Lizentiatsarbeit der Philosophischen Fakultät
der Universität Zürich
im Bereich Computerlinguistik

Eigennamenerkennung über Konzeptsensoren

Geschrieben von Carola Kühnlein
Betreuung: PD Dr. Martin Volk

Dezember 2003

Inhaltsverzeichnis

VORWORT III

DANK IV

KAPITEL I EINLEITUNG I-1

1. INFORMATION RETRIEVAL I-1
1.1. DEFINITION I-1
1.2. GESCHICHTE DES IR I-1
1.3. INFORMATION I-5
1.4. ZIELE UND METHODEN DES IR I-5
1.5. BEWERTUNG VON IR-SYSTEMEN I-8
2. EIGENNAMENERKENNUNG I-10
2.1. STATISTISCHE METHODEN I-17
2.2. REGELN UND LEXIKON I-21
2.3. BEWERTUNG VON EIGENNAMENERKENNUNG I-22
3. PROJEKTBESCHREIBUNG I-23
3.1. ZIEL I-23
3.2. GRUNDLAGEN I-24
3.3. DIE FIRMA *EUROSPIDER* I-24
3.4. EIGENNAMENERKENNUNG FÜR *EUROSPIDER* I-30
3.5. ZEITRAHMEN I-32
4. RESSOURCEN I-33
4.1. DEFINITION I-33
4.2. VERWENDUNG VON RESSOURCEN I-34
4.3. QUELLEN FÜR RESSOURCEN I-36
4.4. GRÖSSEN DER RESSOURCEN I-39

KAPITEL II THEORETISCHE GRUNDLAGEN II-42

1. EIGENNAMEN II-42
1.1. DEFINITION VON EIGENNAMEN II-42
1.2. GESCHICHTE DER EIGENNAMEN II-43
1.3. KATEGORIEN VON EIGENNAMEN II-44
1.4. EIGENSCHAFTEN/MERKMALE UND BEDEUTUNG VON EIGENNAMEN II-48
2. RELEVANZ VON EIGENNAMENERKENNUNG II-53
3. VERWENDETE KATEGORIEN II-55
3.1. EIGENSCHAFTEN DER VERWENDETEN KATEGORIEN II-56
4. ALGORITHMEN II-60
4.1. ÜBERNOMMENE RESSOURCEN II-61
4.2. ÜBERNOMMENE REGELN II-62
4.2.1. Regeln zur Erkennung von Personennamen II-62
4.2.2. Regeln zur Erkennung von geographischen Eigennamen II-63
4.2.3. Regeln zur Erkennung von Firmennamen II-63
4.3. UNTERSCHIEDE DER IMPLEMENTIERUNG ZU DEN BESTEHENDEN ALGORITHMEN II-64

KAPITEL III IMPLEMENTATION III-67

1. EUROSPIDER RETRIEVAL SYSTEM ‚RELEVANCY' III-67
2. KONZEPTSENSORSPRACHE III-75

2.1. DIE AUFGABE DER KONZEPTSENSORSPRACHE III-75
2.2. DIE SYNTAX DER KONZEPTSENSORSPRACHE III-76
2.3. ERWEITERUNGEN IN DER KONZEPTSENSORSPRACHE ZUR ERKENNUNG VON EIGENNAMEN III-82
2.4. PROBLEME INNERHALB DER KONZEPTSENSORSPRACHE UND IHRE GRENZEN III-85
3. RESSOURCEN FÜR DIE ERKENNUNG VON ENTITÄTEN IN DEUTSCHSPRACHIGEN TEXTEN III-89
3.1. RESSOURCEN ZUR ERKENNUNG VON PERSONENNAMEN (DE) III-89
3.2. RESSOURCEN ZUR ERKENNUNG VON GEOGRAPHISCHEN NAMEN (DE) III-91
3.3. RESSOURCEN ZUR ERKENNUNG VON FIRMENNAMEN (DE) III-96
4. IMPLEMENTIERUNG DER EIGENNAMENERKENNUNG IN DEUTSCHSPRACHIGEN TEXTEN III-99
4.1. ERKENNUNG VON PERSONENNAMEN (DE) III-99
4.2. ERKENNUNG VON GEOGRAPHISCHEN NAMEN (DE) III-103
4.3. ERKENNUNG VON FIRMENNAMEN (DE) III-106
5. RESSOURCEN FÜR DIE ERKENNUNG VON ENTITÄTEN IN ENGLISCHEN TEXTEN III-114
5.1. RESSOURCEN ZUR ERKENNUNG VON PERSONENNAMEN (EN) III-114
5.2. RESSOURCEN ZUR ERKENNUNG VON GEOGRAPHISCHEN NAMEN (EN) III-116
5.3. RESSOURCEN ZUR ERKENNUNG VON FIRMENNAMEN (EN) III-119
6. IMPLEMENTIERUNG DER EIGENNAMENERKENNUNG IN ENGLISCHEN TEXTEN III-121
6.1. ERKENNUNG VON PERSONENNAMEN (EN) III-122
6.2. ERKENNUNG VON GEOGRAPHISCHEN NAMEN (EN) III-126
6.3. ERKENNUNG VON FIRMENNAMEN (EN) III-130

KAPITEL IV EVALUATION IV-136

1. EVALUATIONSSKRIPT IV-136
2. TESTKORPUS FÜR DIE EVALUATION IN DEUTSCHSPRACHIGEN TEXTEN IV-138
3. EVALUATION DER EIGENNAMENERKENNUNG IN DEUTSCHSPRACHIGEN TEXTEN IV-141
3.1. EVALUATION VON PERSONENNAMEN (DE) IV-142
3.2. EVALUATION VON GEOGRAPHISCHEN EIGENNAMEN (DE) IV-150
3.3. EVALUATION VON FIRMENNAMEN (DE) IV-155
4. TESTKORPUS FÜR DIE EVALUATION IN ENGLISCHEN TEXTEN IV-161
5. EVALUATION DER EIGENNAMENERKENNUNG IN ENGLISCHEN TEXTEN IV-163
5.1. EVALUATION VON PERSONENNAMEN (EN) IV-164
5.2. EVALUATION VON GEOGRAPHISCHEN NAMEN (EN) IV-168
5.3. EVALUATION VON FIRMENNAMEN (EN) IV-171

KAPITEL V AUSBLICK V-175

1. RESULTATE IM VERGLEICH ZU ANDEREN SYSTEMEN V-175
2. VERBESSERUNGSMÖGLICHKEITEN V-177
3. WEITERVERWENDUNG V-178
4. ZUSAMMENFASSUNG V-179
ANHANG 182
PSEUDOCODE 182
MODUL ZUR PERSONENAMENERKENNUNG IN ENGLISCHEN TEXTEN 189
BIBLIOGRAPHIE 199

Vorwort

Diese Lizentiatsarbeit im Bereich Eigennamenerkennung ist in Zusammenarbeit mit der Firma *Eurospider* entstanden. Die Arbeit soll einen Einblick geben in die Thematik der Eigennamenerkennung, insbesondere in Verbindung mit Dokumentenretrieval, und die Entwicklung eines eigenen Systems zur Eigennamenerkennung dokumentieren. Meine Aufgabe innerhalb der Firma *Eurospider* bestand darin, mit der Konzeptsensorsprache, einer Mustererkennungssprache, die von der Firma selbst entwickelt worden ist, Eigennamenerkennung in deutschen und englischen Texten zu implementieren. Die Eigennamen sind auf die Kategorien *Personenname*, *geographischer Eigenname* und *Firmenname* beschränkt. Als Vorlage diente mir ein von Volk in *Perl* implementiertes Programm zur Eigennamenerkennung in deutschen Texten. Ein zukünftiges Ziel der Firma *Eurospider* ist, diese Eigennamenerkennung in das bestehende Information Retrieval System *Relevancy* zu integrieren. Eigennamenerkennung ist bisher beim Retrievalprozess nicht berücksichtigt worden.

Die folgende Arbeit ist in fünf Teile gegliedert. Kapitel I führt ins Information Retrieval und insbesondere in die Thematik der Eigennamenerkennung ein. Methoden sowie bereits bestehende und getestete Ansätze und Systeme werden vorgestellt. Zusätzlich erläutere ich in diesem Kapitel I die Rahmenbedingungen sowie die Zielsetzung dieses Projekts.

Kapitel II wird die Bedeutung und Relevanz von Eigennamen sowie deren Erkennung in Texten behandelt. Zusätzlich sind die Grundlagen, welche ich in diesem Projekt verwendet habe, genauer ausgeführt. Die Grundlagen, auf denen ich aufbauen konnte, bestanden einerseits aus den Algorithmen, welche Volk für die Eigennamenerkennung in deutschen Texten entwickelt hat, andererseits aus bestehenden Listen mit Wörtern, die ich im Algorithmus zur Erkennung der Eigennamen mit verwendet habe.

In Kapitel III stelle ich die Konzeptsensorsprache vor und diskutiere deren Vor- und Nachteile. Danach folgt eine genaue Beschreibung der Implementierung sowohl für deutsche als auch englische Texte.

Im Kapitel IV unterziehe ich die durch meine Implementierung erreichten Werte einer präzisen Analyse und erläutere die Vor- und Nachteile, die meine Erkennungsmethoden mit sich bringen.

In Kapitel V folgt ein kurzer Ausblick, wie die bestehenden Muster zur Eigennamenerkennung verbessert werden könnten, und wo die Grenzen für eine Eigennamenerkennung über die Konzeptsensorsprache liegen. Die erreichten Resultate werden mit Evaluationswerten von anderen Systemen zur Eigennamenerkennung verglichen. Die Zusammenfassung enthält eine

Kurzbeschreibung des Projekts, des Umfelds, in dem es entstanden ist, und der Problematik bei der Implementierung sowie der Evaluationswerte.

Dank

Ich möchte mich bei folgenden Personen, die mich während der Lizentiatsarbeit tatkräftig unterstützt haben, ganz herzlich bedanken:

Martin Volk, der mich in der Firma *Eurospider* und während der anschliessenden Schreibphase hervorragend betreut hat,

Peter Schäuble, der mir ermöglicht hat, meine Lizentiatsarbeit innerhalb seiner Firma zu schreiben und so viele wertvolle Erfahrungen zu sammeln,

Martin Braschler, der mich innerhalb der Firma betreut und unterstützt hat, damit ich das Projekt erfolgreich abschliessen konnte,

Patrick Frey, bei dem ich bei jedem technischen Problem umgehend Unterstützung erhielt, so dass es mir möglich war, die Tücken des Indexer und der Konzeptsensoren in den Griff zu kriegen.

Mein Dank gilt ebenfalls denjenigen Personen, die mit viel Geduld und Fleiss all die kleinen Fehler entdeckten, die sich beim Schreiben eingeschlichen haben.

Kapitel I Einleitung

1. Information Retrieval

1.1. Definition

Information Retrieval (IR) bedeutet wörtlich übersetzt lediglich „Zurückgewinnen von Information“ ([Gaus, 2000], S. 1). Eine Definition in [Poetzsch, 2001] ist aus Sicht der Informatik weit informativer und genauer:

„Information Retrieval umfasst die Repräsentation, Speicherung, Organisation von und den Zugriff auf Informationen. Ziel des Information Retrieval ist es, all die Information wiederzufinden, die für die Suchfrage des Nutzers relevant ist.“

IR ist folglich eine Wissenschaft, die sowohl Methoden aus der Informatik benötigt, um grosse Datenmengen zu speichern und zu repräsentieren, als auch Methoden aus der Sprachverarbeitung verwendet, um die Daten zu ordnen und relevante Informationen zu erkennen und zu extrahieren.

Repräsentation und Organisation von Daten müssen gut durchdacht sein, um tatsächlich relevante Informationen zu einer Suchfrage zu erhalten. Linguistische Lösungsansätze haben eine wichtige Position in den Bereichen Organisation, Zugriff und Erkennung bzw. Extraktion von relevanten Informationen eingenommen.

Eigennamenerkennung (vgl. Kapitel I 2. „Eigennamenerkennung“) ist sowohl im Bereich der Informationsgewinnung als auch im Bereich der Organisation von Dokumentenkollektionen eine wichtige Komponente. Einerseits können Dokumente, welche Eigennamen enthalten, in spezifische Klassen eingeteilt werden, um bei entsprechenden Anfragen einen schnelleren Zugriff zu erlauben, andererseits ist es notwendig, Eigennamen in Dokumenten zu erkennen, um überhaupt einen Zugriff auf solche Nomen zu ermöglichen.

1.2. Geschichte des IR

Bereits vor beinahe 4000 Jahren haben Menschen Informationen strukturiert, um sie zu einem späteren Zeitpunkt wiederzufinden und weiterzuverwenden. Dies kann als früheste Form des IR bezeichnet werden. Durch das Verfassen von immer dickeren und somit informationsreicheren Büchern nahm die Bedeutung des schnellen Zugriffs auf gesuchte Information zu. Es wurden wichtige Wörter und Konzepte mit entsprechenden Verweisen zu den Textstellen, in denen sie stehen, gesammelt und in einer alphabethischen Liste aufgeführt (gleich einem Index). Jahrhunderte lang sind Indizes von Hand erstellt worden. Durch die rasante Entwicklung in der Informatik ist es unterdessen möglich, grosse Indizes auch automatisch zu generieren. Dies ist eine Notwendigkeit in Anbetracht der ständig wachsenden Informationsflut. Die ur-

sprüngliche Aufgabe des IR, das Erstellen eines Indexes, ist entsprechend erweitert worden. Dokumentenklassifikation, Kategorisierung, Filtern von Daten und Sprachidentifikation[1] sind inhaltliche Themen, die im modernen IR zusätzlich Beachtung gefunden haben. Benutzerschnittstellen, Visualisierung von Daten sowie Systemarchitektur sind Themen, die bei der Implementierung von IR-Programmen von grosser Bedeutung sind und somit ebenfalls einen Platz im modernen IR einnehmen.

Lange Zeit sind IR-Techniken und -Systeme vor allem in Bibliotheken gebraucht und genutzt worden. Dies änderte sich mit der Einführung des WWW (***W**orld **W**ide **W**eb*) Anfang der 90er-Jahre. Die Möglichkeit, uneingeschränkt Informationen mit wenig Aufwand ins Web zu stellen und somit der Öffentlichkeit zugänglich zu machen, liess das WWW beständig wachsen (und lässt es auch jetzt noch wachsen). Das Auffinden von nützlicher Information ist schwierig geworden. Dem Benutzer des WWW's ist wichtig, schnell Information zu einer gegebenen Anfrage zu finden. Dadurch wird ein neues Interesse an IR geweckt. Neu entwickelte Lösungen und Techniken aus dem IR sollen helfen, sich in dieser Informationsmasse effizient und gezielt zu bewegen. IR nimmt die führende Position im Bereich Informationsgewinnung ein, was die Forschung entsprechend schnell vorangetrieben hat (und weiterhin vorantreibt).[2]

Jährlich findet die Konferenz TREC (***T**ext **Re**trieval **C**onference*) statt, gesponsert von NIST (***N**ational **I**nstitute of **S**tandards and **T**echnology*) und DARPA (***D**efense **A**dvanced **R**esearch **P**rojects **A**gency*). TREC ist die wichtigste Konferenz im IR-Bereich. Vom Organisationskomitee wird allen Teilnehmern eine genügend grosse Testkollektion von Texten und Anfragen zur Verfügung gestellt, anhand welcher verschiedene IR-Systeme unter äquivalenten Bedingungen miteinander verglichen werden können. Systeme, welche sich an einer solchen Konferenz bewährt haben, entsprechen dem aktuellen Forschungsstand und werden als gute Systeme allgemein anerkannt. Die erste Konferenz fand im November 1992 statt als Teil des TIPSTER-Text-Programm. Folgende Ziele sollten mit dieser und den folgenden Konferenzen erreicht werden (vgl. [TREC, http://trec.nist.gov], *overview*):

- IR mit speziell grossen Textkollektionen soll gefördert werden.
- Über die Konferenz soll der Austausch von neu entwickelten Ideen, welche in der Industrie, im akademischen Bereich oder in einem politischen Umfeld entstanden sind, gefördert werden.

[1] Mit Sprachidentifikation ist der Prozess gemeint, in dem die Sprache eines Dokuments oder einer Anfrage erkannt wird.
[2] Vgl. [Baeza-Yates und Ribeiro-Neto, 1999], S. 2f. und S. 6ff.

- Die Umsetzung von im Forschungsbereich entstandenen Technologien in kommerzielle Produkte soll beschleunigt werden, indem grundsätzliche Fortschritte in Retrieval Methoden auf tatsächlich bestehende Probleme angewendet werden.
- Neu entwickelte Techniken sollen leichter und schneller für Universitäten und Firmen erhältlich sein, und Einbindungsmöglichkeiten der neu entwickelten Technik in bestehende Systeme sollen gefördert werden.

Die nächste Konferenz wird im November 2003 stattfinden. Man behandelt folgende Themen (vgl. [TREC, http://trec.nist.gov]: *tracks)*:

1. ‚Cross Language Track'[3]

   In ‚Cross Language Retrieval' wird versucht, IR übergreifend in verschiedenen Sprachen zu betreiben. Anfragen werden in einer vom System unterstützten Sprache gestellt, während gesuchte Informationen in Dokumenten einer anderen Sprache[4] stehen.
2. ‚Filtering Track'

   Binäre Entscheidungen sollen entwickelt und optimiert werden, auf Grund deren man neue Dokumente als relevant zu einer Anfrage oder als nicht relevant einstufen kann.
3. ‚Genome Track'

   Man will eine Optimierung des Retrieval aus Texten mit Inhalt Genomanalyse erreichen. Informationen über Gene und Genketten, die sehr zahlreich vorkommen, sollen aus diversen, spezifischen Textarten wie Forschungsberichten oder Laborrapporte gewonnen werden.
4. ‚HARD (***H**igh **A**ccuracy **R**etrieval from **D**ocuments*) Track'

   In diesem Teilbereich soll eine möglichst hohe Präzision im IR erreicht werden, indem einerseits zusätzliche Informationen vom Suchenden, andererseits Informationen über den genaueren Kontext der Anfrage mit einbezogen werden. Diese Informationen fliessen bei der Gewichtung der Dokumente als Metadaten mit ein. Der Benutzer spezifiziert bei der Anfrage, ob die gesuchten Informationen innerhalb einer Phrase oder einer Passage stehen sollen. Eine enge Zusammenarbeit mit dem Suchenden, um einen Teil der Metadaten zu gewinnen, sowie Methoden aus dem Passagen Retrieval, um zur Anfrage relevantes Textmaterial automatisch zu extrahieren, sind notwendig. Im Passagen Retrieval werden nicht die Dokumente als ganzes als relevant zu einer Anfrage gewichtet sondern einzelne Passagen oder Phrasen. Im ‚HARD-Tack' sollen in den gefundenen Dokumenten, die relevanten Passagen zusätzlich markiert werden, wenn

[3] Neu wird dies in einer separaten Konferenz CLEF (***C**ross **L**anguage **E**valuation **F**orum*) behandelt. In früheren TREC-Konferenzen wurde dieses Thema mit behandelt.

[4] Diese Sprachen müssen ebenfalls im System implementiert sein.

dies vom Fragenden gewünscht wird. Die Metadaten haben somit die Aufgabe, einerseits die Rangliste der relevanten Dokumente, andererseits die Gewichtung der einzelnen Passagen zu beeinflussen.

5. ‚Interactive Track'
   In diesem Bereich wird die Interaktion von IR-System und Nutzer genauer untersucht. Studien mit Testpersonen, welche eine gemeinsame Testkollektion und eine gewisse Anzahl von Anfragen als Grundlage erhalten, sollen durchgeführt werden.
6. ‚Novelty Track'
   Hier geht es darum, IR-Systeme auf ihre Fähigkeit zu testen, neu abgeleitete, möglichst nicht redundante Information in Dokumenten zu finden. Dieser Track sollte eine Verbesserung der Resultate bei nicht auf Fakten basierenden, indirekten Anfragen[5] herbeiführen.
7. ‚Question Answering Track'
   Auf Anfragen sollen relevante Textpassagen aus Dokumenten gefunden werden. Mit IR ist hier eher Informationsextraktion (IE) oder Passagen Retrieval gemeint. Bei IE werden einzelne Informationen extrahiert, während im IR ganze Dokumente gefunden werden (vgl. Kapitel I 1.4. „Ziele und Methoden des IR")
8. ‚Robust Retrieval Track'
   Ad hoc Retrieval (vgl. Kapitel I 1.4. „Ziele und Methoden des IR") soll auf Dokumente diverser Themen wirksam angewendet werden.
9. ‚Video Track'
   IR wird auf digitale Videos angewendet.
10. ‚Web Track'
    Es sollen möglichst präzise und relevante Schlüsselwörter zu verschiedenen Themen aus einer Dokumentensammlung extrahiert werden. Bei der Dokumentensammlung handelt es sich um einen Auszug aus dem WWW. Die Schlüsselwörter sollen Suchen von thematisch verwandten Web-Seiten erleichtern und die Resultat entsprechend verbessern.

All die oben aufgezählten Punkte sind Teil des modernen IR und müssen in einem System zumindest teilweise integriert sein. Durch TREC werden immer wieder neue Teilziele (vgl. die oben aufgeführten ‚Track') des IR definiert, welche für diese Disziplin zukunftsweisend sind.

[5] Unter auf Fakten basierenden Anfragen sind Fragen zu verstehen, deren Antworten direkt im Text als Fakten enthalten sind. Meist handelt es sich um kurze Anfragen mit kurzen Antworten.

1.3. Information

Information, welche mit IR-Methoden gefunden werden soll, kann in diversen Formen gespeichert sein, beispielsweise in Texten, Bildern (Fotos), Tonaufnahmen (Tonbänder oder Musik-CDs) oder Bildaufnahmen (Fernsehen oder Video). Die Gewinnung von Information aus all diesen Medien nennt man multimediales IR (vgl. [Schäuble, 1997]). Eigennamenerkennung ist speziell bei Information-Textretrieval wichtig. Textretrieval bedeutet, dass die zu gewinnende Information in Textform vorliegt. Information soll aus Nachrichten, Dokumentationen, Büchern etc. filtriert werden. Informationsgewinnung aus Texten erhält immer grössere Bedeutung, weil die Papierflut beständig wächst. Es werden heutzutage ca. 240 Terabytes Information jährlich in Büchern, Zeitungen und sonstigen Dokumenten produziert. Beinahe 10% dieser Informationen (23 Terabytes) sind in schriftlicher Form, also als Text, festgehalten (vgl. [Lyman und Varian, 2000]). Information-Textretrieval sowie IE (siehe nächste Seite genauere Definition) erhalten eine wichtige Rolle, um sich in dieser Informationsflut noch zurechtfinden zu können.

1.4. Ziele und Methoden des IR

Manche IR- und IE-Systeme verwenden Verarbeitungsprozesse für natürlicher Sprache (engl. ***n**atural **l**anguage **p**rocessing* = NLP), um ihre Ziele zu erreichen. NLP ist eine Sammlung von Methoden, um unstrukturierte Texte (Texte, welche keine expliziten Markierungen wie XML enthalten (vgl. [Sullivan, 2001], S. 29)) zu bearbeiten, z.B. Dokumente, Bücher oder Nachrichten. Interne Strukturen wie Syntax, Semantik und Pragmatik werden in den NLP verwendet. Morphologie kann zusätzlich als Hilfe zum Erkennen dieser Strukturen angewendet werden. Das Finden interner Strukturen sowie das Bearbeiten von unstrukturierten und semistrukturierten Texten (über eine Markierungssprache wie XML oder SGML teilstrukturiert) sind Themen der Computerlinguistik, welche mit NLP partiell gelöst werden können. Das Auffinden von Eigennamen ist mit der Semantik und Syntax eines Textes eng verknüpft und bedient sich somit auch NLP.

Trotz der ähnlichen Methoden in IR und IE sind die Ziele verschieden. IR versucht zu einer Anfrage relevante Dokumente auszuwählen, während IE die zu einer Anfrage relevanten Fakten aus Dokumenten filtriert (vgl. [Pazienza, 1997], S. V). Eigennamenerkennung nimmt sowohl bei IR wie auch bei IE eine zentrale Rolle ein.

Retrievalmodelle werden im klassischen IR in folgende Kategorien eingeteilt ([Poetzsch, 2001], S. 14f.):

1. ‚boole'sches Modell'

   Information wird mit Hilfe von mengentheoretischen Operatoren und Funktionen ($\wedge$, $\vee$, $\neg$ etc.[6]) extrahiert.

2. ‚Vektormodell'

   Information wird über Vektoren errechnet. In moderneren Einteilungen (vgl. [Baeza-Yates und Ribeiro-Neto, 1999]) werden Vektormodelle zu den algebraischen Modellen gerechnet. Algebraische Modelle berechnen Informationen und extrahieren sie über die zuvor berechneten Werte. Neben den Vektormodellen gehören auch neuronale Netzwerke in diese Kategorie.

3. ‚probabilistisches Modell'

   Information wird über Wahrscheinlichkeiten geschätzt und bewertet (z.B. bayes'sche Netzwerke).

Eine Anfrage eines Nutzers kann zwei Arten von Retrieval auslösen:

1. ‚ad-hoc-Retrieval'

   Ein Nutzer formuliert seinen Informationsbedarf mittels einer Anfrage. Die Recherche auf Grund der vorgegebenen Suchfrage wird über den Datenbestand genau einmal durchgeführt (retrospektive Recherche).

2. ‚filtering'

   Ein Nutzer formuliert seinen Informationsbedarf, indem er ihn in ein Nutzerprofil einträgt. Anhand des Nutzerprofils wird in einem definierten Zeitabstand eine Recherche mit neu eingegangenen Texten (z.B. Nachrichten) durchgeführt, deren Ergebnisse dem Auftragsgeber zugestellt werden (selektive Informationsverarbeitung).

In beiden Retrievalarten kann Eigennamenerkennung wichtig sein. Die Software der Firma *Eurospider* [7] unterstützt beide Anfrageformen.

Abhängig von den verschiedenen Methoden, die ein IR-System verwenden kann (vgl. auf vorheriger Seite Abschnitt über Retrieval-Modelle) gibt es verschieden Arten von Anfragen. Neben einer eigenen Retrievalsprache[8], die künstlich erstellt wurde und entsprechend erlernt werden muss, gibt es eine benutzerfreundlichere Variante. Ein Nutzer stellt seine Anfragen in natürlicher Sprache, entweder in ganzen Sätzen oder einzelnen Wörtern. Die Anfrage in na-

---

[6] Die mengentheoretischen Zeichen $\cap$, $\cup$ entsprechen den logischen Operatoren $\vee$, $\wedge$.

[7] Die Firma *Eurospider*, in welcher ich dieses Projekt ausführen durfte, ist in Kapitel I 3.3. „Die Firma *Eurospider*" genauer beschrieben.

[8] „Eine Retrievalsprache ist eine ‚künstliche Sprache', die der Wiederfindung von Information dient. Zur Formulierung der Fragestellung an das IR-System werden die in der Retrievalsprache festgelegten Kommandos (Befehle) benutzt."[Poetzsch, 2001], S. 105 (Bsp. SFQL (***S**tructured **F**ull-**T**ext **Q**uery **L**anguage*) in Anlehnung an SQL (***S**tructured **Q**uery **L**anguage*), die von Air Transport Association und Aircraft Industry Association verwendet wird ([Baeza-Yates und Ribeiro-Neto, 1999], S. 114).

türlicher Sprache nimmt immer stärker an Bedeutung zu, da der Nutzer in diesem Fall keine neue Sprache lernen muss, um Information mit Hilfe eines IR-Systems zu gewinnen. Prinzipiell sind vier verschieden Arten von Anfragen (schlüsselwortbasierte Anfragen[9]) zu unterscheiden:

1. Anfragen über einzelne Wörter

   Einzelne Wörter werden als Suchterm eingegeben, und es wird in der Dokumentenkollektion nach diesen Wörtern gesucht[10]. Je häufiger der Suchterm im Dokument auftaucht, desto relevanter ist es. Dies ist eine sehr einfache Variante. Um befriedigende Retrievalergebnisse zu erhalten, genügt diese Methode nicht.

2. Anfragen unter Berücksichtigung des Kontexts

   Die Anfrage ist eine Phrase, deren einzelne Wörter, falls es sich um bedeutungstragende handelt[11], in gleicher Schreibweise (inklusive Steuerzeichen, Abständen, Kommasetzung etc.) in der Dokumentenkollektion gesucht werden. Es werden folglich nur Dokumente gefunden, die Phrasen in derselben Schreibweise wie Anfrage enthalten.

   Beispiel 1:

   Benutzeranfrage: „*Was ist Information Retrieval*".
   reduzierte Anfrage: *Information Retrieval*
   gefundene Dokumente: ….. *Information Retrieval* ….
   nicht gefundene Dokumente: … *Information-Retrieval*….
   nicht gefundene Dokumente: … *Informatio**nR**etrieval*….
   nicht gefundene Dokumente: … *Informatio**ns** Retrieval*….
   etc.

   Flexibler gehandhabt wird eine Anfrage unter Berücksichtigung von Nähe. Bei der Anfrage, welche aus einer Sequenz von Wörtern oder einer Phrase besteht, muss eine maximale Distanz angegeben werden, in der die Wörter auftauchen dürfen. Die Distanz wird entweder in Buchstaben oder Wörtern bemessen abhängig vom IR-System. Die Rangierung der Dokumente basiert in beiden Methoden auf der Häufigkeit des Auftretens der Anfrageterme innerhalb eines Dokuments.

3. boole'sche Anfragen

   Man verbindet Wörter über boole'sche Operatoren (OR, AND, NOT) miteinander. Die gefundenen Dokumente werden auf Grund der Anzahl der gefundenen Wörter und deren Kombination innerhalb des Dokuments rangiert. Boole'sche Anfragen sind bei den

[9] Unter schlüsselwortbasierter Anfrage versteht man eine Anfrage, aus der Schlüsselwörter bestimmt werden, die für die Anfrage relevant sind und mit der Dokumentenkollektion verglichen werden.
[10] Bei diesem Retrievalmodell können Schlüsselwörter nicht über boole'sche Operatoren miteinander verbunden werden. Eine Anfrage besteht immer nur aus genau einem Wort.
[11] Hochfrequente Wörter (häufig sind dies Funktionswörter) werden in einer Stoppwortliste gesammelt. Die Anfrage wird um diese Wörter reduziert.

IR-Systemen eine beliebte und nach wie vor oft verwendete Lösung (z.B. im Suchsystem *Google*).

4. Anfragen in natürlicher Sprache

   Jedes Wort einer Anfrage, die man als Satz oder Phrase (in natürlicher Sprache) stellt, wird in den Dokumenten einzeln und als Abfolge (Phrase) gesucht. Die Resultate werden auf Grund der Häufigkeit der Wörter und Phrasen, welche in einem Dokument gefunden worden sind, rangiert.[12]

Eine Anfrage, die über Schlüsselwörter mit einer Dokumentenkollektion verglichen wird, kann man mit verschiedenen Techniken verbessern. Zu diesen Techniken zählen unter anderm linguistische Analysen, um die Suchen nach Präfix oder Suffix zu ermöglichen, die Anwendung von regulären Ausdrücken (z.B. Suche nach einem Wort wie *Januar* in Verbindung mit irgendeiner Zahl; die Zahl wird als regulärer Ausdruck und somit über ein spezifisches Muster (z.B. [0-9]) gesucht wird.) oder die Berücksichtigung von Schreibfehlern (*Blume* wird ebenfalls als *Blum e* gefunden)[13].

Linguistische Techniken erhalten im IR sowohl im Bereich der Anfragen, wie auch im Bereich des Indexierens von Dokumenten je länger je mehr eine bedeutende Rolle. Mit statistischen Techniken lassen sich zwar schnell grosse Mengen Text bearbeiten, es fehlt jedoch oft an Präzision, wie die gängigen Suchsysteme wie *Altavista* oder *Google* zeigen. Unter den von einem herkömmlichen Suchsystem gefundenen Dokumenten befinden sich viele, die bezüglich der Anfrage nicht relevant sind. Meiner Meinung nach lassen sich mit statistischen Methoden leicht 80% der gewünschten Ergebnisse erreichen, um die restlichen 20% zu Verbessern genügen die statistischen Methoden nicht. Eigennamenerkennung und -extraktion ist eine dieser linguistischen Techniken und gehört zu den gefragten Komponenten eines IR-Systems.

1.5. Bewertung von IR-Systemen

Die Güte eines IR-Systems wird über Ausbeute (engl. recall) und Präzision (engl. precision) festgelegt. Die beiden Werte sind wie folgt definiert : ([Poetzsch, 2001] S. 17)

Ausbeute:

$$\frac{\text{Zahl der gefundenen Dokumente}}{\text{Zahl aller relevanten Dokumente der Datenbank}}$$

Formel 1

[12] Es gibt noch weitere Modelle, wie Wörter aus Anfragen in Bezug zu den Dokumenten gewichtet werden können (z.B. Vektormodelle, in denen Anfragen und Dokumente als Gewichte von Vektoren verstanden werden).
[13] Vgl. [Baeza-Yates und Ribeiro-Neto, 1999]. S. 99ff.

Präzision:

$$\frac{\text{Zahl der nachgewiesenen, relevanten Dokumente}}{\text{Zahl aller nachgewiesenen Dokumente}}$$

Formel 2

In Formel 1 und Formel 2 wird eine Rangierung der Dokumente nicht berücksichtigt. Man teilt die Dokumente folglich nur in zwei Kategorien ein: relevant und nicht relevant. Um eine Rangierung mitzuberücksichtigen, gibt es andere Formeln zur Berechnung der Güte eines IR-Systems.

Eine Formel, die die Kombination von Ausbeute und Präzision berechnet und bei der Ausbeute oder Präzision je nach Wunsch gewichtet werden kann, ist der F-Wert (engl. f-score):

F-Wert:

$$\frac{(\beta^2 + 1)\ \text{Präzision} * \text{Ausbeute}}{\beta^2\ \text{Präzision} + \text{Ausbeute}}$$

Formel 3

Wenn Präzision und Ausbeute gleich gewichtet sein sollen, ist β 1. Falls die Präzision in der Evaluation stärker gewichtet werden soll, wird β grösser 1 gewählt. Wenn man der Ausbeute den Vorzug geben will, muss β eine Wert kleiner 1 erhalten. Wenn bei einer Evaluation die F-Werten nicht weiter dokumentiert sind, so wird weder Ausbeute noch Präzision der Vorzug gegeben.

Präzision und Ausbeute beeinflussen einander in dem Sinn, dass eine hohe Ausbeute meist eine geringere Präzision verursacht und umgekehrt. Dies kann mit einer Präzisions-Ausbeute-Kurve dargestellt werden ([Gaus, 2000], S. 228).

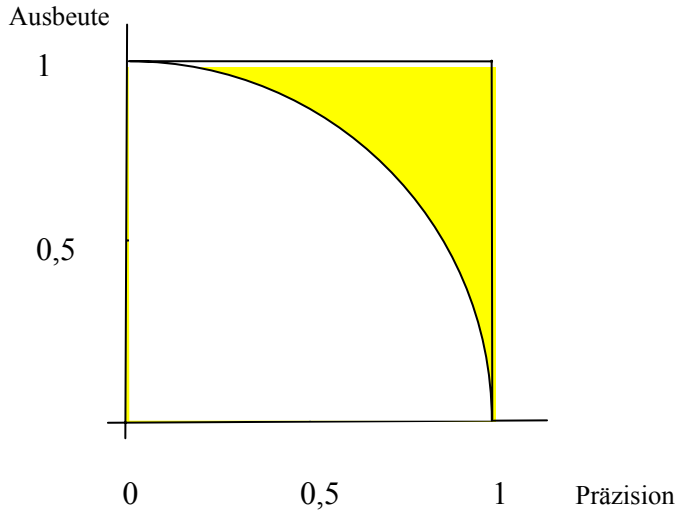


Abbildung 1

Das Ideal von 100% Ausbeute bei 100% Präzision ist bis jetzt von keinem System erreicht worden. Gute Systeme erreichen Werte, welche rechts der Kurve im farbigen Bereich zu liegen kommen.

2. Eigennamenerkennung

Ähnlich den Retrieval Modellen im IR (vgl. Kapitel I 1.4. „Ziele und Methoden des IR“) werden drei verschiedene Methoden zur Eigennamenerkennung verwendet:

1. statistischer Ansatz
2. regel- und lexikonbasierter Ansatz
3. automatisches Erlernen von Regeln (statistisch) mit Verwendung von kleinen Ressourcen[14] (lexikonbasiert)

Es ist zu beobachten, dass heutzutage statistische Methoden häufiger verwendet werden, da sie für grosse Datenmengen geeigneter (vgl. Abbildung 2) sind und zusätzlich sprach- und domänenunabhängig verwendet werden können.

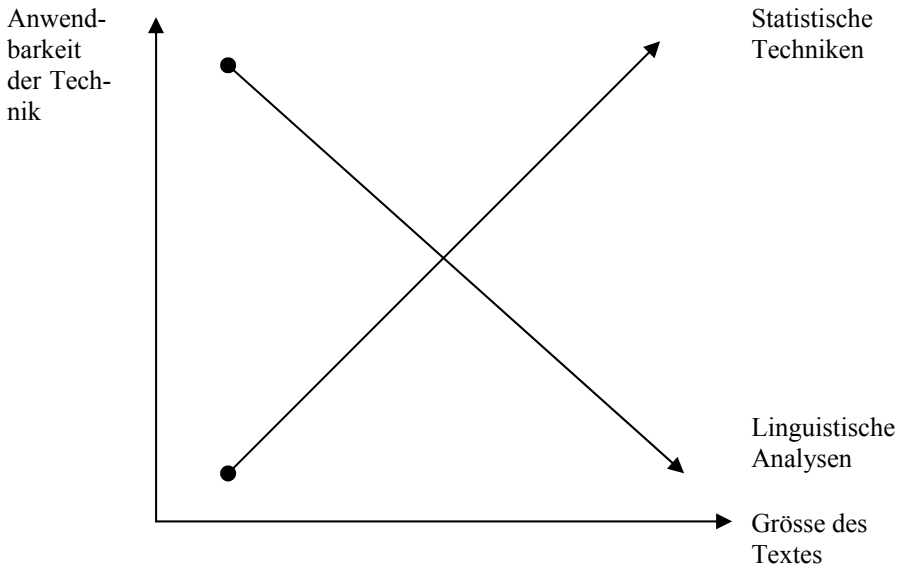


Abbildung 2

(vgl. [Sullivan, 2001] S. 44)

Allerdings benötigen statistische Ansätze ein grosses, von Hand annotiertes Korpus, um das System zu trainieren (beispielsweise automatisches Erlernen von Regeln, in welchem Kontext eine gewisse Wortkategorie auftritt). Ein annotiertes Korpus ist eine Menge von Texten, in denen Wörter oder Phrasen entsprechend gewisser Richtlinien explizit markiert worden sind (z.B. Markierung von Personennamen). Über statistische Methoden kann automatisch gelernt werden, in welchen Kontexten mit grosser Wahrscheinlichkeit ein Personenname auftritt.

[14] Ressourcen sind Listen (z.B. Vornamenlisten) von Wörtern, die einem bestimmten Thema angehören. Die Listen entsprechen einem themenspezifischen Lexikon. Mehr zu Ressourcen in Kapitel I 4. „Ressourcen“.

Diese Regeln werden anschliessend auf nicht annotierte Texte angewendet. Manuell annotierte Texte allerdings sind bis jetzt nur in geringen Mengen erstellt worden. Statistische Methoden gewinnen jedoch an Güte, je grösser das annotierte Korpus ist.

Sowohl Systeme, welche einen statistischen Ansatz wählen, als auch Systeme, welche manuell erstellte Regeln und entsprechenden Ressourcen verwenden, erreichen gute Leistungen. Präzisions- und Ausbeutewerte bewegen sich im Bereich 80-90%.

Zu den Systemen bzw. Ansätzen, welche **statistisch basiert** sind, gehören folgende:

- **IdentiFinder™** (vgl. [Kubala et al., 1998])[15]

IdentiFinder™ wurde in der Firma *BBN Technologies in Cambridge* (Ma) entwickelt. Es handelt sich um ein System zur Erkennung und Extraktion von Personen, geographischen Eigennamen, Firmen, Zahlen, Daten (Bsp. 2.5.03), Geldbeträgen und anderen Kategorien, welche über Beispiele erlernt werden können. Das Programm oder Lizenzen sind kommerziell erwerbbar.

Das System benutzt verdeckte Markov Modelle (HMM = Hidden Markov Models, mehr dazu in Kapitel I 2.1. „Statistische Methoden“), um Informationen zu extrahieren. Dieser Ansatz erlaubt einen Einsatz sowohl sprachunabhängig, als auch domänenunabhängig.

Man trainierte das System mit einer Textkollektion, bestehend aus *braodcast news*, und testete es anhand der HUB-4[16] Testdaten aus dem Jahre 96, der HUB-4 Evaluationsdaten aus dem Jahre 97, *Wall-Street-Journal*-Texte (MUC-6[17]) und *New-York-Times-News-Service*-Texte (MUC-7). Die Resultate sehen wie folgt aus:

New York Times: F-Wert[18] von 94,8% (Text nur in Grossbuchstaben: 92,6%, SNOR (***S**peech **N**ormalized **O**rthographic **R**epresentation*) -Format[19]: 90,5%)

Wall Street Journal: F-Wert von 94,9% (Text nur in Grossbuchstaben: 94,1%, SNOR-Format: 92,1%)

Broadcast News (in SNOR-Format): F-Wert von 87%.

- **[Rössler, 2002]**

Marc Rössler hat an der Universität Duisburg in Deutschland einen Ansatz entwickelt, mit dem Eigennamen (Personen-, Organisationsnamen und geographische Eigennamen) in deut-

---

[15] Mehr Informationen zu *IdentiFinder™* unter [IdentiFinder, http://www.bbn.com/speech/identifinder.html].

[16] HUB-4 hatte das Ziel Informationen aus *brodcast news* zu extrahieren, welche aus Fernseh- oder Radioreportagen stammen. Folglich handelt es sich hierbei um gesprochene Sprache, welche anderen syntaktischen Kriterien unterliegt wie in geschriebene Sprache (z.B. Missachtung von Grossschreibung, keine Satzzeichen, unvollständige Sätze etc.). HUB-4 ist von der *Speech Group* organisiert (vgl. [HUB-4, http://www.nist.gov/speech/tests/bnr/hub4_98/hub4_98.htm]).

[17] Mehr zu MUC (***M**essage **U**nderstanding **C**onference*) in Kapitel II 1.3. „Kategorien von Eigennamen“.

[18] Definition von F-Wert in Kapitel I 2.3. „Bewertung von Eigennamenerkennung“.

