann. 1993

[165] ZELEWSKI, S.: *Organisierte Erfahrung – Wissensmanagement mit Ontologien.* In *Essener Unikate* (11), 2002: S. 63–73

[166] ZELL, A.: *Simulation neuronaler Netze*. Oldenbourg, Bonn, 1994

[167] ZELL, A.; MAMIER, G.; MACHE, M. V. N.; HÜBNER, R.; HERRMANN, S. D. K.-U.; SOYEZ, T.; SOMMER, M. S. T.; HATZIGEORGIOU, A.; SCHREINER, D. P. T.; KETT, B.; WIELAND, G. C. J.; RECZKO, M.; SEEMANN, M. R. M.; RITT, M.; BIEDERMANN, J. D. J.; DANZ, J.; WERNER, C. W. R.; BERTHOLD, M.; ORSIER, B.: *SNNS: Stuttgart Neuronal Network Simulator. User Manual, Version 4.1*. Universität Stuttgart, 1995. IPVR. Report Nr. 6/95, `http://www-ra.informatik.uni-tuebingen.de/SNNS/UserManual/UserManual.html`

[168] ZHAI, C. X.; LAFFERTY, J.: *Two-Stage Language Models for Information Retrieval*. In *Proceedings of the 25st ACM SIGIR Conference on Research and Development in Information Retrieval*. 2002, S. 49–56

[169] ZIMMERMANN, H.-J.: *Fuzzy set theory — and its applications*. Kluwer Academic Publishers, Norwell, Massachusetts, USA, 4. Auflage, 2001

[170] ZÚÑIGA, G.: *Ontology: Its Transformation From Philosophy to Information Systems*. In *Proceedings of Formal Ontology in Information Systems*. 2001, S. 187–197

Index

Ähnlichkeit
  von Dokumenten, **48**

Abdeckung
  beim Parsing, 28
ACID-Prinzip, **19**
Ambiguität, 27
Antonymie, **33**, 92, 173, 175
Architektur integrierter Informationssysteme, **14–16**
ARIS, *siehe* Architektur integrierter Informationssysteme
Attributtyp, **17**
Automatic Global/Local Analysis, 13

Backpropagation Neuronal Network, **79–81**, 107
Bayes
  Theorem, 55, 57, 59
Bayesian Network, 57
Bedienbarkeit, 188
Belief Network Model, **59–60**
Benutzerprofil, **183–186**
Bewertung
  gängiger IF/IR-Modelle, **81–85**
Big-Bang-Vorgehen, 171
Binary Independence Retrieval, **54–56**
BIR, *siehe* Binary Independence Retrieval
BNM, *siehe* Belief Network Model
BNN, *siehe* Backpropagation Neuronal Network
Boolean Model
  Extended ∼, *siehe* Extended Boolean Model
  Standard ∼, *siehe* Standard Boolean Model

Co-Occurrenz, **63**, 84
Computerlinguistik, 20
COSIMIR, **80**
Cranfield-Kollektion, 81

Daten, **1**
  ∼modell, 94–98, 148–151
  ∼sicht, **14**
  semistrukturierte ∼, **8**
  unstrukturierte ∼, **8**
Datenbank
  ∼system, 8
  relationale ∼, **19–20**, 94, 148–167
Derivation, **21**, 88, 108
Dice-Maß, 64
Disambiguierung, **32**, 128, 133
Dokument, **8**, 90, **110**
  ∼enähnlichkeit, 91, 113, **137–139**, 166–167
DV-Konzept, **15**

EBM, *siehe* Extended Boolean Model
Effektivitätsmaße, **188**
Effizienz
  beim Parsing, 28
Effizienzmaße, 188
enhanced Topic-based Vector Space Model, **109–170**
  Anwendung im IF, 183–187
  Anwendung im IR, 182–183
  Ontologien für das ∼, 171–182
Entity-Relationship-Model, **16–18**