[19] Alle Buchstaben sind grossgeschrieben, alle Abkürzungen ausgeschrieben und alle Zahlen als Wörter ausformuliert.

schen Zeitungstexten ohne Verwendung von Ressourcen erkannt werden können. Ein Markov Modell 2. Ordnung wird anhand eines Korpus trainiert, in dem die Wortarten zuvor bestimmt worden sind. Eine Liste mit Wörtern, die mit einer gewissen Wahrscheinlichkeit in enger Verbindung mit Eigennamen stehen, ist über statistische Methoden aus einem annotierten Korpus gewonnen worden und unterstützt die Eigennamenerkennung. Im zu analysierenden Text müssen zuvor die Wortarten annotiert werden. Die Wahrscheinlichkeit, dass ein Wort in der zuvor generierten Liste vorkommt, die Wortart und das Token selbst werden bei der Eigennamenerkennung berücksichtigt. Beim verwendeten Test- und Trainingskorpus handelt es sich um die Computer-Zeitung aus dem Jahre 1996 [Konradin-Verlag, 1998] (vgl. Kapitel IV 2. „Testkorpus für die Evaluation in deutschsprachigen Texten“)[20]. Das Testkorpus wurde über eine zehnfache Cross-Validierung in zehn verschiedene Teile aufgeteilt, um so sicher zu stellen, dass es sich um repräsentative Testdaten handelt und das trainierte Modell mit „neuen“ Texten (andere Texte, als diejenigen, auf denen das Modell trainiert worden ist) getestet werden kann. Für jede Entität oder Teilentität hat Rössler in einem Teil des Korpus gezählt, wie häufig die beiden vorgängigen als auch die beiden folgenden Wörter in Verbindung mit einer Entität vorkommen. Es handelt sich dabei um Trigramme. Um die Liste mit Wörtern, die in den zuvor gebildeten Trigrammen vorkommen, zu begrenzen, gewichtet Rössler die Wörter mit TF*IDF-Mass und setzt einen entsprechenden Schwellwert, der erreicht werden muss, um in die Liste aufgenommen zu werden. TF ist Term Frequenz, welche beschreibt, wie oft ein Wort innerhalb eines Dokuments auftritt. IDF ist die entgegengesetzte Dokument Frequenz, die berücksichtigt, dass Wörter, die selten in einem Dokument vorkommen, höher gewichtet werden (vgl. [Jurafsky und Martin, 2000], S. 651f.). Um zu verhindern, dass in der Liste Wörter stehen, die direkt auf eine spezifische Organisation verweisen und der Ansatz einem lexikonbasierten entspräche[21], wurden diejenigen Einträge entfernt, die nur aus dem Organisationsnamen (z.B. *Microsoft* oder *AOL*) bestehen[22]. Über den TnT-Tagger [TnT, http://www.coli.uni-sb.de/~thorsten/tnt/] sind die Wortarten der Trainingstexte, mit denen das Markov Modell trainiert wird, sowie Wortarten der Texte, die zum Testen des Modells dienen, bestimmt worden. Die Wortarten in den Trainingstexten werden auf eine spezifische Notation reduziert, die nur die Eigennamen und je zwei Wörter davor und dahinter berücksichtigt. Es wurden verschieden Experimente durchgeführt, wobei dasjenige Experiment am bes-

[20] Rössler hat mir sein Korpus zur Evaluation dieses Projekts zur Verfügung gestellt.

[21] Ein lexikonbasierter Ansatz ist insofern problematisch, als die Anwendung auf andere Texte bzw. Textsorten schlechtere Resultate erzielen kann. Fehler entstehen, weil neue Firmennamen nicht in der Liste stehen oder in einem neuen Kontext keine Firmennamen mehr sind. Rössler strebt allerdings einen möglichst unabhängigen Ansatz zur Eigennamenerkennung an.

[22] Die Problematik, dass in der Liste Eigennamen stehen und nicht nur Teile von Eigennamen sind, ist lediglich bei Firmennamen aufgetreten.

ten abschnitt, das sowohl die Token selbst als auch die Wortarten mitberücksichtigt hat. Des weitern hat Rössler mit dem Ansatz „Lernen, Hinzufügen, Vergessen“ von Volk und Clematide seine Resultate verbessern können. Jeder Text wird mit einer Liste, welche die zuvor gefundenen Eigennamen enthält, überprüft. Falls ein Eigenname im ersten Schritt nicht gefunden worden ist, wird er in einem zweiten Durchgang, in dem alle auf Grund einer Regel gefunden Eigennamen unabhängig vom Kontext (und der Regel) erneut gesucht werden, als solcher erkannt und der Liste mit Eigennamen hinzugefügt. Am Ende des Vorgangs werden alle Eigennamen vergessen ([Volk und Clematide, 2001]). Das erlernte Markov Modell, welches aus den Wörtern der Listen, den lexikalischen Wahrscheinlichkeiten sowie den in Bezug auf die Position im Trigramm errechneten Wahrscheinlichkeiten besteht, wird auf einen über den TnT-Tagger annotierten (Test-)Text angewendet, um so die Eigennamen zu erkennen. Mit seinem Ansatze hat Rössler die folgenden Evaluationswerte erreicht: Personennamen: 78% Ausbeute, 71% Präzision, Organisationsnamen: 62%, Ausbeute, 85% Präzision, geographische Eigennamen: 77% Ausbeute, 86% Präzision.

Systeme, welche **Lexikon und Regeln** zur Erkennung und Extraktion von Eigennamen verwenden, sind folgende:

- **TextPro** (vgl. [Appelt und Martin])[23]

Das TextPro-System wurde am *Artificial Intelligence Center SRI International* in Kalifornien entwickelt. Unterdessen handelt es sich um ein ausgereiftes IE-Programm für *Macintosh*-Computer, welches kommerziell erwerbbar ist. Das System wurde insbesondere entwickelt, um Eigennamenerkennung aus gesprochenen englischen Texten[24] zu ermöglichen. Ein kaskadierter endlicher Automat (engl. transducer) interpretiert Regeln, welche auf einem Lexikon aufgebaut sind. In TextPro werden unter anderem Lexika verwendet, welche Ortschaften aus den USA, Personennamen, Namen von prominenten Amerikanern und verschiedenen amerikanischen Organisationen beinhalten. Die Regeln berücksichtigen interne Strukturen der Entitäten (z.B. Namen sind Kombinationen aus Vor-, Mittel- und Nachname) sowie die Umgebung, in der eine Entität steht. In der HUB-4 (vgl. Fussnote 16, S. 11) wurde ein F-Wert von 89% (bei unbekannten Texten) erreicht.

[23] Mehr Informationen zu TextPro unter [TextPro, http://www.ai.sri.com/~appelt/TextPro/].

[24] Bei gesprochenen Texten werden Grosskleinschreibung, Punktion und korrekter Syntax weniger Beachtung geschenkt. Deshalb können diese Merkmale, welche in der geschriebenen Sprache dominant sind, für die Erkennung kaum eingesetzt werden.

- **MUSE** (***Mu**lti-**S**ource **E**ntity finder*) [Maynard et al.] und [Maynard et al., 2001]

MUSE ist an der Universität Sheffield entwickelt worden. Das System baut auf der Architektur von GATE[25] auf und benutzt Annie (***A** **N**early-**N**ew **IE** System*, siehe Abbildung 3), Teil dieser Architektur. Die Verarbeitungsschritte von MUSE sehen entsprechend der Architektur von Gate wie folgt aus:

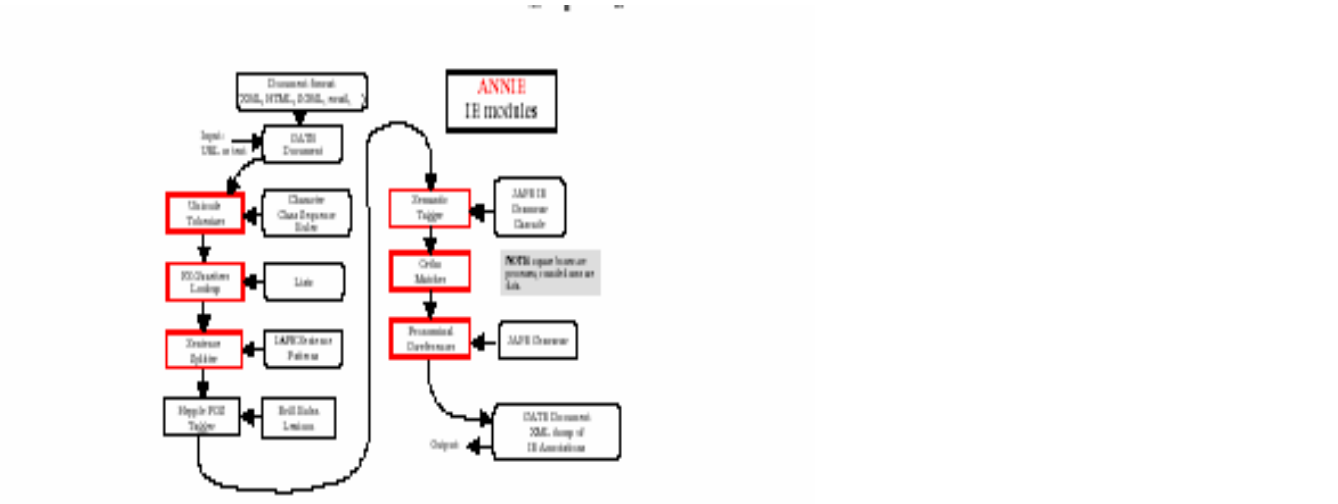


Abbildung 3

[Maynard et al., 2002a] S. 3

Ein Dokument wird zuerst in Bestandteile zerlegt, wie z.B. in Token (tokenizer) oder einzelne Sätze (sentence splitter)), um dann die Wortarten zu bestimmen (tagger[26]).

Erst jetzt setzt der eigentliche Informationserkennung und -extraktionsschritt mit Hilfe des ‚semantischen Taggers' ein. Über einen Grammatik, die zur Erkennung von semantisch definierten Wortarten dient und von Hand geschriebene Regeln enthält, werden spezifische Muster erkannt (beispielsweise Umgebung in der ein Personenamen stehen kann). Die Regeln, welche auf Grund der erkannten Muster entstehen, werden mit Ressourcen (gazetteer) unterstützt (vgl. Kapitel I 1.1. „Definition"). Regeln und Ressourcen dienen somit zur Eigennamenerkennung und erlauben die folgende Extraktion der Entitäten.

Dieses System benutzt folglich den klassischen, regel- und lexikonbasierten Ansatz. Dieser Ansatz profitiert von einer genauen linguistischen Analyse des Textes, inklusiv einer Wortartenbestimmung, welche im Voraus vorgenommen wird. Der ‚semantische Tagger' ist in einer eigenen Mustererkennungssprache namens JAPE (***J**ava **A**nnotation **P**atterns **E**ngine*) geschrieben und wird unter anderem zur Erkennung von Entitäten eingesetzt.

---

[25] GATE wurde im Jahre 1995 an der Universität Sheffield entwickelt. Es ist eine spezifische Architektur und Entwicklungsumgebung für NLP-Prozesse, insbesondere IE-Extraktion. Mehr dazu unter [GATE, http://gate.ac.uk/].

[26] Es handelt sich um eine modifizierte Version des Brill Taggers.

Auf Grund des ACE-Programms[27] im Bereich Eigennamenerkennung im Rahmen des EDT (*the **E**ntity **D**etection and **T**racking*) ergab MUSE bei der Evaluation folgende Wert: Das System erreichte 82-86% Präzision und Ausbeute je nach Texttyp (vgl. [Maynard et al., 2002a]).

- **[Volk und Clematide, 2001]**

Dieser Ansatz zur Eigennamenerkennung, den Martin Volk und Simon Clematide an der Universität Zürich entwickelt haben, baut auf ‚Lernen, Filtern, Hinzufügen und Vergessen' auf. Er wurde insbesondere für deutsche Texte entwickelt. Auf Grund von Ressourcen und Regeln werden Personennamen, geographische Entitäten und Firmennamen erkannt. Ressourcen sind Vornamen, geographische Eigennamen (Städte-, Ländernamen, geographische Adjektive etc.) und diverse Trigger (Auslöser von Eigennamen) für die Erkennung von Personen- (*Dr.*, *Professor*, *von*, *van*, etc.) und Firmennamen (*AG*, *Firma* etc.). Über Regeln wird die Umgebung beschrieben, in der ein Eigenname auftauchen kann, sowie interne Strukturen (beispielsweise folgt gewissen Triggern ein Eigenname oder Eigennamen beginnen immer mit Grossbuchstaben etc.). Teile dieser gelernten Entitäten (Bsp. *Peter Müller* wurde gelernt, *Müller* ist ein Teil davon) werden falls nötig gefiltert und im Text in gekürzter Form erneut gesucht. Als Filter für einfache, d.h. aus einem Wort bestehende Firmenamen, benutzten Volk und Clematide *Gertwol*[28], ein morphologisches Analysesystem für Deutsch.

Dieser Ansatz baut folglich auf Lexika und manuelle Regeln auf, die von vorangegangenen linguistischen Analysen (Wortartenbestimmung und morphologische Analyse) profitieren.

Eine Evaluation von 990 Sätzen aus der Computer-Zeitung [Konradin-Verlag, 1998], welche zuvor von Hand annotiert worden sind, ergab folgendes: Personennamen: 86% Ausbeute, 92% Präzision, geographische Eigennamen: 91% Ausbeute, 81% Präzision, Firmennamen: 81%Ausbeute, 76% Präzision.

Algorithmen und Ressourcen, welche für diesen Ansatz von Volk und Clematide erstellt worden sind, haben mir als Grundlage des Projektes bei *Eurospider* gedient (vgl. Kapitel I 3.2. „Grundlagen") und werden im weiteren Verlauf dieser Arbeit noch genauer erläutert.

Zu Systemen, welche **kleine Ressourcen verwenden und Regeln automatisch mit statistischen Techniken** erlernen, gehören folgende:

---

[27] ACE (***A**utomatic **C**ontent **E**xtraction*) versucht die Entwicklung von robusten NLP-Applikationen zu fördern in dem sie linguistische Ressourcen zu Verfügung stellt. (Mehr darüber unter [ACE, http://www.nist.gov/speech/tests/ace/index.htm]

[28] *Gertwol* ist ein System, das automatische deutsche Wortformen erkennt. *Gertwo*l enthält neben verschiedenen Lexikoneinträgen auch 12000 Eigennamen und 17000 Abkürzungen. Unter [Gertwol, http://www.lingsoft.fi/cgi-bin/gertwol] kann das System online getestet werden.

MENE (***M**aximum **E**ntropy **N**amed **E**ntity*, vgl. [Borthwick et al., 1998])

MENE ist an der Universität von New York (NYU) entwickelt worden. In diesem System versucht man einen statistischen Ansatz (Lernen von Entitäten über maximale Entropie (engl. maximum entropy), mehr dazu in Kapitel I 2.1. „Statistische Methoden“) mit binären Entscheidungen (z.B. 1 für grossgeschriebenes Wort und 0 für kleingeschriebenes Wort) zu verbessern. Die Regeln, welche automatisch errechnet werden, sind von einer solchen Ja(1)-Nein(0)-Entscheidung abhängig. Die kleinen Ressourcen werden ebenfalls über binäre Entscheidungen ins System integriert (1 = ist Teil der Ressource, 0 = ist nicht Teil der Ressource). Die maximale Entropie als Methoden zum Erlernen von Regeln hat Borthewick ein Jahr später zum Thema seiner Dissertation gemacht ([Borthwick, 1999]). Die Idee des automatischen Erlernens von Regeln ist deshalb für das Department für Computerwissenschaft von der NYU so wichtig, weil es sprachunabhängig funktioniert, und somit auch ohne grosse Änderungen für Japanisch (ein weiterer Forschungsschwerpunkt des Institutes) übernommen und getestet werden kann.

MENE erreichte an der MUC-7 folgende Werte:

- Test innerhalb der Domäne, in welcher das System trainiert wurde: F-Wert: 92,20% (Präzision: 96% und Ausbeute:89%)
- Test ausserhalb einer spezifischen Domänen: F-Wert: 84,22% (Präzision: 91% und Ausbeute:78%)

Entitäten, welche erkannt wurden, sind Personen-, Organisationsnamen, geographische Namen, Datums-, Zeit-, Prozentangaben und Geldbeträge (gemäss MUC-7, vgl. Kapitel II „Theoretische Grundlagen“).

- **[Quasthoff et al., 2002]**

Quasthoff entwickelt an der Universität Leipzig in Deutschland einen Ansatz in Eigennamenerkennung, der mit kleinen Ressourcen auskommt. Er erkennt Personenamen durch automatisches Erlernen und Verifizieren. Der Algorithmus beginnt mit einer kleinen Anzahl gesetzter Wörter (z.B. Titel) und Regeln (z.B. Titel, gefolgt von grossgeschriebenem Wort, ist ein Nachname), gewinnt während des Algorithmus mehr Wissen über die Umgebung, in der Eigennamen stehen werden und versucht so neue Kandidaten für Eigennamen zu finden. Je höher die Wahrscheinlichkeit ist, dass ein gewisses Wort vor oder nach einem Eigennamen steht, desto eher wird es in einem andern Text als Auslöser (Trigger) für eine Entität gewertet. Die Berechnung der Wahrscheinlichkeitswerte geschieht über HMM’s. Die Berechnung der Wörter, die mit grösster Wahrscheinlichkeit als Trigger dienen werden, geschieht über den Algorithmus *Expectation Maximation*. Der Lern- sowie der Verifikationsschritt sind iterativ.

Der Algorithmus erreichte folgende Werte bei einer Evaluation mit einem Korpus von 1000 Sätzen: Personenamen in deutschen Texten: 97,5% Präzision, 71,4% Ausbeute.

Die folgende Grafik enthält eine Zusammenstellung aller zuvor vorgestellten Systeme und Ansätze zur Eigennamenerkennung und stellt diese in Beziehung zu den verwendeten Techniken.

	Manuell erstellte Regeln	**Automatisch erlernte Regeln (statistische Methoden)**
lexikonbasiert	Projekt in *Eurospider* TextPro MUSE [Volk und Clematide, 2001]	MENE [Quasthoff et al., 2002]
Ressourcenfrei (ohne Lexikon)[29]		IdentifFinder™ [Rössler, 2002]

Tabelle 1

2.1. Statistische Methoden

Zu statistischen Ansätzen, welche zur Eigennamenerkennung eingesetzt werden gehören unter anderem folgende Methoden:

1. Markov-Modelle und verborgene Markov-Modelle (engl. ***h**idden **m**arkov **m**odels* = HMM)
2. Maximum Entropie

Markov-Modelle und **HMM** werden eingesetzt, um zu errechnen, welches die wahrscheinlichsten Wörter, bzw. Phrasen sind, welche einem Tag oder einer Satzkategorie zugeordnet werden. Auf Eigennamenerkennung angewendet würde das bedeuten, dass über Markov-Modelle berechnet werden kann, welche Wörter am wahrscheinlichsten einem Eigennamen, bzw. welcher Kategorie von Eigennamen (Personenname, geographischer Eigennamen, Firmennamen etc.), zugeordnet werden können.

Ein Markov-Modell besteht per Definition ([Krenn und Samuelsson, 1997] aus folgenden Punkten:

1. einer endlichen Menge von Zuständen $\Omega = \{s_1,....,s_n\}$,

   Dies würde bei einer Analyse von Wörtern den Wortarten entsprechen, bei einer Analyse von Phrasen, den grammatikalischen Kategorien.
2. einem Ausgabealphabet $\Sigma = \{\sigma_1,...,\sigma_n\}$,

   Dies würde dem Output, also den Wörtern, bzw. den Phrasen, entsprechen.

[29] Ressourcenfrei bedeutet, dass keine Entitäten oder Teile einer Entität in Listen manuell zusammengetragen worden sind (z.B. Erstellung einer Vornamensliste zum Erkennen von Personennamen), um Eigennamen unmittelbar zu erkennen.

3. „einer [n∗n]-Zustandsübergangsmatrix P“ ([Brants et al., 2000]),
   Diese Matrix enthält die Wahrscheinlichkeiten der möglichen Übergänge von einem Tag- bzw. Phrase zur nächsten.
4. „einer [n∗k]-Ausgabematrix A, die für jedes Zustand-Ausgabe-Paar die Wahrscheinlichkeit festlegt, mit der der Zustand diese Ausgabe macht“,
   Konkret werden in dieser Matrix die Wahrscheinlichkeiten ausgerechnet, mit welcher ein Wort einer Wortart zugeordnet wird.
5. „einem Anfangs-Wahrscheinlichkeitsvektor v“.
   Dieser gibt die Wahrscheinlichkeit an, dass vom Zustand s_0 (Startzustand) in den Zustand s_1 (Folgezustand) gewechselt wird und in alle weitern Zustände i-n.

Jedes Markov-Modell lässt sich in einem Graphen veranschaulichen (vgl. Abbildung 4):

Geg.: $\Omega = \{s_1,s_2\}$, $\Sigma = \{a,b,c\}$

$$P \begin{array}{cc} \begin{bmatrix} 0{,}2 & 0{,}8 \\ 0{,}5 & 0{,}5 \end{bmatrix} & \begin{array}{c} s_1 \\ s_2 \end{array} \\ \begin{array}{cc} s_1 & s_2 \end{array} & \end{array} \qquad A \begin{array}{cc} \begin{bmatrix} 0{,}5 & 0{,}5 & 0{,}0 \\ 0{,}0 & 0{,}5 & 0{,}5 \end{bmatrix} \\ \begin{array}{ccc} a & b & c \end{array} \end{array}$$

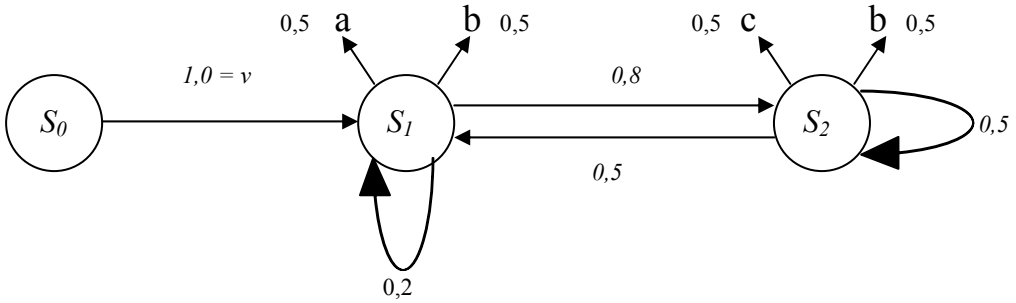


Abbildung 4

(vgl. [Brants et al., 2000], S. 4)

In einem linguistischen Beispiel muss man sich nun vorstellen, dass s_1 und s_2 Kategorien von Eigennamen sind, denen eine gewisse Anzahl Wörter, also a, b, c, mit einer gewissen Wahrscheinlichkeit zugeordnet werden.

Das HMM ist eine spezielle Sorte von einem Markov-Modell und heisst so, weil nur die Ausgabe bekannt ist, alle Wahrscheinlichkeit (in Abbildung 4 die Matrizen *P* und *A*) aber verdeckt.

Das soeben beschriebene Markov-Modell ist ein Modell 1. Ordnung. Dies bedeutet, dass es nur von einem bestehenden Zustand abhängig ist (s_1). Um nun ein Modell 2. Ordnung zu berechnen, oder auch ein Modell n-ter Ordnung, kann man durch Zusammenfassen der jeweils n

Vorgänger-Zustände zu einem einzigen, neuen Zustand jedes Markov-Modell höherer Ordnung in ein Modell 1.Ordnung umwandeln.

Systeme, welche Eigennamen vollständig automatisch erkennen, ohne Verwendung von Ressourcen, behelfen sich oft mit entsprechenden Markov Modellen oder HMM. Für Wortartenbestimmung werden sie bereits erfolgreich eingesetzt (vgl. [TnT, http://www.coli.uni-sb.de/~thorsten/tnt/]).

Bei kaskadierten (engl. cascaded) Markov-Modellen wird schichtweise aufeinander aufgebaut (vgl. Abbildung 5).

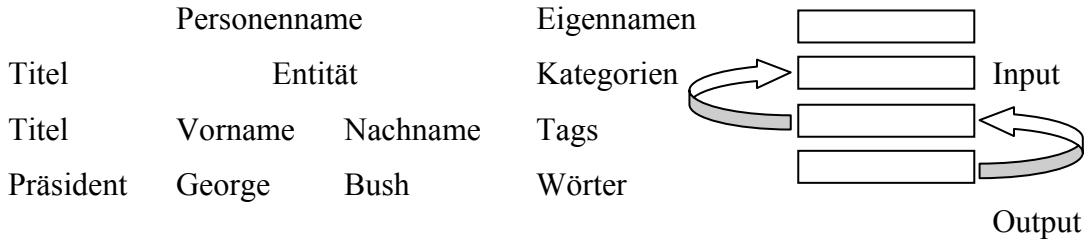


Abbildung 5

Die nächste Schicht wird jeweils unter Berücksichtigung von lexikalischer Wahrscheinlichkeit und kontextueller Wahrscheinlichkeit berechnet. Die ‚lexikalische Wahrscheinlichkeit' entspricht der Wahrscheinlichkeit, dass ein Wort bzw. Token (z.B. *George*) einer Kategorie (z.B. *Vorname*) zugeordnet wird (p(VORNAME | George)). Mit der Übergangswahrscheinlichkeit wird berechnet, ob ein Wort oder Token (z.B. *Bush*) einer Kategorie (z.B. *Nachname*) angehört, unter Berücksichtigung der davor stehenden Wörter (z.B. *Präsident*, *George*), die ebenfalls Kategorien angehören ((p(Bush | Präsident, George)). In der Regel wird die Übergangswahrscheinlichkeit (kontextuelle Wahrscheinlichkeit) anhand von Trigrammen (3 Wörter) berechnet.

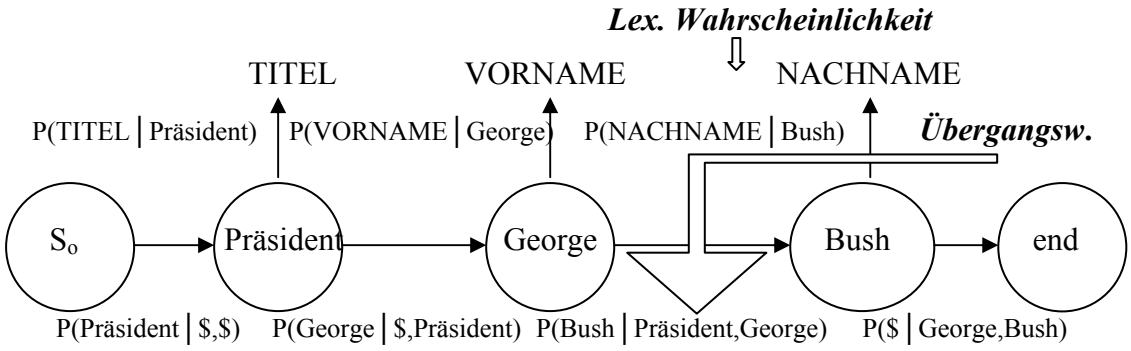


Abbildung 6

Das *$*-Zeichen bedeutet jeweils Anfangs- und Endposition. Das Fenster wird über drei Wörter gespannt, weshalb vorne zwei *$*-Zeichen vorkommen, also zwei Anfangspositionen angenommen werden.

Um zu verhindern, dass ambige Wörter (z.B. *Bus*h) nicht falsch analysiert werden, werden alle Wahrscheinlichkeiten in einer Schicht berechnet und der nächsten Schicht weiter gegeben. Es muss nicht von Anfang entschieden werden, dass in einem N-Gramm (in diesem Beispiel ein Trigramm) ein Wort definitiv einer Kategorie angehört. Erst unter Berücksichtigung des weiteren Kontextes, in dem das entsprechende Wort steht, fällt der definitive Entscheid. Um diesen Entscheid zu fällen, wird über den Viterbi-Algorithmus der beste Weg automatisch und effizient berechnet. Als erstes wird ein Wahrscheinlichkeitsraster über alle Wahrscheinlichkeit erstellt.

	PERSONENNAME (95%)	
	VORNAME (5%)	
		VORNAME (5%)
		STADT (10%)
	NACHNAME (40%)	KEINE ENTITÄT (50%)
TITEL 100%	*VORNAME* (60%)	*NACHNAME* (35%)
0	**1**	**2** **3**
Präsident	**George**	**Bush**

Abbildung 7[30]

In diesem Gitter muss nun der beste Weg vom Knoten 0 bis zum Endknoten gefunden werden. Der beste Weg entspricht nicht immer der Aufsummierung der höchsten Wahrscheinlichkeiten in jeder Spalte (z.B. Titel + Vorname + Keine Entität). Der wahrscheinlichste und somit beste Weg wird über die

Formel 4 berechnet:

$$\delta_j(i) \overset{\text{def}}{=} \max_{t1\ldots ti-1} p(w_1\ldots w_{i-1}, t_1\ldots t_{i-1} = j)$$

Formel 4

δ ist eine Funktion (ein Algorithmus). In der Funktion δ wird für jedes Wort *w* berechnet, mit welcher Wahrscheinlichkeit *p* es einer Kategorie *t* zugeordnet wird. Am Ende maximiert man das Ergebnis, um die optimale Funktion zu berechnen

Eine andere statistische Methode, welche das automatische Berechnen von Regeln erlaubt, ist die Maximierung der Entropie (engl. ***m**aximum **e**ntropy* = **ME**). Entropie ist ein Wert für Information, der allerdings in NLP nur schätzbar ist. Die Idee der ME besteht darin, dass keine andere Annahme gemacht wird ausser, dass ein Wort einer möglichen Klasse einer Klasse in

[30] Es handelt sich hier um ein fiktionales Beispiel. Alle Zahlen sind frei erfunden.

einem Modell mit einer gewissen Wahrscheinlichkeit entspricht. Die abhängige Entropie wird wie folgt berechnet:

$$H_{n,c} = -\Sigma\, p(n, c) \log p(c \mid n) \quad = \quad -\Sigma_{n}\, \tilde{p}(n)\, \Sigma_{c}\, p(c \mid n) \log p(c \mid n)$$

Formel 5

C ist eine bestimmte Klasse, *n* ein Wort. Die Entropie, welche in Abhängigkeit von einer bestimmten Klasse und einem Wort berechnet wird, muss am Ende maximiert werden. Der maximierte Wert entspricht dem Entscheidungsgehalt. Je grösser er ist, umso wahrscheinlicher wird das Wort *n* einer gebildeten Klasse *c* zugeordnet. Mehr Information zu den verschieden statistischen Methoden ist in [Manning und Schütze, 2002] zu finden.

2.2. Regeln und Lexikon

Regeln können mit statistischen Methoden (z.B. *ME*) automatisch errechnet oder manuell entwickelt und in einer Mustererkennungssprache (z.B. *Perl*) ausgedrückt werden. In diesem Projekt habe ich teils manuell Regeln neu entwickelt, teils Volks Algorithmen (vgl. [Volk, 2002]) übernommen oder erweitert und in der Konzeptsensorsprache (eine Mustererkennungssprache, die für interne Verwendung in der Firma entwickelt worden ist) ausgedrückt. Die Konzeptsensorsprache wird im Kapitel III 2. „Konzeptsensorsprache“ genauer vorgestellt und soll als Beispiel für Mustererkennungssprachen dienen. Algorithmen, welche die Erkennung von Eigennamen erlauben (vgl. Abbildung 8), müssen in eine entsprechende Mustererkennungssprache übersetzt werden. Sie bilden die Regeln. Das Lexikon, bzw. die Lexika oder Ressourcen, bilden die Grundlage für die Regeln. In der Abbildung 8 werden externe Listen (z.B. *first name form list*) und Listen, welche so kurz sind, dass sie direkt im Algorithmus aufgeführt werden können (z.B. *salutation name part*), als Ressource verwendet. Der Algorithmus besagt, wie ein *vollständiger Name* in einem Text erkannt wird[31]. Mehr zu Ressourcen und Lexika ist im Kapitel I 4. „Ressourcen“ zu finden.

[31] Dies ist ein Auszug des Pseudocodes, welchen ich für dieses Projekt erstellt habe. Der Algorithmus beschreibt, wie ein vollständiger Personenname gefunden werden kann. Ein *vollständiger Personenname* besteht aus einem Vornamen oder Initial, einem Mittelinitial oder einem Teil des Nachnamens und dem Nachnamen selbst (vgl. Kapitel III 4.1. „Erkennung von Personennamen“). Der vollständige Pseudocode befindet sich im Anhang.

```
## Bsp.: Vincent van Gogh

recog_full_pers_name
IF
W1 = first name from list OR hyphenate word (first name)
AND W2 = middle initial OR salutation name part (von, van, de)
AND W3 = capitalized word
AND W4 /= firma-trigger-word
        THEN {save FAMILYNAME (W3)
                AND mark PERS (W1, W2, W3)
                AND mark PERSNAME (W1-W3)}
ELSIF
W1 = first name from list OR hyphenate word (first name)
AND W2 = capitalized word
AND W3 /= firma-trigger-word
        THEN {save FAMILYNAME (W2)
                AND mark PERS (W1, W2)
                AND mark PERSNAME (W1-W2)}
```

Abbildung 8

2.3. Bewertung von Eigennamenerkennung

Bei der Eigennamenerkennung werden gleich wie bei IR-Systemen (vgl. Kapitel I 1.5. „Bewertung von IR-Systemen“) Ausbeute und Präzision zur Evaluation verwendet. Die Bedeutung der Begriffe ist folgende:

Ausbeute:

$$\frac{\text{Zahl der gefundenen Eigennamen in einem Dokument}}{\text{Zahl aller Eigennamen in einem Dokument (Schlüssel)}}$$

Formel 6

Präzision:

$$\frac{\text{Zahl der korrekt gefundenen Eigennamen in einem Dokument}}{\text{Zahl aller Eigennamen in einem Dokument (Schlüssel)}}$$

Formel 7

Präzision und Ausbeute hängen in gleicher Weise wie bei den IR-Systemen von einander ab. Eine hohe Ausbeute hat meistens eine geringere Präzision zur Folge und umgekehrt. Ziel eines guten Systems zur Eigennamenerkennung ist, eine möglichst hohe Ausbeute bei möglichst hoher Präzision zu erreichen.

Ein Wert, der Präzision und Ausbeute gemeinsam berücksichtigt, ist der F-Wert (engl. f-score). Er wird wie folgt berechnet:

F-Wert:

$$\frac{(\beta^2 + 1)\ \text{Präzision} * \text{Ausbeute}}{\beta^2\ \text{Präzision} + \text{Ausbeute}}$$

Formel 8

Wenn Präzision und Ausbeute gleich gewichtet sein sollen, ist β eins. Falls die Präzision in der Evaluation stärker gewichtet werden soll, wird β grösser 1 gewählt, wenn der Ausbeute der Vorzug gegeben wird, wird β kleiner 1 gewählt. Wenn bei einer Evaluation beim F-Wert nichts weiter vermerkt ist, so wird in der Regel weder Ausbeute noch Präzision der Vorzug gegeben.

Ausbeute, Präzision und F-Wert werden gleich wie im IR berechnet. Der Unterschied besteht darin, dass beim Berechnen nicht Dokumente gezählt werden, sondern Wörter oder Wortkombinationen und die Entscheidungen relevant oder nicht durch Entscheidungen wie ‚zugehörig zu einer Kategorie' oder nicht ersetzt werden (vgl. Kapitel I 1.5. „Bewertung von IR-Systemen").

Die Güte eines Systems wird mit Präzision, Ausbeute oder F-Wert evaluiert. Über diese Zahlen ist ein Vergleich verschiedener Systeme zur Erkennung von Eigennamen möglich, unter der Voraussetzung, dass dieselben Grundbedingungen (gleicher Textkorpus, gleiche Anfragen) gelten.

3. Projektbeschreibung

3.1. Ziel

Folgende Ziele sollten in diesem Projekt bei der Firma *Eurospider* erreicht werden:

1. Erarbeitung der Grundlagen, um Eigennamenerkennung zu realisieren[32].
2. Entwicklung von Konzepten in der firmeneigenen Konzeptsensorsprache, um Erkennung von Personennamen, geographischen Bezeichnungen und Firmennamen in deutschsprachigen Texten zu ermöglichen. Bei den Texten handelt es sich um Nachrichten. Allerdings sollte die Erkennung nicht auf eine bestimmte Textsorte beschränkt sein, sondern möglichst allgemein gehalten werden.
3. Weiterentwicklung der Konzepte zur Eigennamenerkennung in deutschsprachigen Texten (vgl. zweites Ziel), so dass neue Konzepte entstehen, welche Eigennamenerkennung sprachspezifisch in englischen Texten erlauben.

[32] Vgl. Kapitel I 3.2. „Grundlagen", S. I-24.

4. Evaluation aller Konzepte anhand eines deutschen bzw. englischen Textkorpus. Bei der Evaluation der Konzepte zur Eigennamenerkennung sollte eine möglichst hohe Ausbeute (engl. recall) mit einer möglichst hohen Präzision (engl. precision) erreicht werden.

3.2. Grundlagen

Als Grundlage für dieses Projekt dienen Martin Volks Habilitationsschrift [Volk, 2002] und von ihm entsprechend entwickelte Algorithmen zur Eigennamenerkennung (Personennamen, geographische Bezeichnungen und Firmennamen) in deutschen Texten. Die Algorithmen sind von Volk ursprünglich in der Programmiersprache *Perl* implementiert worden und müssen in eine programmiersprachenunabhängige Form (Pseudocode) umgesetzt werden. Dieser Pseudocode wird an die Konzeptsensorsprache von *Eurospider* angepasst und erweitert. Diese Erweiterungen sind nötig, weil die ursprünglichen Algorithmen zu spezifische Muster abhängig vom Computer-Zeitung*s*-Korpus (Computer-Zeitung 1993-1995 und 1997) beschreiben und folglich Eigennamen auf Grund einer Umgebung (Kontext) falsch erkennen.

Volks Algorithmen bauen einerseits auf Lernmechanismen auf (beispielsweise Erlernen eines Nachnamens auf Grund eines bekannten Vornamens), andererseits auf Ressourcen (z.B. eine Liste von Vornamen). Neben den Algorithmen dienen mir diese Ressourcen als zweite wichtige Grundlage. Volks Ressourcen enthalten mehrheitlich eine beschränkte Anzahl von Einträge und sind von mir entsprechend erweitert worden.[33]

3.3. Die Firma *Eurospider*

Das vorliegende Projekt ist bei der Firma *Eurospider* entstanden, die Eigennamenerkennung in einer späteren Phase in ihr bestehendes Retrieval System integrieren möchte.

Die Firma wurde im Jahre 1995 als Spin-off-Unternehmen der ETH Zürich von Professor Dr. Peter Schäuble gegründet. Es handelt sich um eine Kleinfirma mit ca. 20 Mitarbeitern, die ihren Sitz in Zürich hat. Die Software-Firma ist auf IR-Methoden spezialisiert und versucht insbesondere, sprachübergreifende Lösungen (Informationssuche in mehrsprachigen Dokumentenkollektionen) in diesem Gebiet zu entwickeln.

Eurospider verwendet neben einfachen, grundsätzlichen IR-Methoden (z.B. Wortsegmentation oder boole'sche Suche) auch ausgereifte und komplizierte Prozesse (z.B. Konzepterkennung oder sprachübergreifendes Retrieval). In Abbildung 9 platziert sich die Firma nach eigener Einschätzung.

[33] Genaueres dazu in Kapitel III 3. „Ressourcen für die Erkennung von Entitäten in deutschsprachigen Texten" und Kapitel III 5. „Ressourcen für die Erkennung von Entitäten in englischen Texten".

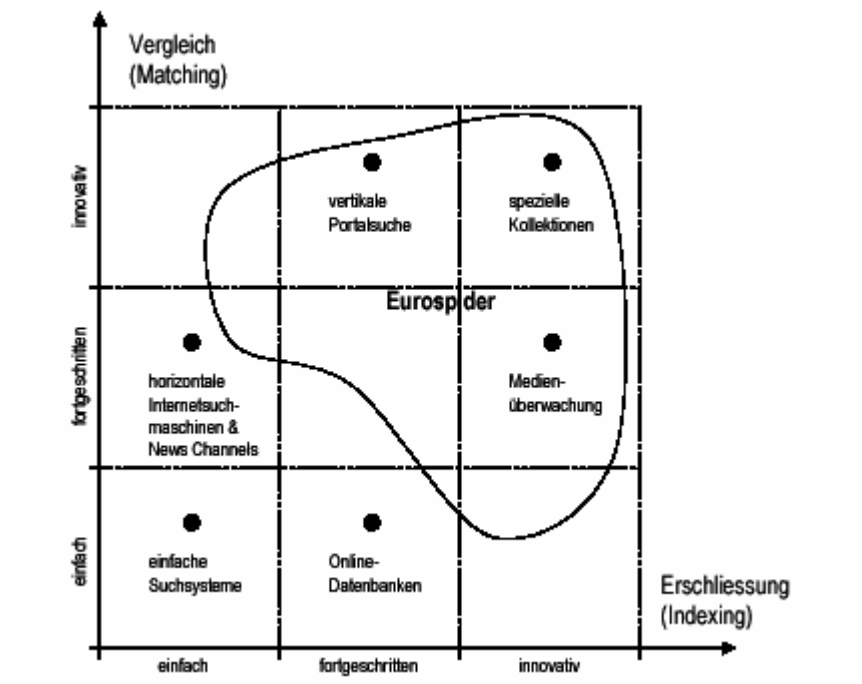


Abbildung 9

(Vgl. [Schäuble und Braschler, 2002])

Indexieren (engl. indexing) bedeutet Erschliessung von Dokumenten (manchmal auch als Informationsobjekten bezeichnet). Unter Erschliessung ist die Reduktion eines Dokumentes auf relevante Wörter (z.B. durch entfernen von Stoppwörtern wie Artikel oder Präpositionen), Normalisierung dieser Wörter (z.B. aus Pluralformen Singularformen bilden) sowie Anreicherung mit neuen themenspezifischen Informationen (z.B. eine Person zusätzlich als Eigenname markieren) zu verstehen. Indexieren sind verschiedene Vorbereitungsschritte, die mit einem Dokumente oder einer Dokumentensammlung sowie mit der Anfrage ausgeführt werden. Der Suchprozess (engl. matching) beinhaltet den Vergleich von (indexierter) Anfrage und Indextermen.

Zu den einfachen Erschliessungsmethoden gehören:

1. Konversion von Dokumenten- und Dateiformaten
   Office-Formate wie Word, Excel etc sowie Austauschformate wie HTML, SGML etc. sollen in eine für das IR-System geeignete Form konvertiert werden.
2. Konversion von Zeichensätzen
   Information in Textform, welche in einem bestimmten Codierungssystem (ASCII, ISO Latin etc.) geschrieben ist, muss in ein internes Format konvertiert werden.
3. Wortsegmentierung
   Wörter müssen erkannt und als einzelne Elemente aus dem Text extrahiert werden.

4. Entfernen von Stoppwörtern (vgl. Definition *Indexieren* im gleichen Kapitel weiter oben).
5. Extraktion derjenigen Teile, welche für die Informationssuche relevant sind
   Die Dokumente werden auf ‚relevante' Teile reduziert. Text (bzw. Code), der lediglich Formatierungshinweise enthält, wird entfernt.

Die einfachen Erschliessungsmethoden sind Grundlage für weitere Erschliessungsmethoden und sollten in jedem IR-System realisiert werden.

Zu den fortgeschrittenen Erschliessungsmethoden zählen:

1. Wortnormalisierung
   Diakritischen Zeichen (è, ç etc.) werden in eine allgemeine Form (e, c etc.) umgewandelt bzw. historische Buchstabenformen (z.B. ä, β etc.) ausgeschrieben (ae, ss, etc.).

   Beispiel 2: M**ü**ller → M**ue**ller

2. Kompositazerlegung
   Zusammengesetzte Nomen werden in ihre Einzelteile zerlegt.

   Beispiel 3: *Eigennamenerkennung* → *Eigennamen*, *Erkennung*

3. N-Gramme
   N-Gramme ermöglichen einen fehlertoleranten Vergleich, indem dem System unbekannte Wörter in kleinere Einheiten zerlegt werden und ein bekanntes Wort gesucht wird, das möglichst viele übereinstimmende Einheiten enthält. In der Regel handelt es sich bei den N-Grammen um Trigramme[34]. Die Trigramme werden nicht nur verglichen, sondern zusätzlich entsprechend der Wahrscheinlichkeit, dass die Buchstabenfolge in einer gewissen Sprache vorkommt, gewichtet. Im Deutschen gibt es beispielsweise Buchstabenfolgen, die oft vorkommen (z.B. *deut**sch***), andere die nie (oder sehr selten) auftreten (z.B. italienisch: *pi**cc**olo*). Häufigen Buchstabenfolgen werden entsprechend mit einer höheren Wahrscheinlichkeit gewichtet. Im folgenden Beispiel ist die Gewichtung der Trigramme nicht berücksichtigt.

   Beispiel 4:

   Eigename (dem System unbekannt, weil es einen Schreifehler enthält)
   Trigrammanalyse:
   Eig ige gen ena **nam ame**
   Trigrammanalyse von *Eigenname* (ohne Schreibfehler):
   Eig ige gen enn nna **nam ame**
   → 5 Trigramme von 6 stimmen überein. *Eigename* wird mit *Eigenname* ersetzt.

[34] Die unbekannten Wörter werden jeweils in Dreiereinheiten zerlegt.

4. Fuzzy Matching

   Ähnlich tönende oder auch ähnlich geschriebene Wörter werden bei einer Anfrage gefunden. Dies erlaubt das Auffinden von Dokumenten, welche eine andere oder inkorrekte Schreibweise eines Suchbegriffs (Begriff, der in einer Anfrage gestellt wird) enthalten.

   Beispiel 5:

   Zu der Anfrage *Hannover* werden auch Dokumente gefunden, welche die falsche Schreibweise *Hanover* enthalten

   Fuzzi Matching hat eine ähnliche Funktion wie N-Gramme, kann jedoch sprachunabhängig[35] eingesetzt werden. Die Erweiterung der Anfrage um zusätzliche Varianten mit ähnlicher Schreibweise findet im IR-System *Relevancy* von *Eurospider* während des Indexierungsprozesses statt. Deshalb zählt Fuzzi Matching zu den Erschliessungsmethoden. Eine Ansiedlung bei den Vergleichsmethoden wäre allerdings ebenfalls möglich.

5. Sprachidentifikation

   Über statistische Methoden kann berechnet werden, ob und welcher implementierten Sprache ein Dokument, eine Phrase oder ein Wort (falls es genügend lang ist) angehört.

6. Statistische Textkategorisierung

   Um das Auffinden von Information zu erleichtern, werden zunehmend Dokumente möglichst automatisch in Kategorien eingepasst, welche hierarchisch gegliedert sein können. Mit Hilfe von statistische Verfahren (wie z.B. Clustering[36]) ist eine automatische Kategorisierung.

7. Strukturen erkennen

   Interne Strukturen (z.B. TITEL, VORNAME, NACHNAME → Personenname) sollen erkannt und ausgewertet werden, um später einen gezielten Zugriff auf die Informationen zu erlauben.

Um ein IR-System zu entwickeln, das hohe Präzisions- und Ausbeutewerte[37] erreicht, gehören neben den einfachen Erschliessungsschritten diese Methoden zu den Grundlagen. *Eurospider* unterstützt die fortgeschrittenen Erschliessungsmethoden mehrheitlich.

---

[35] Sprachunabhängig ist in dem Sinn gemeint, dass kein spezifisches Sprachwissen (wie z.B. häufige Buchstabenfolgen einer Sprache) benötigt wird, um die Methode anzuwenden.

[36] ‚Clustering' bedeutet, dass Dokumente in bestimmte Gruppen (engl. clusters) eingeteilt werden. Automatisch kann man über Ähnlichkeitsberechnung der Dokumente Gruppierungen erstellen.

[37] Vgl. Kapitel I 1.5. „Bewertung von IR-Systemen".

Zu den „innovativen“[38] Erschliessungsmethoden zählen:

1. Anwendung von Konzepten

   Konzepte sind Regeln, welche Wörter abhängig von ihrem Kontext analysieren und somit eine genauere Indexierung ermöglichen. *Eurospider* hat zur Erkennung von Konzepten eine eigene Programmiersprache entwickelt[39].
2. Phrasenerkennung

   Phrasen, Kombinationen aus mehreren Wörtern, die als Ganzes eine spezifische Bedeutung tragen, werden als Einheit erkannt und weiterverarbeitet.
3. Eigennamenerkennung

   (Vgl. Kapitel I 3.4. „Eigennamenerkennung für *Eurospider*“)

Das IR-System von *Eurospider* unterstützt diese Methoden nur teilweise. Eigennamenerkennung existiert bis jetzt nur als ad-hoc-Lösung (Nachschauen in Listen mit vielen Eigennamen, wobei keine linguistischen Analysen berücksichtigt werden) und soll mit diesem Projekt fundiert untersucht und als Prototyp implementiert werden.

Universitäten und Laboratorien versuchen mehrheitlich Methoden zu verwenden, die über die fortschrittlichen Ansätze hinausgehen. Eine hohe Präzision der Ergebnisse zu Erreichen ist oft ein wichtiges Ziel. Kommerzielle Systeme verwenden meist minimale Methoden, um maximale Information zu extrahieren. Eine hohe Ausbeute ist wichtiger als eine grosse Präzision[40] Hohe Präzision oder hohe Ausbeute kann bereits durch fortschrittliche Erschliessungsmethoden erreicht werden. Deshalb bewegen sich kommerzielle Systeme, nach meiner Erfahrung, seltener als universitäre über den fortschrittlichen Bereich hinaus.

Methoden, welche beim Vergleich von Anfrage und Dokument (engl. matching) angewendet werden, - dies entspricht der senkrechten Achse in Abbildung 9 - sind folgende:

Einfache Vergleichsmethoden:

1. Boole’sches Retrieval

   Suchterme werden mit Boole’schen Operatoren wie AND, OR und NOT verbunden. (vgl. dazu *boole’ sches Modell*“ Kapitel I 1.4. „Ziele und Methoden des IR“)
2. Coordination Level Matching

   Eine Rangliste wird über einzelne Abschnitte erstellt, welche gemäss der Häufigkeit der gefundenen Suchbegriffe geordnet sind. Bei einer 4-Wort-Anfrage erscheinen unabhängig von anderen Gewichtungsmethoden (z.B. anhand der entgegengesetzte Do-

[38] Das Wort innovativ ist von der Firma *Eurospider* geschaffen worden. Es umschliesst alle Methoden, die über die einfachen und fortgeschrittenen Methoden hinausführen.
[39] Mehr zu der Konzeptsensorsprache von *Eurospider* ist im Kapitel III 2. „Konzeptsensorsprache“ zu finden.
[40] Diese Aussage stammt aus der Firma *Eurospider*.

kument Frequenz = IDF[41]) zuerst diejenigen Dokumente, welche alle 4 Begriffe enthalten, dann diejenigen, welche 3 Begriffe enthalten etc.

Fortgeschrittene Vergleichsmethoden

1. Berücksichtigung von Wahrscheinlichkeiten
   (vgl. dazu *pobabilistisches Modell,* Kapitel I 1.4. „Ziele und Methoden des IR“)
2. Vergleich auf Grund von Regeln
   Regeln werden erstellt, welche einen Vergleich zwischen Anfrage und Dokumenten ermöglichen und die Erstellung einer Rangliste von Dokumenten beeinflussen, welche zur Anfrage relevant sind.
3. Relevanzrückkopplung
   Der Benutzer kann die Suchresultate entsprechend seinen Interessen bezüglich Relevanz bewerten. Das System verfeinert die Anfrage auf Grund des Feedbacks.
4. Möglichkeiten zur Erweiterung/Verfeinerung von Anfragen
   Die Anfrage wird automatisch erweitert, um verwandte Begriffe (Synonyme, Hyperonyme, Hyponyme etc.) in den Vergleich miteinzubeziehen.
5. Einbindung von Metadaten
   Einschränkung der Dokumentenkollektion durch zusätzliche Angaben von Metadaten (Datum, Quelle etc.)
   Falls z.B. ein Datum angegeben wird, soll der Vergleich zwischen Anfrage und Dokument auf Dokumente eingeschränkt werden, welche zum gefragten Zeitpunkt erstellt worden sind. Solche Spezifizierungen funktionieren allerdings nur bei semistrukturierten Dokumenten (vgl. Kapitel I 1.4. „Ziele und Methoden des IR“), weil jene eine entsprechende Formatierung enthalten, die Metadaten kennzeichnen.
6. Einteilung der Daten/Dokumente in Subkollektionen
   Falls die Dokumentensammlung in eine nach Themen oder anderen Kriterien geordnete Hierarchie eingeteilt wurde, erhält der Benutzer zusätzlich die Möglichkeit, entsprechend eines Themas nur in einem Teil der Dokumentenkollektion zu suchen.
7. Automatische Elimination von Duplikaten (in einer Dokumentenkollektion)
   In einer Rangliste müssen Duplikate eliminiert werden. Dokumente, welche in einer Kollektion doppelt vorkommen, muss man folglich erkennen, um sie dann zu eliminieren. Dies geschieht über einen automatischen Prozess.

[41] Vgl. Kapitel I 2. „Eigennamenerkennung“.

8. Einschränkung des Zugriffs auf geschützte Informationen
   Falls in einer Firma ‚geheime' Informationen existieren, welche nicht jedem Benutzter zugänglich sein dürfen, müssen solche Informationen in einem IR-System entsprechend geschützt werden.

„Innovative"[42] Vergleichsmethoden

1. Sprachübergreifendes Retrieval (engl. cross language information retrieval)
   (Vgl. *Cross Language Retrieval,* Kapitel I 1.2. „Geschichte des IR")
2. Passagen Retrieval
   (Vgl. *sprachübergreifendes Retrieval* sowie *Passagen-Retrieval,* Kapitel I 1.2. „Geschichte des IR")

Ein Teil dieser Vergleichsmethoden werden von *Eurospider* in ihrem IR-System *Relevancy* verwendet. Je nach Kundenwunsch wird *Relevancy* angepasst und entsprechende Methoden kommen zur Anwendung oder werden versucht, neu zu implementieren[43].

3.4. Eigennamenerkennung für *Eurospider*

Wie bereits im vorherigen Teil dieser Arbeit erläutert wurde, ist Eigennamenerkennung eine wichtige Komponente im IR. Es wird versucht, Informationen aus Texten zu extrahieren. Bei den gesuchten Informationen handelt es sich oft um Entitäten (Eigennamen, entsprechend meiner Definition, sind Entitäten). Dies wird in der Studie von Paul Thompson und Christopher C. Dozier [Thompson und Dozier, 1999] belegt. Sie haben natürlichsprachliche Anfragen an eine Datenbank, welche verschiedene Nachrichtentexte enthält, über eine Periode von einigen Tagen im Jahre 1995 auf Vorkommen von Personen-, Firmennamen und anderen Entitäten untersucht. Dies waren die Resultate:

- 67,83% aller Anfragen an das *Wall St. Journal* enthielten Entitäten
  (36,23% Firmennamen, 13,57% Personennamen)
- 83,4% aller Anfragen an die *Los Angeles Times* enthielten Entitäten
  (18% Firmennamen, 38,2% Personennamen)
- 38,8% aller Anfragen an die *Washington Post* enthielten Entitäten
  (15% Firmennamen, 17,3% Personennamen)
- 91,6% aller Anfragen an *Allnews* enthielten Entitäten
  (34,65% Firmennamen, 29,3% Personennamen)

Die Zahlen werden von Thompson und Dozier nicht weiter ausgeführt. Die Unterschiede kommen vermutlich durch die verschiedenen Themenbereiche zustande, auf die sich die Zei-

[42] Vgl. Fussnote 38, S. 28.
[43] Mehr zu *Relevancy* in Kapitel III 1. „Eurospider Retrieval System ‚Relevancy'".

tungen jeweils spezialisiert haben. Eine Zeitung wie das *Wall St. Journal* publiziert Meldungen aus dem Wirtschafts- und Firmenbereich. Anfragen an diese Zeitung enthalten folglich oft Firmennamen, da Informationen zu den entsprechenden Firmen in dieser Zeitung vermutet werden. Wie die Zahlen zeigen, enthalten Anfragen an eine Zeitung, unabhängig davon, zu welchem Thema die Zeitung hauptsächlich publizieren, eine gewisse Anzahl von Eigennamen, so dass der Schluss gezogen werden kann, dass Eigennamen eine entsprechende Relevanz zukommt. Eine Erkennung und Zuordnung der Eigennamen zu einer bestimmten Klasse bereits in der Anfrage, könnte den gesamten IR-Prozess verbessern, da so Ambiguität (ob z.B. *Dennis Banks* ein Personenname oder Firmenname ist) verringert werden kann.

Da Eigennamen einen grossen Prozentsatz in Anfragen von IR-Systemen ausmachen (so wie Thomson und Dozier in ihrer Studie gezeigt haben) und in Textkollektionen ebenfalls häufig auftreten[44], ist es für ein IR-System wünschenswert, die Erkennung von Eigennamen oder zumindest einen Teil der Eigennamen in den Indexier- und Suchmechanismus zu integrieren.

Bei den folgenden drei Anwendungsbeispielen sind Eigennamen in einem IR-System von Bedeutung:

1. Kategorisierung

   Dokumente werden mit Vorteil in Kategorien wie Personen-, Länder- oder Firmennamen eingeteilt, um eine entsprechende Anfrage schnell über die eine oder andere Kategorie zu beantworten. Wenn z.B. ein Dokument über *Lukas Mühlemann* gefunden werden soll, so muss nur in der Kategorie *Personenname* gesucht werden.

2. Anfrage

   Bei der Anfrage können durch Eigennamen nicht nur spezifische Anfragen (*Lukas Mühlemann)*, sondern auch kategoriale Anfragen gestellt oder Verbindungen zwischen spezifischen und kategorialen Elementen gefunden werden. So würde beispielsweise eine mögliche Anfrage ans System lauten: „Suche jegliches Vorkommen der *Person Lukas Mühlemann* (spezifisch) in Verbindung mit irgendeiner *Firma* (kategorial)“.

3. Highlighting

   Wenn man in Dokumenten Eigennamen finden und markieren will, ist es wichtig, alle Vorkommen dieser Entität zu finden. Beispielsweise soll nicht nur der Personenname *Peter Müller* gefunden werden, sondern auch gekürzte Formen wie der Nachname *Müller* alleine. Im IR kann ein Dokument eine falsche Gewichtet erhalten[45], wenn an-

[44] Dies wird in Kapitel II 2. „Relevanz von Eigennamenerkennung“ genauer erläutert.

[45] Dokumente werden im IR auf Grund der Vorkommenshäufigkeit eines Suchbegriffs (ein Wort, das Teil der Anfrage ist) innerhalb eines Dokumentes gewichtet. Wenn somit der Name *Peter Müller* in einem Dokument dreimal gefunden wird, im andern fünfmal, so ist das zweite relevanter als das erste und erscheint in einer Rangliste entsprechend weiter oben. (Nicht alle IR-Systeme nehmen eine Gewichtung der Dokumente vor.).

dere Formen eines Eigennamens aus der Anfrage nicht gefunden werden. Durch einen solchen Fehler sinkt die Relevanz eines Dokumentes zu einer Anfrage drastisch und das Dokument erscheint unberechtigterweise im hinteren Teil einer Rangliste.

Neben der Erkennung von Eigennamenvarianten ist zusätzlich die Ambiguität gewisser Eigennamen zu berücksichtigen. Die Berufsbezeichnung *Müller* sollte nicht als Entität markiert werden, der Nachname *Müller* allerdings sehr wohl.

In der Retrievalsoftware von *Eurospider* ist Eigennamenerkennung bis jetzt noch nicht realisiert worden. Mit diesem Projekt sollten die Grundlagen geschaffen werden, dass Personennamen, geographische Bezeichnungen und Firmennamen über Konzeptsensoren[46] erkannt werden. Die Konzeptsensoren werden über die Konzeptsensorsprache, welche eigens von der Firma entwickelt worden ist, im IR-System *Relevancy* von *Eurospider* implementiert. Um eine vollständige Version des erweiterten IR-Systems zu erhalten, muss unter anderem die geographische Benutzeroberfläche (GUI) erweitern werden. Dies ist jedoch nicht mehr Teil meines Projektes. Es geht lediglich darum, eine Erkennung von gewissen Entitäten über die Konzeptsensoren zu ermöglichen.

3.5. Zeitrahmen

Da dieses Projekt in einer „wirtschaftlichen“ Umgebung und nicht in universitärem Umfeld realisiert worden ist, bestand eine entsprechende Zeitlimite. Innerhalb von 5 Monaten ist die Eigennamenerkennung in der Konzeptsensorsprache von mir implementiert worden. Dies bedeutet, dass die Zeit für eine ausführliche Evaluation und somit die Möglichkeit zur Verbesserung der Erkennung eingeschränkt ist. Deshalb sind Präzisions- und Ausbeutewerte um 80% durchaus befriedigend. Eine 100% Ausbeute bei 100% Präzision ist nicht erreichbar, weil immer Ambiguität zwischen verschiedenen Eigennamenkategorien oder zwischen Entitäten und Nomen vorhanden ist, die nicht vollständig aufzulösen ist. In diesen Fällen hängt es vom willkürlichen Entscheid des Annotators ab, wie die Nomen eingeordnet werden. Die Feststellung, dass eine 100% korrekte Kategorisierung durch Ambiguität verunmöglicht wird, ist ebenfalls von Appelt und Martin bezüglich ihres *TextPro*-Systems gemacht worden: „We believe that some of the categories of error […] represent a nearly irreducible source of error.“ (vgl. [Appelt und Martin, 1999] S. 3.)

Der gesetzte Zeitrahmen ist leicht überzogen worden, da verschiedene Probleme auftraten, welche im Voraus nicht genügend bedacht worden sind. Die Erkennung von Personennamen, geographischen Namen und Firmennamen ist am Ende jedoch gelungen.[47]

---

[46] Vgl. Kapitel III 2. „Konzeptsensorsprache“.
[47] Vgl. Kapitel IV „Evaluation“.

4. Ressourcen

4.1. Definition

In der Eigennamenerkennung spricht man von Ressourcen, wenn die Erkennung über Regeln und Lexika erfolgt. Das Lexikon ist in diesem Fall die Ressource. Entitäten, welche in Texten erkannt werden sollen, können in Form grosser Listen gespeichert werden. Das Lexikon besteht dann aus einer grossen Anzahl von Entitäten. Die einzelnen Einträge solcher Listen werden meist in Grundform oder in einer sonstigen Form gespeichert, auf welche sich die Entitäten reduzieren lassen (keine Vollformen). Geographische Eigennamen stehen z.B. in den Ressourcen im Nominativ, währenddem die Genitivform, welche in einem Dokument steht, trotz der anderen Flexionsendung als Ableitung vom Nominativ erkannt werden muss. Der Nominativ ist in diesem Fall die reduzierte Form.

Beispiel 6:

„Debitel, **Deutschland**s größter Mobilfunk-Service-Provider, hat Festnetz-ambitionen."
[Konradin-Verlag, 1998]
Ressourceneintrag: **Deutschland**

Deutschlands muss als flektierte Form des Nominativs *Deutschland* erkannt werden.

Listen mit geographischen Bezeichnungen (geographisches Namenverzeichnis) werden in der Fachliteratur mit dem englischen Wort *gazetteer* bezeichnet.[48] Solche geographischen Lexika sind nichts anderes als Teilressourcen, die ein bestimmtes Themengebiet abdecken. Nicht nur Listen von Entitäten fallen unter den Begriff *Ressource*, sondern auch Listen von Attributen. „Attribute charakterisieren Personen oder Sachverhalte hinsichtlich bestimmter Merkmale,…." [Bussmann, 2002]. Im Fall von Eigennamen sind Attribute (von mir *Trigger* genannt) Wörter, die meistens mit einer bestimmten Kategorie von Eigennamen (*Personenname*, *geographischer Name*, *Firmenname*) auftreten und somit ein Wort oder eine Wortfolge als zugehörig zu einer Eigennamenkategorie markieren.

Beispiel 7:

„Doch obwohl die Road-Pricing-Lösung der Bundesrepublik "eine besondere industriepolitische Chance" eröffnen könnte, "da es hier um einen weitgehend unbesetzten Weltmarkt geht", wie ***Professor*** *Gerhard Barth* von der *Daimler-Benz* ***AG*** betont, ist die Methode umstritten: …"
[Konradin-Verlag, 1998]

Trigger können zur Entität (vgl. Beispiel 8, *Gerhard*) gehören. Es gibt jedoch auch Fälle, in denen der Trigger nicht Teil der Entität ist (vgl. Beispiel 8, -*Tochter*).

[48] *Gazetteer* ist ursprünglich nur ein alphabetisches Ortsverzeichnis. In [Mikeev et al., 1999] wird das Wort aber auf Listen, welche jegliche Art von Entität (Personennamen, Firmennamen etc.) enthalten, ausgedehnt.

Beispiel 8:

„Gerhard *Sundt* wurde zum neuen Geschäftsführungsmitglied der *Telekom-* **Tochter** …."
[Konradin-Verlag, 1998]

Grundsätzlich sind Ressourcen eine Auflistung von Wörtern in verschiedensten Formen, welche als Teil von Regeln (Trigger, z.B. -*Tochter*) oder direkt (Eintrag einer vollständigen Entität, z.B. *New York*) eine Erkennung von möglichen Entitäten erlauben oder eine solche einschränken (z.B. Stoppwortlisten).

4.2. Verwendung von Ressourcen

Eine Eigennamenerkennung mit Regeln und Ressourcen zu realisieren, ist dann sinnvoll, wenn es sich bei den Ressourcen um eine konstante Liste von beschränkter Grösse handelt. Unter einer konstanten Liste ist folgendes zu verstehen:

1. Vollständigkeit; Häufigkeit

Eine Liste sollte zu einem gewissen Thema oder einer Kategorie (beispielsweise Vornamen) möglichst alle verschiedenen Vorkommen enthalten. Je genauer das Thema definiert ist, umso wahrscheinlicher ist die Vollständigkeit einer Liste. Wenn man beispielsweise die Vornamen auf den europäischen Sprachraum einschränkt, so handelt es sich um eine kleinere Liste, welche bezüglich Vollständigkeit innerhalb dieses neuen Themas weit besser überprüft werden kann.

Am Ende handelt es sich bei den Ressourcen jedoch meist um keine vollständigen Listen, da die Anzahl der zu erfassenden Begriffe je nach Thema meist zu gross ist. Man versucht zumindest die wichtigsten Vertreter eines Themas aufzulisten. Eine Ressource, die häufige Vorkommen zu einem Thema enthält, bringt bei der Eigennamenerkennung bessere Resultate, als Listen, die zwar vollständig sind, dafür aber riesig. Mikheev, Moens und Grover [Mikeev et al., 1999] haben für die MUC-7 Untersuchungen bezüglich Nützlichkeit und Grösse von Listen mit Entitäten (Personen-, Firmennamen, geographische Namen etc.) durchgeführt, aus welchen folgende Aussage entstand: „[…] it is sufficient to use relatively small gazetteers of well-known names, rather than large gazetteers of low-frequency names."[49]
([Mikeev et al., 1999], S. 1)

[49] Diese These habe ich ebenfalls in meinem Projekt bestätigt gefunden. Zur Erkennung von geographischen Entitäten in englischen Texten habe ich die Ressourcen im Vergleich zur Erkennung in deutschen Texten vergrössert (Vgl. Kapitel I 4.4. „Grössen der Ressourcen"). Bei den ersten Tests hat sich herausgestellt, dass viele Fehler wegen Mehrdeutigkeit der Einträge (z.B. *Bush* als Person, *Bush* als Stadt und *Bush* als Busch) produziert worden sind. In diesem Fall waren weitere Regeln zur Einschränkung der Ressourcen nötig (z.B. falls vor *Bush* ein Vorname steht, dann handelt es sich um einen Personennamen etc.).

2. Beständigkeit

Ressourcenlisten zu einem Thema zu erstellen, ist erst dann sinnvoll, wenn nicht andauernd Neubildungen von Wörtern in diesem Bereich kreiert werden. Firmen werden beispielsweise ständig neu gebildet. Jede neu gebildete Firma gibt sich einen Namen, der möglichst noch nicht auf dem Markt vertreten ist. Eine Liste von Firmennamen zu bilden, würde eine ständige Erweiterung dieser Liste bedeuten. Dies ist für Ressourcen jedoch nicht wünschenswert. Sie sollen mindestens über einige Zeit ihre Aktualität und Korrektheit ohne ständige Anpassungen behalten. Abgesehen von politisch bedingten Änderungen bringen Städte- oder Ländernamen diese Eigenschaft mit sich und sind somit zur Erstellung von Listen geeignet.

3. Thematische Abgrenzung

Eine klare Klassifikation der Ressourcen ist notwendig, damit keine Redundanz entsteht. Eine Klassifikation setzt jedoch eine mögliche Abgrenzung von Begriffen voraus. Ressourcen sollten folglich thematisch abgegrenzt werden. Es ist sinnvoll, geographische Ressourcen in Städtenamen, Nationennamen, Kantonsnamen etc. zu unterteilen. Eine thematische Abgrenzung, falls ein Thema spezifisch gewählt wird (beispielsweise *Schweizer Kantone*), erlaubt wiederum eine bessere Kontrolle der Vollständigkeit.

Die Grösse einer Ressource ist nur durch die Speicherkapazität des Computers beschränkt. Bei grossen Mengen jedoch wird die Kontrolle der Vollständigkeit schwierig, da der Ersteller der Ressource die Übersicht verliert. Redundanz kann nicht mehr ausgeschlossen werden.[50] Ausserdem müssen beim Anwenden von sehr grossen Listen zur Eigennamenerkennung erneut Regeln gebildet werden, welche garantieren, dass das Nomen, welches in einem Text dem in der Liste stehenden Wort entspricht, tatsächlich eine Entität ist. Wenn zum Beispiel *Halle* in einer Liste mit Städtenamen aufgeführt ist, so muss zusätzlich über eine Regel festgestellt werden, ob *Halle* im Text einen Ort in einem Gebäude beschreibt und es sich folglich um keinen Eigennamen handelt, oder ob die Stadt in Deutschland gemeint ist. Solche Desambiguierungsregeln müssen zwar für kleine Listen ebenfalls erstellt werden, sind aber in geringerer Zahl nötig.

In diesem Projekt habe ich festgestellt, dass grössere Ressourcen einen Anstieg solcher Ambiguitäten mit sich bringen und es viel Aufwand braucht, um sie korrekt und vollständig (eher unwahrscheinlich) aufzulösen. Folglich ist es sinnvoll, die Grösse der Ressourcen zu beschränken.

---

[50] Die Problematik der Redundanz könnte über eine Datenbank gelöst werden. Beim Eintragen von neuen Begriffen in eine Datenbank wird automatisch überprüft, ob der Begriff bereits vorhanden ist. Fehlende Wörter können jedoch nicht automatisch von der Datenbank gefunden werden. Somit kann Vollständigkeit nicht garantiert werden.

Wenn mehr als eine Liste zur Eigennamenerkennung verwendet wird, sollte sichergestellt werden, dass Einträge nicht in mehr als einer Liste auftauchen. Städtenamen können beispielsweise auch als Kantonsbezeichnung auftauchen (Stadt *Zürich* vs. Kanton *Zürich*). Wenn bei der Eigennamenerkennung eine solche Ambiguität nicht aufgelöst wird, sollte in den Ressourcen darauf geachtet werden, dass der Eintrag (z.B. *Zürich*) nur an einem Ort auftaucht, da sonst redundante Information entstehen würde. Um sicher doppelte Einträge in Ressourcen zu vermeiden, sollte eine Reihenfolge innerhalb der Ressource erstellt werden. Im Fall dieses Projektes habe ich die Reihenfolge, in der die Einträge der geographischen Ressourcen geordnet sind, wie folgt gewählt: Städte > Nationen > Kantone/Staaten/Länder[51] > Regionen/Inseln. Diese Reihenfolge habe auf Grund der Auftretenshäufigkeit im Text sowie meiner eigenen Wichtigkeitseinschätzung festgelegt. Städte- Dorfnamen sowie Nationsnamen treten meiner Meinung nach in Nachrichtentexten häufiger auf als Kantonsnamen oder sonstige Regionsbezeichnungen und erhalten eine grössere Gewichtung. Ausserdem sind Städte- und Dorfnamen mit konkreteren Objekten verbunden als Kantons- oder Regionsnamen. Beim Wort *New York* denkt man, meiner Meinung nach, zuerst an die Stadt und erst in zweiter Linie an den Staat, der nach seiner grossen Hauptstadt benannt wurde. Deshalb kommt Städtenamen eine wichtigere Bedeutung als Kantons- bzw. Staatsnamen zu. Entsprechend habe ich die Reihenfolge meiner Ressourcen gewählt.

Reihenfolgen müssen nur dann bestimmt werden, falls die Listen thematisch sehr nahe beieinander liegen (z.B. geographische Eigennamen) und es deshalb zu Überschneidungen kommen könnte.

4.3. Quellen für Ressourcen

Es gibt drei Möglichkeiten, um an Ressourcen für Eigenamenerkennung zu gelangen.

1. Listen aus dem Internet zusammenstellen

Das Internet ist eine bedeutende Quelle für Informationen jeglicher Art. So kann man nach intensiver Suche Aufstellungen von Städtenamen, Anredeformen bei Personennamen und Ähnliches finden. Geeignet sind Systeme mit Kategorien zu einem bestimmten Thema, in welche Nachrichten oder Internetadressen eingeteilt werden. Ein gutes Beispiel ist das Suchsystem *Yahoo*[52], welches das Thema *Reisen* in Kategorien wie *Länder*, *Städte*, *Regionen* etc. einteilt. Mit Vorteil sollten solche Listen kopiert werden können, damit man sie nicht einzeln von Hand eintippen muss. Bei grossen Datenmengen, welche bei guten und vollständigen

[51] Mit Ländern sind hier, entsprechend meiner Definition, nicht Nationen gemeint, sondern regionale Unterteilungen innerhalb von Nationen, beispielsweise die Bundesländer in Deutschland oder Österreich.
[52] [Yahoo, http://www.yahoo.com].

Ressourcen schnell erreicht werden, ist eine Einzelaufnahme sehr aufwendig. Man würde zu viel Zeit benötigen, so dass die Herstellungskosten zu hoch wären.

Punkte, welche bei der Verwendung des Internets als Quelle beachtet werden müssen, sind folgende:

1. Schreibweisen

   Falls man Listen über Kopieren der Daten erstellt, muss die Schreibweise vereinheitlicht und müssen unsichtbare Steuerzeichen wie Zeilenumbrüche entfernt werden. Um doppelte Einträge aus Listen zu erkennen und entsprechend löschen zu können, dürfen sich die verdeckte sowie die sichtbare Schreibweise (beispielsweise Umlautschreibung oder Zeilenumbrüche) nicht unterscheiden.

2. Orthographie

   Man muss definieren, in welcher Schreibweise diakritische Zeichen in den Listen umgesetzt werden sollen (ob für *ü ue* geschrieben wird oder umgekehrt). Diese neu festgesetzte Schreibweise ist in allen Ressourcen einheitlich zu verwenden.

3. Redundanz vermeiden

   Falls Einträge doppelt vorhanden sind, muss einer gelöscht werden, da sonst die Ressourcen unnötigerweise wachsen. Falls ein Eintrag später aus den Ressourcen gelöscht oder geändert werden muss, sollte eine Bearbeitung an einer Stelle genügen. Vermeiden von doppelten Einträgen hilft somit, Übersichtlichkeit und Vollständigkeit der Ressourcen zu wahren.

4. Urheberrechte

   Vieles im Internet steht kostenlos zur Verfügung. Man sollte jedoch gerade beim Kopieren von grossen Datenmengen die Urheberrechte in Erfahrung bringen und falls nötig abklären, was unter welchen Umständen wie verwendet werden darf. Urheberrechte gelten auch für das Internet und müssen entsprechend beachtet werden.

Falls man im Internet für Ressourcenerstellung fündig wird und all die oben aufgezählten Punkte berücksichtigt, ist das Internet eine ergiebige Adresse zur Erweiterung von neuen Einträgen einer entsprechenden Liste.

Die geographischen Ressourcen (Stadt-, Länder-, Regions-, Inselnnamen etc.) sind in diesem Projekt grösstenteils von der Internetadresse http://www.yahoo.com und den länderspezifischen Homepages (http://www.yahoo.de, http://www.yahoo.ca etc.) desgleichen Anbieters gesammelt worden.

2. Extraktion von Entitäten, welche in einer speziellen Textsorte in einem starren Muster vorkommen.

Es gibt Textsorten, welche nach bestimmten Mustern aufgebaut sind und somit eine Struktur aufweisen. Bei Nachrichten ist es z.B. häufig der Fall, dass der Ort des Geschehens als erstes im Text erscheint:

> **Beispiel 9:**
>
> **„München** (pg) - Der Online-Anbieter Compuserve verpaßte seinen Kunden eine saftige Gebührenerhöhung, die in der ersten Mitteilung des Unternehmens als Umstellung auf die jeweilige Landeswährung verpackt wurde."
> [Konradin-Verlag, 1998]

Wenn Wörter immer in der gleichen Umgebung auftauchen (vgl. Beispiel 9), dann können sie auf Grund des Musters, in welchem sie auftreten, extrahiert werden. Mit Muster sind Dinge wie Kommasetzung, Abschnittsbeginn oder ähnliche Textsegmentierungsmittel gemeint. *Perl* ist eine geeignete Programmiersprache, um entsprechende Wörter zu extrahieren.

Bedingungen zur Erstellung von Ressourcen über Extraktion von Wörtern in bestimmten Erscheinungsmustern sind folgende:

1. Strukturierung

   Der Text, aus dem die Wörter gewonnen werden, muss das Muster, in welchem die Wörter erscheinen, konstant über die ganze Länge aufweisen.

2. Frequenz

   Wörter, welche nur einmal in einem grossen Korpus (Texte aus denen die Wörter extrahiert werden) vorkommen, sollten überprüft werden, da es sich um Kandidaten mit Rechtschreibefehler handelt. Wenn Wörter mehr als einmal falsch geschrieben werden, ist es denkbar, sie trotz des Fehlers in die Ressource aufzunehmen, da sie vermutlich oft entsprechend falsch geschrieben und trotzdem als Entität erkannt werden sollten. Die Erkennung wird so robuster.

Eine sorgfältige Kontrolle der extrahierten Ressourcen ist notwendig, um falsche Einträge zu vermeiden, da die Muster, welche das zu extrahierende Wort umgeben, auch in anderen Wortumgebungen auftauchen können und das extrahierte Wort in diesem Fall keine Entität ist.

Nachrichtentexte eignen sich insbesondere zur Extraktion, da sie formal starke Strukturen aufweisen (Autorenangabe, Ort des Geschehens, Vorspann etc.). Zur Extraktion von Städtenamen haben mir Texte von SDA (***S**chweizerische **D**epeschen**a**gentur*)[53] gedient.

[53] *Eurospider* besitzt eine grosse Sammlung von SDA-Texten zum firmeninternen Gebrauch.

3. Einkaufen von bereits vorhandenen Listen

Wenn einer Firma die Zeit zum Erstellen von entsprechend grossen Ressourcen fehlt, hat sie die Möglichkeit, Ressourcen zu kaufen, welche von anderen Anbietern erstellt worden sind. Indem die Firma beträchtliche Arbeitszeit (das Erstellen von grossen und guten Ressourcen braucht mehrere Tage bis Monate Arbeitsaufwand) spart und die Ressourcen in einem Programm entsprechend häufig verwendet, werden die ausgegebenen Gelder wieder kompensiert. In diesem Projekt wurde die Vornamenliste vom Institut für Informatik an der Universität Zürich zusammengestellt und von *Eurospider* für die Eigennamenerkennung erworben.

4.4. Grössen der Ressourcen

In der folgenden Aufstellung sind die Grössen der in diesem Projekt verwendeten Ressourcen angegeben. Eine detaillierte Ausführung der verwendeten Ressourcen ist in Kapitel III „Implementation“ unter 3. „Ressourcen für die Erkennung von Entitäten in deutschsprachigen Texten“ und 5. „Ressourcen für die Erkennung von Entitäten in englischen Texten“ nachzulesen.

DEUTSCH

	Grösse	**Beispiel**
Ressource		
PERS insgesamt*	**20500**	
Anreden	11	Ing. Professor, Herr
Vornamen	20440 (3242 Genitivformen)	Anna, Hans-Peter, Thian Seong
Titel	8	van, vom, d’
Affix	0	
Stoppwörter	41	Tor, August, Mile
GEO insgesamt*	**18552**	
Städte	15192 (1839 Genitivformen)	Zuerich, Frankfurt/M., Rust am Neusiedlersee
Nationen	544 (186 Genitivformen)	Schweiz, Deutschlands, Vereinigte Staaten von Amerika
Kantone/Länder/Staaten	77	Aargau, South Dakota, Demokratische Republik Kongo
Regionen/Inseln	75	Allgaeu, Skandinavische Laender, Nord-Europa
Kontinente	11 (5 Genitivformen)	Europa, Asiens, Australien
Abkürzungen	187	AUT, VS
Adjektive	2466	afrikanisch, New Yorker, baden-wuerttembergisch

FA insgesamt*	**110**	
Trigger vorne	60	Firma, Netzwerkspezialist
Trigger hinten	22	AG, Partners
Stoppwörter vorne	22	Der, Spezialist
Stoppwörter hinten	3	Trade, Unternehmen
Bindestrichsuffixe	3	-tochter, -chef
Filter für FA insgesamt	**46431**	
Abkürzungen	714	usw. Abb. Apoth.
Lexikon	45660	ABRECHNEN, youth
Spezielle Liste	49	WWW, ABS, Ram
Stoppwörter	10	Sun, Shell

* PERS = Personenname, GEO = geographische Entität, FA = Firmennamen

Tabelle 2

ENGLISCH

	Grösse	**Beispiel**
Ressource		
PERS insgesammt	**17306**	
Anreden	45	Mr., Baroness, Herr
Vornamen	17172	Abby, Al'for, Odd Bijorn
Titel	12	di, MC, van
Affix	2	Jr. Sr.
Stoppwörter	75	As, Love
GEO insgesamt	**29128**	
Städte	28111	Hope, Sainte-Roch-de-L'Achigan, St. Lucia
Nationen	336	US, Cote d'Ivoire, Saudi-Arabien
Kantone/Länder/Staaten	53	Newfoundland, Schleswig-Holstein, Nova Scotia
Regionen/Inseln	116	Caribbean, Allgaeu, Central-Asia
Kontinente	6	Africa, Europe, Australia
Abkürzungen	200	WA, Calif., Va
Adjektive	0	
FA insgesamt	**39**	
Trigger vorne	12	company, provider, firm
Trigger hinten	26	Holding, Cos., AG
Stoppwörter vorne	1	The
Stoppwörter hinten	0	
Bindestrichaffixe	0	

Filter für FA insgesamt	**46431**	
Abkürzungen	714	usw. Abb. Apoth.
Lexikon	45660	ABRECHNENE, youth
Spezielle Liste	49	WWW, ABS, Ram
Stoppwörter	10	Sun, Shell

* PERS = Personenname, GEO = geographische Entität, FA = Firmennamen

Tabelle 3

Grössen von Ressourcen, welche zur Eigennamenerkennung verwendet werden, sind sehr verschieden. Ein grosses Problem für Systeme, welche Eigennamenerkennung über grosse Ressourcen betreiben, ist die Beschaffung dieser Daten, insbesondere wenn Ressourcen für verschiedene Sprachen benötigt werden (vgl. [Mikeev et al., 1999]).

Kapitel II Theoretische Grundlagen

1. Eigennamen

1.1. Definition von Eigennamen

Bussmann grenzt Eigennamen ab als „eine semantisch definierte Klasse von Substantiven, die Objekte und Sachverhalte im Kontext eindeutig benennen, bzw. beschreiben“ ([Bussmann, 2002], S. 185). Eigennamen sind somit keine grammatikalischen Einheiten. Als Klasse können sie erst durch Berücksichtigung des Kontexts definiert werden, der manchmal im Text selbst als Zusatzinformation zu einem solchen Nomen steht, manchmal aber lediglich als Teil des Allgemeinwissens dem Leser bekannt sein muss. Um die semantische Klasse *Eigenname* in einem Text zu erkennen, muss somit neben viel aktuellem Allgemein- und Weltwissen eine sorgfältige Analyse des Umfelds, in dem der Eigenname steht, mitberücksichtigt werden. Da Eigennamen über ihren semantischen Inhalt definiert werden, genügt eine grammatikalische Analyse des Kontexts selten. Inhaltliche Komponenten müssen mit berücksichtigt werden. Dies erschwert die Erkennung von Eigennamen über automatische Algorithmen stark.

Wie Eigennamen Objekte genauer benennen bzw. beschreiben und welche Voraussetzungen mit einer Beschreibung erfüllt sein müssen, hat Blanár ([Blanár, 2001], S. 13ff.) in den folgenden zwei Punkten festgehalten:

1. Eigennamen sind an ein spezifisches Sprachsystem gebunden und entstehen in fortwährender Interaktion mit dem übrigen Wortschatz dieses Sprachsystems. Dies bedeutet, dass Eigennamen auf eine gewisse Sprache oder auf einen Sprachraum eingeschränkt sind. Nomen, die in der einen Sprache als Eigennamen gelten, können in anderen Sprachen unter Umständen der Eigennamenklasse nicht mehr angehören.
2. Es besteht ein ständiges Zusammenspiel zwischen Appellativen[54] und Eigennamen. Eigennamen entstehen in der Regel aus Appellativen (z.B. der Nachname *Müller* ist ursprünglich dem Vornamen einer Person mit dem Beruf *Müller* zugefügt worden). Bei Eigennamen muss eine Identifikation auf ein spezifisches, singuläres Objekt gewährleistet sein (z.B. *Müller* als Personenname). Appellative hingegen verweisen auf eine Menge von Objekten und fassen die Objekte auf Grund derselben Funktion zusammen (z.B. *Müller* als Berufsbezeichnung).

Diese beiden Punkte bedeuten einerseits, dass die Eigennamenerkennung bis zu einem gewissen Punkt an eine Sprache oder einen Kulturraum gebunden ist, und andererseits, dass der

[54] Ich verwende in meiner Arbeit grundsätzlich eine verdeutsche Form (Appellativ) von Appellativum (lateinisch).

Unterschied zwischen Appellativ (Eigenschaftsausdruck) und Proprium (Eigenname) nur über die Funktion des Nomens an dieser spezifischen Stelle des Textes bestimmt wird. Die Unterscheidung zwischen Proprium und Appellativ ist jedoch nur in der Theorie möglich. Die Grenzen dieser beiden Kategorien sind fliessend, da ein Nomen sowohl propriale Eigenschaften als auch appellative Eigenschaften tragen kann und eine Zuordnung somit nur noch eine Frage der Gewichtung dieser Eigenschaften ist (vgl. [Blanár, 2001], S. 35f.). Die Feststellung Blanárs (vgl. Punkt 1, S. II-42), dass Eigennamen an einen gewissen Sprachraum gebunden sind, ist nur bedingt korrekt. Meiner Meinung nach gibt es viele allgemeingültige Eigennamen, die lediglich bezüglich der Entstehung an ein Sprachgebiet gebunden sind (z.B. geographische Eigennamen), unterdessen aber so bekannt sind, dass sie über Sprachgrenzen hinweg als gültige Eigennamen anerkannt werden. Deshalb ist die Einschränkung auf ein Sprachgebiet, solange es sich um den europäischen Kulturkreis[55] handelt, bei der Eigennamenerkennung weniger dominant[56].