Entitytyp, **17**
ERM, *siehe* Entity-Relationship-Model
eTVSM, *siehe* enhanced Topic-based Vector Space Model
Evaluationsmaße, 188–190
Evaluierung
  qualitativ, 81–85
  quantitativ, 187–192
Expected-Mutual-Information-Measure, 78
Extended Boolean Model, **52–54**

Fachkonzept, **15**
Fehlerrate, **189**
Flexion, **21**, 66, 83, 88, 107
FSM, *siehe* Fuzzy Set Model
Funktionssicht, **15**
Fuzzy Set Model, **73–76**, 105

Generalized Vector Space Model, **68–69**, 105
GermaNet, **175–179**
Grammatik, 23–26
  kontextfrei, **24**
Graph, 116
GVSM, *siehe* Generalized Vector Space Model

Homographen, 181
Homographie, **32**, 108, 112, 174, 183
  Repräs. der ∼, **132–134**
Homonymie, **32**
Homophonie, **32**
Hyponymie, **33**, 67, 84, 88, 108, 173, 175, 176

Immanenz, 63
Implementierung, **15**
Inference Network Model, **56–58**
Information, **1**
  ∼-Filtering, 2, 7, **10–11**, 183
  ∼-Retrieval, 2, 7, **8–10**, 182
  ∼süberflutung, **2**
  ∼sbedarf, 183
INM, *siehe* Inference Network Model
Interpretation, **111**, **128–139**, 175
  ∼sähnlichkeiten, **129–130**, 158–159
  von Worten, 30

Jaccard-Maß, 63

k-nearest neighbour, 48, 186
Kardinalität, **18**
Keyword-Connection-Matrix, **74**, 105
Klassifikation, 38
Komposition, **21**, 66, 88, 108
Kompositum, **21**
Konsistenzkriterien, 106, 116
Kosinus-Maß, 64
Kosten, 188

Language Model, **60–63**
Latent Semantic Index, **69–70**
Lautlehre, *siehe* Phonologie
Lemma, *siehe* Wort
Lemmatisierung, **22**
Lexem, *siehe* Wort
Linguistik, *siehe* Computerlinguistik
LM, *siehe* Language Model
Logical-Imaging, 76
LSI, *siehe* Latent Semantic Index

Machine Learning for User Modeling, 13
Makrobewertung, **189**
Maximalität, 116
Meronymie, **33**, 67, 84, 108, 173, 175, 176
Metonymie, **32**, 88
  Repräs. der ∼, **136**
Mikrobewertung, **190**
Minterm, **68**, 105
ML4UM, 13
Modell, **14**
  ∼vergleich, 81–85, 104–108, 168–170
  algebraisches ∼, **43**
  der Interaktion, 13
  der Repräsentation, 12, 43–85
  des Information-Filtering, **10**
  des Information-Retrieval, **8**
  mengentheoretisches ∼, **43**

mit Terminterdependenzen
immanent, **63–72**, 84
transzendent, **72–81**, 84
ohne Terminterdependenzen, **49–63**, 83
probabilistisches ∼, **44**
Morphologie, **21–23**, 83

Navigational Searches, **182**
Network Model
Backpropagation Neuronal ∼, *siehe* Backpropagation Neuronal Network
Belief ∼, *siehe* Belief Network Model
Inference ∼, *siehe* Inference Network Model
Spreading Activation Neuronal ∼, *siehe* Spreading Activation Neuronal Network
Normalisierung, *siehe* Stemming
Normierung, 116
nutzerorientierter Ansatz, *siehe* Makrobewertung

Ontologie, **35–42**
∼n für das eTVSM, 171–182
hier verw. Def., **37**
Beispiel, 144–146
Erstellung einer ∼, 171–172
eTVSM und ∼, **143–146**
in der Informatik, 36
in der Philosophie, 36
Modellierungssprache, 38, 143
nach GRUBER, **36**
nach ZELEWSKI, **36**
Organisationssicht, **15**