Die enge Verwandtschaft zwischen Proprium und Appellativ zeigt sich auch darin, dass in den Wissenschaften ein enger und ein weiter Eigennamen-Begriff bestehen. In der Linguistik und Onomastik wir üblicherweise der enge Begriff verwendet, bei dem Eigennamen „die Bezeichnungseinheiten für Personen (Vornamen wie *Hans*, *Peter*, *Thomas*, Familiennamen wie *Müller*, *Meier*, *Schmidt*) Orte (*Hamburg*, *Paris*, *Weiterstadt*), Tiere (*Harras* als Hundenamen, *Samson* als Katzenname), Gegendstände (z.B. Autos) und Ereignisse (z.B. Waterloo)“ ([Wimmer, 1995], S. 373) umfassen. Die Semiotik und philosophische Logik orientiert sich mehrheitlich am weiten Eigennamen-Begriff. „Danach gehören auch Appellativa wie *Berg*, *Gebirge*, Nominalgruppen wie *der höchste Berg Europas*, *die Hauptstadt von Polen* und *Pronomina* wie *du*, *er*, *das* zu den Eigennamen“ ([Wimmer, 1995], S. 373f.). Die wahrheitsfunktionale Semantik der philosophischen Logik geht sogar soweit, jeden Namen, der einen als existent vorausgesetzten Gegenstand bezeichnet, als Eigennamen (bzw. Name) zu definieren. Für die Linguistik ist jedoch ein weit gefasster Eigennamen-Begriff unbrauchbar, da er zu wenig differenziert (vgl. [Wimmer, 1995], S. 375f.). In meiner folgenden Arbeit ist der Eigennamen immer durch den eng gefassten Begriff definiert.

1.2. Geschichte der Eigennamen

Eigennamen werden in der Namenforschung (engl. onomastic) genauer untersucht. Die Namenforschung befasst sich im wissenschaftlichen Rahmen mit der Erforschung der Entste-

[55] Im europäischen Kulturkreise schliesse ich die amerikanischen Staaten sowie Kanada mit ein, da ihre Kultur durch die Kolonisation stark von Europa geprägt ist.
[56] Die erfolgreiche Umwandlung der Eigennamenerkennung in deutschen Texten auf englische Texte im Rahmen dieses Projekts, hat diese Aussage bestätigt.

hung, der Bedeutung und der geographischen Verbreitung von Eigennamen. Für die Eigennamenerkennung insbesondere ist die Erforschung der Bedeutung von Eigennamen wichtig. Wie die Definition bereits gezeigt hat, sind die Merkmale, die für einen Eigennamen bezeichnend sind, sehr schwierig zu erörtern. Der Eigenname sowie dessen Bedeutung und Funktion hat seine Wurzeln bereits in der Antike. Die Bibel berichtet in der Genesis, dass Gott dem Menschen den Auftrag zur Benennung der Tierwelt gegeben hat. „Der Mensch betrachtete die Tiere und benannte sie“ (Erstes Buch Mose (Genesis), Kapitel 2, Vers 20). Bereits in dieser Urschrift wird eine Verbindung zwischen Name[57] und Objekt auf Grund des Aussehens gemacht. Der Name, das benannte Objekt und die Abgrenzung als Klasse beschäftigten bereits bedeutende Philosophen der Antike wie Platon, Homer, Cicero und Quintilian. Im Mittelalter steigt das Interesse besonders bei den Chronisten. Es entstand eine Pseudo- bzw. Volksmythologie. Insbesondere wurden Personennamen und geographische Namen sowie deren Herkunft genauer untersucht. Gottfried Wilhelm Leibnitz (1646-1716) wies erstmals darauf hin, dass Eigennamen auf Appellativen beruhen. Weil der enge Zusammenhang, der zwischen Kultur und Eigennamen besteht, erkannt wurde, erstellte man Sammlungen von Orts- und Personennamen. Nach dem zweiten Weltkrieg dominieren linguistische Einflüsse in der Namenforschung immer stärker. Der Eigenname wird als sprachliches Zeichen interpretiert. Man versucht vermehrt, Funktion des Zeichens wie Individualisierung und Identifikation zu untersuchen, theoretisch zu analysieren und genauer zu beschreiben (vgl. [Eichler, 1995], S. 1ff.). Die Namenforschung wird eine selbständige linguistische Teildisziplin. Der Eigenname wird zu einem wichtigen Untersuchungsobjekt in allen linguistischen Richtungen (historische und synchrone Dialektologie, Computerlinguistik, Soziolinguistik, Textlinguistik, Pragmatik etc. (vgl. [Eichler et al., 1995], S. 14f.).

1.3. Kategorien von Eigennamen

Die klassischen Kategorien, die in der Eigennamenforschung untersucht werden, sind Orts- und Personennamen. Mit der Zeit hat man Klassen wie temporale Eigennamen ([Harweg, 1999], S. 1ff.), Firmen-, Organisations-, Produktenamen etc. in die Forschung mit einbezogen. Unter temporalen Eigennamen versteht Harweg Monatsnamen, Jahreszahlen, Datumsangaben etc. Im Bereich der Informationsextraktion und Informationsretrieval wird Eigennamenerkennung entsprechend dem modernen Forschungsstand mit verschiedensten Klassen betrieben. Für den Aufgabenbereich, Lösungen zur sprachunabhängigen Eigennamenerkennung zu finden, werden Eigennamen von der CoNNL 2002 (***C**ongerence **o**n **N**atural **L**angua-*

[57] *Name* ist in dieser Arbeit synonym mit *Eigenname* verwendet worden. Nach [Bussmann, 2002], S. 464, ist *Name* ein Obergriff für *Orts*- und *Personennamen* und sonstige Untersuchungsgegenstände der Namenforschung.

*ge **L**earning*)[58] mit folgenden Klassen umschrieben: "Named entities are phrases that contains the names of persons, organizations, locations, times and quantities" [CoNLL, http://cnts.uia.ac.be/conll2002/ner/]. Die CoNLL ist eine jährliche Konferenz, mit der versucht wird, die Forschung im Bereich des automatischen Erlernens natürlicher Sprachen voranzutreiben. Ziele dieser Konferenz sind Entwicklung von Modellen der natürlichen Sprache für den Computergebrauch, Entwicklung von Computermodellen zur Erforschung von Sprache, Entwicklung automatischer Lernmethoden von Computern und entsprechenden Theorien mit Verwendung von Prozessen zur Verarbeitung von natürlicher Sprache (NLP) und weiteren Methoden (z.B. logische Programmierung, Entscheidungsbäume, neuronale Netze, etc.). Die CoNLL wurde im Jahre 1997 erstmals von SIGNLL (ACL's[59] ***S**pecial **I**nterest **G**roup on **N**atural **L**anguage **L**earning*)[60] als spezifische Konferenz abgehalten und danach jährlich weiter geführt. Ziele von SIGNLL sind dieselben der Konferenz CoNLL. Der Aufgabenbereich *Eigennamenerkennung* ist 2002 an der CoNNL erstmals untersucht worden. Man hat verschiedene Systeme und Methoden vorgestellt, mit denen man Eigennamen möglichst sprachunabhängig erkennen kann. Die Ansätze sollten möglichst leicht auf andere Sprachen anwendbar sein. In dieser Konferenz wurden Eigennamen, nicht zum ersten Mal, neben den herkömmlichen Kategorien wie Personennamen und geographische Eigennamen auch in Organisationsnamen, Zeit- und Quantitätsangaben unterteilt.

Die am meisten verwendete Richtlinie bezüglich der Einteilung von Eigennamen in Kategorien wurde für die MUC 6 (***M**essage **U**nderstanding **C**onference*) im Jahre 1995 festgelegt. Entitäten (Klasse *ENAMEX*, engl. **nam**ed **en**tities **ex**pression) werden in Organisationsnamen, Personennamen, geographischen Eigennamen, Zeitausdrücke (Datums- und Uhrzeitangaben, entspricht der Klasse *TIMEX*, engl. **tim**e **ex**pression) und Zahlenausdrücke (Prozentzahlen und Geldbeträge, entspricht der Klasse *NUME*X, engl. **num**ber **ex**pression) eingeteilt (vgl. [MUC-6, http://www.cs.nyu.edu/cs/faculty/grishman/muc6.html]) Bei Zeit- und Zahlenausdrücken handelt es sich im Grunde genommen (bei einem engen Entitätsbegriff) nicht um Eigennamen[61]. Sie werden allerdings mit denselben Methoden aus einem Text extrahiert, so dass sie bei der Aufgabenstellung *Eigennamenerkennung* mit eingeschlossen sind. Die MUC wurde 1987 erstmals abgehalten, um die Erforschung von verschiedenen Algorithmen in der Informationsextraktionstechnologie im Englischen voranzutreiben. Weniger mit der Konfe-

[58] [CoNLL, http://cnts.uia.ac.be/conll2002/ner/].
[59] ACL (*The **A**ssociation for **C**omputational **L**inguistics*) ist eine internationale Vereinigung von Leuten aus dem universitären sowie kommerziellen Bereich, die insbesondere an computerlinguistischen Problemen arbeiten ([ACL, http://www.aclweb.org/]).
[60] [SIGNLL, http://ilk.kub.nl/~signll/].
[61] Weil kein in der Welt eigentliches Referenzobjekt besteht, ist es fraglich die Ausdrücke zu den Eigennamen zu zählen.

renz, sondern insbesondere mit der Evaluation verschiedener Systeme, die entsprechend einer bestimmten Aufgabenstellung Informationen aus Textmaterial extrahieren mussten, machte sich MUC einen Namen. Vor der Konferenz wurde entwickelte Software getestet, indem die Resultate der Systeme mit einem von Hand annotierten Schlüssel verglichen wurden. Die MUC 1-6 wurde in den Jahren 1987, 1989, 1991, 1992, 1993, 1995 von NOSC (***N**aval **O**cean **S**ystems **C**enter*)[62] mit Unterstützung von DARPA (***D**efense **A**dvanced **R**esearch **P**rojects **A**-gency*) organisiert (vgl. [Grisham und Sundheim, 1995]). MUC-7 fand als Teil des Tipster-Text-Programm („a multi-agency, multi-contractor program") das von DARPA unter Zusammenarbeit mit NIST geführt wurde, im Jahre 1998 zum letzten Mal statt (vgl. [MUC, http://www.itl.nist.gov/iaui/894.02/related_projects/muc/index.html]). Weil eine Phase des TIPSTER-Text-Programm im gleichen Jahr abgelaufen war und am Ende der MUC-7 erklärt wurde, dass man die Problematik der Informationsextraktion in der englischen Sprache genügend gut erkannt und erforscht hatte ([Robinson et al., 1999], S. 2), wurden keine weiteren MUC-Konferenzen abgehalten. Die MUC-6 ist für die Eigennamenerkennung von grosser Bedeutung, da für diese Konferenz neben den Themen Koreferenzauflösung, Schablonenbildung[63], Entwicklung von Szenarien zur Schablonenbildung und Extraktion von Beziehung zwischen Schablonen und insbesondere Eigennamenerkennung neu eingeführt wurde. Die Anforderungen entsprechend der Themen von MUC-6, welche eingereichte Software möglichst gut erfüllen sollte, wurden für die MUC-7 übernommen. Aus den MUC-Konferenzen ist eine selbständige Aufgabenstellung MET (***M**ulty **L**ingual **E**ntity **T**ask*) entstanden, die insbesondere auf die sprachunabhängige Eigennamenerkennung[64] Wert legt. Sie wurde in diversen Konferenzen und Workshops immer wieder als Thema aufgegriffen Die Richtlinien zur Eigennamenerkennung, welche für die MUC-6 bestimmt worden sind, werden in späteren Konferenzen und Veranstaltungen, wie z.B. HUB-4 (vgl. [HUB-4, http://www.nist.gov/speech/tests/bnr/hub4_98/hub4_98.htm][65]) im Jahre 1998, immer wieder übernommen und teilweise leicht abgeändert. Die Klasseneinteilung in *ENAMEX*, *TIMEX* und *NUMEX* bleibt bestehen.

Das von NIST unterstützte ACE-Programm, dessen Ziel die Entwicklung von Systemen und Technologien zur automatischen Inhaltsextraktion aus natürlichsprachlichen Texten (OCR-

[62] Später wurde aus der NOSC die NCCOSC.
[63] Mit Schablonen (engl. templates) sind Muster gemeint, die es erlauben, aus einem Text spezifische Information, wie z.B. den Hauptakteur eines Textes oder den Ort eines Ereignisses, zu extrahieren.
[64] Sprachunabhängige Eigennamenerkennung ist relativ, da Systeme nur Eigennamen erkennen können, falls die entsprechenden Sprachen im System in irgendeiner Art und Weise implementiert sind.
[65] Mehr Informationen zu HUB in Fussnote 16, S. 11.

Texte[66], gesprochene Texte[67] und geschriebene Texte) ist, hat in einem Zeitraum zwischen 1999 und 2000 Methoden und Systeme zur Eigennamenerkennung sowie Identifikation untersucht und entwickelt. ACE verwendet im Gegensatz zu den gewählten Ansätzen der Systeme für die MUC-Evalutionen, die hauptsächlich linguistische Textanalysen verwendet haben, zusätzlich semantische Textanalysen. Für das Pilotprojekt (1. Phase) sowie die folgende Phase ist die EDT–Aufgabenstellung (***e**ntity **d**etection and **t**racking*) entwickelt worden. Es gilt neben der reinen Erkennung einer limitierten Anzahl von Eigennamenklassen auch Attribute zu Eigennamen (z.B. Pronomen) zu extrahieren. Es sollen Zusammenhänge zwischen den Entitäten erkannt werden, was eine Eigennamenidentifikation[68] voraussetzt. Die zu erkennenden Entitäten gehören den folgenden fünf verschiedenen Klassen an:

1. Personennamen
2. Organisationsnamen
3. geopolitische Entitäten (GPE)

   Bei den geopolitischen Entitäten handelt es sich um Eigennamen, die politisch definierte Regionen bezeichnen (z.B. Städte- oder Nationsnamen).
4. geographische Entitäten

   Die geographischen Entitäten sind auf Namen von geographisch definierten Gebieten und Landmassen, Gewässern und geologische Formationen (Gebirge, Täler etc.) beschränkt.
5. Eigennamen von Einrichtungen

   Mit Einrichtung sind vom Menschen geschaffene Gebäude oder sonstige Gebilde gemeint, die von Architekten oder Bauingenieuren errichtet worden sind (z.B. Flughafennamen oder Namen eines Regierungsgebäudes etc.).

Für den EDT wurden im Vergleich zu den MUC-Richtlinien mit Ausnahme der Eigennamen von Einrichtungen keine neuen Klassen definiert, da es sich bei den GPE lediglich um eine Unterklasse der geographischen Eigennamen handelt. ACE befindet sich bis Oktober 2003 in der Phase 3, in der Eigennamenerkennung unter Beibehaltung des EDT's als wichtige Aufgabenstellung in verschiedensten Sprachen (z.B. Englisch, Chinesisch und Arabisch) untersucht und implementiert wird (vgl. [ACE, http://www.nist.gov/speech/tests/ace/index.htm]).

---

[66] OCR-Texte sind mit Hilfe eines Texterkennungsprogramms gescannt worden.

[67] Mit gesprochenen Texten sind Kurznachrichten gemeint, die von einem Sprecher ohne Vorlage gesprochen werden und entsprechend nicht immer in einer korrekten Syntax vorliegen und teilweise keine graphischen Markierungen wie z.B. Grossschreibung aufweisen (vgl. Kapitel IV 4. „Testkorpus für die Evaluation in englischen Texten").

[68] Bei der Eigennamenidentifikation muss sichergestellt werden, dass die Referenz auf eine spezifische Person / spezifisches Objekt gewährleistet ist.

Eine weitere Eigennamenklasse, die bisher unerwähnt blieb, jedoch gleiche Methoden zur Extraktion erfordert, sind Adressen. Zu Adressen zählen Telefonnummern, Wohnadressen, Emailadressen, URL's etc. Bei Adressen handelt es sich im Grunde genommen um keine Eigennamen, da sie nicht mit einem Objekt identifiziert werden können, sondern eher mit der Funktion, wo das bestimmte Objekt zu finden ist. Die Erkennung von Adressen wird trotzdem zur Eigennamenerkennung gerechnet und ist z.B. im System MUSE implementiert (vgl. [Maynard et al., 2002a]). Adressen sollten eine eigene Klasse zusätzlich zu den Unterteilungen von *ENAMEX*, *TIMEX* und *NUMEX,* gemäss den Richtlinien der MUC-Konferenzen, erhalten. Eine Unterscheidung zwischen klassischen Eigennamen (*ENAMEX*) und den Informationen, die mit denselben Methoden aus den Texten filtriert werden können, wäre so gewährleistet.

Es gibt diverse Klassifizierungsmöglichkeiten von Eigennamen. Zu den drei Hauptklassen, die am weitesten verbreitet sind, zählen Personennamen, geographische Eigennamen und Organisationsnamen. Bezüglich der Aufgaben der Eigennamenerkennung, -identifikation und der Einteilung in Klassen hat die MUC-6-Konferenz Richtlinien verfasst, die in diesem Forschungsgebiet als Vorlagen weiterhin verwendet werden. Die Klassen der zu erkennenden Eigennamen habe ich in meiner Lizentiatsarbeit entsprechend auf Personennamen, geographische Eigennamen und Firmennamen, die eine Untermenge der Organisationsnamen bilden, begrenzt. Es handelt sich dabei um die wichtigsten Klassen.

1.4. Eigenschaften/Merkmale und Bedeutung von Eigennamen

Das einzige, äusserliche Merkmal, welches Eigennamen besitzen, ist die Grossschreibung, welche allerdings nur in Sprachen, welche Klein- und Grossbuchstaben kennen, vorhanden ist. Ansonsten gibt es gewisse Merkmale (z.B. Setzung des bestimmten bzw. unbestimmten Artikels), die für Subklassen von Eigennamen spezifisch sind (z.B. Rechtschreibung von Personennamen[69]), nicht aber als allgemeines Merkmal von Eigennamen dienen. Zu den Alphabeten, die zwischen Gross- und Kleinbuchstaben unterscheiden, gehören das griechische, lateinische, kyrillische und armenische Schriftsystem ([Harweg, 1999], S. 200). Diese Alphabete bilden die Mehrheit aller verwendeten Schriftsysteme. Insbesondere im asiatischen Raum sind Sprachen üblich, die keine Gross- und Kleinbuchstaben kennen und teilweise über Silben und Laute definiert sind (z.B. Chinesisch oder Japanisch). Merkmale von Eigennamen beste-

[69] Personennamen müssen entsprechend ihrer Schreibweise zum Versteinerungszeitpunkt geschrieben werden. Der Versteinerungszeitpunkt ist dasjenige historische Datum, an dem willkürliche Änderungen der Namen verboten und die Namenform definiert worden ist (in Deutschland ist dieser Zeitpunkt zwischen 1666 und 1775 anzusiedeln). Normierungsregeln gelten für Personennamen nicht: ***Goe**the* ist nicht dieselbe Person wie ***Gö**the* (vgl. [Koß, 2002], S. 210f.).

hen dort unter anderem in Akzenten. Darauf möchte ich allerdings nicht näher eingehen (mehr dazu in [Harweg, 1999], S. 196ff.).

Die Tatsache, dass als Grundlage der Fixierung einer Sprache ein Alphabet dient, das Gross- und Kleinbuchstaben besitzt, bedeutet noch nicht, dass die Grossschreibung als Merkmal allein für Eigennamen verwendet wird. Im Deutschen wird die Grossschreibung neben der Markierung eines Satzanfanges auch zur Markierung aller Nomen/Substantiva verwendet. Eigennamen werden in der klassischen Grammatik als Subkategorie der Nomen verstanden (vgl. Abbildung 10)

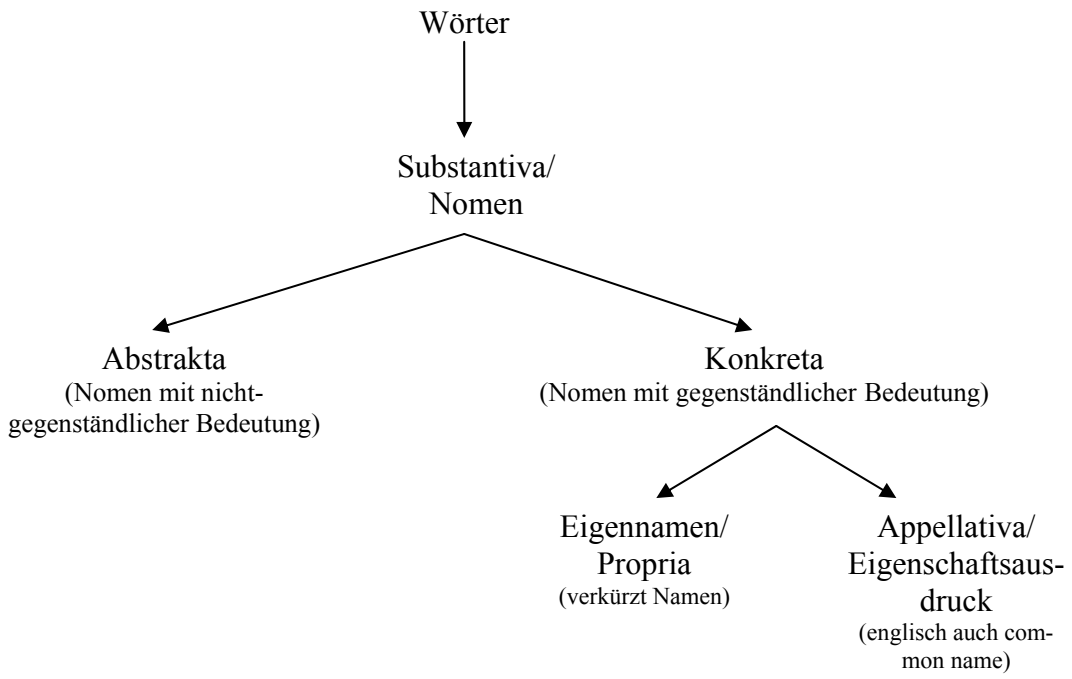


Abbildung 10

[Koß, 2002], S. 59

Es bestehen verschiedene Ansichten, ob die Grossschreibung tatsächlich ein Merkmal der Eigennamen ist oder ein Merkmal von Substantiven im Allgemeinen. Die Grossschreibung von geographischen Adjektiven wie z.B. *Zürcher* spricht für die Annahme, dass die Grossschreibung ein Merkmal für Eigennamen ist. Es gibt allerdings auch geographische Adjektive, die kleingeschrieben werden, wie z.B. *schweizerisch*. Ich bin der Meinung, dass die Grossschreibung sehr wohl ein wichtiges Merkmal von Eigennamen ist, da die Mehrheit aller Eigennamen grossgeschrieben wird. Kleingeschriebene Eigennamen bilden die Ausnahme. Solche Ausnahmen gibt es auch in anderen Sprachen wie Französisch oder Spanisch (Wochentags-, Monats- und Sprachnamen werden kleingeschrieben). Bei den Ausnahmen handelt es sich immer um jene Eigennamenklassen, die bezüglich ihrer Situierung als Eigennamen in der Forschung immer stark umstritten waren und noch immer sind. Eine mögliche Erklärung für

dieses Phänomen wäre, dass jede Sprache ihre eigene Definition von Eigennamen besitzt und durch Grossschreibung der Eigennamen markiert, welche Ausdrücke tatsächlich Eigennamen sind und welche nicht (vgl. [Harweg, 1999] S. 200ff.). Eine allgemeine, sprachunabhängige Definition von Eigennamen wäre in diesem Fall nicht mehr möglich. Englisch ist diejenige Sprache, in der es am wenigsten Ausnahmen bezüglich der Grossschreibung von Eigennamen gibt. Da Englisch als Weltsprache ein gewisse Vorbildsfunktion besitzt, werden die Eigennamenklassen entsprechend der dort bestehenden Grosschreibung definiert und auf die anderen Sprachen übertragen, obwohl die Grossschreibung dort divergieren kann und somit nicht immer als Merkmal für Eigennamen besteht.

Grosschreibung ist somit für europäische Sprachen ein wichtiges Merkmal von Eigennamen, wenn auch nicht interferenzfrei. Satzanfänge werden zusätzlich durch Grossbuchstaben gekennzeichnet sowie gewisse Personalpronomen (z.B. engl. *I*) oder die Höflichkeitsform von Pronomen (z.B. dt. *Sie*). Insbesondere im Deutschen ist die Unterscheidung von Appellativen und Eigennamen durch die Grossschreibung nicht gegeben. Und diese Unterscheidung ist eine der schwierigsten.

Die Unübersetzbarkeit von Entitäten ist eine Eigenheit, welche erst bei sprachübergreifenden Analysen von Bedeutung ist. Bei Eigennamen handelt es sich um einen sprachunabhängigen Wortschatz. Wenn man versuchen würde, Eigennamen zu übersetzen, so würde ihre Bedeutung verloren gehen. Dies ist durch den Titel eines Zeitungsberichts aus dem Jahre 1977 verdeutlicht (vgl. [Koß, 2002], S. 56f.). Der übersetzte Titel lautet: „***Kirchhügels*** Witte gestorben“. Bei dieser prominenten Persönlichkeit handelt es sich um Winston *Churchill* (*Church* = *Kirche*; *Hill*= *Hügel*). Durch die Übersetzung ist der Bezug zur Referenzperson nicht mehr gewährleistet. Der Eigenname hat keinen Inhalt mehr. Appellative dagegen, müssen übersetzt werden. Weil im Fall von *Churchill* der Name in Einzelkomponenten aufgeteilt wurde, geht der Leser davon aus, dass es sich um Appellative handelt, nicht aber um Eigennamen. Es entsteht ein Missverständnis. Eigennamen werden mit Ausnahme von Länder- und Kontinentsnamen nicht übersetzt. Für die Eigennamenerkennung bedeutet dies, dass Listen mit Eigennamen, welche als Ressourcen dienen, sprachunabhängig verwendet werden können. Regeln, die auf Grund interner Strukturen (Muster, die innerhalb des Eigennamens bestehen, z.B. *J.D. Edwards &* ***Co***) gebildet werden, gelten somit auch für andere Sprachen und können übernommen werden. Regeln, die über externe Strukturen (Muster, die nicht Teil der Entität sich, aber zur Erkennung verwendet werden, z.B. „**die Firma** *Electronic Book Technologies“*) erstellt werden, müssen jedoch an die Sprache angepasst werden.

Die Bedeutung eines Eigennamens festzulegen, ist ein lang diskutiertes Thema der Namenforschung. Weder von philosophischer noch von sprachtheoretischer Seite ist man zu einem allgemein befriedigenden Resultat gekommen. In der Forschung bestehen verschiedenste Ansichten bezüglich der Bedeutung von Eigennamen. Dass ein Eigenname gleich wie alle Wörter als ein sprachliches Zeichen definiert ist, wird mehrheitlich anerkannt. Das sprachliche Zeichen ist nach Saussure in einen graphischen Teil (*signifiant* = Bezeichnendes), der die geschriebene Seite eines Wortes repräsentiert, und in einen Bedeutungsteil (*signifié* = Bezeichnetes) unterteilt, der den allgemeingültigen Sinn (Vorstellung), der mit dem Lautbild verbunden ist, enthält. Was der Sinn eines Eigennamens ist und welche allgemeingültigen Eigenschaften mit einem Namen verbunden sind, ist ein vieldiskutiertes Problem der Onomastik (vgl. Abbildung 11).

Sprachliches Zeichen

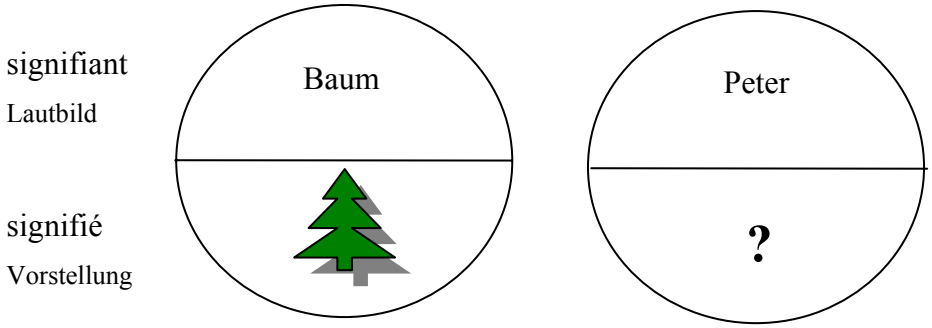


Abbildung 11

Insbesondere in der wahrheitswertfunktionalen Semantik der philosophischen Logik erhält die Bedeutung (Sinn) von Eigennamen einen wichtigen Stellenwert, da damit der Wahrheitswert einer Aussage verknüpft ist. Entsprechend der Terminologie, welche der Philosoph Frege (erstmals 1892) verwendet hat, enthalten Worte oder Aussagen sowohl Sinn als auch Bedeutung. Mit Sinn ist die Funktion gemeint, welche ein Eigenname ausführt, auch Intension genannt. Mit Bedeutung wird die Referenz auf ein bestimmtes Objekt beschrieben. Im semiotischen Dreieck[70], welches neben dem Zeichen selbst auch das Objekt der realen Welt miteinander in Beziehung setzt, sind Sinn und Bedeutung entsprechend Abbildung 12 anzuordnen. Mit Bedeutung eines Zeichens wird im allgemeinen Sprachgebrauch sowohl Referenz als auch Sinn gemeint. In der Semantik wird oft versucht, ein Zeichen möglichst unabhängig vom Referenzobjekt zu untersuchen und entsprechend den allgemeingültigen Sinn zu erforschen. In der Pragmatik dagegen ist der Bezug zwischen Zeichen und Referenzobjekt durch einen

[70] Das semiotische Dreieck ist 1923 von Ogden und Richards entworfen (vgl. [Bussmann, 2002], S. 596). In der Theorie Ch. Morris erlangt das semiotische Dreieck durch die Umsetzung auf Zeichen, Bezeichnetes und Zeichenbenutzer einen hohen Bekanntheitsgrad.

Benutzer geben, der die Referenz auf Grund einer gegebenen Situation zieht. Es handelt sich dabei um eine handlungsbezogene Betrachtungsweise. Je nach Theorie variiert die Bedeutung eines Eigennamens.

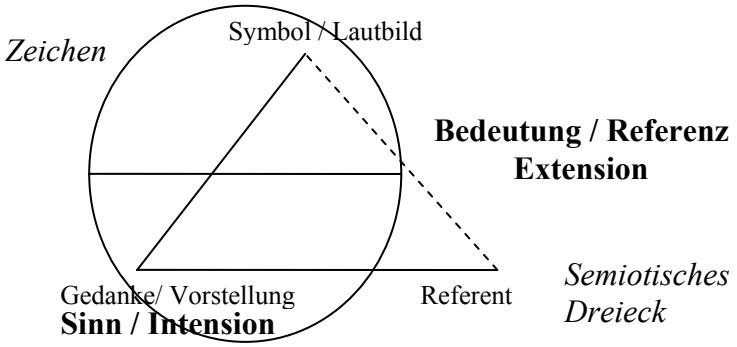


Abbildung 12

John Stuart Mill argumentiert im Jahre 1956, dass der Eigenname sich vom Appellativ unterscheidet, weil jener keine Intension besitze. Wenn dies allerdings zuträfe, so würde die Bedeutung eines Eigennamens lediglich in der Extension als Referenz liegen, was wiederum bedeuten würde, dass ein Name auf jedes beliebige Objekt quasi als eine Variable eingesetzt werden könnte. Dies trifft nicht zu. Vielmehr ist anzunehmen, dass die Bedeutung eines Eigennamens sowohl in der Intension als auch in der Extension liegt. Die Extension besagt, dass ein Eigenname auf ein spezifisches Objekt verweist, das in einer Welt existiert oder existiert hat. Die Intension ergibt sich dadurch, dass der Eigenname durch einen Vorgang, wie z.B. einen Taufakt[71], auf den Referenten fixiert und somit mit einen spezifischen Bedeutungsinhalt gefüllt wird (vgl. [Lötscher, 1995], S. 450). Ob es sich tatsächlich so verhält, ist genauer zu diskutieren. Es steht jedenfalls fest, dass Eigennamen in der Forschung seit Beginn einen Sonderstatus einnehmen, was zu vielen Diskussionen bezüglich ihrer Bedeutung in allen Disziplinen geführt hat.

Ich möchte jedoch jetzt nicht näher auf diese Diskussionen eingehen, da diese die automatische Eigennamenerkennung wenig beeinflusst haben. Mehr über die semantisch-erkenntnistheoretische Sicht ist in [Brüning, 1996] zu lesen. Eine Einführung in die Onomastik und ein Überblick der neuralgischen Punkte in der Diskussion über die Semantik der Eigennamen wird in [Koß, 2002] gegeben. Die populäre Ansicht über die Bedeutung von Eigennamen besagt, dass ein Eigenname auf ein in einer Welt bestehendes Objekt verweist. Falls dieses Objekt mit seinen Eigenschaften jemandem gänzlich unbekannt ist, so hat der

[71] Der Philosoph Kripke hat die Bedeutung eines Eigennamens im Taufakt und den darauf folgende Ereignisketten begründet. Wenn es einem gelingt durch Rückverfolgung von Ereignisketten den genauen Zeitpunkt des Taufaktes herauszufinden, dann besitzt der Eigenname einen Sinn und erhält einen Wahrheitswert (mehr dazu in [Brüning, 1996]).

Eigenname für diese Person keine Bedeutung. Die entstandene Wissenslücke könnte aber durch Sammeln von Informationen über das entsprechende Objekt, welche z.B. in einem Lexikon stehen, wieder geschlossen werden, so dass der Eigenname für die entsprechende Person eine Bedeutung erhält. Damit jedoch eine Person erkennen kann, dass ein unbekanntes Wort ein Eigenname sein könnte und bei fehlendem Referenzobjekt die Person ihre Wissenslücke schliessen könnte, müssen Merkmale entweder im Kontext oder im Eigennamen selbst bestehen, die es erlauben, ein Wort der Kategorie *Eigenname* zuzuordnen. Diese Merkmale sind diejenigen, welche für eine Eigennamenerkennung relevant sind (z.B. die Grossschreibung).

2. Relevanz von Eigennamenerkennung

Im Klappentext des Internationalen Handbuchs zur Namenforschung ([Eichler et al., 1995]) sind Eigennamen als „eine besondere Art von Wörtern“ über alle Sprachen hinweg beschrieben. Diese Aussage ist damit zu begründen, dass Eigennamen in direkter Beziehung zu einzigartigen Objekten der Welt bestehen[72]. Eigennamen besitzen die Eigenschaft, auf ein bestehendes Objekt in der „realen“ Welt sowie dessen Eigenschaften zu verweisen. Eigennamen verknüpfen somit einen Text (mit Ausnahme von literarisch fiktionalen) mit der Realität in unserer bestehenden Welt. Wenn ein Name einer Person in einem Text steht, so wird eine (und genau eine) Person, die in der „realen“ Welt lebt oder gelebt hat, über den Namen identifiziert. Jeder Eigenname ist mit genau einem einzigartigen Objekt (das Objekt kann auch eine Gruppe von Menschen oder eine Institution sein) in unserer Welt verbunden. Eigennamen haben somit die Eigenschaft, Einmaligkeits- und Existenzpräsupposition auszulösen. Die Existenzpräsupposition besagt, dass in der Welt ein Objekt existiert. Die Einmaligkeitspräsupposition besagt, dass dieses Objekt einmalig ist und die Summe aller Eigenschaft nur genau bei diesem Objekt vorkommt. Folglich ist ein Eigenname nicht nur eine Folge von Buchstaben/Lauten, sondern an ein real existierendes Objekt gebunden, dessen Eigenschaften mit der graphischen Abbildung des Eigennamens verknüpft sind. Der Sonderstatus des Eigennamens ist ebenfalls dadurch bedingt, dass er der Klasse der Konkreta (vgl. Abbildung 10) angehört und somit real existierende Objekte und keine fiktiven Objekte benennt. Eigennamen sind somit Elementarausdrücke (es besteht eine 1:1-Beziehung zwischen Eigennamen und bestehendem Objekt in der Welt). Ein Eigenname ist somit ein zweigeteiltes, arbiträres Zeichen (vgl. Abbildung 11), das aus einem Lautbild (*signifiant*) und aus dem Objekt selbst sowie dessen Eigenschaften (*signifié*) besteht. Eigennamen beinhalten somit einen semantischen

[72] Dies ist nur der Fall, wenn es sich nicht um fiktionale Eigennamen handelt. Mit Welt ist hier die „real bestehende“ Welt gemeint.

Wert: [[San Francisco]] (das Lautbild) = SAN FRANCISCO (die in den USA existierende Grossstadt entspricht dem Objekt selbst). Der semantische Wert definiert das real existierende Objekt. Das real existierende Objekt wird wiederum durch die Summe seiner Eigenschaften in der Welt definiert. Damit folglich ein Eigenname wahr ist und der Klasse der Eigennamen angehört, muss die Existenzpräsupposition erfüllt sein. Wenn kein Objekt zu irgendeinem Zeitpunkt in der Welt besteht oder bestanden hat, auf das ein Eigennamen referiert, so handelt es sich um keinen Eigennamen, da der Eigenname keinen semantischen Wert besitzt.[73] Wenn folglich die Existenzpräsupposition nicht erfüllt ist, existiert kein entsprechendes Objekt und der Text, in dem der vermeintliche Eigenname steht, hat keine Beziehung zur realen Welt und ist somit erfunden. Um den Wahrheitswert eines Textes zu überprüfen, ist es somit wichtig, dass Eigennamen erkannt werden.

Die Einmaligkeitspräsupposition ist ebenfalls eine grundlegende Eigenschaft von Entitäten. Ein Eigenname wie z.B. *Halle* muss auf eine tatsächlich existierende Ortschaft verweisen. Es handelt sich um genau jene Ortschaft, die unter anderem die Eigenschaft besitzt, sich in Deutschland zu befinden. Wenn *Halle* jedoch ein Appellativ ist und somit auf die allgemeine Eigenschaft, ein grosser Raum zu sein, verweist, handelt es sich trotz desselben Lautbildes nicht um einen Eigennamen. Die Einmaligkeitspräsupposition ist nicht erfüllt. Wenn folglich in einem Text ein Eigenname erkannt wird, schliesst man daraus, dass in der Welt ein Objekt mit einzigartigen Eigenschaften existiert.

Daneben, dass Eigennamen die Verknüpfungsstelle zur realen Welt bilden, treten sie in Texten relativ häufig auf. Paul Thompson und Christopher C. Dozier haben in einer empirischen Studie festgestellt, dass Entitäten einen entscheidenden Prozentsatz eines Textes ausmachen. Sie postulieren, dass auf Grund der Auftretenshäufigkeit eine Eigennamenerkennung und -extraktion sinnvoll ist. Sie haben festgestellt, dass in einem Korpus bestehend aus 720 Gesetzestexten - weshalb zu erwarten wäre, dass relativ wenig Eigennamen vorkommen - 5,2% aller Token Eigennamen sind (vgl. [Thompson und Dozier, 1999], S. 268).[74] Eigennamen treten somit so häufig auf, dass eine präzise Analyse eines Textes auf die Erkennung von Eigennamen angewiesen ist.

Eigennamen enthalten über die Welt relevante Information, die ein wichtiger Bestandteil der Grundinformation in eine Text bilden und folglich, um den Text zu verstehen, erkannt werden müssen. Über Eigennamen kann ein Informationsgerüst aus einem Text filtriert werden. Auf

[73] Es ist mir klar, dass es auch Eigennamen gibt, die keinen semantischen Wert im oben beschriebenen Sinn besitzen, jedoch trotzdem zu den Eigennamen gehören (z.B. mythische Gestalten wie Pegasus). Wie der Wahrheitswert eines solchen Objekts zu errechnen ist, wird bis jetzt noch von den Philosophen und Sprachanalytikern heftig diskutiert (Mehr dazu in. [Brüning, 1996]).

[74] Mehr zu dieser Studie in Kapitel I 3.4. „Eigennamenerkennung für Eurospider“.

die Fragen *Wer kommt vor*, *Wo befindet man sich*, *Wann spielt sich etwas ab* etc., die gestellt werden, um einen Text zu situieren, geben Eigennamen Antwort. Diese Informationen entsprechen dem grundlegenden Bedürfnis eines Menschen, sich in der Welt zurechtzufinden oder ein Ereignis in den Verlauf der Welt einzuordnen. Durch die Eigennamenerkennung in einem Text können somit die grundlegenden Informationen schnell erkannt werden.

3. Verwendete Kategorien

Um die Grösse des Projektes einzuschränken, werden über einen Algorithmus die folgenden drei Eigennamenklassen extrahiert: Personennamen, geographische Eigennamen und Firmennamen. Entsprechend den Grundlagen von Volk ([Volk, 2002]) sind die drei Eigennamenklassen, die ich in diesem Projekt über einen spezifischen Algorithmus erkenne, wie folgt definiert:

1. Personennamen

   Personennamen sind Namen, die einem Individuum (Vorname, Familienname und Beiname) oder einer Gruppe von Individuen (Familienname, der auf die Gruppe verweist, Völkernamen) gegeben werden und somit darauf verweisen. Das Individuum kann auch eine fiktionale Gestalt (mythische Namen) sein (vgl. [Bussmann, 2002], S. 507).

2. geographische Eigennamen

   Geographische Eigennamen sind Namen von politisch oder geographisch definierten Gebieten und Landmassen (z.B. Städtenamen, Kantons- bzw. Ländernamen, Nationsnamen, Regionsnamen, Kontinentsnamen). Nicht dazugezählt habe ich Quartier- und Strassennamen[75] sowie Namen von Gewässern und geologische Formationen.[76]

3. Firmennamen

   Firmennamen sind Namen von Handelskaufleuten oder Handelsgesellschaften, die offiziell im Handelsregister eingetragen sind. Firmennamen sind eine Subkategorie von Organisationsnamen, welche ich allerdings in diesem Projekt nicht erkennen werde. Organisationen sind Zusammenschliessungen von Menschen zu einer Gruppe, die dieselben Ziele verfolgt, unabhängig davon, ob die Ziele aus dem Handelsbereich stammen. Es muss eine hierarchische Struktur innerhalb der Gruppe bestehen.

---

[75] Quartier- und Strassennamen sind politisch definierte Gebietsbezeichnungen und zählen somit zu den geographischen Eigennamen. Im Projekt habe ich sie jedoch nicht berücksichtigt, da sie auf Grund ihrer Grösse und regionalen Abhängigkeit mir weniger relevant erschienen.

[76] Die nicht berücksichtigten Unterkategorien würden sich mit gewissem Aufwand in die von mir erstellten Algorithmen implementieren lassen.

Die Wahl fiel in diesem Projekt auf die soeben beschriebenen drei Eigennamenklassen, weil sie zu den grundlegendsten gehören. Eigennamen drücken Individualität von Menschen, Menschengruppen, Familien, Örtlichkeiten, und Einrichtungen aus ([Koß, 2002], S. 71). Somit kann anhand von Personennamen, geographischen Namen und Firmennamen ein Text ziemlich umfassend situiert werden. Die zentralen Fragen *wer* und *wo* können durch die diese drei Eigennamenklassen bereits beantwortet werden.

Ein weiteres Kriterium, das zur Auswahl dieser drei Eigennamenklassen geführt hat, sind die bestehenden internen und externen Strukturen. Diese Eigennamen treten oft in gleichartigem Kontext (mit Kontext sind die Wörter gemeint, welche unmittelbar vor und nach dem Eigennamen stehen) auf (externen Strukturen) und weisen häufig innerhalb des Eigennamens für die entsprechende Klasse spezifische, gleichartige Strukturen auf (interne Strukturen). Es ist somit möglich, über Regeln und Wortlisten diese Eigennamen zu erkennen.[77]

3.1. Eigenschaften der verwendeten Kategorien

In den folgenden Algorithmen werden drei verschiedene Kategorien von Eigennamen erkannt: Personennamen, geographische Bezeichnungen und Firmennamen. Jede dieser Kategorien ist durch spezifische Eigenschaften und Merkmale ausgezeichnet, die zur Erkennung dieser Eigennamen verwendet werden können. In den folgenden Abschnitten werde ich die Eigenschaften kurz vorstellen.