Paradigma, **21**
Parsing, **26**, 45–48, 98, **159–166**
Performanz, 188
Phonologie, 20
Polysemie, **31**
Possible-World-Semantics, 76
Pragmatik, 33–35
Precision, **189**
Probabilistic Model, *siehe* BIR

Query Expansion, 13, 52, 84

RbLI, *siehe* Retrieval by Logical Imaging
Recall, **189**
Relationshiptyp, **17**
Research Searches, **182**
Retrieval by Logical Imaging, **76–78**, 107

SANN, *siehe* Spreading Activation Neuronal Network
SBM, *siehe* Standard Boolean Model
Semantik, **29**
Diskurs∼, 29
lexikalische ∼, **30–33**, 83
Satz∼, 29
Semasiologie, *siehe* Semantik
Singular Value Decomposition, 69
Smoothing, 62
Spreading Activation Neuronal Network, **70–72**
SQL, *siehe* Standard Query Language
Standard Boolean Model, **49–50**
Standard Query Language, 20, 96–97, 152–167
Statistical Language Model, 60
Stemming, **21–23**, 83, 162, 173
∼-Lemma, **101**
Durchführen von ∼, 46
Over-/Under-∼, **23**
Repräs. des ∼, **140**
Strong-∼, **22**
Weak-∼, **22**
Steuerungssicht, **15**
Stoppwort, 30
∼-Lemma, **99**
Anwendung von ∼listen, 46
Repräs. von ∼listen, **139**
Suche
Arten von ∼n, **182**
Suchstrategien, 183

Sukzessives-Vorgehen, 171
Supportterm, **132**
  Definition von ∼, **134**
SVD, *siehe* Singular Value Decomposition
Symmetrie, 106, 116
Synonym
  ∼-Lemma, **103**
  Anwendung von ∼ersetzung, 47
Synonymie, **30**, 66, 84, 88, 107, 111, 174, 183
  Partielle ∼, **31**
  Repräs. partieller ∼, **134**
  Repräs. totaler ∼, **131**
  Totale ∼, **31**
Synset, **176**
Syntax, **23**
Systematik, 38
systemorientierter Ansatz, *siehe* Mikrobewertung

Taxonomie, 38
Term, 45, 90, **111**
  ∼ähnlichkeit, 64, 88, 90
  ∼gewichte, **83**
  ∼interdependenzen, 45
  ∼vektor, 90
  Orthogonalität von ∼en, **49**
  Unabhängigkeit von ∼en, **49**
Testdaten, 188, 190
tf-idf, 51
Thema, **112**
Themen
  ∼-Ähnlichkeiten, **115–128**, 157
  ∼blatt, **121**, 151–155
  ∼knoten, **121**, 155–157
  ∼komplex, 117
  ∼strukturen, 116
  ∼vektor, **121–122**
  elementares ∼gebiet, 90
Thesaurus, 39
  semiotischer ∼, 39
Topic-based Vector Space Model, **87–108**
Transitivität
  schwache ∼, 106, 116
Transzendenz, 72
TVSM, *siehe* Topic-based Vector Space Model

User Relevance Feedback, 13

Vector Space Model, **50–52**, 104, 180
  enhanced Topic-based ∼, *siehe* enhanced Topic-based Vector Space Model
  Generalized ∼, *siehe* Generalized Vector Space Model
  Topic-based ∼, *siehe* Topic-based Vector Space Model
Vektorraum, 89
VSM, *siehe* Vector Space Model

Wissensbasis, 38
word form, 175
word meaning, 175
WordNet, **174–179**
Wort, **21**, **110**, 175
  ∼bildung, **21**
  ∼form, **21**
  ∼gewichte, **83**
  ∼gruppe, 67, 83, **83**, 108, 111
  ∼interpretation, 30
    nicht-wörtlich, 32
    Variabilität, 31
  ∼netz, 39
  ∼stamm, **111**
Wortschatz-Lexikon, **172–174**

Zykel, 117, 122