Personennamen bestehen in der Regel aus mindestens einem Element, nämlich dem Nachnamen oder einem allein stehenden Vornamen. Vor einem Nachnamen können ein oder mehrere Vornamen bzw. Initialen stehen. Ein Personenname kann zusätzlich durch eine Anrede (z.B. *Herr*) oder eine Berufsbezeichnung (z.B. *Präsident*) eingeleitet werden. Ein Nachname kann durch einen Titel (z.B. *von* oder *van*) spezifiziert werden. Der Titel gehört zum Nachnamen. Personennamen besitzen somit die folgende Struktur (vgl. Abbildung 13):

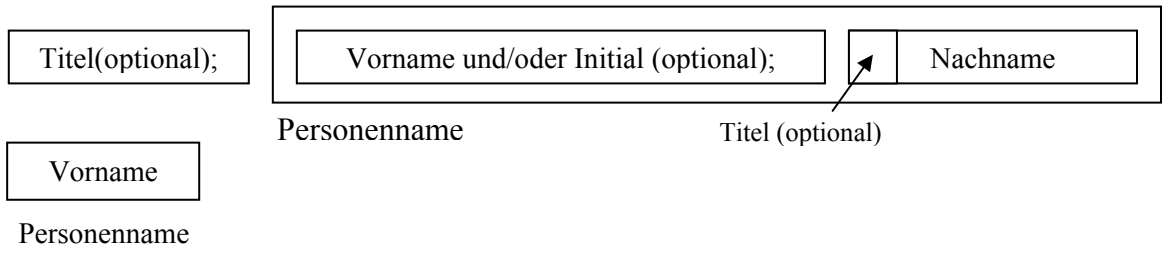


Abbildung 13

[77] Genaueres zu den Regeln und Ressourcen ist in Kapitel II 4. „Algorithmen“ zu finden.

Die Anrede bzw. die Berufsbezeichnung zählt in meiner Definition nicht zum Eigennamen. Als externe Struktur dient er als Trigger zur Erkennung des folgenden Personennamens. Die Anrede nicht zum Eigennamen zu zählen, entspricht den an den MUC-6 erstellten Annotationsrichtlinien (vgl. Kapitel II 1.3. „Kategorien von Eigennamen“). Bei Systemen wie z.B. MUSE[78] gehört die Anrede entgegen den Richtlinien zum Personennamen. Es handelt sich somit um eine willkürliche Entscheidung, ob die Anrede Teil der Entität ist oder nicht.

Ein allgemeines Merkmal, das Personennamen besitzen, ist die Grossschreibung. Jeder Teil eines Personennamens (mit Ausnahme des Titels) beginnt mit einem Grossbuchstaben, sofern es sich um eine Sprache handelt, die Gross- und Kleinbuchstaben unterscheidet. Dieses Merkmal ist im Englischen weit dominanter als im Deutschen, weil es weniger Interferenzen aufweist (vgl. Kapitel II 1.4. „Eigenschaften/Merkmale und Bedeutung von Eigennamen“).

Vornamen gibt es nur in einer beschränkten Anzahl und sie werden selten neu gebildet. Bei der Namengebung eines Kindes wählen die Eltern in der Regel aus einem bereits bestehenden Inventar (Namen aus dem Bekanntenkreis oder aus Namenbüchern etc.) den Namen ihres Kindes aus. Wegen der Beständigkeit und der beschränkten Anzahl eignen sich die Vornamen, um in eine konstante Liste aufgenommen zu werden (vgl. Kapitel I 4. „Ressourcen“) Vornamen können als Trigger verwendet werden, um den folgenden Nachnamen (das nächste grossgeschriebene Wort) zu erkennen.

Nachnamen haben die Eigenschaft, dass sie in verschiedensten Schreibformen (z.B. *Meier*, *Mayer*, *Maier* etc.) auftreten und teilweise sogar neu gebildet werden können (z.B. Künstlernamen oder durch Heirat entstandene Doppelnamen). Auf Grund der grossen Anzahl, die unter anderem wegen der verschiedenen Schreibweise zustande kommt, eignen sich Nachnamen nicht, um in konstante Listen gesammelt zu werden (vgl. Kapitel I 4. „Ressourcen“)[79].

Titel bzw. Berufsbezeichnungen, die Teil des Nachnamens sind, gibt es nur in beschränkter Anzahl. Titel zumindest werden nicht neu gebildet. Sie eignen sich zum Auflisten und können als interne Indikatoren (Trigger, die Teil der Entität sind) zur Namenerkennung benutzt werden.

Die Klasse der geographischen Eigennamen umfasst in diesem Projekt Städte-, Regions-, Nationen-, Länder- und Kontinentnamen sowie geographische Adjektive. Geographische Eigennamen besitzen nur wenige interne oder externe Strukturen, die es erlauben würden, den Ei-

[78] Vgl. Kapitel I 2. „Eigennamenerkennung“.

[79] Es gibt zwar Bücher, in denen Nachnamen einer Region oder Sprache gesammelt werden. Diese sind einerseits sehr umfangreich, andererseits auf ein spezifisches Thema hin gesammelt und durch die Themenwahl begrenzt (sei es nur auf Grund der Sprache). Es ist folglich möglich die Nachnamen zu sammeln, allerdings ist Vollständigkeit in dem Sinne, dass alle Nachnamen, die auf der Welt vorkommen, enthalten sind, nur schwierig zu erreichen.

gennamen als solchen zu erkennen. Es gibt Lexeme, die als eigenständige Substantive klar dem appellativen Bereich zugeordnet werden (z.B. *Weide* oder *Dorf*). Diese Lexeme kommen relativ oft als Teile von Städte- oder Dorfnamen (z.B. *Weiden* oder *Urdorf* etc.) vor. Dort geht ihre appellative Eigenschaft verloren, da der Eigennamen für eine Ortschaft steht, die zu einem früheren Zeitpunkt entsprechende Eigenschaften besessen hat, zum jetzigen Zeitpunkt mit ihnen jedoch nicht mehr in Verbindung gebracht wird (vgl. [Koß, 2002], S. 58ff.). Die Ortschaft *Urdorf* (im Kanton *Zürich*) ist beispielsweise schon lange kein *Dorf* mehr, sondern in Folge der Einwohnerzahl eine *Stadt*. Diese Lexeme, typisch für geographische Eigennamen, sind jedoch eher selten in den geographischen Entitäten anzutreffen. Dies hat sich auch bei Rössler bestätigt, der über statistische Methoden solche Strukturen unter anderem für geographische Eigennamen von einem System automatisch zu lernen versuchte (vgl. Kapitel I 2. „Eigennamenerkennung“) und gerade bei den geographischen Eigennamen wenig Erfolg hatte (vgl. [Rössler, 2002]). Der Grund der schlechten Evaluationswerte lag seiner Meinung nach darin, dass die geographischen Eigennamen nur wenige Strukturen aufweisen und deshalb nur schlecht über Regeln gelernt werden können. In meinem Projekt habe ich ausser zur Unterscheidung zwischen Eigennamen und Wörtern aus dem Allgemeinwortschatz (vgl. Kapitel III 3.2. „Ressourcen zur Erkennung von geographischen Namen“), die gleich lauten, weder interne noch externe Strukturen verwendet.

Weil geographische Eigennamen die Eigenschaft besitzen, in beschränkter Anzahl und konstanter Schreibweise vorzukommen, kann man sie zur Eigennamenerkennung in Ressourcen auflisten. Ein konstantes Vorkommen ist insofern gewährleistet, als eine Stadt heutzutage nur selten neu gegründet wird und deshalb einen neuen Namen erhält. Ausnahmen bilden politische Veränderungen eines Landes, die eine neue Namensgebung mit sich bringen können (z.B. als der kommunistische Staat Sowjetrussland oder die DDR aufgelöst wurden). Aber selbst in diesen Fällen wird wieder auf Namen zurückgegriffen, die bereits in früherer Zeit einmal für das Objekt verwendet worden sind, z.B. *St. Petersburg* (vorher *Leningrad*) oder *Chemnitz* (vorher *Karl-Marx-Stadt*).

Problematisch bei geographischen Eigennamen ist, dass oft zwei verschiedene Subkategorien (z.B. Städtenamen und Kantonsnamen) mit dem gleichen Lautbild bezeichnet werden. Beispielsweise gibt es sowohl die Stadt *Zürich* als auch den Kanton *Zürich*. Trotz des gleichen Eigennamens handelt es sich um zwei verschiedene Objekte, die sich allein durch ihre Grösse unterscheiden. Die Auflösung einer solchen Ambiguität setzt eine Eigennamenidentifikation voraus. Um eine Identifikation eines Eigennamens zu gewährleisten, ist sowohl Weltwissen als auch Kontextwissen nötig. Somit benötigt ein System, das automatische Eigennameniden-

tifikation betreibt, ausgereiftere Methoden, als sie bei der Eigennamenerkennung verwendet werden. Die Eigennamenidentifikation ist momentan ein viel diskutiertes und aktuelles Thema in der Forschung.[80] Im Mai dieses Jahres wurde spezifisch zu diesem Thema ein Workshop im Rahmen der Human-Language-Technology-Konferenz in Edmonton (Kanada) abgehalten. Die Konferenz wurde von NAACL (***N**orth **A**merican Chapter of the **A**ssociation for **C**omputational **L**inguistics*), ACM SIGIR, ISCA (*The **I**nternational **S**peech **C**ommunication **A**ssociation*) und HLT (***H**umen **L**anguage **T**echnology*) organisiert. Ziele dieser jährlichen Konferenz sind, die Zusammenarbeit zwischen Computer und Menschen, die sich der natürlichen Sprache bedienen, zu ermöglichen, und somit Themen wie automatische Übersetzung, automatische Spracherkennung, IR, automatische Textzusammenfassung und IE zu diskutieren und zu erforschen. Thema des Workshops war es, automatisch Referenzen von geographischen Eigennamen zu analysieren (mehr dazu ist unter [NAACL'03-Workshop, http://www.sims.berkeley.edu/research/conferences/hlt-naacl03/] zu finden). Als Resultat des Workshops wurde die Notwendigkeit erkannt, die Thematik in weiteren Projekten zu vertiefen.

Thema meines Projekts und somit auch der Lizentiatsarbeit ist jedoch Eigennamenerkennung das Thema gewesen, so dass die Problematik von ambigen Eigennamen nur insofern berücksichtigt werden muss, als dass keine doppelten Einträge (ein Eintrag für *Zürich* als Kanton und ein Eintrag für *Zürich* als Stadt) die Ressourcen unnötig vergrössern (vgl. Kapitel III 3.2. „Ressourcen zur Erkennung von geographischen Namen“) und sie deshalb aus den Listen gestrichen werden müssen, falls eine lexikonbasierte Eigennamenerkennung angestrebt wird.

Firmennamen enthalten ebenfalls eher wenig interne oder externe Strukturen, die zur Erkennung dienen könnten (vgl. Abbildung 14). Es gibt eine beschränkte Anzahl von Wörtern, die oft vor einem Firmennamen stehen oder intern in der Entität am Ende auftauchen.

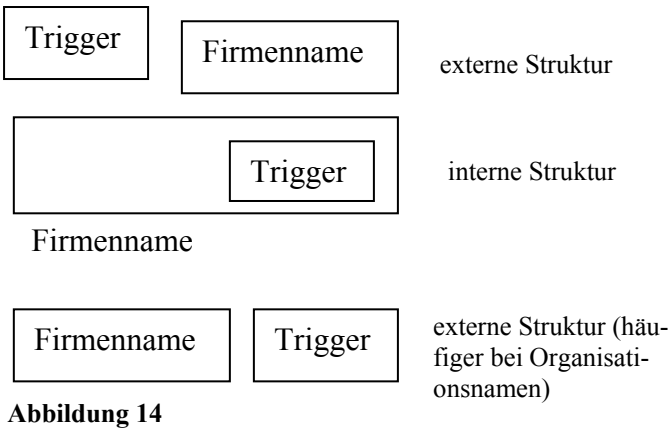


Abbildung 14

[80] Human Language Technology Conference, 2003 [NAACL'03-Workshop, http://www.sims.berkeley.edu/research/conferences/hlt-naacl03/].

Ein Teil der Firmennamen kann über solche Wörter erkannt werden, falls er grossgeschrieben wird oder mit einer Zahl oder einem Sonderzeichen ((&, -,…) beginnt. Weil die verwendeten Buchstaben und Zeichen in einem Firmennamen stark divergieren, ist die Definition einer typischen Zeichenfolge schwierig. Die unkonstante Länge, welche ein Firmennamen besitzen kann, erschwert zusätzlich die Erkennung. Die Länge kann meiner Erfahrung nach zwischen zwei bis drei Buchstaben oder Zeichen (z.B. *CS*) und ca. sieben Wort- oder Buchstabenkombinationen schwanken (z.B. *computer telefon service (cts) petersen & partner ohg*). Bei den Firmennamen handelt es sich nicht um eine konstant bleibende Menge, die zusätzlich aus verschiedensten Zeichen (auch Phantasiezeichen) besteht. Die Gründung neuer Firmen sowie deren Namensgebung ist ein ständig anhaltender Prozess, der zu keinem Zeitpunkt abgeschlossen ist. Deshalb können Firmennamen nicht einfach in einer Liste aufgeführt werden. Die Liste wäre niemals aktuell, weil sie ständig mit neu entstandenen Firmennamen ergänzt werden müsste. Es macht lediglich Sinn, einen Kontext (mit Kontext ist die direkte Wortumgebung um den Firmennamen gemeint) zu finden, in dem möglichst häufig Firmennamen auftauchen oder interne Strukturen zu finden sind, die konstant Teil von Firmennamen sind.
Die Erkennung von Firmennamen ist das schwierigste Unterfangen für die Eigennamenerkennung, weil sehr wenig externe und interne Strukturen vorhanden sind, die es erlauben würden, allgemeine Regeln aufzustellen.

4. Algorithmen

Die Algorithmen zur Erkennung von Personennamen, geographischen Namen und Firmennamen sind ursprünglich von Martin Volk im Rahmen seiner Habilitationsschrift (vgl. [Volk, 2002]) entwickelt worden. Die Algorithmen basieren auf Eigennamenerkennung in deutschen Texten. In diesem Projekt habe ich die bestehenden Algorithmen leicht abgeändert und erweitert, um sie mit der Konzeptsensorsprache[81] von der Firma *Eurospider* implementieren zu können und eine entsprechend hochgradige[82] Eigennamenerkennung zu ermöglichen. Zusätzlich habe ich die Algorithmen, welche nur teilweise in anderen Sprachen verwendet werden können, für die Eigennamenerkennung in englischen Texten angepasst. Die Algorithmen bestehen aus Regeln, welche mit Hilfe von aufgelisteten Wörtern (Ressourcen) eine Erkennung der Eigennamen erlauben. Es handelt sich somit um eine regel- und ressourcenbasierte Lösungsansatz (vgl. Kapitel I 2. „Eigennamenerkennung“). Die Regeln mussten fürs Englische nur leicht angepasst werden, während die Ressourcen entsprechend dem englischen Vokabular grosse Abänderungen erforderten. Im folgenden Abschnitt werde ich kurz die von Volk

[81] Näheres zu Konzeptsensorsprache ist in Kapitel III 2. „Konzeptsensorsprache“ zu lesen.
[82] Hochgradig bedeutet hier eine möglichst hohe Präzision bei möglichst hoher Ausbeute zu erreichen.

übernommenen Algorithmen und die darin verwendeten Ressourcen sowie die vorgenommenen Änderungen beschreiben.

4.1. Übernommene Ressourcen

Ressourcen hat Volk in seinem Algorithmus, den er in *Perl* implementiert hat, einerseits ausserhalb des Codes in externen Listen eingetragen, andererseits in internen Listen innerhalb des Codes aufgeführt. Die folgenden, externen Listen konnte ich mit leichten Veränderungen in meiner Implementierung übernehmen:

1. Vornamen-Liste

   Die Liste enthält hauptsächlich Vornamen aus dem europäischen Sprachraum sowie einige aus dem asiatischen Raum. Die Liste ist von mir für die Eigennamenerkennung in diesem Projekt nur leicht abgeändert worden.

2. Städte-Liste

   Die Liste diverse Städtenamen. Insbesondere sind Städtenamen aus Deutschland enthalten. Zur Erkennung von Eigennamen in deutschen Texten habe ich die Liste insbesondere um Städte- und Dorfnamen aus der Schweiz, aus Deutschland und Österreich erweitert. Für die Erkennung in englischen Texten sind Namen aus den USA, aus Kanada und Grossbritannien hinzugekommen.

3. Nationen-Liste

   Sie enthält diverse Ländernamen. Mit Ländern sind in meiner Kategorisierung Nationen gemeint und nicht Kantons- oder Bundesländernamen. Die Länderliste wurde von mir stark erweitert.

4. geographische Adjektiv-Listen

   Diese Listen enthalten Einträge mit geographischen Adjektiven, welche auf *isch* oder auf *er* enden. Die Adjektive stammen hauptsächlich aus Deutschland. Die Adjektivlisten wurden von mit wenig erweitert.

5. Lexikon Englisch – Deutsch

   Es handelt sich um ein kleines Wörterbuch, das einen Standardwortschatz in Deutsch und Englisch enthält. Die Einträge des Wörterbuchs habe ich übernommen. In einem Lexikoneintrag hat sowohl der deutsche als auch der englische Teil einen eigenen Eintrag in den Ressourcen erhalten.

6. Abkürzungsverzeichnis

   Die Liste enthält gängige Abkürzungen im Deutschen. Dieses Abkürzungsverzeichnis ist gleich übernommen worden.

Die Ressourcen von Volk sind von mir mit Kantons- und Ländernamen, mit Regions- und Inselnamen sowie Kontinentnamen zusätzlich ergänzt worden (vgl. Kapitel III 3. „Ressourcen für die Erkennung von Entitäten in deutschsprachigen Texten“ und Kapitel III 5. „Ressourcen für die Erkennung von Entitäten in englischen Texten“).

4.2. Übernommene Regeln

Die Regeln für die Eigennamenerkennung sind von Volk insbesondere für deutsche Texte entwickelt worden. Somit enthalten sie gewisse sprachspezifische Muster, wie beispielsweise die Erkennung von Firmennamen über Bindestrichsuffixe. Weil allerdings das Hauptmerkmal, die Grossschreibung von Eigennamen, für alle Sprachen zutrifft, die Gross- und Kleinbuchstaben verwenden, sind viele Regeln für andere Sprachen ebenfalls anwendbar. Dies hat sich ebenfalls bei der Implementierung der Eigennamenerkennung für englische Texte herausgestellt, da hauptsächlich die Ressourcen angepasst werden mussten. Die Regeln konnten mehrheitlich gleich übernommen werden. Volk hat die Eigennamenerkennung in *Perl* implementiert. Um die Algorithmen in eine allgemeine Form zu bringen, habe ich einen Pseudocode erstellt, der sich im Anhang dieser Arbeit befindet. In den folgenden Abschnitten sind die von Volk verwendeten Algorithmen, welche mir als Grundlage für meine Implementierung gedient haben, genauer erläutert. Für eine ausführliche Version sei auf den Pseudocode verwiesen.

4.2.1. Regeln zur Erkennung von Personennamen

Personennamen sind auf Grund drei verschiedener Muster erkannt worden: „*vollständige Personennamen*“, „*Nachnamen“* und „*abgekürzte Personennamen“*. Ein „*vollständiger Personenname*“ besteht aus einer Anrede (z.B. *Herr*), einem optionalen Vornamen (z.B. *Vincent*), einem optionalen Titel (z.B. *van*) und einem Nachnamen (z.B. *Herr* ***Vincent van Gogh***). Anreden und Titel sind in internen Listen aufgeführt[83], Vornamen in einer externen Liste. Als Vornamen gelten auch Kombinationen aus zwei Vornamen, die durch einen Bindestrich getrennt werden (z.B. *Hans-Peter*). Als Nachnamen gelten alle Nomen, die grossgeschrieben werden und im entsprechenden Muster an entsprechender Position auftauchen. Die Anrede gehört nicht zum Eigennamen. Das Muster „*Nachname“* besteht aus einem Vornamen; einem optionalen Titel und einem Nachnamen (z.B. *Lukas Mühlemann*). Ein abgekürzter Personenname besteht aus einem Initial; einem optionalen Titel und einem Nachnamen (z.B. *L. Mühlemann*). Als Initial wird ein Grossbuchstabe gefolgt von einem Punkt verstanden. Als Initial

[83] Der Inhalt der internen Listen ist aus dem Pseudocode ersichtlich, welcher sich im Anhang dieser Arbeit befindet.

wird ebenfalls die Kombination von Initial und Vorname, getrennt durch einen Bindestrich (z.B. *K.-Heinz*), akzeptiert.

Nachnamen, die durch eines der oben aufgeführten Muster erkannt wurden, werden innerhalb des Dokuments erneut gesucht. Nachnamen, die im Dokument wiederholt verwendet werden, jedoch ohne Anrede oder Vorname, da eine Identifizierung der entsprechenden Person durch die Verwendung des vollständigen Namens bereits geschehen ist, werden durch den zweiten Suchschritt trotzdem gefunden. Alle Nachnamen werden am Ende eines Dokuments wieder vergessen.

4.2.2. Regeln zur Erkennung von geographischen Eigennamen

Bei der Erkennung von geographischen Eigennamen werden keine Regeln angewendet. Die Erkennung basiert lediglich auf den Ressourcen. Falls ein Eintrag in den Listen gefunden wird, so handelt es sich um einen geographischen Eigennamen. Alle Ressourcen sind in externen Listen abgespeichert. Zusätzlich zu den bestehenden Nominativformen werden alle Einträge der geographischen Ressourcen durch Hinzufügen des Buchstaben *s* am Ende ebenfalls im Genitiv gesucht. Weil *isch*-Adjektive flektiert werden, muss bei der Suche der Wörter im Text berücksichtigt werden, dass verschiedene Endungen des Adjektivs vorkommen können und es trotzdem der Kategorie *Eigenname* angehört. Die Adjektive müssen folglich in der Grundform gesucht werden.

4.2.3. Regeln zur Erkennung von Firmennamen

Volk hat vier verschiedene Muster zur Firmennamenerkennung entwickelt. Davon habe ich allerdings nur drei (ein Firmenname folgt einem Trigger, ein Firmenname steht vor einem Trigger, ein Firmennamen steht vor einem Bindestrichsuffix) verwendet, da das vierte Muster zu spezifisch auf den Wortschatz der Computer-Zeitung ausgerichtet war. Die Anwendung des Musters auf andere Texte hat gezeigt, dass oft Fehlanalysen entstehen. In der folgenden Beschreibung werde ich nur auf die von mir verwendeten drei Muster genauer eingehen. Das erste Muster erkennt einen Firmennamen auf Grund eines Triggers (z.B. *Firma*), dem der Eigenname folgt (z.B. „die **Firma** *Electronic Book Technologies*“). Die Trigger sind in internen Listen[84] gesammelt. Als Firmennamen gelten alle grossgeschrieben Wörter, die unmittelbar dem Trigger folgen und nicht in einer Stoppwortliste enthalten sind. Die Stoppwortliste ist eine interne Liste, welche Wörter enthält, die grossgeschrieben werden, oft nach einem entsprechenden Trigger folgen, jedoch nicht Teil des Firmennamens sind (z.B. *Software* oder *Trade*). Das zweite Muster erkennt Firmennamen auf Grund eines Triggers (z.B. *Co.*), der

[84] Der Inhalt aller internen Listen ist aus dem Pseudocode im Anhang ersichtlich.

dem Firmennamen folgt (z.B. „… von *J.D. Edwards & **Co***“). Als Firmennamen gelten wiederum alle grossgeschriebenen Wörter, die unmittelbar vor dem Trigger stehen und nicht in einer Stoppwortliste enthalten sind. Die Stoppwortliste ist in diesem Fall eine interne Liste, welche Wörter enthält, die grossgeschrieben werden, oft vor einem Firmennamen auftauchen, jedoch nicht Teil des Firmennamens sind (z.B. *Die* oder *Hersteller*). Das dritte Muster erkennt Firmennamen auf Grund eines Bindestrichsuffixes (z.B. *-Tocher*) hinter der Entität (z.B. „die Zukunft der *France-Télécom-***Tochter** ist ...“). Die Bindestrichsuffixe sind in einer internen Liste gesammelt. Als Firmennamen gelten alle grossgeschriebenen Wörter, die vor dem Suffix stehen und mit einem Bindestrich voneinander getrennt werden, ausser es handelt sich um das Präfix *Ex* (z.B. ***Ex**-Apple-Chef*). Es dient als Stoppwort.

Das Erkennen von Firmennamen ist somit ein zirkulärer Vorgang, der abbricht, sobald ein kleingeschriebenes Wort oder ein Eintrag aus einer Stoppwortliste auftaucht. Die Länge des Firmennamens ist entsprechend nicht auf eine bestimmte Anzahl Wörter begrenzt.

Alle Firmennamen werden auf Grund eines der drei oben beschriebenen Muster gelernt und abgespeichert. Aus den gelernten Firmennamen werden Kurzformen (z.B. *Escom AG* → ***Escom***) und Akronyme (z.B. *United Bank of Switzerland* → ***UBS***) extrahiert und der bestehenden Liste hinzugefügt. Die einteiligen Firmennamen dieser Liste filtert man. Falls ein Name im Englisch-Deutsch-Lexikon oder im Abkürzungsverzeichnis steht oder von *Gertwol*[85] als ein Nomen/Substantiv erkannt wird, entfernt man den Namen aus der Liste, da es sich nicht um einen Eigennamen handelt. Die gefilterte Liste dient zur Erkennung weiterer Firmennamen in der gesamten Textkollektion. Die gelernten und gefilterten Firmennamen werden folglich, im Gegensatz zu den Nachnamen, über Dokumentgrenzen hinweg gesucht.

4.3. Unterschiede der Implementierung zu den bestehenden Algorithmen

Ich habe die folgenden Änderungen an Volks Algorithmen vorgenommen, um sie in der Konzeptsensorsprache implementieren zu können und möglichst hohe Ausbeute bei möglichst hoher Präzision zu erreichen.

Bei der Personennamenerkennung wurden anhin nur Kombinationen von Vornamen und Nachnamen sowie Nachnamen alleine erkannt. Weil jedoch in Texten ein Vorname allein als Personenname vorkommen kann und dieser ebenfalls der Kategorie Personenname angehört, habe ich, um die Personennamenerkennung zu vervollständigen, beschlossen, auch Vornamen, wenn sie alleine auftreten, als Personennamen zu markieren. Vornamen alleine kommen sehr selten vor, da die Identifikation des Eigennamens mit einem Objekt nur durch die vorherige Einführung einer längeren Form desselben Namens gewährleistet werden kann. In man-

[85] Vgl. Fussnote 28, S. 15.

chen Fällen wird der Eigenname mit einer allgemein bekannten Persönlichkeit (z.B. *Aristoteles*) verbunden und muss somit weder davor noch danach im Text genauer spezifiziert werden. In antiken Schriften oder in der Bibel ist es sogar üblich, dass die Personen nur mit einem Namen, nämlich dem Vornamen genannt werden (z.B. *Moses*). Es gibt allerdings auch Fälle, in denen ein Eigenname zum Appellativ wird, weil die Eigenschaften dieser Person so ins Weltwissen eingeflossen sind, dass nun der vermeintliche Eigenname nicht für die Person selbst steht, sondern als Charakterzug für eine Gruppe von Menschen verwendet wird (z.B. *Krösus*[86]) In diesem Fall käme es zu einer Fehlanalyse.

Bindestrichvornamen wurden anhin definiert als eine Folge, bestehend aus zwei Vornamen (z.B. *Hans-Peter*) oder einem Vornamen und einem Initial, getrennt durch einen Bindestrich (z.B. *Hans-P.*). In seltenen Fällen wird jedoch auch der erste Teil des Vornamens durch ein Initial ersetzt, so dass ein Bindestrichvornamen aus Initial oder Vornamen in jeglicher Kombination besteht (z.B. *H.-P.* oder *H.-Peter*). Um mehr Personennamen zu erkennen, habe ich die Definition der Bindestrichnamen erweitert und lasse nun jegliche Kombinationen von Initial und Vorname zu (z.B. *Hans-Peter*, *H.-Peter*, *Hans-P.* oder *H.-P.*)

Da die geographische Eigennamenerkennung nicht auf Regeln basiert, sind lediglich die Ressourcen von mir erweitert worden. Alle Genitivformen musste ich in einer separaten Liste aufführen, weil in der Konzeptsensorsprache keine direkte Stringmanipulation vorhanden ist (vgl. Kapitel III 2.4. „Probleme innerhalb der Konzeptsensorsprache und ihre Grenzen“).

Die Muster zur Erkennung von Firmennamen sind von mir gleich übernommen worden. Die Wiedererkennung der über ein Muster gelernten Firmennamen über Dokumentgrenzen hinweg kann in der Konzeptsensorsprache nicht implementiert werden. Deshalb habe ich die Firmennamenerkennung nur ansatzweise implementiert (vgl. Kapitel III 4. „Implementierung der Eigennamenerkennung in deutschsprachigen Texten“). Der Filterschritt sowie die Akronymgewinnung weichen vom ursprünglichen Algorithmus ab. Weil in der Konzeptsensorsprache keine einzelnen Buchstaben von Wörtern abgetrennt oder an Wörter angefügt werden können, ist es unmöglich, Akronyme von gelernten Firmennamen abzuleiten. Sie müssen als entsprechende Wörter definiert werden. Meine Definition von Akronymen entspricht der Abbildung 15.

```
Acronym (W) {
IF      (W == (([A-Z]+) OR ([A-Z] AND &)))
THEN TRUE
}
```

Abbildung 15

[86] *Krösus* war König der Lyder um 560-547 v. Chr. und besass unermesslichen Reichtum. Die Bezeichnung wird entsprechend für die Eigenschaft ‚reich zu sein’ verwendet.

Ein Akronym ist demzufolge eine Kombination bestehend aus mindestens zwei Grossbuchstaben oder einem Grossbuchstaben und einem &. Da diese Definition sich während des Projekts als unzureichend erwiesen hat, ist sie von mir erweitert worden (vgl. Kapitel III 2.3. „Erweiterungen in der Konzeptsensorsprache zur Erkennung von Eigennamen"). Trotz diverser Versuche konnte allerdings nie dieselbe Präzision erreicht werden, wie sie über die tatsächliche Ableitung der Firmennamen entstanden wäre.

Da kein Wortanalysesystem wie *Gertwol* zur Verfügung stand, musste ich den Schritt vereinfachen, in dem einteilige, falsch gelernte oder fälschlicherweise abgeleitete Firmennamen rausgefiltert wurden, weil sie von *Gertwol* als ein allgemeines Nomen/Substantiv (Abstrakta oder Appellativ) erkannt worden sind oder im Englisch-Deutsch-Lexikon oder Abkürzungsverzeichnis standen. Mein Filter besteht lediglich aus dem Englisch-Deutsch-Lexikon und der Liste mit Abkürzungen (vgl. Kapitel III 4.3. „Erkennung von Firmennamen"). Diese beiden Ressourcen sind weit weniger mächtig als *Gertwol* und die Präzision der Firmennamenerkennung leidet stark unter dem Fehlen eines besseren Filters (vgl. Kapitel IV 3.3. „Evaluation von Firmennamen" und Kapitel IV 5.3. „Evaluation von Firmennamen"). Um die Firmennamenerkennung möglichst eng an Volks Algorithmen anlehnen zu können, wäre eine ‚Zwei-Weg-Lösung' notwendig. Diese würde bedingen, dass in einem ersten Schritt die Firmennamen über die drei oben beschriebenen Muster innerhalb der Konzeptsensorsprache erkannt und extrahiert, dann in einem nächsten Schritt ausserhalb der Konzeptsensorsprache entsprechend bearbeitet würden (Bilden von Akronymen, Kurzformen und Genitivformen), um anschliessend wieder mit Hilfe der Konzeptsensorsprache gefiltert (nur einteilige Firmennamen) und im Text erneut gesucht zu werden. Die Problematik, dass der Filter unzureichend ist, kann nach wie vor mit einer entsprechenden ‚Zwei-Weg-Lösung' nicht behoben werden.[87]

[87] Mehr zu der Problematik, welche die Konzeptsensorsprache mit sich bringt, ist im Kapitel III 2.4. „Probleme innerhalb der Konzeptsensorsprache und ihre Grenzen"zu lesen.

Kapitel III Implementation

1. Eurospider Retrieval System ‚Relevancy'

Das Retrieval System *Relevancy* wurde von der Firma Eurospider für IR entwickelt. Es handelt sich um eine Grundversion, die fortlaufend verbessert wird. Im Verkauf befindet sich zurzeit die 5.Version von *Relevancy*. Das System besteht aus Grundmodulen, die den bestehenden Kundenwünschen angepasst werden können. In einem Projekt in Zusammenarbeit mit der Zentralbibliothek Zürich hat *Eurospider* beispielsweise *Relevancy* eingesetzt, um eine Abfrage derjenigen Bücher zu ermöglichen, die über den Zettelkatalog von Hand archiviert worden waren. Man musste *Relevancy* entsprechend anpassen, damit die Verarbeitung von OCR-Texten möglich war. Die folgenden Module gehören zur Grundversion von *Relevancy*: ein Indexer, der Dokument und Anfrage auf einen Index reduziert, um einen präzisen Abgleich von Dokument und Anfrage zu erlauben (engl. matching), eine Kategorisierungskomponente von Dokumenten sowie Benutzer- und Systemschnittstellen (z.B. eine graphische Oberfläche für den Benutzer (GUI)). Die Eigennamenerkennung wurde in diesem Projekt innerhalb des Indexers implementiert. Die restlichen Komponenten sind nicht weiter angepasst worden. Das gesamte System ist in der Programmiersprache *C* geschrieben.

Der Indexer ist das Herzstück des *Relevancy.* Das Ziel des Indexers ist es, Anfrage und Dokumente in eine vergleichbare Form (eine Index) zu bringen. Im Indexierungsprozess wird jedes Wort eines Dokuments sowie Teile des Wortes entsprechend der definierten Einstellungen bearbeitet. Früher bestand der Indexierungsprozess lediglich aus Reduktion und Transformation von Daten und Datensätzen. Im modernen IR werden die Daten zusätzlich angereichert. Indexieren von Wörtern, Phrasen, Sätzen oder Dokumenten setzt sich aus folgenden Komponenten zusammen:

Indexieren = Reduktion + Transformation + Anreicherung

Formel 9

Die Reduktion beinhaltet Schritte wie Elimination von nicht inhaltstragenden Elementen (z.B. Anweisungen zu einem Dokument), Entfernung von Satzzeichen oder nicht bedeutungstragenden Wörtern (Stoppwörtern) oder Entfernung von Formatierungselementen für Schriftart, Schriftgrösse etc.

Unter Transformation versteht man Formatveränderungen (Verwandeln eines Dokuments in das vom System intern verwendete Format), Übersetzungen von einer in eine andere Sprache oder Umwandlung von Wörtern in eine Normform (z.B. Umwandlung von diakritischen Zeichen in Wörtern ***Ö**rlikon* → ***Oe**rlikon*).

Zu Anreicherungsschritten zählen Erweiterung von Token einer Anfrage mittels Thesauri oder Kategorisierung und Markierung von spezifischen Wörtern (z.B. Entitäten), um erweiterte Anfragen in Verbindung mit den Kategorien zu stellen.

Der Indexer von *Relevancy* ist wie eine Pipeline aufgebaut. Er enthält verschiedene Boxen (Module) mit Bearbeitungsmethoden, welche miteinander verlinkt sind. Unter Bearbeitungsmethoden sind Reduktions-, Transformations- und Anreicherungsschritte zu verstehen. Auf die Boxen kann man einzeln zugreifen und sie für den gesamten Prozess aktivieren, bzw. deaktivieren. Jede Box besitzt eine eigene Eingabe und erzeugt eine Ausgabe, die auf einem entsprechenden Speicherplatz (Ebene) abgespeichert wird. Andere Boxen können jederzeit auf eine Ausgabe zugreifen, um sie als neue Eingabe zu verwenden.

Die wichtigsten Schritte beim Indexieren eines Textes sind folgende:

1. Entfernen der SGML-Markierungen und Abspeicherung des Textinhalts sowie der SGML-Markierungen auf je getrennten Ebenen (*Tag* und *Content*). In der Inhaltsebene stehen die einzelnen Elemente. Die einzelnen Elemente sind nicht zwingend nur Wörter, sondern Zeichenfolgen, welche durch Zwischenräume (im Beispiel mit '_' markiert) von einander abgegrenzt werden (z.B. Zuürich oder *ROSE,.*). Die Wortsegmentierung erfolgt später.

   Beispiel 10:

   <poem><title>The SICK ROSE</title>
   Tag-Ebene: <poem> <title> </title>
   Inhalts-Ebene: The_SICK_ROSE,

   Die extrahierten Elemente werden jeweils mit Positionsangabe abgespeichert, so dass jedes Element in Beziehung zum ganzen Dokument gesetzt werden kann.

2. Extraktion des Titels, indem der Text aus der Inhaltsebene genommen wird, der zwischen den Markierungen *TI* steht. Es entsteht eine neue Ebene *TITEL*.

   Beispiel 11:

   <title>The_SICK_ROSE</title>
   Titel-Ebene: The_SICK_ROSE,

   Die Wortsegmentierung wird in einem späteren Schritt ausgeführt. In der Titelebene stehen Elemente, die nicht zwingend Wörter sein müssen (vgl. Inhalts-Ebene).

3. Einzelne Wörter werden abgetrennt und entsprechend abgespeichert (in der Ebene *ENTS_WORD*).

   Beispiel 12

   Titel-Ebene: The_SICK_**ROSE,**
   Titel-Ebene nach Wortsegmentierung: The SICK **ROSE**

4. Umwandlung von Wörtern mit HTML-Schriftzeichen (*ENTS_WORD*) in *Standard Latin (*Abspeicherung in der Ebene *WORD)*.

   Beispiel 13:

   Zu**ü**rich → Zuerich
   Aber Partner&Co. → Partner & Co.

5. Der Text wird einer Sprache zugeordnet. Auf der Inhalts-Ebene wird ein Dokument automatisch einer bestimmten Sprache zugeordnet. Die Information ist in der Sprachebene (*LANGUAGE*) abgespeichert.
6. Alle Wörter werden in Kleinbuchstaben umgewandelt.

   Beispiel 14:

   Zuerich → zuerich

7. Elimination von Wörtern, die eine gewisse Länge überschreiten, um zu vermeiden, dass unsinnig lange Aneinanderreihungen von Buchstaben oder Buchstabenfolgen, die als Wort analysiert wurden, unnötig Speicherplatz verbrauchen. So können beispielsweise Internetseiten, die z.B. Programmiercode enthalten oder mit zusammengeschmolzenen Wörtern Aufsehen erregen wollen (Spam-Mails), als irrelevant erkannt werden. Diese Methode beruht auf der Annahme, dass Wörter auf eine gewisse Länge begrenzt sind, falls es sich um solche aus einem natürlich sprachlichen Wortschatz (Deutsch, Englisch) handelt. Die Länge in *Relevancy* ist auf maximal 100 Buchstaben begrenzt worden.

   Beispiel 15:

   HALLOHALLOHALLOHALLOHALLOHALLOHALLO

8. Stoppwörter werden extrahiert und auf einer eigenen Ebene abgespeichert (*STOPWORD*).

   Beispiel 16:

   „Das hat die repräsentative Befragung **eines** kalifornischen Marktforschungsinstituts **über di**e Chancen **der** Infogesellschaft ergeben.“
   [Konradin-Verlag, 1998]
   Stoppwörter-Ebene: Das, hat, die, eines, über, die, der

9. Wörter werden in einer Grundform abgespeichert. Die Grundform ist nicht zwingend eine linguistische Form. Es handelt sich um keine Lemmatisierung, sondern um ein gezieltes Kürzen von Wörtern (Trunkierung).

   Beispiel 17:

   scheitern, gescheitert, scheiterte
   Grundform: scheiter

10. Diakritische Zeichen werden in eine Normform umgewandelt. Das Umwandeln diakritischer Zeichen geschieht im Deutschen bereits nach Schritt 5. In den restlichen Sprachen wird diese Umwandlung erst nach der Reduktion auf einen Wortstamm durchgeführt, weil sonst Lesarten eines Wortes verloren gingen.

    Beispiel 18:

    Zürich → Zürich
    Groβvater → Grossvater

11. Speicherung spezieller Kategorien, welche zuvor als Konzepte oder reguläre Ausdrücke definiert worden sind (z.B. Zahlen, Geldbeträge etc.)

    Beispiel 19:

    „Das hat die repräsentative Befragung eines **kalifornischen** Marktforschungsinstituts über die Chancen der Infogesellschaft ergeben.“
    [Konradin-Verlag, 1998]
    Geographisches Adjektiv: kalifornischen[88]

Die verschiedenen Speicherungsebenen verweisen über Positionsangaben der Wörter innerhalb des Dokuments aufeinander. Die Indexierungsmethoden bauen kaskadenförmig aufeinander auf. Die Ausgabe der einen Methode wird als Eingabe der nächsten verwendet. Dies ist in Abbildung 16 illustriert.

[88] Geographische Adjektive zählen in diesem Projekt zu den geographischen Eigennamen und werden nicht als eigene Kategorie markiert.

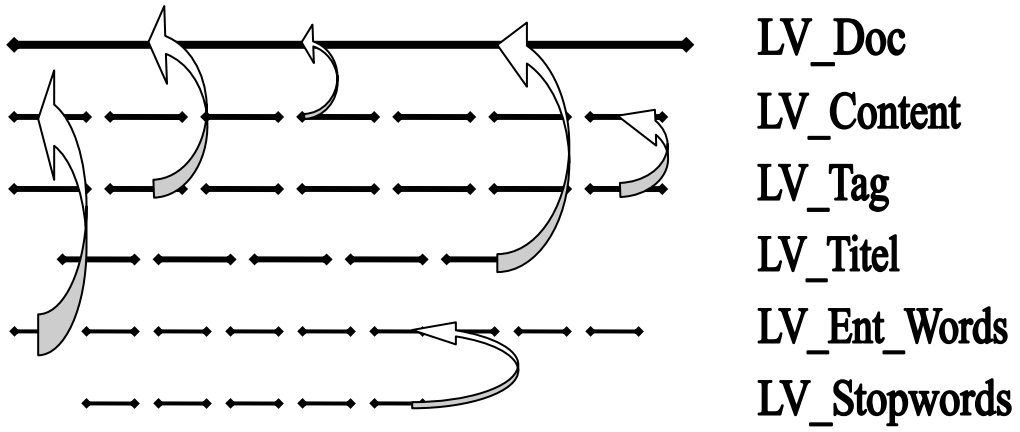


Abbildung 16

Die Inhalts-Ebene wird aus der Dokumenten-Ebene extrahiert. Wörter entstammen der Inhalts-Ebene extrahiert. Die Stoppwörter wiederum werden aus der Wort-Ebene gewonnen. Extraktion bedeutet nicht, dass die Einträge aus der bestehenden Ebene entfernt werden, sondern, dass sie kopiert und in einer neuen Ebene abgelegt werden. So ist ein Zugriff jederzeit auf alle Ebenen möglich. Für die Eigennamenerkennung wurde meist die Wort- oder Inhalts-Ebene verwendet; da die Wörter in möglichst wenig reduzierter Form zur Analyse vorliegen sollten.

Diese modulartige Struktur (Aufteilung des Indexiervorgangs in einzelne Schritte, welche als einzelne Methoden realisiert sind), in welcher der Indexer aufgebaut ist, braucht zwar viel Speicherplatz, gewährt jedoch eine flexiblere Handhabung bei Veränderungen oder Erweiterungen des Systems als eine Architektur, in der alle Indexiervorgänge in einem einzigen, riesigen Modul vereinigt sind.

Der *Synchronizer* bestimmt den Ablauf der einzelnen Normalisierungsvorgänge und versteht sich somit als Schaltstelle zwischen den Dokumenten der Textsammlung bzw. den Anfragen und dem Indexer selbst, der die Normalisierungsvorgänge ausführt.

Relevancy unterstützt verschiedene Indexierungsvarianten.

1. Sprachunabhängige Indexierung

   Dokumente werden unabhängig davon indexiert, in welcher Sprache sie verfasst sind.

2. Sprachabhängige Indexierung

   Es wird ein verfeinerter Index entsprechend einer implementierten Sprache (Deutsch, Englisch, Französisch, Italienisch, Spanisch) über das gesamte Dokument erstellt. In diesem Fall können sprachspezifische Stoppwörter eliminiert werden.

3. Spracherkennung bis auf Satzebene, Wortnormalisierung und Kompositazerlegung
   Die wahrscheinlichste Sprache, in der ein Satz geschrieben ist, wird anhand von Trigrammodellen errechnet. Ein Stemmer reduziert Wortflexionen mit einfachen Regeln auf eine normalisierte Form (vgl.Beispiel 17, S. III-70). Komposita werden zerlegt, falls sie im internen Wörterbuch stehen und eine Mindestlänge besitzen (z.B. *Aktiengesellschaft* in *aktie* und *gesellschaft*).
4. Fuzzy-Matching
   Fuzzy-Matching als Lösung zur Handhabung von Schreibfehlern und Schreibvarianten[89]. Es wird nur auf die Anfrage (nicht aber auf das gesamte Dokument[90]) angewendet, um so trotz eines Schreibfehlers vom Fragenden relevante Dokumente zu erhalten.
5. N-Gramm-Indexierung für OCR-Texte
6. Indexieren der Strukturelemente bei strukturierten Texten
   Durch die Indexierung der Strukturelemente (Tags) sowie des Inhalts, können zu einem späteren Zeitpunkt Information exakter gefunden werden, z.B. Extraktion der Titelzeile (vgl. Beispiel 10, S. III-68) damit in späteren Analysen direkt auf sie zugegriffen werden kann.

Der Indexierungsprozess kann durch Einbinden von domänenspezifischen Wörterbüchern oder Thesauri, domänenspezifischen, automatisch generierten Phrasen und kundenspezifischen Stoppwörtern erweitert werden. Dasselbe Dokument kann man mit verschiedenen Indexierungsmethoden automatisch analysieren.

Um neue Analysen in den Indexer zu integrieren, wird entweder der Indexer selbst erweitert oder es werden Konzepte über die Konzeptsensorsprache in den Indexer eingefügt, welche bei der Firma entwickelt worden ist. Die Konzeptsensorsprache ist ein Mittel, um leicht domänenspezifischen Kontext in die Grundanalyse des Indexers zu integrieren. Mehr zur Konzeptsensorsprache in Kapitel III 2. „Konzeptsensorsprache“.

Ein Index wird sowohl über dem Dokument als auch über die Anfrage erstellt. Der Abgleichungsprozess zwischen Anfrage und Dokument wird nur über den Index durchgeführt. Um den Abgleichungsprozess von Anfrage und Dokument bezüglich Präzision und Ausbeute zu optimieren, unterstützt *Relevancy* verschiedene Funktionen:

---

[89] Im Prinzip zählt Fuzzy-Matching zu den Methoden zum Abgleich von Frage und Dokument. Das Verändern von einzelnen Buchstaben innerhalb der Anfrage, um ein in einer Sprache bestehendes Wort zu erhalten, wird jedoch im Indexer durchgeführt.

[90] Der Abgleichprozess kann nur exakt sein, da das Dokument indexiert wurde. In einem Index ist eine unscharfe Suche bis jetzt noch nicht möglich, weshalb die Anfrage um gewünschte Varianten erweitert werden muss.

1. Kombination von Wörtern als Anfrage mit boole'schen Operatoren
   Durch die Kombination von Wörtern kann die Anfrage spezifischer gestellt werden, so dass die Präzision optimiert wird (vgl. Kapitel I 1.4. „Ziele und Methoden des IR").
2. Integration von wahrscheinlichkeitsbasierten Ansätzen (Lnu.ltn und Okapi[91])
   Lnu.ltn und Okapi sind Gewichtungsschemata. Sie machen eine Aussage darüber, welche Begriffe in welchen Dokumenten höher gewichtet werden sollen. Damit wird die Effektivität eines Resultats optimiert und folglich sowohl Präzision als auch Ausbeute verbessert.
3. Einschränkung der Dokumente auf spezifische Kategorien
   Die Kategorien werden durch die Wahl des Benutzers festgelegt. Die Präzision wird optimiert.
4. automatische Expansion der Anfrage
   Die Anfrage wird automatisch expandiert, indem Synonyme und reduzierte Wortformen mit berücksichtigt werden. Durch die Expansion erhält man neue Dokumente und die Ausbeute wird somit optimiert.
5. Rangierung gefundener Dokumente
   Dokumente, welche zu einer Anfrage gefunden werden, rangiert man über Regeln und nach speziellen Benutzerwünschen, so dass wichtigere Dokumente am Anfang einer Auswahlliste stehen. Die Präzision wird erhöht, weil relevante Dokumente am Anfang der Rangliste erscheinen.
6. Relevanzrückkopplung
   Wenn ein relevantes Dokument zu einer Anfrage gefunden worden ist, wird aus den Daten (Schlüsselwörter etc.) des gefundenen Dokuments eine neue Anfrage formuliert, zu der weitere, relevante Dokumente gefunden werden.
   Die Relevanzrückkopplung wird nicht automatisch mit jeder Anfrage ausgeführt, sondern muss vom Benutzer als Option manuell angewählt werden. Die Relevanzkopplung hat eine Optimierung der Ausbeute zur Folge.
7. Elimination von Duplikaten
   Duplikate werden eliminiert und Dokumente über statistische Clusteringverfahren automatisch nach Ähnlichkeit in Gruppen eingeteilt. Die Gruppierung hat eine Optimierung der Präzision zur Folge. Falls Dokumente aus derselben Gruppe beim Abgleich von Anfrage und Dokument mitberücksichtigt werden, wird eine Optimierung der Ausbeute zusätzlich erreicht.

[91] Mehr Informationen zu den beiden Methoden in [Singhal et al., 1996] und [Robertson et al., 1994].

Dokumente, welche zum Abgleich einer Anfrage verwendet werden, sind in einer Datenbank (*Oracle*, *PostgreSQL*) oder in einem Datei-System (*Unix*, *Windows NT*) abgespeichert. *Relevancy* kann bis zu 250 verschiedene Dokumentformate (*SGML*, *HTML*, *Postscript* etc) verarbeiten.

Das IR-Programm benutzt verschiedene System- und Benutzerschnittstellen, welche eine Einbindung in andere komplexe Umgebungen erlauben (z.B. Systemschnittstellen) und dem Benutzer den Umgang mit dem System erleichtern (z.B. webbasierte Benutzerschnittflächen).

Um eine Dokumentensammlung zu erstellen, indem im Internet effizient nach relevanten Dokumenten gesucht und deren verlinkte Unterdokumente verfolgt werden, hat *Eurospide*r in Verbindung mit *Relevanc*y einen Webcrawler entwickelt. Mit den gleichen Methoden wie beim Indexier wird ein Index erstellt, auf Grund dessen eine Web-Seite weiter verfolgt und gespeichert oder als irrelevant eingestuft wird. Einschränkungen (z.B. dass das Dokument in einer spezifischen Sprache geschrieben sein muss) werden vom Benutzer definiert und dienen nach dem Indexieren als Filter.

Eine Kategorisierung der Dokumente ist nötig, um grosse Datenmengen besser zu verwalten und entsprechende Filter, welche beim Abgleichungsprozess von Anfrage und Dokument verwendet werden, in *Relevancy* zu integrieren. Ein Modul zur Kategorisierung ist folglich Teil des Systems. Dokumente werden über die statistische Methode *KNN* (***K n****earest* ***n****eighbours*) automatisch in Kategorien eingeteilt. Merkmale, die Zugehörigkeit zu den Kategorien anzeigen, werden anhand manuell zugeordneter Trainingsdokumente gelernt.

Über die Konzeptsensorsprache lassen sich ebenfalls Konzepte erstellen, anhand derer Dokumente in Klassen eingeteilt werden. Es gibt eine Anzahl bereits vorgefertigter Konzepte, die Benutzern mit einer entsprechenden Oberfläche zur Verfügung stehen, um Dokumente zu einem Thema (z.B. Unternehmensergebnisse, Wirtschaft, Branchen etc.) automatisch klassifizieren zu lassen.

Einem Benutzer wird mit der Komponente „*Search Guide*“ die Formulierung einer präzisen Anfrage erleichtert, indem die Anfrage automatisch, mit Hilfe einer Ontologie, welche domänenspezifisch erstellt worden ist, expandiert wird oder Beziehungen zwischen den Begriffen mit Hilfe von statistischen Analysen mitberücksichtigt werden. Für den Benutzer besteht zusätzlich die Option, eine allgemeine Frage auf Grund eines Vorschlags vom System einzuschränken oder spezifische Fragen auszuweiten.

Relevancy läuft auf *Sun Solaris 2.6*, *2.7*, *2.8*, auf Linux-Versionen basierend auf *Kernel-Version 2.2* oder *2.4* und auf *Windows NT* und *XP* von *Microsoft*. Das System benötigt als

Fremdsoftware einen Webserver (*Apache 1.3* oder *Microsoft IIS5*) sowie eine Datenbank (*Oracle* oder *PostgreSQL 7.1*) oder Datenschnittstelle (*ODBC*).

2. Konzeptsensorsprache

Die Konzeptsensorsprache ist eine Sprache, die es erlaubt, Muster zu definieren, die in einem Text oder einer Phrase vorkommen. Konzepte, welche in dieser Sprache formuliert werden können, sind Regeln, die eine Erkennung von Elementen in einem Kontext erlauben. Elemente können Wortfolgen, Wörter, Buchstaben, Zeichen oder Zahlen sein.

2.1. Die Aufgabe der Konzeptsensorsprache

Die Konzeptsensorsprache wurde innerhalb der Firma *Eurospider* entwickelt, um über Muster spezifische, komplexe Merkmale (z.B. Eigennamen, Geldbeträge etc.) in Texten zu finden, die beim Indexierungsvorgang (vgl. Schritt 11 Kapitel III 1. „Eurospider Retrieval System ‚Relevancy'", S. III-70) in einer eigenen Ebene abgespeichert werden. Die Mustererkennung kann zur Kategorisierung von Texten in neue Themen benutzt werden. Mit Muster ist eine starre Umgebung (Kontext) eines Ausdrucks (z.B. Wort) gemeint, die in einer gewissen Art und Weise gestaltet sein muss, damit der Ausdruck einer bestimmten Kategorie angehört.
Wenn zum Beispiel einer Zahl das Wort *Prozent* folgt und davor möglicherweise ein Partikel steht, dann handelt es sich um eine Prozentangabe (vgl. Abbildung 17).

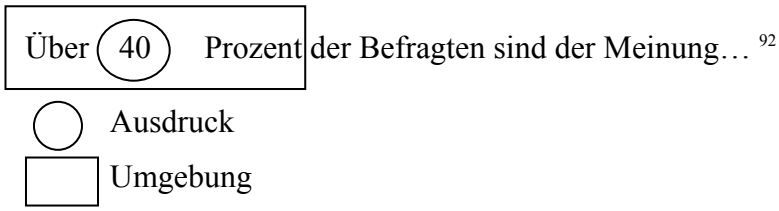


Abbildung 17

Mit der Konzeptsensorsprache können solche Muster (vgl. Abbildung 19) einfach beschrieben werden.

[92] Die Phrase stammt aus der Computerzeitung [Konradin-Verlag, 1998].

Die Konzeptsensorsprache ist in den Indexer des *Relevany*-Systems wie folgt eingebunden:

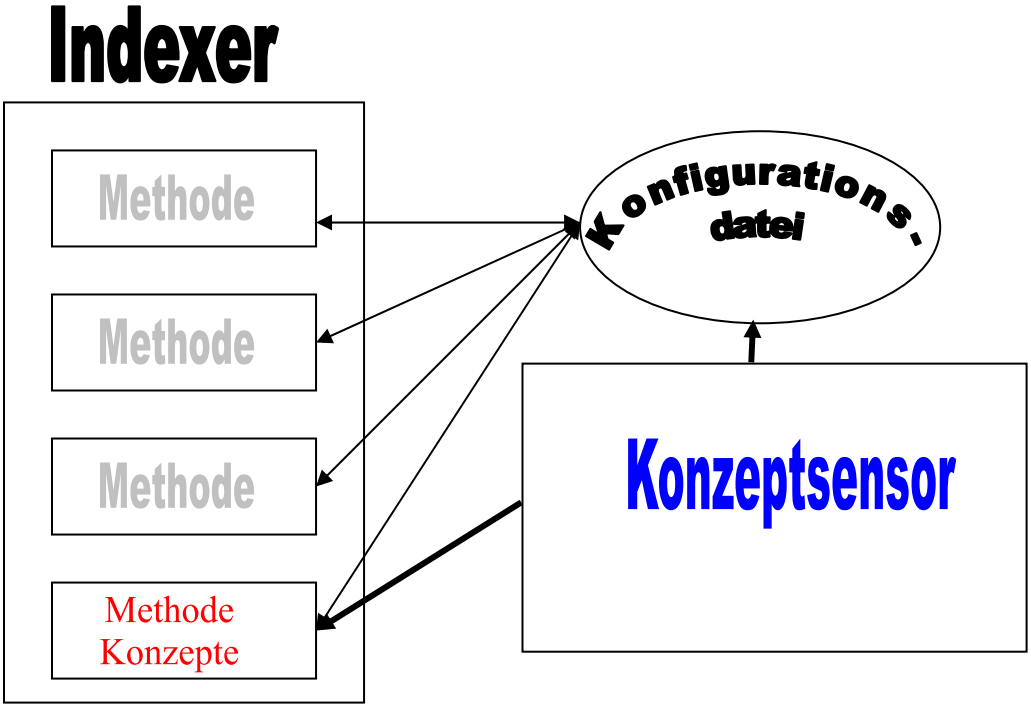


Abbildung 18

Der Indexer enthält verschiedene Methoden (vgl. Kapitel III 1. „Eurospider Retrieval System ‚Relevancy'"), welche Dokument und Anfrage für den Retrieval-Prozess analysieren. Die Konzepte, welche zur Mustererkennung geschaffen und in der Konzeptsensorsprache implementiert worden sind, werden als eine Methode, die in Submethoden unterteilt ist, im Indexer verankert. Die Abspeicherung der Ausgabe (engl. output) erfolgt in einer eigenen Ebene (engl. level) und steht anderen Methoden zur Weiterverarbeitung zur Verfügung (vgl. Kapitel III 1. „Eurospider Retrieval System ‚Relevancy'") In der Konfigurationsdatei werden die verschiedenen Methoden miteinander verbunden und es wird festgelegt, ob im Indexierungsvorgang eine Methode verwendet werden soll (aktiv) oder nicht (inaktiv).

2.2. Die Syntax der Konzeptsensorsprache

Konzepte sind zweigeteilt. Neben dem Namen und der Sprachdeklaration besitzen sie immer einen Beschreibungsteil (‚IF-Teil'), der die Muster enthält, welche erkannt werden sollen, und einen Bedingungsteil (‚CONDITION-Teil'), in dem mögliche Einschränkungen stehen, wann das Muster gefunden werden soll, sowie explizit formulierte Aktionen, die bei erfolgreicher Mustererkennung ausgeführt werden. Der Bedingungsteil ist optional. Falls ein Muster, das im Beschreibungsteil definiert ist, in einem Text gefunden wird und im Bedingungsteil keine explizit formulierte Aktion steht, wird das Konzept automatisch als wahr erkannt und keine weitere Aktion folgt. Die Mustererkennung ist gelungen. Indem eine explizit aufgeführte Ak-

tion *create* definiert wird, kann als Folge eine eigene Kategorie definiert werden, der die gefundenen Elemente des Konzept zugeordnet werden (vgl. Abbildung 21, S. III-80).

Über eine solche Syntax lassen sich Dinge ausdrücken wie „Wenn (*IF*) einem Vornamen ein grossgeschriebenes Wort folgt, dann (*CONDITION*) ist es ein Nachname, falls es nicht ein Auslöser für einen Firmennamen ist“. Die Syntax eines Konzepts sieht folgendermassen aus:

CONCEPT „*Name*“ LANGUAGE „*Sprache*“
IF
Regeln zur Erkennung von Mustern
CONDITION
Bedingungen, die die Mustererkennung einschränken
END

Abbildung 19

Die Konzeptsensorsprache kann auf alle Sprachen angewendet werden, die der Indexer bereitstellt (Deutsch, Englisch, Französisch, Italienisch und Spanisch). Gleiche Konzepte können auf mehr als eine Sprache angewendet werden, falls dies unter LANGUAGE deklariert wurde.

Die gesuchten Muster, welche im Regelteil definiert werden, bestehen aus Wert-Merkmal-Paaren, die über boole'sche Operatoren miteinander verknüpft werden können. Ein Wert-Merkmal-Paar besteht aus einem Term und einer dazugehörigen Kategorie. Terme können einzelne Wörter (*von*, *Peter*, *company*), einzelnen Zeichen (*&*, -, */* etc.), Zahlen (*0*, *10*, *333* etc.) oder Konzeptnamen (Konzepte, die an einer anderen Stelle definiert wurden) sein. Die Kategorie definiert den Term genauer, z.B. *“von“:WORD* oder *“8“:NUMBER*. Wert-Merkmal-Paare können auch mit einem beliebigen Term einer Kategorie definiert werden, indem der Term durch ein Fragezeichen ersetzt wird und so einen beliebigen Wert der im Wert-Merkmal-Paar definierten Kategorie annehmen kann (z.B. *?:WORD*).

Die folgenden booleschen Operatoren sind im Regelteil erlaubt:

1. logisches *und* (AND)

   Beispiel 20:

   ("aargauisch":STEM **AND** $?:WORD)

   Das Wort *aargauisch* soll in allen Flexionsarten gefunden (genauer: der fiktiv gebildete Stamm wird gesucht) und zur Verarbeitung in derjenigen Form weitergereicht werden, in der es im Text steht (flektierte Form).

2. logisches *oder* (*OR*)

   Beispiel 21

   ($"de":WORD) **OR** ($"van":WORD)

   Es soll *de* oder *van* gefunden werden.

3. Negation (*NOT*)

   Beispiel 22:

   $?:POSFANAMEPART AND **NOT** "all_filter":CONCEPT

   Das gesuchte Wort der selbst definierten Kategorie „*möglicher Firmennameteil*“ darf nicht im Konzept „all_filter“ (fasst mehrere Filter zusammen), gefunden werden.

4. Sequenz von Wörtern (*;*)

   Beispiel 23:

   (($"New":CAPWORD)**;**($"York":CAPWORD))

   Gesucht wird das Muster *New* direkt gefolgt von *York*.

5. Suchfenster (*?Zahl*)

   Beispiel 24:

   ("triggerword_in_front":CONCEPT);
   ("pos_fa_name_part_in_quotation_marks":CONCEPT
   AND $?:POSFANAMEPART);
   ("":SIGN **?2**)

   Gesucht ist ein Firmentriggerwort, gefolgt von einem möglichen Firmennamen in Anführungszeichen, dem ein Anführungszeichen nach höchstens zwei Wörtern folgt.

Ein einfaches Konzept, welches keine zusätzlichen Bedingungen beachten muss, sieht folgendermassen aus:

```
CONCEPT "stopwords_for_filter"
IF      ("apple":STEM)
        OR ("bull":STEM)
        OR ("seagate":STEM)
        OR ("sun":STEM)
        OR ("sharp":STEM)
        OR ("shell":STEM)
        OR ("tulip":STEM)
        OR ("touchstone":STEM)
        OR ("sunrise":STEM)
END
```

Abbildung 20

Das Konzept zählt Wörter auf, die im Filter für einteilige Firmennamen vorkommen, jedoch trotzdem zu den Firmennamen zählen und folglich nicht aus der Firmennamenliste mit einteiligen Wörtern eliminiert werden sollten.

Im Bedingungsteil (CONDITION) eines Konzepts, stehen Einschränkungen der im ersten Teil beschriebenen Regeln. Diese Einschränkungen beziehen sich jeweils global auf das ganze Dokument oder auf einen Abschnitt, dessen Grösse definiert werden kann, während die Regeln lokal, von der direkten Umgebung abhängig, beschrieben werden. Im Bedingungsteil können die Konditionen wie folgt beschrieben werden:

1. **find(„Name“:CONCEPT)**

   Die im „IF-Teil“ definierten Regeln werden nur ausgeführt, wenn irgendwo im Dokument ein bestimmtes Konzept gefunden wird. Die Suche über das gesamte Dokument kann nur mit der Klasse CONCEPT ausgeführt werden.

   Beispiel 25:

   CONDITION **find("initial_of_name":CONCEPT)**

   Das Konzept, welches Initialen definiert, soll innerhalb des Dokuments gefunden werden.

2. **find(„Term“:Kategorie, -/+N)**

   Die im „IF-Teil“ definierten Regeln treffen nur zu, falls ein Term einer gewissen Kategorie in der Distanz N (eine positive Zahl) gefunden wird. Minus bedeutet, dass die Einschränkung vor der Regel gesucht werden soll, Plus, dass die Einschränkung nach der Regel im Raum N gesucht werden soll. Die Suche des Wert-Merkmal-Paars, welches zur Einschränkung der Regeln dient, ist hier auf einen Teil des Dokuments beschränkt und somit gezielter angewendet als *find* (vgl. Punkt 1. S. III-79) über das gesamte Dokument.

   Beispiel 26:

   NOT **find("triggerword_in_front":CONCEPT, -1)**

   Um einen gültigen Personennamen zu finden, der über die Regeln im „IF-Teil“ genauer definiert ist, darf dem möglichen Personenamen kein Firmentrigger folgen, sonst handelt es sich um einen Firmennamen.

3. **overlaps_with(„Term“:Kategorie)**

   Die Regeln im „IF-Teil“ werden nur dann angewendet, wenn das Wert-Merkmal-Paar im Bedingungsteil teilweise oder ganz in den Wert-Merkmal-Paaren der Regeln eingeschlossen ist.

Beispiel 27:

NOT **overlaps_with("stopwords_for_filter":CONCEPT)**

Die aus den Regeln generierten Wörter dürfen nicht mit den Stoppwörtern des Filters (vgl. Beispiel 27) überlappen.

Die boole'schen Operatoren des „IF-Teils" (z.B. NOT, OR) dienen gleichermassen zur Verbindung einzelner Konditionen im Bedingungsteil der Konzepte. Es gibt noch weitere Bedingungen, die nicht aufgelistet werden, da sie für die in diesem Projekt erzeugten Konzepte irrelevant sind.

Innerhalb der Konzeptsensorsprache besteht die Möglichkeit, über Konzepte neue Kategorien zu generieren, die in Wert-Merkmal-Paaren weiter verwendet werden können. Im Regelteil definiert man die Terme, die der neu kreierten Kategorie angehören sollen. Über die Variable *$* werden die definierten Terme weitergereicht, um im Bedingungsteil der neuen Kategorie über den Befehl *create* zugeordnet zu werden.

```
CONCEPT "double_names"
IF
        (($"New":CAPWORD);($"York":CAPWORD))
CONDITION
        find("capitalized_word":CONCEPT);
        create("$0 $1":GEONAME)
END
```

Abbildung 21

Wenn Kategorien nicht innerhalb eines Konzepts über Regeln beschrieben werden können, so besteht die Möglichkeit, sie über einen regulären Ausdruck innerhalb der Konfigurationsdatei zu definieren. Über reguläre Ausdrücke besteht die Möglichkeit, ein gesuchtes Muster mit einzelne Buchstaben zu definieren, während in Konzepten das Muster über die vorgegebenen Kategorien (vgl. Tabelle 4, S. III-81) beschrieben werden muss. Die Kategorie CAPWORD, welche z.B. in Abbildung 21 Verwendung findet, wird innerhalb der Konfigurationsdatei über einen regulären Ausdruck beschrieben, der besagt, dass ein grossgeschriebenes Wort immer mit genau einem Grossbuchstaben beginnt und mindestens ein Kleinbuchstabe folgen muss.

Konzepte werden in der Konzeptsensorsprache als Module zusammengefasst. Jedes Modul erhält einen einmaligen Namen. Die Angabe, auf welche Sprache das Konzept bzw. mehrere Konzepte angewendet werden sollen, kann zusammen mit dem Konzeptnamen oder mit dem Modulnamen angegeben werden (LANGUAGE). Falls die Sprache im Modulnamen definiert wird, so gilt sie für alle Konzepte in diesem Modul und muss nicht jedes Mal wieder angegeben werden (vgl. Abbildung 22)

```
MODULE „name“ LANGUAGE „language“
IMPORT „Name“
TYPEDEF NEWTYPE
CONCEPT “Name1”
IF
….
CONDITON
…
END
CONCEPT “Name2”
…
ENDMODULE
```

Abbildung 22

Die Konzeptmodule werden im aktiven Konzeptsensor kompiliert und als Methode in den Indexer eingefügt. Es besteht die Möglichkeit, in Modulen andere aufzurufen und deren Funktionalität mitzuverwenden. Selbst definierte Kategorien können in anderen Modulen durch einen entsprechenden Aufruf, ohne dass man sie neu definieren müsste, verwendet werden.

Zu den Kategorien, welche in der Standardeinstellung der Konzeptsensorsprache innerhalb der Konfigurationsdatei definiert worden sind, zählen folgende:

Kategorie	**Beschreibung**	**Beispiel**
NUMBER	eine Ziffer (0-9) oder eine Sequenz von Ziffern (10-∞)	1, 12, 123 etc.
SIGN	spezifische Zeichen	-, /, &, “, , etc.
STEM	Ein Wort wird über den Stemmer von Relevancy auf eine künstliche Stammform reduziert und umfasst alle Vorkommen der gekürzten Form des Worts.	works → works, Work, worker etc.
WORD	Wörter, die in Kleinschreibung umgewandelt wurden und keine diakritischen Zeichen mehr besitzen	zuerich, peter, firma etc.
TAG	Markierungen zur Strukturierung eines Texts (SGML, HTML etc.)	Titel, Document etc.
CONCEPT	Konzepte	capitalized_word, stop_words_for_filter, filter etc.

Tabelle 4

In diesem Projekt wurden alle Kategorien ausser TAG verwendet, da die Erkennung von Eigennamen auch in unstrukturierten Texten möglich sein sollte und somit keine Gebundenheit an eine bestimmte Textannotation entstehen soll.

Die Kategorie *STEM* umfasst keine linguistischen Grundformen (Lemmata), sondern lediglich Formen, welche aus spezifischen Reduktionsregeln der Wörter hervorgegangen sind (vgl. Kapitel III 1. „Eurospider Retrieval System ‚Relevancy’“, S. III-70). Falls ein Wort weder

über eine Regel gekürzt werden kann, noch im Lexikon steht, wird keine Grundform generiert. Bei Fremdwörtern oder Eigennamen geschieht dies recht häufig. Entsprechende Probleme sind in Kapitel III 2.4. „Probleme innerhalb der Konzeptsensorsprache und ihre Grenzen“ genauer erläutert.

2.3. Erweiterungen in der Konzeptsensorsprache zur Erkennung von Eigennamen

Wie oben beschrieben, können zusätzlich neue Kategorien in der Konzeptsensorsprache definiert werden. Weil Mustererkennung mit der Konzeptsensorsprache bis jetzt ohne Gross-und Kleinschreibung funktionierte, gibt es keine Kategorien, die eine Unterscheidung zwischen Grossbuchstaben und Kleinbuchstaben machen. Da aber die Grossschreibung eines Worts eine zentrale Rolle in den Algorithmen zur Erkennung von Eigennamen einnimmt, mussten entsprechende Kategorien über einen regulären Ausdruck innerhalb der Konfigurationsdatei neu geschaffen werden. Neben den beiden Kategorien *grossgeschriebenes Wort* und *Ausdruckt in Grossbuchstaben* musste ein Initial und ein Akronym (Firmenakronym) definiert werden.

Kategorie	**Definition**	**Beispiel**
CAPWORD (capitalized word)	Genau ein Grossbuchstabe gefolgt von Kleinbuchstaben	Meier, Sonne
CAPLET (capital letters)	Eine Sequenz von Grossbuchstaben	AG, BERLIN
INITIAL	Ein Grossbuchstabe, gefolgt von einem Punkt, oder ein Grossbuchstabe und ein Kleinbuchstabe, gefolgt von einem Punkt	P. Ch.
ACR (acronym)	Buchstabenfolge aus mindestens zwei Grossbuchstaben oder einem & und Grossbuchstaben (über regulären Ausdruck definiert)	UBS, AT&T

Tabelle 5

Die Kategorie *Akronym* wird zusätzlich zum regulären Ausdruck über Konzepte definiert, die wiederum unter anderem die Kategorien *CAPLET* und *CAPWORD* verwenden. In der erweiterten Definition von Akronymen sind auch Kombinationen von Grossbuchstaben oder Wörtern mit Zahlen erlaubt (z.B. *3Com*).

Die folgenden Kategorien wurden über Konzepte neu definiert und sind zum internen Gebrauch bestimmt. „Intern“ bedeutet, dass die neu definierte Kategorie von anderen Konzepten weiter verwendet werden kann, der Indexer jedoch keine sichtbare Ausgabe erzeugt.

Namen	Funktion	Beispiel	Verwendung
LASTNAMEPART	Teil eines Nachnamens, der früher, den Stand, der Person, bzw. ihre Herkunft, markiert hat	van, de, O'	Personennamenerkennung
NAMEPART	Vornamen	Peter, Anna	Personennamenerkennung
POSLASTNAME	Regel, aus welchen Teilen ein Nachname zusammengesetzt sein könnte	Grund-Ludwig, Müller Maguh, Kanzler	Personennamenerkennung
LASTNAME	Nachname, der neben dem gesammelten Personennamen gelernt wird	Grund-Ludwig, Müller Maguh, Göbel	Personennamenerkennung
HNAMEPART	zwei Vornamen oder Initialen mit Bindestrich verbunden	Hans-Peter, H.-Peter, Peter-H., P.-H.	Personennamenerkennung
DNAMEPART	eine Kombination, bestehend aus zwei Vornamen	Anna Barbara	Personennamenerkennung (fürs Englische auf Kombination aus Initial und Vorname erweitert)
GEOABBR	geographische Abkürzung	F, ZH, CH	geographische Namenserkennung
GEOADJ	geographisches Adjektiv	afrikanisch, Zuger	geographische Namenserkennung (nur im Deutschen)
GEONAME	geographischer Eigenname	Zuerich, Asien, Allgaeu	geographische Namenserkennung
POSFANAMEPART	Regel, wie ein Teil eines Firmennamens aussehen könnte	Apple, ABB, J.	Firmennamenerkennung
FANAMEPART	Trigger, die Teil des Firmennamens sind	AG, Company, B.V.	Firmennamenerkennung

Tabelle 6

Neben den Standardausgaben (Grundform): *TYPE=1, POS=21431, VALUE='beschaftigung'*[93] und Wortform im Text: *TYPE=10, POS=21431, VALUE='halbtagsbeschaeftigung'* sind die Endresultate der Eigennamenerkennung, nämlich Personennamen, geographische Namen und Firmennamen, vom Indexer als Ausgabe zusätzlich generiert worden.

[93] Die Grundform ist hier grammatikalisch unkorrekt. Von ihr können aber die restlichen Formen problemlos abgeleitet werden.

Kategorie	Definition	Beispiel einer Ausgabe
PERS	Personenname	TYPE=96, POS=23522,40727,58722, VALUE='Michael Vogel' TYPE=96, POS=63420,63455,63479, VALUE='Schroeder' TYPE=96, POS=17108, VALUE='Ulrich von Weizsaecker'
GEO	geographischer Name	TYPE=95, POS=42360, VALUE='Osnabrueck' TYPE=95, POS=68841,68841, VALUE='Orem / Utah' TYPE=95, POS=49438, VALUE='North Sioux City'
FA	Firmennamen	TYPE=97, POS=59589, VALUE='Acorn' TYPE=97, POS=42267, VALUE='Andersen Consulting' TYPE=97, POS=11799, VALUE='CAD & Art'

Tabelle 7

Zur Evaluation wird diese Ausgabe weiter verwendet (vgl. Kapitel IV „Evaluation“).

Da auf Grund der Algorithmen zur Eigennamenerkennung die Notwendigkeit bestand, über spezifische Muster gefundene Namen (z.B. *Bill Gates*) in einem andern Kontext erneut zu finden (‚meinte *Gates* nach …’), musste das ‚lokale *find’* (vgl. Punkt 2., S. III-79) im Bedingungsteil entsprechend erweitert werden. Der zweite Suchvorgang wurde nachgeahmt, indem ein Term einer gewissen Kategorie innerhalb eines definierten Suchraums (+/-N) erneut gefunden werden musste, damit er über den Befehl *create* einer anderen Klasse zugeordnet werden konnte (vgl. Abbildung 23). Der Wert 10000 für N im folgenden Konzept ist von mit empirisch ermittelt worden. Grössere Suchbereiche haben mehr Fehlanalysen gebracht, während in kleinere Suchbereiche zu viele Nachnamen nicht gefunden worden sind.

```
CONCEPT "find_last_name"
IF
/*Falls irgendwo im Dokument ein möglicher Nachname gefunden wird,*/
        ($?:POSLASTNAME)
CONDITION
/*und dieser mögliche Nachname im Bereich 10000 Wörter davor als Nachname er-
kannt worden ist,*/
        find("$0":LASTNAME, -10000);
/*Ort, an dem die Kategorie LASTNAME definiert wurde.*/
                (find("full_name":CONCEPT)
        OR find("last_name":CONCEPT)
        OR find("suffix_name":CONCEPT));
/*Ort, an dem die Kategorie POSLASTNAME definiert wurde.*/
        find("pos_last_name":CONCEPT);
/*Der Personenname soll nur generiert worden, falls er nicht bereits in einem anderen
Konzept erkannt worden ist.*/
        NOT overlaps_with("full_name":CONCEPT);
        NOT overlaps_with("last_name":CONCEPT);
        NOT overlaps_with("suffix_name":CONCEPT);
/*Zuweisung des Terms zur Kategorie Personenname.*/
        create("$0":PERS)
END
```

Abbildung 23

Diese Erweiterung ist im Projekt verwendet worden, um Teile von gelernten Personennamen (Nachnamen) und Firmennamen wiederzufinden, und wurde entsprechend in den Modulen *learn_pers_name* und *learn_fa* implementiert.

Mit der Vorgehensweise, welche in der Erweiterung vom ‚lokalen *find'* (vgl. Punkt 2, S. III-79) im Bedingungsteil gewählt worden ist, kann nicht garantiert werden, dass der gefundene Name tatsächlich einer bestimmten Kategorie angehört. Falls z.B. in einem Dokument die Person *Harrison Ford* vorkommt, der Nachname *Ford* und der Firmenname *Ford*, so wird die Firma *Ford*, falls sie von keinen Firmentriggern umgeben ist, fälschlicherweise als Personenname erkannt. Mehr zu dieser allgemeinen Problematik ist in Kapitel III 2.4. „Probleme innerhalb der Konzeptsensorsprache und ihre Grenzen" zu finden.

Insgesamt sind in diesem Projekt 17 Module für Deutsch und 16 Module für Englisch neu entstanden (3 Module zur Personenerkennung, 8 bzw. 9 Module zur Erkennung von geographischen Eigennamen und 5 Module zur Erkennung von Firmennamen). Die Module enthalten Konzepte, welche in Kapitel III 3. „Ressourcen für die Erkennung von Entitäten in deutschsprachigen Texten", Kapitel III 4. „Implementierung der Eigennamenerkennung in deutschsprachigen Texten", Kapitel III 5. „Ressourcen für die Erkennung von Entitäten in englischen Texten" und Kapitel III 6. „Implementierung der Eigennamenerkennung in englischen Texten" genauer beschrieben sind.

2.4. Probleme innerhalb der Konzeptsensorsprache und ihre Grenzen

Die Konzeptsensorsprache soll auf eine leichte und verständliche Art und Weise das Erstellen von Regeln erlauben, um eine Anfrage bei einem Retrievalvorgang gezielter stellen zu können und das Auffinden relevanter Dokumente zu erleichtern. Somit ist die Architektur der Sprache sehr einfach gehalten. Das bringt jedoch mit sich, dass gewisse Strukturen wie beispielsweise Iteration oder Rekursion, die in höheren und komplexeren Programmiersprachen (z.B. *Perl* oder *Java*) durchaus gängig sind, nicht in der Konzeptsensorsprache formuliert werden können. Sie müssen auf einfachere Strukturen reduziert werden, welche allerdings an Mächtigkeit einbüssen. Die Länge der Firmennamen musste beispielsweise auf sieben Elemente[94] beschränkt werden, da es keine Möglichkeit gab, innerhalb einer Schlaufe auszudrücken, dass so lange Teile von Firmennamen aneinandergereiht werden sollen, bis ein Stoppwort oder ein kleingeschriebenes Wort erscheint. In meinen Textkorpora sind nie Firmennamen vorgekom-

[94] Als Elemente von Firmennamen gelten folgende Teile: ein grossgeschriebenes Wort, ein oder mehrere aneinander geschriebene Grossbuchstaben, ein Akronym, ein Initial, eine Zahl gefolgt von Grossbuchstaben, ein grossgeschriebenes Wort oder ein oder mehrere Grossbuchstaben, gefolgt von einer Zahl oder einem Zeichen (*&*, -, /), gefolgt von einem grossgeschriebenen Wort oder einem oder mehreren Grossbuchstaben (vgl. Tabelle 6, S. III-83).

men, deren Länge sieben Elemente überschritten hat. Weil aber neue Firmennamen fortlaufend und möglichst einzigartig gebildet werden, ist es denkbar, dass längere Namen in Zukunft vorkommen werden. Die Iteration ist in Abbildung 24 als Pseudocode dargestellt.

```
Search_Forward {
IF (sequence_of_tokens(W0,W1,...WN))
THEN            IF (Trigger_in_Front (W0))
        THEN FOREACH (W1,...WN)
                IF (Posible_FA (W1,...WN))
                        AND (W1,...,WN) != Stopword_for_Trigger_in_Front (W)
THEN {mark FA (W1,...,WN)}
}
```

Abbildung 24

In der Konzeptsensorsprache musste für jedes Element, das nach einem Firmentrigger (vgl. Kapitel II 4. „Algorithmen“) folgte, ein eigenes Konzept geschrieben werden. Somit musste man pro Algorithmus sieben verschiedene Konzepte schreiben, die allerdings nur Firmennamen bis zur Länge sieben korrekt erkennen (vgl. Abbildung 25). Firmennamen, welche aus mehr als sieben Teilen bestehen, werden nur teilweise erkannt und bei der Evaluation als falsch erkannte Entitäten gewertet.

```
CONCEPT "seven_fa_names_follow"
IF
    ("triggerword_in_front":CONCEPT);
    ("pos_fa_name_part":CONCEPT AND $?:POSFANAMEPART);
    ("pos_fa_name_part":CONCEPT AND $?:POSFANAMEPART);
    ("pos_fa_name_part":CONCEPT AND $?:POSFANAMEPART);
    ("pos_fa_name_part":CONCEPT AND $?:POSFANAMEPART);
    ("pos_fa_name_part":CONCEPT AND $?:POSFANAMEPART);
    ("pos_fa_name_part":CONCEPT AND $?:POSFANAMEPART);
    ("pos_fa_name_part":CONCEPT AND $?:POSFANAMEPART)
CONDITION
    NOT overlaps_with("stop_words_for_fa_name_follows":CONCEPT);
    NOT overlaps_with("seven_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT find("triggerword_behind":CONCEPT, +1);
create("$0 $1 $2 $3 $4 $5 $6":FA)
END
```

Abbildung 25

Die Personennamenerkennung büsst durch Reduktion einer Schleifenkonstruktion in einzelne Konzepte ebenfalls an Mächtigkeit ein. Es ist nicht möglich, so lange nach Vornamen oder Initialen zu suchen, bis ein unbekanntes, grossgeschriebenes Wort auftritt, das einen möglichen Nachnamen verkörpert. Die Anzahl der Vornamen oder Initialen vor einem Nachnamen musste auf zwei (De) bzw. vier (En) beschränkt werden.

Das Wiederfinden gelernter Entitäten ist in den ursprünglichen Algorithmen von Volk (vgl. Kapitel II „Theoretische Grundlagen“) ein wichtiges Element. Während das Wiederfinden

von Personennamen (Nachnamen) innerhalb eines Dokuments geschieht, soll der Suchvorgang von Teilen von gelernten Firmennamen (Akronyme) über alle Dokumente erfolgen. In der Konzeptsensorsprache kann der Wiederfindungsprozess mit einem zuvor gelernten Wort nur innerhalb der Eingabedatei erfolgen. Falls die Eingabedatei aus mehreren Dokumenten besteht[95], wird folglich die Personennamenerkennung ungenau Falls die Eingabedatei aus genau einem Dokument besteht, kann keine dokumentenübergreifende Suche von Firmennamen implementiert werden.

Die Lösung dieses Problems besteht darin, die Eingabedatei auf genau ein Dokument zu beschränken, da man meist die Ausgabe genau eines Dokuments benötigt, um zu bestimmen, ob es relevant zu einer Anfrage ist oder nicht. Die Firmennamenerkennung lässt sich in diesem Fall nicht vollständig in der Konzeptsensorsprache realisieren. Eine ‚Zwei-Weg-Lösung' wird notwendig. Dann würde in einem ersten Schritt ein Teil der Firmennamen, die mit Triggern erkannt worden sind, in einer Liste abgespeichert. Anschliessend ergänzt man die entsprechende Liste mit neu erzeugten Akronymen und filtriert daraus falsche, einteilige Firmennamen. Das Erzeugen der Akronyme kann nicht mit der Konzeptsensorsprache gelöst werden, sondern muss in eine Programmiersprache implementiert werden, die Bearbeitung von einzelnen Buchstaben in bestehenden Wörtern (vgl. nächster Abschnitt) erlaubt (z.B. *Perl*). Die neue, ergänzt und filtrierte Liste dient als Ressource, um weitere Firmennamen in demselben Dokument zu finden, in dem zuvor bereits ein Teil der Firmennamen erkannt worden ist. Ob dieser Suchvorgang wiederum in der Konzeptsensorsprache implementiert wird, steht offen. Diese ‚Zwei-Weg-Lösung' ist nicht Teil dieses Projekts und wurde bis jetzt noch nicht realisiert.

Die Konzeptsensorsprache ist zum Suchen von Elementen (Wörter, Phrasen etc.) konzipiert, bietet jedoch nicht die Möglichkeit, gefundene Elemente als Buchstabenfolge zu erkennen und entsprechend zu bearbeiten (Stringbearbeitung). So gibt es beispielsweise keine Möglichkeit, ein über ein Konzept gefundenes Wort um gewisse Buchstaben zu reduzieren oder zu erweitern und einen erneuten Suchvorgang mit diesem bearbeiteten Wort zu starten.

Beispiel 28:

... Chief Executive Office ***Michael Spindler*** *...*
Der Nachname wird auf Grund des Vornamens gelernt.
Mit ***Spindlers*** *Ausmusterung verliert ...*[96]

[95] Die Eingabedatei wird vom System als ein grosses Dokument verstanden. Falls mehrere Dokumente als Eingabedatei dienen sollen, dann müssen die Dokumente in eine Datei geschrieben werden, weil das System Dokumentgrenzen innerhalb der Datei ignoriert.

[96] Die Phrasen stammen aus der Computerzeitung [Konradin-Verlag, 1998].

Der Nachname im Genitiv in Beispiel 28 sollte ebenfalls durch Hinzufügen eines *s* gefunden werden. Eine Manipulation des gelernten Nachnamens ist jedoch nicht möglich, weshalb der zweite Name im Dokument nicht erkannt wird.

Im Prinzip gäbe es die Möglichkeit, in der Konzeptsensorsprache ein Wort in Grundform zu suchen. Allerdings beschränkt sich diese Möglichkeit auf Wörter, die im Lexikon des Stemmers stehen oder auf die gewisse Regeln angewendet werden können. Zu diesen zählt nicht die Abtrennung des Buchstabens *s*, da es schwierig festzustellen ist, wann es sich tatsächlich um ein Genitiv-*s* handelt und wann *s* Teil des Wortes ist. Dazu wäre eine Lemmabestimmung nötig, die nicht durchgeführt wird. Eine Möglichkeit, Buchstabenfolgen mit Hilfe eines Programms zu manipulieren, wäre wünschenswert, ist aber in der Konzeptsensorsprache nicht vorgesehen. Somit mussten Genitivformen für die Erkennung der Eigennamen in separaten Listen zusätzlich aufgeführt werden.

In wenigen Fällen (geographische Adjektive und Firmentrigger) konnte der Stemmer eingesetzt werden. Um ein Wort in gestemmter Form innerhalb eines Dokuments zu finden, in der Ausgabe jedoch die ursprüngliche Schreibweise des Wortes zu erhalten, muss das Wort in Grundform (STEM) gesucht, aber in ursprünglicher Form (WORD) dem Output übergeben werden (vgl. Abbildung 26).

```
CONCEPT "isch_adj"
IF
        ("afghanisch":STEM AND $?:WORD)
        …
CONDITION
        create("$0":GEOADJ)
END
```

Abbildung 26

In Abbildung 26 wird das Wort *afghanisch* in Dokument als Grundform gesucht (Deklinationen des Wortes werden ebenfalls gefunden) und in derjenigen Form (deklinierten Form) der Kategorie *geographisches Adjektiv* übergeben, wie sie im Text steht. Auch hier ist allerdings zu bemerken, dass der Stemmer nicht für alle Adjektive eine Grundform kennt und somit Fehler entstehen.

Lemmatisierung oder gar Wortartenbestimmung, die weder in der Konzeptsensorsprache, noch im Indexer implementiert sind, würde die Eigennamenerkennung erleichtern und präzisieren. Da bis jetzt der Stemmer von *Eurospider* ausgereicht hat, hielt man es für unnötig, weiter linguistische Analysesysteme zu erwerben, um sie in *Relevancy* einzusetzen. Die Kosten sind bis jetzt zu hoch gewesen, verglichen mit dem Nutzen, den solche vertieften Analysen bringen würden. Im Fall von Eigennamenerkennung wäre der Nutzen jedoch gegeben.

3. Ressourcen für die Erkennung von Entitäten in deutschsprachigen Texten

In den Ressourcen für Eigennamenerkennung in deutschsprachigen Texten wurden Trigger und Entitäten aufgelistet. Als Grundlage dienten die Ressourcen von Volk (vgl. Tabelle 8), welche ich in diversen Bereichen mit Hilfe des Internets und einer Sammlung von Kurznachrichten (SDA)[97] erweitert habe. Einige Ressourcen konnten für die Eigennamenerkennung in englischen Texten ebenfalls übernommen werden. Näheres dazu folgt in den nächsten Kapiteln.

Name	**Beispiele**	**Grösse**
Titel/Anreden	Prof., Ing.	5 Einträge
Vornamen	Aaron, Thomas, Zygmund	ca.18'007 Einträge
Teile des Nachnamens	von, van, de	3 Einträge
Städte- /Ortschafts- und Ländernamen	Halle, Zürich	ca. 10'000 Einträge
isch-Adjektive	japanisch, sächsisch	ca. 100 Einträge
geo-Adjektive auf *–er*	Münchner, Basler	ca. 460 Einträge
Bindestrichkompositum als Trigger für Firmennamen	-Tochter, -Chef, -Präsident	3 Einträge
Trigger am Anfang eines Firmennamen	Firma	1 Eintrag
Trigger am Ende eines Firmennamens	GmbH, B.V., Group	22 Einträge
Begrenzer von Firmennamen vorne	Software, Trade, Investitionen	3 Einträge
Begrenzer von Firmennamen hinten	Die, Gesellschaft, Lieferant	13 Einträge
Abkürzungsverzeichnis	d.h., Zimm., …	714 Einträge
Englisch-Deutsch-Lexikon	Sonne – sun, until – bis	22'830 Einträge
Stoppwörter für Filter	Apple, Shell	8 Einträge

Tabelle 8

3.1. Ressourcen zur Erkennung von Personennamen (De)

Die Ressourcen zur Erkennung von Personennamen wurden prinzipiell von Volk übernommen und entsprechend erweitert. In der Konzeptsensorsprache habe ich die Ressourcen als Modul mit Konzepten implementiert, in welchen die entsprechenden Wörter aufgelistet sind. Bindestrichvornamen und doppelte Vornamen bilden die Ausnahme, da es sich nicht um eine Liste, sondern um eine Regel handelt, die die Liste der Vornamen weiterverwendet. Die Regel

[97] vgl. Fussnote 53, S. 38.

für Bindestrichkomposita besagt, dass zwei Vornamen, zwei Initiale oder eine Kombination aus Initial und Vorname, getrennt durch einen Bindestrich, ein Bindestrichkompositum bilden, das die gleiche Funktion wie ein einzelner Vorname übernimmt (vgl.Abbildung 27)[98]

```
/*Bindestrich-Vorname: Kompositum aus Vornamen oder Initial (z.B. Hans-Jürg, H.-
Jürg, Jürg-H. oder H.-J.*/
CONCEPT "hyphenated_pre_pers_name"
IF
        (("pre_pers_name":CONCEPT AND $?:NAMEPART) OR ($?:INITIAL));
        ($"-":SIGN);
        (("pre_pers_name":CONCEPT AND $?:NAMEPART) OR ($?:INITIAL))
CONDITION
        find("initial_of_name":CONCEPT);
        create("$0 $1 $2":HNAMEPART)
END
```

Abbildung 27

Die Regel für doppelte Vornamen schreibt vor, dass zwei aufeinander folgende Vornamen gleich wie ein allein stehender Vorname fungieren. Genitive mussten in eine eigene Liste aufgenommen werden, da in der Konzeptsensorsprache keine Möglichkeit besteht, einzelne Buchstaben eines Wortes abzutrennen oder hinzuzufügen. Die Möglichkeit, Vornamen auf ihre Grundform zu reduzieren (stemmen), ist ebenfalls nicht anwendbar. Da die Vornamen weder einer Wortart zugeordnet werden können, noch im Wörterbuch aufgelistet sind, kann der Stemmer keine Reduktionsregeln durchführen. Der in *Relevancy* implementierte Stemmer ist für Personennamen nicht brauchbar.

Name	**Funktion**	**Beispiel**	**Kategorie**
pers_name_ resources	Ressourcen zur Erkennung von Personennamen		Modul
sal_pers_name	Liste mit Anreden (spezifisch für Deutsch).	Prof., Herr	Konzept
pre_pers_name	Liste mit Vornamen (aus dem englischen, deutschen, französischen und asiatischen Bereich).	Anna, Zheng	Konzept
pre_pers_double_ name	Regel, um zwei folgende Vornamen zu erkennen.	Anna Barbara Thian Seong	Konzept
hyphenated_pre_ pers_name	Regel, um zwei Vornamen oder Initiale, getrennt durch einen Bindestrich, zu erkennen.	Hans-Peter Marie-Ann H.-P.	Konzept
last_name_part	Liste mit Titeln und anderen Teilen von Nachnamen.	Müller Müller-Meier	Konzept
genitiv_pre_pers_ name	Liste mit Vornamen im Genitiv.	Annas, Zhengs	Konzept
stop_word_list_for _single_names	Liste mit Vornamen, welche selten Vornamen sind, wenn sie alleine vorkommen.	Tor, August	Konzept

Tabelle 9

[98] Die Funktion eins Vornamens ist, als Trigger für einen Nachnamen zu dienen.

Die ursprüngliche Liste der Anreden (*sal_pers_name*) und die Liste der Titel (*last_name_part*) von Volk ist durch eigenes Wissen leicht erweitert worden. Gewisse Anreden sind speziell auf den deutschen Sprachraum ausgerichtet (z.B. *Herr* oder *Professor*). Die Liste der Titel umfasst gemischte Einträge aus verschiedenen Sprachregionen (*van*, *della* etc.). Die Vornamensliste wurde um wenige seltene Vornamen gekürzt, welche zu falschen Analysen führten, weil sie zugleich einen Teil des Allgemeinwortschatzes ausmachen, z.B. *Abo* oder *Ach*. Wenn einem solchen möglichen Vornamen ein grossgeschriebenes Wort folgt, so handelt es sich eher selten um einen Eigennamen, und unter Beibehaltung des besagten Vornamens würden viele Fehlanalysen entstehen. Die Vornamensliste habe ich möglichst umfassend erstellt. Sie enthält Einträge aus dem englischen, deutschen, französischen und asiatischen Sprachraum.

Die Liste der Vornamen im Genitiv entstand aus der Nominativliste. Indem einfache Regeln (Anfügen von *es* bei einer Endung auf *s*, Anfügen von *s* in allen anderen Fällen) auf die Vornamen im Nominativ angewendet wurden, entstanden mögliche Vornamen im Genitiv. Um fehlerhafte Generierungen der Genitivform zu vermeiden und die Ressource nicht unnötig zu vergrössern, sind alle Vornamen mit der SDA-Textkollektion (vgl. Fussnote 53, S. 38) verglichen worden. Die Vornamen müssen mindestens einmal in der Textkollektion auftauchen, um in die Liste aufgenommen zu werden. Somit ist die Anzahl der Vornamen im Genitiv (3242 Einträge) kleiner als diejenige im Nominativ (17198 Einträge). Die grosse Differenz ist unter anderem damit zu erklären, dass Vornamen im Genitiv nur dann auftreten, wenn sie alleine, ohne Nachnamen, verwendet werden (z.B. ***Peters*** *Haus*). Dies kommt in Zeitungstexten eher selten vor.

3.2. Ressourcen zur Erkennung von geographischen Namen (De)

Die Erkennung von geographischen Eigennamen basiert nur auf Listen. Folglich erhalten hier die Ressourcen eine besonders wichtige Funktion. Alle Eigennamen, welche nicht in einer der folgenden Listen stehen, werden in einem Text nicht gefunden. Die Grundsammlung von Volk wurde über Internetrecherchen und Extraktion aus dem SDA-Textkorpus von mir ergänzt und vervollständigt. Die Ressourcensammlung ist nicht vollständig. Weitere Kategorien wie Gewässer-, Quartier- oder Gebirgsnamen zählen per Definition ebenfalls zu geographischen Entitäten, werden in diesem Projekt jedoch nicht beachtet, da diese Erweiterung einen erneut grossen Zeitaufwand mit sich bringen würde. Geographische Eigennamen habe ich bewusst auf die unten aufgezählten Kategorien (vgl. Tabelle 10) beschränkt, da diese, meiner Meinung nach, zu den wichtigsten und häufigsten Vertretern von geographischen Eigennamen

zählen. Die geographischen Eigennamen stammen mehrheitlich aus dem deutschen und englischen Sprachraum.

Weil die Listen der einzelnen Kategorien teilweise sehr gross sind, ist jede Kategorie innerhalb eines eigenen Moduls implementiert worden. Die Unterteilung in verschiedene Konzepte ist damit zu begründen, dass bei mehrteiligen, geographischen Eigennamen jedes Wort einzeln als Teil des gesamten Konstrukts (*geographischer Eigenname*) erkannt werden muss. Der einzelne Name *Zürich* wird im Bedingungsteil eines Konzepts der Variable *$0* zugewiesen und als geographischer Eigenname kreiert, während ein mehrteiliger Name wie *New York* den Variablen *$0* und *$1* zugewiesen und entsprechend als „Kompositum“ kreiert wird.

Name	**Funktion**	**Beispiel**	**Kategorie**
geo_resources_city_list	**Ressource mit Städtenamen hauptsächlich aus der Schweiz, Deutschland, Österreich, USA und Kanada**		**Modul**
single_city_names	Liste mit einteiligen Städtenamen	Aachen, Zuerich	Konzept
special_single_city_names	Liste mit einteiligen Städtenamen, welche Teil des Allgemeinwortschatzes sind und somit nur in spezifischem Kontext als Entität gelten	Zug, Laufen	Konzept
single_city_names_genitiv	Liste mit einteiligen Städtenamen im Genitiv	Aachens, Zuerichs	Konzept
special_single_city_names_genitiv	Liste mit einteiligen Städtenamen im Genitiv, welche Teil des Allgemeinwortschatzes sind und somit nur in spezifischem Kontext als Entität gelten	Zugs, Damms	Konzept
hyphenated_names	Liste mit Bindestrich-Städtenamen	Aathal-Seegraeben, Zuerich-Kloten	Konzept
double_names	Liste mit zweiteiligen Städtenamen	Abu Dahbi, Yuka Flat	Konzept
triple_names	Liste mit dreiteiligen Städtenamen	Affoltern am Albis, North Sioux City	Konzept
quadruple_names	Liste mit vierteiligen Städtenamen	Rust am Neusiedler See, Scheffau am Wilden Kaiser	Konzept
slash_citiesI	Liste mit Städtenamen, welche einen Schrägstrich als Abkürzung verwenden	Affoltern a/Albis, Ronco s/Ascona	Konzept
slash_citiesII	Liste mit dreiteiligen Städtenamen, welche mit einem Schrägstrich geschrieben werden	Korat/Nakhon Ratchasima	Konzept
slash_citiesIII	Liste mit zweiteiligen Städtenamen, welche mit einem Schrägstrich geschrieben werden	Breil/Brigels, Frankfurt/M.	Konzept
st_names	Liste mit Städtenamen, welche mit St. beginnen	St. Lucia, St. Ingberg	Konzept
geo_resources_nationen	**Ressource mit Ländernamen (=Nationen) aus aller Welt**		**Modul**
single_nation_names	Liste mit einteiligen Ländernamen	Afghanistan, Zimbabwe	Konzept

single_nation_names_ genitiv	Liste mit einteiligen Ländernamen im Genitiv	Afghanistans, Zimbabwes	Konzept
double_nation_names	Liste mit zweiteiligen Ländernamen	American Samoa, Volksrepublik China	Konzept
triple_nation_names	Liste mit dreiteiligen Ländernamen	McDonalds Islands, Vereinigte Arabi-sche Emirate	Konzept
quadruple_nation_names	Liste mit vierteiligen Ländernamen	British Indian Ocean Territory, Vereinigte Staaten von Amerika	Konzept
hyphenated_nation_names	Liste mit zweiteiligen Ländernamen, die mit einem Bindestrich kombiniert werden	Bosnien-Herzegowina, Saudi-Arabien	Konzept
geo_resources_kantone_ laender	**Ressource mit Kantons- und Länder-namen (=Staaten, Bundesländer etc.) hauptsächlich aus der Schweiz, Österreich, Deutschland, USA und Kanada**		**Modul**
single_kantone_laender_ names	Liste mit einteiligen Kantons- und Ländernamen	Aargau, Wyoming	Konzept
single_kantone_laender_ names_genitiv	Liste mit einteiligen Kantons- und Ländernamen im Genitiv	Graubuendens, Wy-omings	Konzept
hyphenated_kantone_ laender_names	Liste mit einteiligen Kantons- und Ländernamen, die über einen Bindestrich kombiniert werden (Nomi-nativ- und Genitivformen)	Baden-Wuerttemberg, Schleswig- Holsteins	Konzept
double_kantone_laender_ names	Liste mit zweiteiligen Kantons und Ländernamen (Nominativ- und Geni-tivformen)	Appenzell Ausserrhodens, South Dakota	Konzept
triple_kantone_laender_ names	Liste mit dreiteiligen Kantons- und Ländernamen	Demokratische Re-publik Kongo	Konzept
geo_resources_regions_ islands	Ressource mit Region- und Inselnamen aus der Schweiz, Deutschland und den USA		Modul
single_regions_islands_ names	Liste mit einteiligen Regionen- und Inselnamen	Allgaeu, Turksinseln	Konzept
single_regions_islands_ names_in_genitiv	Liste mit einteiligen Regionen und Inselnamen im Genitiv	Suedostasiens, Zentraleuropas	Konzept
hyphenated_regions_ islands_names	Liste mit einteiligen Regionen- und Inselnamen, die über einen Bindestrich kombiniert werden	Nord-Europa, Ostwestfalen-Lippe	Konzept
double_regions_islands_ names	Liste mit zweiteiligen Regionen- und Inselnamen (in gestemmter Form)	Bayrischer Wald, Skandinavische Laender	Konzept
geo_resources_continents	**Ressource mit Kontinenten**		**Modul**
continent_names	Liste mit Kontinenten (Nominativ- und Genitivformen)	Europa, Australiens	Konzept
geo_resources_adj	**Ressource mit geographischen Ad-jektiven haupsächlich aus der Schweiz, Deutschland, Österreich, USA und Kanada**		**Modul**
isch_adj	Liste mit einteiligen, geographischen Adjektiven, die auf *-isch* enden (in gestemmter Form)	afrikanisch, ostasia-tisch	Konzept

isch_adj_II	Liste mit zweiteiligen, geographischen Adjektiven, die auf *-isch* enden und mit einem Bindestrich unterteilt werden (Teil nach dem Bindestrich in gestemmter Form)	baden- wuerttembergisch, us-amerikanisch	Konzept
er_adj_typeI	Liste mit einteiligen, geographischen Adjektiven, die auf *-er* enden (immer unflektiert und grossgeschrieben)	Giessener, Zuozer	Konzept
er_adj_typeII	Liste mit dreiteiligen, geographischen Adjektiven, die auf *-er* enden und über einen Bindestich miteinander verbunden sind (immer unflektiert und grossgeschrieben)	Baden-Badner, West-Berliner	Konzept
er_adj_typeIII	Liste mit zweiteiligen, geographischen Adjektiven, die auf *-er* enden (immer unflektiert und grossgeschrieben)	Bad Harzburger, New Yorker	Konzept
geo_resources_abbr	**Ressource mit geographischen Abkürzungen aus aller Welt**		**Modul**
geo_resources_abbr	Liste mit geographischen Abkürzungen	AUT, VS	Konzept

Tabelle 10

Da in der Konzeptsensorsprache keine Stringbearbeitung möglich ist und die geographischen Eigennamen dem Stemmer unbekannt sind und somit nicht reduziert werden können[99], mussten die Genitivformen jeweils in eigene Konzepte eingefügt werden. Um die Ressource nicht unnötig mit seltenen Genitivformen aufzubauschen, wurden die Genitivlisten, welche aus den Nominativlisten generiert worden sind, über dem SDA-Textkorpus (vgl. Fussnote 53, S. 38) abgeglichen. Entsprechend der Generierung der Vornamen im Genitiv, wurden Regeln aufgestellt, die Genitivformen aus den bereits vorhandenen Nominativformen erzeugen.

Beispiel 29:

Regeln zur Generierung von Genitivformen
Zürich + s → Zürichs
Grimselpass + es → Grimselpasses
**Paris + es → Parises*

Um falsche oder seltene Eigennamen im Genitiv aus der Liste zu eliminieren, wurden nur diejenigen Namen in der Liste belassen, welche im gesamten SDA-Textkorpus mindestens einmal aufgetreten sind.

Das Vorgehen, um die Liste der geographischen Adjektive zu erweitern, glich demjenigen zur Gewinnung von Genitivformen. Prinzipiell galt, dass aus Städtenamen (*New York*, *Amsterdam* etc.) Adjektive auf die Endung *er*, mit einem Grossbuchstaben beginnend, abgeleitet werden, aus den restlichen Namen (*Schweiz*, *Europa*, *Baden-Wuerttemberg* etc.) Adjektive mit der Endung *isch*.

[99] Vgl. Kapitel III 2.4. „Probleme innerhalb der Konzeptsensorsprache und ihre Grenzen“.

Beispiel 30:

Regeln zur Generierung von geographischen Adjektiven auf er:

Normalfall: Anhängen der Endung *er*
London → Londoner

Wenn Städte auf *e* enden, dann wird ein *r* angehängt.
Karlsruhe → Karlsruher
Wenn Städtenamen auf *en* enden, dann wird die Endung entfernt und *er* angehängt.
Bremen → Bremer

Die Liste mit den Ableitungen der Adjektive auf die Endung *–isch* habe ich von manuell erstellt, weil es für eine Ableitung fast keine Regelmässigkeiten gibt (z.B. *Schweden* → *schwedisch*, *Kalifornien* → *kalifornisch*, *Deutschland* → *deutsch* etc.).

Die Ableitungsregeln waren sehr einfach gestaltet und generierten nur teilweise korrekte Adjektive. Spezialfälle wie *Münchner* oder *Hannoveraner* mussten am Ende von Hand hinzugefügt werden. Falsche Adjektive oder sehr seltene wurden von mir über einen Filterschritt mit Hilfe des SDA-Textkorpus eliminiert. Das Adjektiv musste mindestens einmal in einem SDA-Text gefunden werden, damit es nicht aus der Liste gestrichen wurde.

Gewisse Eigennamen lassen sich in verschiedene Module einteilen (z.B. *New York,* der Staat, und *New York*, die Stadt oder *Zürich*, der Kanton und *Zürich*, die Stadt). Um die Ressourcen nicht unnötig mit doppelten Einträgen in verschiedenen Modulen zu vergrössern, habe ich eine Reihenfolge erstellt, nach der geographische Eigennamen eingeordnet werden. Gleichlautende Namen erhalten nur in einer der folgenden Kategorien entsprechend der Reihenfolge eine Eintrag: Städtenamen > Namen von Nationen > Namen von Kantonen/Staaten/Ländern > Regions-/Inselnamen (vgl. Kapitel I 4.2. „Verwendung von Ressourcen“). Entsprechend dieser Reihenfolge sind ebenfalls die englischen Ressourcen in den Konzepten geordnet.

Die Liste *special_single_city_names* enthält Städtenamen, die bezüglich ihrer Kategorie ambig sind. Je nach Kontext kann es sich um eine Entität oder ein Nomen handeln. Die Liste wurde durch einen Vergleich mit dem Deutsch-Englisch-Lexikon gewonnen, das als Filter bei der Firmennamenerkennung dient. Wörter, die sowohl im Lexikon als auch in der Liste mit Städtenamen vorkamen, wurden bei den Städtenamen gelöscht und in die Liste *special_single_city_names* eingefügt. Einige Wörter musste ich anschliessend ergänzen, weil es sich um flektierte Formen handelte, die nicht im Lexikon stehen, oder weil sie öfter einer anderen Kategorie wie geographische Eigennamen angehören (z.B. Firmennamen). Die erstellte Liste weist allerdings einige Lücken auf, wie die Evaluation gezeigt hat (vgl. Kapitel IV 3.2. „Evaluation von geographischen Eigennamen (De)“).

3.3. Ressourcen zur Erkennung von Firmennamen (De)

Die Ressourcen zur Firmennamenerkennung bestehen aus einer geringen Zahl Triggern und Stoppwörtern. Die als Grundlage dienenden Listen von Volk habe ich vor und während des Evaluationsschritts kontinuierlich mittels eigener Recherchen erweitert. Neue Triggerwörter entstanden dadurch, dass Wörter, welche in der Umgebung eines Firmennamens auftreten, aus dem Computer-Zeitungs-Korpus [Konradin-Verlag, 1998] gesammelt und entsprechend der Auftretungsfrequenz geordnet worden sind. Hochfrequente Wörter bezüglich der Compute Zeitung wurden in die entsprechende Triggerwortliste eingefügt.

Name	Funktion	Beispiel	Kategorie
fa_name_resources	**Ressourcen für die Erkennung von Firmennamen**		**Modul**
pos_fa_name_part	Definition, wie ein Teil eines Firmennamens aussehen könnte.	Microsoft, J&	Konzept
pos_fa_name_part_in_ quotation_marks	Definition, wie ein Teil eines Firmennamens, der in Anführungszeichen steht, aussehen könnte.	„Microsoft, „AT	Konzept
triggerword_in_front	Liste mit Wörtern, denen ein Firmenname folgen kann, die aber nicht Teil der Entität sind.	Firma, Netzwerkspezialist	Konzept
triggerword_ in_front_part_of_fa	Liste mit Wörtern, denen ein Firmenname folgen kann, die selbst Teil der Entität sind.	CCS, Universitaet	Konzept
hyphaneted_suffix	Liste mit Wörtern, die als Bindestrichsuffix einen Firmennamen auslösen.	-tochter, -chef	Konzept
triggerword_behind	Liste mit Wörtern (ein und zweistellig), die am Ende eines Firmennamens stehen und Teil der Entität sind.	Partners, B.V.	Konzept
triggerword_behind_accr	Definition, dass Akronyme am Ende eines Firmennamens als Trigger fungieren.	NV, BV	Konzept
stop_words_for_fa_name _ follows	Liste mit Wörtern, welche einen Firmennamen begrenzen, der auf Grund eines Triggers vor einem Firmennamen gefunden wurde.	Trade, Unternehmen	Konzept
stop_words_for_fa_name _ in_front	Liste mit Wörtern (ein- und mehrteilig), welche einen Firmennamen begrenzen, der auf Grund eines Triggers am Ende eines Firmennamens gefunden wurde.	Der, Konzern	Konzept
filter	**Ressource, welche ein Englisch-Deutsch-Lexikon enthält, das zum Filtrieren von einteiligen Firmennamen dient.**		**Modul**
filterI	Liste mit englischen und deutschen Wörtern aus dem Allgemeinwortschatz in Grossbuchstaben, gängigen Abkürzungen und spezifisch zusammengestellten Wörtern.	ABRECHNEN, usw., WWW	Konzept
filter	Alle Einträge aus filterI und filterII ohne Wörter aus der Liste stopwords_for_filter	ABRECHNEN, abrechnen, usw.	Konzept
stopwords_for_filter	Liste mit Wörtern, welche im Filter stehen, aber trotzdem häufige Firmennamen sind.	Sun, Shell	Konzept

filterII	**Ressource, welche ein Englisch-Deutsch-Lexikon enthält, das zum Filtrieren von einteiligen Firmennamen dient (filterII hat dieselbe Funktion wie filter, musste allerdings abgetrennt werden, da sonst das Modul zu gross wird und nicht mehr funktioniert).**		**Modul**
filterII	Liste mit englischen und deutschen Wörtern aus dem Allgemeinwortschatz in Grundform.	abrechnen, youth	Konzept

Tabelle 11

Die Definitionen *pos_fa_name_part* und *pos_fa_name_part_in_quotation_marks* gehören im Prinzip nicht zu den Ressourcen, da es sich hierbei nicht um Listen mit Wörtern handelt, sondern eher um Regeln. Da diese Konzepte aber Wörter beschreiben, die innerhalb einer Ressource gesammelt werden könnten, dies jedoch wegen der grossen Anzahl nicht getan wird, sind sie bei den Ressourcen angesiedelt worden.

Triggerwörter, welche vor einem Firmennamen erscheinen, sind in manchen Fällen Teil der Entität (z.B. Akronyme), in andern nicht (z.B. *Firma*). Um dies zu unterscheiden, wurden zwei Konzepte (*triggerword_in_front* und *triggerword_in_front_part_of_fa*) angelegt. Die Idee, dass ein Trigger auch Teil des Firmennamens sein kann, ist neu entstanden, so dass die Ressourcenliste von keiner bestehenden Liste profitieren konnte. Sie entstand auf Grund eigener Recherchen. Die Liste der Trigger, welche nicht Teil der Entität sind, habe ich stark erweitert. Allerdings sind nur wenige Einträge grundsätzlich neu. Viele Trigger (z.B. *Konzern*, *Spezialist*) mussten als Komposita (z.B. *Mischkonzert*, *Consultingfirma*, *Netzwerkspezialist*) nochmals aufgeführt werden, da sie als Fremdwort dem Stemmer unbekannt waren und dieser somit keine reduzierte Form generieren konnte (vgl. Kapitel III 2.3. „Erweiterungen in der Konzeptsensorsprache zur Erkennung von Eigennamen“). Im Gegensatz zu den Triggern, die vor einem Firmennamen stehen, sind Trigger, die am Ende eines Firmennamen stehen weniger sprachabhängig. Die gängigen Abkürzungen (*AG*, *Co.*) sind Standard in vielen Sprachen und stammen entsprechend aus aller Welt.

Ich habe eingeführt, dass Trigger, die am Ende eines Firmennamens stehen, immer Teil der Entität sind, weil es sich bei den Triggern um spezifizierende Information zu einem Firmennamen handelt (z.B. *Corp.* oder *Holding*). Wörter, die zwingend einem Firmennamen folgen, jedoch nicht Teil der Entität selbst sind, kommen in der Computer-Zeitung praktisch nie vor (vgl. Beispiel 31). Meist folgen den Firmennamen Verben wie *haben*, *gehen* oder *sein,* die hinter jedem beliebigen Nomen stehen können und somit als Trigger ungeeignet sind.

Beispiel 31:

„Nur die Spitze des Eisbergs ist <org>Apples</org> **Verlust** von 69 Millionen Dollar im ersten Quartal des neuen Geschäftsjahres."
[Konradin-Verlag, 1998]

Verlust könnte ein Trigger sein. In der Computer-Zeitung würde das Wort jedoch genau zu einer korrekten und einer falschen Analyse führen und ist somit unbrauchbar…
Der **Verlust** lag bei 2,4 Millionen…*

Für Akronyme, die hinter einem Firmennamen als Trigger fungieren, musste ein eigenes Konzept erstellt werden, da es sonst zu Doppelanalysen[100] derselben Entität gekommen wäre, weil die Definition von Akronym sich teilweise mit den Firmentriggern überschneidet. *AG* kann als Kombination aus zwei Grossbuchstaben einmal als Akronym, einmal als Firmentrigger gedeutet werden. Das Erkennen der Firmennamen über Trigger wird dem Erkennen über Akronyme vorgezogen. Diese Restriktion kann nur vorgenommen werden, wenn zwei Konzepte, so wie in diesem Fall, vorhanden sind.

Die Stoppwörter zur Begrenzung von Firmennamen wurden grösstenteils so belassen und aus den Algorithmen von Volk übernommen. Sie sind stark sprachabhängig und mussten fürs Englische entsprechend angepasst werden.

Das Modul *filter* dient zum Filtern von falsch gelernten einteiligen Firmennamen. Einteilige Firmennamen sind in diesem Projekt einerseits über Trigger gelernt worden, andererseits durch Ableitungen von mehrteiligen Firmennamen entstanden. Ableitung bedeutet, dass die einzelnen Teile sowie Kombinationen dieser Teile und Akronyme ebenfalls als Firmennamen gelten.

Beispiel 32:

Ableitung von Firmennamen:
Siemens Nixdorf Informationssysteme AG →
Siemens Nixdorf Informationssysteme
Siemens Nixdorf
Siemens
Nixdorf Informationssysteme
Informationssysteme[101]

In diesem Fall wird der Filter verwendet, um Informationssysteme aus der neu generierten Liste von Firmennamen zu streichen, da das Wort kein Firmenname ist.

Der Filter (*filter* und *filterII*) enthält Einträge aus einem kleinen Deutsch-Englisch-Lexikon und einem Abkürzungsverzeichnis, das ich von Volk übernehmen konnte. Das Lexikon muss-

[100] Unter Doppelanalyse ist gemeint, dass dieselbe Entität über zwei verschiedene Regeln erkannt wird und in der Ausgabe somit zweimal erscheint.
[101] Dies ist eine falsche Ableitung, die entsprechend gefiltert werden muss.

te in eine neue Form umgewandelt werden, in der auf jeder Zeile ein eigenes Wort steht. Für den Filter ist es nicht von Bedeutung, welches deutsche Wort mit einem englischen übersetzt wird oder umgekehrt (Übersetzungsfunktion eines Lexikons), sondern, dass die einzelnen Wörter dem deutschen oder englischen Allgemeinwortschatz angehören. Das Lexikon enthält mit 45660 Wörtern zu wenige Einträge, als dass der gesamte Allgemeinwortschatz von Deutsch und Englisch enthalten wäre. Das Lexikon musste deshalb mit Abkürzungen und einer spezifischen Liste ergänzt werden. Die spezifische Liste enthält vor allem Produktenamen (*NT*, *OS* etc.) oder Abkürzungen aus dem Computerbereich (*PC*, *WWW* etc.).

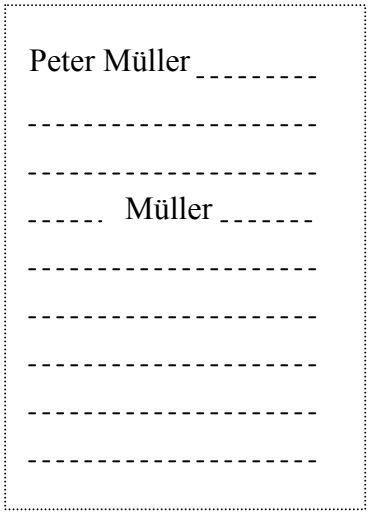
Die Evaluation hat gezeigt, dass sich die verwendeten Ressourcen sowohl für Firmen- und Personennamenerkennung als auch für Erkennung von geographischen Entitäten mit Ausnahme des Filters bezüglich Grösse und Vollständigkeit als gut erwiesen haben. Obwohl der Filter als grösste Ressource in diesem Projekt integriert wurde, hat er sich als unzureichend erwiesen. Problematisch sind Einträge, die in einem Lexikon nicht vorkommen (z.B. Produktenamen), grossgeschrieben werden und trotzdem keine Firmennamen sind. Genaueres zur Evaluation von Ressourcen und Regeln ist in Kapitel IV „Evaluation“ zu finden.

4. Implementierung der Eigennamenerkennung in deutschsprachigen Texten

Die Strategie, die prinzipiell zur Eigennamenerkennung angewendet worden ist, lautet entsprechend der Idee von [Volk und Clematide, 2001] *Lernen, Anfügen* bzw. *Anwenden, Vergessen* bzw. *Filtern*. Die Strategie wird für Personennamen, geographische Namen und Firmennamen jeweils unterschiedlich eingesetzt.

4.1. Erkennung von Personennamen (De)

Die Strategie zur Personennamenerkennung lautet *Lernen, Anwenden, Vergessen.* Personenamen werden über einen Kontext erkannt. Dabei lernt das System Nachnamen. Diese Nachnamen werden erneut innerhalb des Dokuments gesucht, um weitere Personennamen zu finden, die in keinem der oben beschriebenen Kontexten stehen. Am Ende des Dokuments „vergisst“ das System alle Nachname wieder. Der Anwendungsschritt (das erneute Suchen von Nachnamen) ist somit auf einen lokalen Bereich (Dokument) beschränkt. Damit soll vermieden werden, dass falsche Personennamen erkannt werden. Man geht davon aus, dass ein Nachname, der im Kontext eines Vornamens oder einer Anrede bereits einmal im Dokument aufgetaucht ist, wenn er erneut alleine (ohne Trigger) vorkommt, es sich immer noch um einen Personennamen handelt. Er soll entsprechend erkannt und zugeordnet werden (vgl. Abbildung 28).



Dokument

Mit grosser Wahrscheinlichkeit handelt es sich hier um den Personennamen *Müller* und nicht um die Berufsbezeichnung.

Abbildung 28

Das erneute Suchen und Finden eines bereits gelernten Nachnamens ist Eigennamenerkennung und keine Eigennamenidentifikation. Man kann nicht feststellen, ob *Peter Müller* und *Müller* dieselbe Person sind. Dass Eigennamenerkennung betrieben wird und keine Identifikation, gilt für das gesamte Projekt. Eine Identifikation würde weit mehr Probleme mit sich bringen und ist vermutlich nur mit der Konzeptsensorsprache nicht zu lösen (vgl. Kapitel III 2.4. „Probleme innerhalb der Konzeptsensorsprache und ihre Grenzen“).

Die Algorithmen zum Erlernen eines Personennamens wurden so übernommen, wie ich sie in Kapitel II 4. „Algorithmen“ erläutert habe. Die Länge eines Personennamens musste allerdings beschränkt werden, da zirkuläre Konstruktionen in der Konzeptsensorsprache nicht zu implementieren sind (vgl. Kapitel III 2.4. „Probleme innerhalb der Konzeptsensorsprache und ihre Grenzen“).

Die folgende Tabelle enthält alle für dieses Projekt erstellten Module und Konzepte sowie deren Funktionen.

Name	**Funktion**	**Kategorie**
pers_name_rec	**Erkennung von Personennamen**	**Modul**
pos_last_name	Regel zur Erkennung von möglichen Nachnamen	Konzept
full_pers_name	Regel zur Erkennung von ‚vollständigen Personennamen’: Titel; Vor-, Bindestrich- oder Doppelname; Nachname	Konzept
full_pers_name_type1	Regel zur Erkennung von ‚vollständigen Personennamen’: Anrede; Vor-, Bindestrich- oder Doppelname; Initial; Nachname	Konzept
full_pers_name_type2	Regel zur Erkennung von ‚vollständigen Personennamen’: Anrede, Vor-, Bindestrich- oder Doppelname; Titel; Nachname	Konzept

full_pers_name_type3	Regel zur Erkennung von ‚vollständigen Personennamen': Anrede, Vor-, Bindestrich- oder Doppelname; Initial; Titel; Nachname	Konzept
full_pers_name_type4	Regel zur Erkennung von ‚vollständigen Personennamen': Anrede, Vorname, Bindestrich, Initial, Titel, Nachname	Konzept
abbr_pers_name	Regel zur Erkennung von ‚abgekürzten Personennamen': Anrede; Initial; Nachname	Konzept
abbr_pers_name_type1	Regel zur Erkennung von ‚abgekürzten Personennamen': Anrede; Initial; Titel; Nachname	Konzept
abbr_pers_name_type2	Regel zur Erkennung von ‚abgekürzten Personennamen': Anrede; Initial; Initial; Titel; Nachname	Konzept
abbr_pers_name_type3	Regel zur Erkennung von ‚abgekürzten Personennamen': Anrede; Initial; Initial; Nachname	Konzept
last_name	Regel zur Erkennung von ‚Personennamen ohne Anrede': Initial; Initial; Nachname	Konzept
last_name_type1	Regel zur Erkennung von ‚Personennamen ohne Anrede': Vor-, Bindestrich-, Doppelname oder Initial; Nachname	Konzept
last_name_type2	Regel zur Erkennung von ‚Personennamen ohne Anrede': Vor-, Bindestrich-, Doppelname oder Initial; Titel; Nachname	Konzept
last_name_type3	Regel zur Erkennung von ‚Personennamen ohne Anrede': Vor-, Bindestrich- oder Doppelname; Initial; Nachname	Konzept
pre_name	Regel zur Erkennung von allein stehenden Vor-, Bindestrich- oder Doppelnamen	Konzept
genitiv_pre_name	Regel zur Erkennung von allein stehenden Vornamen im Genitiv	Konzept
genitiv_h_pre_name	Regel zur Erkennung von allein stehenden Bindestrichnamen im Genitiv	Konzept
genitiv_i_pre_name	Regel zur Erkennung von allein stehenden Bindestrichnamen im Genitiv, bei denen der erste Teil ein Initial ist.	Konzept
find_last_name	Ein Nachname soll erkannt werden, falls er in einer Kombination aus Vornamen, Titeln, Anreden etc. (vgl. restliche *pers_rec*-Konzepte) im Dokument vorgekommen ist.	Konzept

Tabelle 12

Die Definition eines Nachnamens, dass er nur aus grossgeschriebenen Wörtern besteht, ist erweitert worden, um auch zweiteilige Nachnamen mit und ohne Bindestrich zu erfassen. So gelten in der Implementierung ein grossgeschriebenes Wort (z.B. *Dreyer*), zwei aufeinander folgende, grossgeschriebene Wörter (z.B. *Meier Mül*ler) oder zwei grossgeschriebene Wörter mit Bindestrich getrennt (z.B. *Schmidt-Bleek*) als mögliche Nachnamen (*pos_last_name*).

Entsprechend den ursprünglichen Algorithmen gibt es in der Implementierung drei Grundmuster, um einen Personennamen zu erkennen: *full_pers-name*, *abbr_pers_name* und *last_name*.

Ein vollständiger Personenamen (*full_pers-name)* besteht immer aus einer Anrede und einem Nachnamen, Vorname und Titel sind optional. Ein abgekürzter Personenname (*abbr_pers_name)* beinhaltet als obligatorische Elemente Initiale und Nachnamen, optional kann ein Titel vor dem Nachnamen stehen. Die Anrede fällt weg. Beim Konzept *Nachnamen* (*last_name)* fällt ebenfalls die Anrede weg. Der Regelteil dieses Konzepts besagt, dass ein Vornamen oder Initial, gefolgt von einem Nachnamen, gefunden werden muss, wobei vor dem Nachnamen ein Titel stehen kann. Die Unterscheidung verschiedener Typen in diesen drei Klassen erfolgt, weil in der Konzeptsensorsprache Optionalität jeweils über zwei verschiedene Konzepte aufgelöst werden muss.

Die Erkennung von allein stehenden Vornamen ist von mir neu implementiert worden. Da mit einem Vornamen immer eine konkrete Person gemeint ist und der Vorname somit die Hauptfunktion eines Eigennamens (Benennung einer Sache) erfüllt, entspricht ein Vorname einer Entität und sollte somit als Personenname erkannt werden. Problematisch ist diese Erweiterung, da in Produktenamen, Firmennamen oder Titeln für Theaterstücke oder Filme oft Vornamen (oder allgemein Personennamen) vorkommen, denen aber eher einen appellativischen Charakter zukommt. Appellative sind Nomen, die mehrere Objekte über einen Namen zusammenfassen, z.B. *Müller* als Berufsbezeichnung (fasst alle Personen zusammen, die diesen Beruf ausüben). Eigennamen beziehen sich auf genau ein Objekt und identifizieren es (z.B. *Peter* ***Müller***). Die Hauptfunktion des Namens in einem Titel besteht in der appellativistischen Funktion, das heisst die Benennung einer Sache. Die Hauptfunktion eines Eigennamens ist jedoch Identifikation eines Objektes geben einer Klasse, die man auch propriale Eigenschaft nennt. Somit gehört in diesem Fall der Name nicht zur Klasse *Eigenname*. Eine klare Abgrenzung von Appellativ und Eigennamen ist bis anhin in der Forschung noch nicht gelungen (vgl. [Blanár, 2001], S. 35ff.).

In der Implementierung kann nicht zwischen Eigenname und Appellativ unterschieden werden, da ein Unterschied nur über Identifikation festgestellt werden kann. Eine solche ist je-

doch nicht Teil meines Projekts. Somit werden alle Namen als Eigennamen erkannt ohne Berücksichtigung, ob sie Teil eines Titels oder Firmennamens sind. Personennamen, welche in Firmennamen auftreten, werden nicht als Personennamen erkannt, sondern als Firmennamen.

Mit dem Konzept *find_last_name* wird die auf das Dokument begrenzte Suche von zuvor gelernten Nachnamen implementiert. Der Suchraum, in dem sowohl der Nachname allein als auch in einer Kombination mit Triggern gefunden werden muss, ist über eine absolute Zahl (N) definiert und muss somit manuell der Dokumentgrösse angepasst werden (vgl. Kapitel III 2.3. „Erweiterungen in der Konzeptsensorsprache zur Erkennung von Eigennamen“). Wenn der Suchraum entsprechend grossgewählt wird, ist sichergestellt, dass möglichst alle Nachnamen, die zuvor auf Grund von Triggern gelernt worden sind, auch tatsächlich gefunden werden. Problematisch bei diesem Ansatz war, dass, wenn ein falscher Nachname über die Trigger gelernt worden ist, dieser Fehler sich über die erneute Suche des falschen Nachnamens wiederholt. Ein Fehler zieht somit viele Fehler nach sich. Bei der Evaluation der deutschen Konzepte habe ich festgestellt, dass 15 von 535 verschiedenen Personennamen fehlten, weil sie von den bestehenden Konzepten nicht erkannt worden sind (vgl. Kapitel IV 3.1. „Evaluation von Personennamen (De)“Durchschnittlich werden bei einer Erkennung durch einen Trigger 2-3 weitere Namen durch erneutes Suchen des Nachnamens gefunden. Deshalb ist es wichtig, dass die Konzepte *full_pers_name*, *abbr_pers_name* und *last_name* korrekte Entitäten liefern.

4.2. Erkennung von geographischen Namen (De)

Die Strategie *Lernen, Hinzufügen* wurde hier nur zur Erweiterung der Ressourcen angewendet (vgl. Kapitel III 3.2. „Ressourcen zur Erkennung von geographischen Namen (De)“). Der Suchvorgang von geographischen Eigennamen läuft lediglich über den Vergleich von Wörtern im Text mit Listen, die geographische Eigennamen enthalten. Es werden keine Regeln verwendet, ausser derjenigen, die besagt, dass in einer Liste nachgeschaut werden muss, ob das vorliegende Wort ein geographischer Eigenname ist. Folglich werden nur geographische Eigennamen erkannt, die in den Ressourcen aufgeführt sind.

Die Erkennung von geographischen Entitäten konnte nicht direkt über die Ressourcenauflistung geschehen, sondern musste in einem eigenen Modul (*geo_rec*) implementiert werden, um zirkuläre Abhängigkeiten zwischen den Modulen zu vermeiden (vgl. Abbildung 29). Module dürfen nicht zirkulär aufeinander zugreifen, weil sonst unklar ist, wo, wann und wie die Mustererkennung beginnen soll. Eine korrekte Funktionsweise ist nicht mehr garantiert.

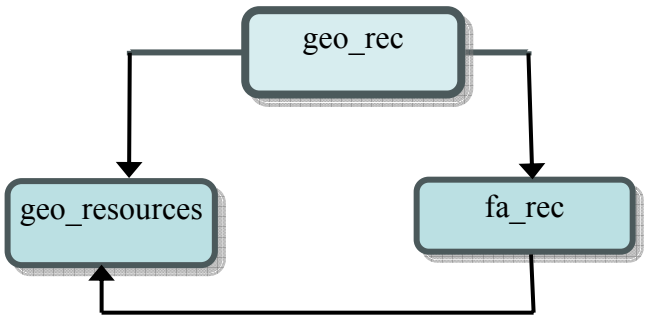


Abbildung 29

Wenn in Abbildung 29 das Modul *geo_rec* mit *geo_resource*s zusammenfällt entsteht eine solche zirkuläre Abhängigkeit (vgl. Abbildung 30). In den Konzepten ist die Abhängigkeit gleich wie in Abbildung 29 implementiert.

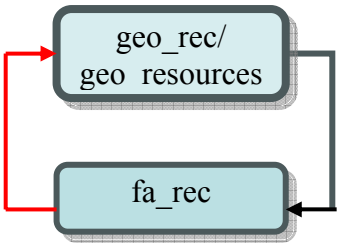


Abbildung 30

Die geographischen Eigennamen hängen sowohl von den Firmennamen als auch von den Personennamen ab. Es gelten folgende Regeln:

1. Geographische Namen (GEO), die in Verbindung mit einer Anrede (*Dr.*, *Herr* etc.) oder einem Vornamen oder beidem stehen, sind keine GEO sondern Personennamen (PERS).

   Beispiel 33: *Herr Rom*bach

2. Mehrteilige GEO, welche ebenfalls ein PERS sein könnten, bei ihnen aber keine Anrede oder kein Vorname davor steht, gelten als GEO.

   Beispiel 34: *St. Petersburg*

   St. wird von den Konzepten als Initial erkannt. Somit könnte *St. Petersburg* auch Kombination aus Initial und folgendem, grossgeschriebenem Wort gedeutet werden (*last_nametype1*) und die Fehlanalyse *Personenname* wäre die Folge.

3. Ein allein stehender, einteiliger GEO, der gleichzeitig ein Vorname sein könnte, wird immer als GEO analysiert, weil allein stehende Vornamen viel seltener vorkommen.

   Beispiel 35: *Austin*

4. Falls ein geographisches Adjektiv vor einem Firmennamen steht, dann gehört dieses Adjektiv nicht zum Firmennamen.

   Beispiel 36: ***Die*** *Düsseldorfer* ***Mannesmann AG***[102]

Die Module und darin enthaltenen Konzepte, welche ich zur Erkennung von geographischen Eigennamen in deutschen Texten verwendet habe, sind in der folgenden Tabelle aufgezählt:

Name	**Funktion**	**Kategorie**
geo_rec	**Regeln zum Erkennen von geographischen Entitäten**	**Modul**
find_single_geoname	Regel zum Finden eines einteiligen geographischen Namens	Konzept
find_hyphenated_geonames	Regel zum Finden eines geographischen Namens, der einen Bindestrich besitzt	Konzept
find_double_names	Regel zum Finden eines zweiteiligen geographischen Namens	Konzept
find_triple_names	Regel zum Finden eines dreiteiligen geographischen Namens	Konzept
find_quadruple_names	Regel zum Finden eines vierteiligen geographischen Namens	Konzept
find_slash_citiesI	Regel zum Finden eines Städtenamens, der einen Schrägstrich als Abkürzung verwendet	Konzept
find_slash_citiesII	Regel zum Finden eines dreiteiligen Städtenamens, der mit Schrägstrich geschrieben wird	Konzept
find_slash_citiesIII	Regel zum Finden eines zweiteiligen Städtenamens, der mit Schrägstrich geschrieben wird	Konzept
find_st_names	Regel zum Finden eines Städtenamens, der mit *St.* beginnt	Konzept
single_geoname_with_abbr	Regel zum Finden eines einteiligen geographischen Namens, dem eine geographische Abkürzung (als Präzision) folgt	Konzept
hyphenated_geonames_with_abbr	Regel zum Finden eines geographischen Namens, der einen Bindestrich besitzt und dem eine geographische Abkürzung (als Präzision) folgt	Konzept

[102] Es gibt keine klaren Regeln, die besagen, dass das geographische Adjektiv Teil des Firmennamens ist oder nicht. Dies ist jeweils abhängig von der Entscheidung der Person, die einen Text annotiert. Marc Rössler hat sich in diesem Fall entschieden, das geographische Adjektiv nicht als Teil des Firmennamens zu analysieren. Diese Entscheidung wurde von mir übernommen.

double_names_with_abbr	Regel zum Finden eines zweiteiligen geographischen Namens, dem eine geographische Abkürzung (als Präzision) folgt	Konzept
triple_names_with_abbr	Regel zum Finden eines dreiteiligen geographischen Namens, dem eine geographische Abkürzung (als Präzision) folgt	Konzept
quadruple_names_with_abbr	Regel zum Finden eines vierteiligen geographischen Namens, dem eine geographische Abkürzung (als Präzision) folgt	Konzept
slash_citiesI_with_abbr	Regel zum Finden eines Städtenamens, der einen Schrägstrich als Abkürzung verwendet und dem eine geographische Abkürzung (als Präzision) folgt	Konzept
slash_citiesII_with_abbr	Regel zum Finden eines dreiteiligen Städtenamens, der mit Schrägstrich geschrieben wird und dem eine geographische Abkürzung (als Präzision) folgt	Konzept
find_st_names_abbr	Regel zum Finden eines Städtenamens, der mit *St.* beginnt und dem eine geographische Abkürzung (als Präzision) folgt	Konzept
find_isch_adj	Regel zum Finden einteiliger, geographischer Adjektive, welche auf *-isch* enden	Konzept
find_isch_adj_II	Regel zum Finden mehrteiliger, geographischer Adjektive, welche auf *-isch* enden und über einen Bindestrich miteinander verbunden sind	Konzept
find_er_adj_typeI	Regel zum Finden einteiliger, geographischer Adjektive, welche auf *-er* enden	Konzept
find_er_adj_typeII	Regel zum Finden mehrteiliger, geographischer Adjektive, welche auf *-er* enden und über einen Bindestrich miteinander verbunden sind	Konzept
find_er_adj_typeIII	Regel zum Finden zweiteiliger, geographischer Adjektive, welche auf *-er* enden	Konzept

Tabelle 13

4.3. Erkennung von Firmennamen (De)

Die Strategie, welche zum Erkennen von Firmennamen angewendet worden ist, lautet *Lernen, Filtern, Anwenden*. Die Firmennamen werden auf Grund von Triggern vor oder nach einem Namen erkannt. Der so erhaltene Bestand wird erweitert, indem Teile der gelernten Firmennamen und Kombinationen mit Teilen der vorhandenen Liste angefügt werden (vgl. Beispiel 32) Weil bei der Erweiterung der Liste auch falsche Firmennamen hinzukommen, müssen die

einteiligen Firmennamen gefiltert werden, bevor man sie zur erneuten Suche im Text einsetzt (vgl. Kapitel II 4. „Algorithmen“).

Die folgende Tabelle enthält die Konzepte und Module, in welchen die Strategie umgesetzt worden ist.

Name	**Funktion**	**Kategorie**
fa_rec	**Regeln zum Erkennen von Firmennamen**	**Modul**
one_fa_name_follows	Regel, um einen einteiligen[103] Firmennamen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
one_fa_name_follows_in_ quotation_marks	Regel, um einen einteiligen Firmennamen in Anführungszeichen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
two_fa_names_follow	Regel, um einen zweiteiligen Firmennamen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
two_fa_names_follow_in_ quotation_marks	Regel, um einen zweiteiligen Firmennamen in Anführungszeichen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
three_fa_names_follow	Regel, um einen dreiteiligen Firmennamen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
three_fa_names_follow_inquot ation_marks	Regel, um einen dreiteiligen Firmennamen in Anführungszeichen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
four_fa_names_follow	Regel, um einen vierteiligen Firmennamen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
four_fa_names_follow_in_ quotation_marks	Regel, um einen vierteiligen Firmennamen in Anführungszeichen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
five_fa_names_follow	Regel, um einen fünfteiligen Firmennamen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
five_fa_names_follow_in_ quotation_marks	Regel, um einen fünfteiligen Firmennamen in Anführungszeichen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
six_fa_names_follow	Regel, um einen sechsteiligen Firmennamen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
six_fa_names_follow_in_ quotation_marks	Regel, um einen sechsteiligen Firmennamen in Anführungszeichen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept

[103] Als Teil eines Firmennamens zählen alle Kombinationen, welche in den Konzepten *pos_fa_name_part* und *pos_fa_name_part_in_quotation_marks* definiert werden. Einteilig muss nicht genau einem Wort entsprechen.

seven_fa_names_follow	Regel, um einen siebenteiligen Firmennamen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
seven_fa_names_follow_in_quotation_marks	Regel, um einen siebenteiligen Firmennamen in Anführungszeichen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
one_fa_name_in_front	Regel, um einen einteiligen Firmennamen zu erkennen, der auf Grund eines Triggers am Ende erkannt wird	Konzept
two_fa_names_in_front	Regel, um einen zweiteiligen Firmennamen zu erkennen, der auf Grund eines Triggers am Ende erkannt wird	Konzept
three_fa_names_in_front	Regel, um einen dreiteiligen Firmennamen zu erkennen, der auf Grund eines Triggers am Ende erkannt wird	Konzept
four_fa_names_in_front	Regel, um einen vierteiligen Firmennamen zu erkennen, der auf Grund eines Triggers am Ende erkannt wird	Konzept
five_fa_names_in_front	Regel, um einen fünfteiligen Firmennamen zu erkennen, der auf Grund eines Triggers am Ende erkannt wird	Konzept
six_fa_names_in_front	Regel, um einen sechsteiligen Firmennamen zu erkennen, der auf Grund eines Triggers am Ende erkannt wird	Konzept
seven_fa_names_in_front	Regel, um einen siebenteiligen Firmennamen zu erkennen, der auf Grund eines Triggers am Ende erkannt wird	Konzept
one_fa_name_in_front_of_hyphaneted_suffix	Regel, um einen einteiligen Firmennamen zu erkennen, der auf Grund eines Bindestrich-Triggers vor dem Firmennamen erkannt wird	Konzept
two_fa_names_in_front_of_hyphaneted_suffix	Regel, um einen zweiteiligen Firmennamen zu erkennen, der auf Grund eines Bindestrich-Triggers vor dem Firmennamen erkannt wird	Konzept
three_fa_names_in_front_of_hyphaneted_suffix	Regel, um einen dreiteiligen Firmennamen zu erkennen, der auf Grund eines Bindestrich-Triggers vor dem Firmennamen erkannt wird	Konzept
four_fa_names_in_front_of_hyphaneted_suffix	Regel, um einen vierteiligen Firmennamen zu erkennen, der auf Grund eines Bindestrich-Triggers vor dem Firmennamen erkannt wird	Konzept
five_fa_names_in_front_of_hyphaneted_suffix	Regel, um einen fünfteiligen Firmennamen zu erkennen, der auf Grund eines Bindestrich-Triggers vor dem Firmennamen erkannt wird	Konzept
six_fa_names_in_front_of_hyphaneted_suffix	Regel, um einen sechsteiligen Firmennamen zu erkennen, der auf Grund eines Bindestrich-Triggers vor dem Firmennamen erkannt wird	Konzept
seven_fa_names_in_front_of_hyphaneted_suffix	Regel, um einen siebenteiligen Firmennamen zu erkennen, der auf Grund eines Bindestrich-Triggers vor dem Firmennamen erkannt wird	Konzept
pos_fa_name_part_for_listing	Definition, welche Firmennamen sich zur Aufzählung eignen	Konzept

listing_of_fa_in_front	Regel, um Firmennamen in einer zweiteiligen Aufzählung zu finden, bei der der zweite Firmenname bereits von einem der oberen Konzepte erkannt worden ist	Konzept
listing_of_fa_in_front_II	Regel, um Firmennamen in einer mehrteiligen (bis zu siebenteilig) Aufzählung zu finden, bei der der letzte Firmenname bereits von einem der oberen Konzepte erkannt worden ist	Konzept
listing_of_fa_behind	Regel, um Firmennamen in einer zweiteiligen Aufzählung zu finden, bei der der erste Firmenname bereits von einem der oberen Konzepte erkannt worden ist	Konzept
listing_of_fa_behind_II	Regel, um Firmennamen in einer mehrteiligen (bis zu siebenteilig) Aufzählung zu finden, bei der der letzte Firmenname bereits von einem der oberen Konzepte erkannt worden ist	Konzept
stop_words_for_fa_name_in_front_adj	Regel, dass geographische er- Adjektive ebenfalls zu den Begrenzern von Firmennamen gehören, die auf Grund eines Triggers am Ende gefunden worden sind (Um zirkuläre Beziehungen zu vermeiden, steht dieses Konzept nicht in den Ressourcen)	Konzept
learn_fa	**Regeln zum Wiederfinden von Teilen von bereits gelernten Firmennamen (Akronymen)**	**Module**
learn_fa	Regeln zum Wiederfinden von Teilen von bereits gelernten Firmennamen (Akronymen)	Konzept

Tabelle 14

Da in der Konzeptsensorsprache jede Iteration in einem neuen Konzept ausgedrückt werden muss, erscheinen in der Implementierung sieben Konzepte (*x_fa_names_follow* und *x_fa_name_in_front*[104]), denen jeweils dieselbe Regel zu Grunde liegt. Wichtig ist, zu berücksichtigen, dass der Firmennamen nur in seiner längsten Extension erkannt wird und in der Ausgabe erscheint. Zum Beispiel sollte *J.D. Edwards & Co.* nur als fünfteiliger Name in der Ausgabe erscheinen, nicht aber als dreiteiliger (*Edwards & Co*) oder vierteiliger (*D. Edwards & Co*). Um doppelte Analysen zu vermeiden, muss der Bedingungsteil mit der Einschränkung (*NOT overlaps_with*) erweitert werden, dass das kürzere Konzept (ein Firmenname mit weniger Teilen[105]) nur angewendet wird, falls die Eigennamenerkennung nicht über ein längeres Konzept (ein Firmenname aus mehreren Teilen) zuvor stattgefunden hat (vgl. Abbildung 31).

[104] Für x kann eine beliebige Zahl zwischen 1 und 7 eingesetzt werden.
[105] Wie ein Teil eines Firmennamens definiert wird, ist in Fussnote 94, S. 85 beschrieben.

```
CONCEPT "one_fa_name_follows"
IF
    ("triggerword_in_front":CONCEPT);
    ("pos_fa_name_part":CONCEPT AND $?:POSFANAMEPART)
CONDITION
    NOT overlaps_with("stop_words_for_fa_name_follows":CONCEPT);
    NOT overlaps_with("two_fa_names_follow":CONCEPT);
    NOT ovelaps_with("two_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("three_fa_names_follow":CONCEPT);
    NOT overlaps_with("three_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("four_fa_names_follow":CONCEPT);
    NOT overlaps_with("four_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("five_fa_names_follow":CONCEPT);
    NOT overlaps_with("five_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("six_fa_names_follow":CONCEPT);
    NOT overlaps_with("six_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("seven_fa_names_follow":CONCEPT);
    NOT overlaps_with("seven_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("one_fa_name_follows_in_quotation_marks":CONCEPT);
    NOT find("triggerword_behind":CONCEPT, +1);
    create("$0":FA)
END
```

Abbildung 31

Zusätzlich zum oben genannten Punkt muss sichergestellt werden, dass ein Firmenname, der sowohl einen Trigger vor wie auch nach dem Namen besitzt, nicht zweimal erkannt wird. Durch die Eingrenzung, dass kein Firmentrigger hinter dem Konzept stehen darf (*NOT find("triggerword_behind":CONCEPT, +1)*) und ein Firmentrigger kein möglicher Firmennamenteil (*pos_fa_name_part*) sein darf, entsteht eine Reihenfolge, in der die Konzepte angewendet werden. Falls ein Firmenname von Triggern umgeben ist, wird dem Trigger am Ende der Vorzug gegenüber dem Trigger am Anfang gegeben.

In der Implementierung sollte neben einem Firmennamen, der einem Trigger folgt, auch ein Firmenname erkannt werden, der mit Anführungs- und Schlusszeichen umgeben ist und folglich nicht direkt einem Trigger folgt. Ein Firmenname wird nur in Anführungs- und Schlusszeichen gesetzt, wenn ein Triggerwort davor steht oder der Firmenname ohne Trigger vorkommt[106]. Um zu ermöglichen, dass Firmennamen, welche nicht direkt an den Trigger anschliessen, als Entitäten erkannt werden, sind mögliche Firmennamenteile definiert worden, die mit einem Anführungszeichen beginnen (*pos_fa_name_part_in_quotation_marks*). In den Konzepten *x_fa_names_follow_in_quotation_marks*[107] werden dieselben Regeln verwendet

[106] Diese Aussage wird einerseits durch die Algorithmen von Volk gerstützt, der Anführungszeichen ebenfalls als Merkmal von Firmennamen verwendet hat, andererseits durch meine eigenen Erfahrungen mit dem englischen und deutschen Testkorpus.

[107] Für x muss eine Zahl zwischen 1 und 7 eingesetzt werden, je nachdem, wie viele Firmennamenteile einem Trigger folgen sollen.

wie in den Konzepten *x_fa_names_follow*, mit dem Unterschied, dass der erste, gesuchte Firmennamenteil ein Teil eines Firmennamens in Anführungszeichen sein muss.

Bei Firmennamen gilt meist die Regel, dass Trigger, welche vor einem Firmennamen stehen (z.B. *Firma*, *Spezialist*), nicht Teil der Entität sind, während Trigger, die am Ende eines Firmennamens stehen (*AG*, *Group*), Teil der Entität sind. Wenn Firmennamenerkennung auf Organisationen ausgeweitet werden soll, dann gibt es diverse Trigger, die Teil der Entität sind.

Beispiel 37:

Universitäten München
Institut für sozialökologische Forschung und Bildung
Zentrum für Interaktive Medien
französische **Regierung**

Indem jeweils ein zweites Konzept *x_fa_names_follow* mit dem gleichen Namen geschrieben wurde, ist versucht worden, Trigger am Anfang einer Entität als Teil des Eigennamens mit zu berücksichtigen. Die Firmennamenerkennung ist versuchsweise auf gewisse Organisationen wie Institute und Universitäten erweitert worden, um bei der Evaluation bessere Resultate zu erreichen, weil im Testkorpus (vgl. Kapitel IV 2. „Testkorpus für die Evaluation in deutschsprachigen Texten“) neben Firmennamen auch Organisationsnamen annotiert worden sind. Präzisions- und Ausbeutewerte sind allerdings nur leicht gestiegen, weil die Erkennung von Organisationen weitere Probleme mit sich bringt, wie beispielsweise keine konstante Grossschreibung (vgl. Kapitel IV „Evaluation“).

In deutschen Texten kommt es vor, dass mit einem Firmennamen ein Bindestrichkompositum gebildet wird (z.B. „Noch in diesem Jahr will die ***Debis*-Tochter** über …“ [Konradin-Verlag, 1998]). Der zweite Teil solcher Bindestrichkomposita dient als Trigger für Firmennamen in den Konzepten *x_fa_names_in_front_of_hyphaneted_suffix*[108], wobei der Trigger in diesem Fall nicht Teil der Entität ist. Bindestrichkomposita kommen spezifisch im Deutschen vor.

Die Länge der Firmennamen wird nicht nur durch die Anzahl möglicher Firmennamenteile begrenzt, sondern auch durch Stoppwörter, die per Definition zu möglichen Firmennamenteilen zählen, jedoch, wenn sie am Anfang oder Ende einer Entität stehen, nicht Teil dieser Entität sind (*stop_words_for_fa_name_in_front, stop_words_for_fa_name_behind*).

Beispiel 38:

„**Die** Deutsche Telekom kommt nicht …
..., bietet die IBM Business Recovery Services **Unternehmen** seit kurzem das Servicepaket …“
[Konradin-Verlag, 1998]

[108] Vgl. Fussnote 107.

In den Erkennungsregeln habe ich diese Stoppwörter integriert, indem das Konzept fehlschlägt, falls ein entsprechendes Wort an einer beliebigen Stelle im Suchmuster auftaucht (vgl. Abbildung 32).

```
CONCEPT "one_fa_name_in_front"
IF
        ("pos_fa_name_part":CONCEPT AND $?:POSFANAMEPART);
        (("triggerword_behind":CONCEPT AND $?:FANAMEPART)
        OR ("triggerword_behind_accr":CONCEPT AND $?:FANAMEPART))
CONDITION
        NOT overlaps_with("stop_words_for_fa_name_in_front":CONCEPT);
        NOT overlaps_with("stop_words_for_fa_name_in_front_adj":CONCEPT);
        NOT overlaps_with("two_fa_names_in_front":CONCEPT);
        NOT overlaps_with("three_fa_names_in_front":CONCEPT);
        NOT overlaps_with("four_fa_names_in_front":CONCEPT);
        NOT overlaps_with("five_fa_names_in_front":CONCEPT);
        NOT overlaps_with("six_fa_names_in_front":CONCEPT);
        create("$0 $1":FA);
        create("$0":ACR)
END
```

Abbildung 32

Bei einem Fehlschlag eines Konzepts (z.B. *seven_fa_names_in_front*) kommt das nächst kürzere (z.B. *six_fa_name_in_front*) zum Zuge.

Wenn Stoppwörter allerdings innerhalb eines Firmennamens auftreten, wird der Firmennamen zu stark gekürzt und man erzeugt einen Fehler.Teilanalysen werden bei der Evaluation als Teilanalysen gewertet.

Beispiel 39:[109]

PC-**Spezialist** Systemzentrale AG
→ Systemzentrale AG

Solche Fehler sind im Testkorpus nie aufgetreten, sind jedoch durchaus denkbar. In der Implementierung war es nicht möglich, die Stelle zu bezeichnen, an der ein Stoppwort innerhalb eines Musters als Stoppwort tatsächlich fungieren soll. Deshalb wurde die allgemeinere Variante gewählt, dass ein Stoppwort, das in einem Muster auftritt, immer zum Fehlschlag eines Konzeptes führt.

Weil die Methoden, einen Firmennamen über einen Trigger zu erkennen eine geringe Ausbeute geliefert haben, mussten andere Merkmale wie Trigger zu Hilfe gezogen werden. In den Methoden *listing_of_fa_in_front* und *listing_of_fa_behind* wird die Aufzählung von Firmennamen genutzt. Sobald ein Firmenname über einen Trigger erkannt worden ist, wird davor bzw. danach geschaut, ob in Aufzählungsform ein weiterer möglicher Firmenname gefunden

[109] Dies ist ein hypothetisches Beispiel, aber durchaus denkbar.

werden kann. ‚In Aufzählungsform' bedeutet, dass die Firmennamen mit *und*, *oder* oder Komma voneinander getrennt werden. Die Länge eines Firmennamens ist wiederum auf sieben Teile beschränkt (vgl. *listing_of_fa_in_front_II* und *listing_of_fa_behind_II)*[110] Es wäre zu überlegen, nicht nur zweiteilige Aufzählungen mit zu berücksichtigen. Allerdings bringt eine solche Erweiterung neben einigen neuen, korrekten Firmennamen auch viele falsche Analysen (z.B. von Produktenamen), wodurch der Präzisionswert stark abnimmt.

In Ermangelung der Möglichkeit, Firmennamenakronyme abzuleiten (vgl. Beispiel 40) wurden Akronyme definiert (vgl., S. III-82: ACR). Diese dienen beim Erkennen von Firmennamen als Trigger am Ende bzw. am Anfang einer Entität. Wenn sie alleine auftreten und nur Wörter in Kleinbuchstaben angrenzen, werden Akronyme als Firmennamen analysiert, falls sie nicht im Filter stehen. Allein stehende Akronyme als Firmennamen zu analysieren, bringt jedoch die Problematik mit sich, dass viele Produktenamen oder sonstige Akronyme fälschlicherweise als Firmennamen zählen. Der Filter müsste entsprechend verbessert werden, um diese Fehlanalysen zu vermeiden.

Beispiel 40:

Firma Agri → Agri
Agri AG → Agri
CSS ... AG → CSS
United Bank of Switzerland → UBS

Das Wiederfinden von Teilen von gelernten Firmennamen, die ebenfalls als Akronyme bezeichnet werden, habe ich nur mit wenig Erfolg implementiert. Dieser Anwendungsschritt, der als Ersatz für die ‚Zwei-Weg Lösung' (vgl. Kapitel III 2.4. „Probleme innerhalb der Konzeptsensorsprache und ihre Grenzen") dienen sollte, hat grosse Verluste in der Präzision der Konzepte, jedoch nur geringe Verbesserung der Ausbeute mit sich gebracht. Es werden zu viele falsche Namen gelernt, weil der Filter zu wenige eliminiert. Ausserdem werden Genitivformen nur gefunden, falls sie als Genitiv bereits von den anderen Konzepten erkannt worden sind. Dies ist nur selten der Fall und somit werden viele Firmenamen in diesem zweiten Lernschritt nicht berücksichtig. Eine ‚Zwei-Weg-Lösung' ist, wie in diesem Projekt klar wurde, unumgänglich, um Teile von gelernte Firmennamen mit möglichst hoher Präzision bei möglichst hoher Ausbeute wiederzufinden und so bessere Evaluationswerte zu erreichen (vgl. Kapitel IV „Evaluation").

[110] Die Längenbeschränkung dadurch zu begründen, dass in der Konzeptsensorsprache keine Schlaufenstrukturen (zirkuläre Abhängigkeiten oder rekursive Vorgänge) implementiert werden können.

5. Ressourcen für die Erkennung von Entitäten in englischen Texten

Die Ressourcen zur Erkennung von Eigennamen in deutschen Texten wurden im Bereich der geographischen Eigennamen um Einträge aus dem englischen Sprachgebiet (USA, Kanada, Grossbritannien) erweitert. Einträge aus dem deutschen Sprachraum sind belassen worden, da nicht davon ausgegangen werden kann, dass Eigennamen, wenn überhaupt möglich, in die Sprache des Dokuments übersetzt werden. Oft stehen Eigennamen in der Schreibweise derjenigen Sprache, in der sie ihren Ursprung besitzen.

Beispiel 41:

„Bei "Big Blue" geht die globale Neuorganisation weiter.“
[Konradin-Verlag, 1998]

Die Firma *Big Blue* wird nicht mit „*Der Grosse Blaue*“ übersetzt oder ähnlich. Städtenamen besitzen keine Übersetzung. Personennamen darf man nicht übersetzen, sonst ist die Funktion des Bezeichnens nicht mehr gewährleistet. Ländernamen bilden eine Ausnahme. Sie besitzen in den verschiedenen Sprachen verschiedene Schreibweisen. Deutsche Schreibweisen haben jedoch in diesem Projekt die Präzision nicht verschlechtert. Deshalb habe ich sie in den Ressourcen belassen.

Die Ressourcen zur Eigennamenerkennung sind somit nur teilweise sprachspezifisch und können, abgesehen von leichten Anpassungen, in andere (indogermanische) Sprachen[111] übernommen werden.

5.1. Ressourcen zur Erkennung von Personennamen (En)

Bis auf die Erkennung eines Personennamens über ein Suffix (*Jr.* oder *Sr.*) sind die Regeln und somit die dafür benötigten Ressourcen im Vergleich zur Erkennung von Personennamen in deutschen Texten gleich geblieben. Genitivformen von Vornamen müssen nicht mehr als eigene Ressource aufgeführt werden, da diese über Hinzufügen von Apostroph und *s* gebildet werden. Eine solche Komposition (z.B. *Peter's*) wird vom Indexer in drei Teilen analysiert (*Peter*, *'*, *s*). Der erste Teil enthält die Grundform, so wie sie in den Ressourcen steht. Der Apostroph gehört nicht zum nach meiner Definition nicht zum Personenamen.

[111] Für Sprachen, die einem anderen Grammatiksystem als die indogermanischen Sprachen unterliegen (z.B. Chinesisch), können die Ressourcen nicht übernommen werden, da dort der Aufbau von Eigennamen (z.B. Vornamen, Nachname etc.) anders geartet ist und die Entitäten nicht mit Grosschreibung makriert werden. In diesem Fall müssten ganz neue Muster gefunden werden.

Name	Funktion	Beispiele	Kategorie
pers_resources	**Ressourcen zur Erkennung von Personennamen**		**Modul**
pos_last_name	Regel zur Erkennung von möglichen Nachnamen	Müller, Müller-Meier	Konzept
pre_name	Definition, dass ein Vorname allein stehen muss	Abby, Al'for, Ek'vt'ime	Konzept
sal_pers_name	Liste mit Anreden	Mr., Baroness, Maj.	Konzept
suffix_names	Liste mit Namensuffixen	Jr., Sr., Senior	Konzept
pre_pers_name	Liste mit Vornamen (ein- bis fünfteilig)	Abby, Al'for, Ek'vt'ime	Konzept
pre_pers_double_ name	Definition eines zweiteiligen Namens (Zweierkombination aus Vornamen oder Initialen)	Odd Bjorn, O. Bijorn, O. B.	Konzept
hyphenated_pre_ pers_name	Definition eines Bindestrich-Vornamens (Zweierkombination aus Vornamen oder Initialen)	Jean-Jacques, Marie-A.	Konzept
last_name_part	Liste mit Titeln und anderen Teilen von Nachnamen	di, MC, O'	Konzept
stop_word_list_for_ single_names	Liste mit Vornamen, welche keine Vornamen sind, wenn sie alleine vorkommen	As, Love, Xenos,	Konzept

Tabelle 15

Die Definition eines Nachnamens (*pos_last_name*), die Liste mit Vornamen (*pre_pers_name*) sowie die Liste mit Bindestrichvornamen (*hyphenated_pre_pers_name*) sind unverändert von den deutschen Ressourcen übernommen worden. Die Liste mit Anreden (*sal_pers_name*) und Titeln (*last_name_part*) ist um verschiedene Formen aus dem Englischen (z.B. *Ms.*, *Mc*) und Spanischen (*Senor*[112], *di*) ergänzt worden. Die Erweiterung ist damit zu begründen, dass bei Personennamen der Titel, in gewissen Fällen auch die Anrede, in derjenigen Sprache geschrieben wird, die in der Region, aus welcher der Namen stammt, gesprochen wird. Die Anreden wurden spezifisch durch militärische Grade (z.B. *Corporal*, *Comandor*), Adelstitel (z.B. *Duke*, *Lord*) und religiöse Titel (z.B. *Bishop*, *Pastor*) ergänzt, da auf solche Auszeichnungen im englischen Sprachraum grösserer Wert gelegt wird und sie somit in einem Text häufiger in Verbindung mit einem Personennamen auftreten.

In englischen Texten bilden Ergänzungen wie *Jr.* oder *Sr.* eine Teilkomponente eines Namens. Sie sind von mir zusätzlich zum Erkennen von Nachnamen verwendet worden. Die Ressource *Suffix-Namen* (*suffix_names*) enthält eine entsprechende Liste.

Die Liste mit denjenigen Namen, die, wenn sie alleine auftreten, meist keine Vornamen bezeichnen, musste der Sprache gemäss neu angelegt werden (z.B. *Angel*, *He*, *Since*[113]). Im Eng-

[112] Alle Ressourcen sind in einer Normform festgehalten, in der diakritische Zeichen einer Fremdsprache (keine Umlaute) in die einfachste Form umgewandelt werden (ñ → n).

[113] Grossschreibung als Merkmal für Entitäten ist in diesem Fall unzureichend, da die Wörter oft auch am Anfang eines Satzes auftreten und somit grossgeschrieben werden müssen, ohne eine Entität zu sein.

lischen erzeugen andere Wörter als im Deutschen Ambiguität. Deshalb ist eine sprachspezifische Erneuerung immer notwendig. Überschneidungen mit geographischen Namen (z.B. *Hamliton*) konnten aus den deutschen Ressourcen übernommen werden, neue Überschneidungen, die auf Grund der Erweiterung der geographischen Ressourcen entstanden sind, mussten ergänzt werden (z.B. *Dallas*).

Im Gegensatz zur Erkennung von Personennamen in deutschen Texten (vgl. Kapitel III 3.1. „Ressourcen zur Erkennung von Personennamen (De)“), habe ich in den englischen Texten ein Initial an einer beliebigen Stelle als Abkürzung eines Vornamens erlaubt. Dies bringt eine Erweiterung der doppelten Vornamen (*pre_pers_double_name*) mit sich, so dass jegliche Kombination von Initial oder Vornamen erlaubt ist. Um zu verhindern, dass zusammengesetzte Vornamen (*pre_pers_double_name,* z.B. *Hans Peter* oder *hyphenated_pre_pers_name,* z.B. *Hans-Peter*) einerseits als einzelne Vornamen (*pre_pers_name*, z.B. *Hans* oder *Peter*), andererseits als Gefüge (z.B. *Hans Peter*) erkannt werden, musste eine weitere Regel (*pre_name)* implementiert werden. Sie besagt, dass vor einem Vornamen weder ein Bindestrich noch ein anderer Vornamen, noch ein Initial stehen darf. Bei der Personennamenerkennung (vgl. Kapitel III 6.1. „Erkennung von Personennamen (En)“) wird folglich nur noch die eingeschränkte Definition von Vornamen (*pre_name*) verwendet, nicht aber die Ressource (*pre_pers_na*me) direkt.

5.2. Ressourcen zur Erkennung von geographischen Namen (En)

Die geographischen Ressourcen wurden mit Städte-, Regions- und Kantonsnamen (bzw. Staatsnamen) aus den Gebieten USA, Kanada und Grossbritannien erweitert. Die Erweiterungen sind aus Listen von den Internetadressen www.yahoo.com, www.yahoo.ca, www.yahoo.co.uk gesammelt worden. Unter der Rubrik *Reisen* werden die Internetadressen in einem Katalogssystem nach Ländern, Städten und Regionen geordnet.

Kontinentnamen sind ins Englische übersetzt worden. Die Ressource *geo_resources_ continents* enthält nur englische Formen. Namen von Nationen konnten übernommen werden und sind zusätzlich mit englischen Schreibweisen angereichert worden. Die deutschen Formen habe ich in der Ressource belassen, da in einem Text denkbar ist, dass auch eine andere Schreibweise eines geographischen Namens (meist diejenige Schreibweise in der Landessprache einer Region) erscheint. Es ist zu keinen Fehlanalysen gekommen, die durch deutschsprachige Einträge ausgelöst worden sind.

Aus demselben Grund wie bei der Erkennung von Personenamen (vgl. Kapitel III 5.1. „Ressourcen zur Erkennung von Personennamen (En)“) müssen Genitivformen nicht separat abgespeichert werden. Eine Liste der Nominativformen genügt.

Auf das Erkennen von geographischen Adjektiven wurde im Englischen ganz verzichtet, da bei den Ableitungen die Bezeichnung des Landes nur noch eine sekundäre Funktion einnimmt. Mit *New Yorker* ist beispielsweise in erster Linie eine Person gemeint, in zweiter Linien erst das Herkunftsland dieser Person. Mit *german* wird eine Sprache beschrieben, in zweiter Linie erst die Herkunft dieser Sprache. Es ist allerdings ein willkürlicher Entscheid, wo die Grenzen bezüglich geographischer Adjektive und Eigennamen gezogen werden. Da es sich bei geographischen Adjektiven um einen Grenzfall handelt, ob sie zu den Entitäten zählen oder nicht, habe ich beschlossen, sie ganz wegzulassen (vgl. Kapitel II 1.1. „Definition von Eigennamen“). In den annotierten Textkorpora werden unter anderem geographische Adjektive und Nominalisierungen entsprechend anderer Richtlinien als Entitäten markiert.

Name	**Funktion**	**Beispiele**	**Kategorie**
geo_resources_ citieslist			**Modul**
single_city_names	Liste mit einteiligen Städtenamen	Aachen, Zionsville	Konzept
special_single_city_ names	Liste mit einteiligen Städtenamen, welche Teil des Allgemeinwortschatzes sind und somit nur in spezifischem Kontext als Entität gelten	Bush, Love, Hope	Konzept
hyphenated_names	Liste mit Bindestrich-Städtenamen (ein- bis fünfteilig)	Wood-Ridge, Annandale-on-Hudson, Sainte-Roch-de-L'Achigan	Konzept
double_names	Liste mit zweiteiligen Städtenamen	Arlington Heights, New York	Konzept
triple_names	Liste mit dreiteiligen Städtenamen	City of Albany, New York City	Konzept
quadruple_names	Liste mit vierteiligen Städtenamen	Rust am Neusiedler See, Lake in the Lake	Konzept
slash_citiesI	Liste mit Städtenamen, welche einen Schrägstrich als Abkürzung verwenden	Affoltern a/Albis, Wangen a/Aare	Konzept
slash_citiesII	Liste mit dreiteiligen Städtenamen, welche mit mindestens einem Schrägstrich geschrieben werden	Korat/Nakhon, Ratchasima	Konzept
slash_citiesIII	Liste mit zweiteiligen Städtenamen, welche mit einem Schrägstrich voneinander getrennt werden	Breil/Brigels, Athens/Ohio	Konzept
st_names	Liste mit Städtenamen, welche mit *St.* beginnen	St. Lucia, St. Thomas	Konzept

geo_resources_nations	**Ressource mit Ländern (=Nationen)**		**Modul**
single_nation_names	Liste mit einteiligen Ländernamen	Austria, US	Konzept
double_nation_names	Liste mit zweiteiligen Ländernamen	American Samoa, Vatican City	Konzept
triple_nation_names	Liste mit dreiteiligen Ländernamen	Central Africa Republic, Republic of Yugoslav	Konzept
quadruple_nation_names	Liste mit vierteiligen Ländernamen	British Indian Ocean Territory, Cote d'Ivoire[114]	Konzept
quintuple_nation_names	Liste mit fünfteiligen Ländernamen	Democratic Republic of the Congo	Konzept
hyphenated_nation_ names	Liste mit zweiteiligen Ländernamen, die mit einem Bindestrich kombiniert werden	Bosnien-Herzegowina, Saudi-Arabien	Konzept
geo_resources_ provinces_territories	**Ressource mit Kantons- und Ländernamen (=Staaten, Bundesländer etc.)**		**Modul**
single_provinces_ territories_names	Liste mit einteiligen Kantons- und Ländernamen	Bayern, Newfoundland	Konzept
hyphenated_provinces_ territories_names	Liste mit einteiligen Kantons- und Ländernamen, die über einen Bindestrich kombiniert werden	Baden-Wuerttemberg, Schleswig-Holstein	Konzept
double_provinces_ territories_names	Liste mit zweiteiligen Kantons- und Ländernamen	North Dakota, Nova Scotia	Konzept
triple_provinces_ territories_names	Liste mit dreiteiligen Kantons- und Ländernamen	Prince Edward Island	Konzept
geo_resources_regions_ islands	**Ressource mit Regions- und Inselnamen**		**Modul**
single_regions_islands_ names	Liste mit einteiligen Regions- und Inselnamen	Caribbean, Mediterranean	Konzept
hyphenated_regions_ islands_names	Liste mit einteiligen Regions- und Inselnamen, die über einen Bindestrich kombiniert werden	Middle-Europe, Central-Asia	Konzept
double_regions_islands_ names	Liste mit zweiteiligen Regions- und Inselnamen	Far East, Baltic Provinces	Konzept
tripple_regions_islands_ names	Liste mit dreiteiligen Regions- und Inselnamen	Eastern European states	Konzept
geo_resources_ continents	**Ressource mit Kontinentnamen**		**Modul**
continent_names	Liste mit Kontinenten	Africa, Europe	Konzept
geo_resources_abbr	**Ressource mit geographischen Abkürzungen**		**Modul**
geo_resources_abbr	Liste mit geographischen Abkürzungen (ein- und zweiteilig)	WA, Calif.	Konzept

Tabelle 16

Die grösste Anpassung der Ressourcen auf englische Texte musste innerhalb der Liste mit spezifischen Städtenamen (*special_single_city_names*) vorgenommen werden, weil durch den

[114] Das Apostroph ' muss in der Konzepten als eigenes Zeichen definiert werden und gilt somit als selbständiger Teil.

Gebrauch eines anderen Wortschatzes (Englisch) andere Wörter ambig geworden sind. Da die Liste mit Städtenamen im Vergleich zu den deutschen Ressourcen stark angewachsen ist, gab es mehr ambige Wörter, die ich in der Spezialliste aufnehmen musste. Um herauszufinden, welche Städtenamen ambig sein können, habe ich das Englisch-Deutsch-Lexikon mit der vorhandenen Städteliste verglichen. Wörter, welche sowohl im englischen Allgemeinwortschatz wie auch in der Liste mit Städtenamen vorkommen, müssen aus der Städteliste entfernt und in die Spezialliste eingefügt werden. Die Spezialliste ist während der Evaluationsschritte weiter mit Wörtern ergänzt worden, die sowohl eine geographische Entität als auch einen Firmennamen (z.B. *Mercedes*) oder einen Personennamen (z.B. *Clinton*) beschreiben.

5.3. Ressourcen zur Erkennung von Firmennamen (En)

Triggerwörter, welche am Ende eines Firmennamens stehen, werden sprachunabhängig verwendet. Somit konnte ich diese Ressource unverändert vom Deutschen übernehmen. Weil der Filter bereits auf Englisch und Deutsch abgestimmt ist, musste dieser nicht verändert werden. Die restlichen Ressourcen habe ich spezifisch für Englisch neu zusammengestellt.

Name	**Funktion**	**Beispiele**	**Kategorie**
fa_name_resources	**Ressourcen für die Erkennung von Firmennamen**		**Modul**
pos_fa_name_part	Definition wie ein Teil eines Firmennamens aussehen könnte	ITAR-Tass, AP	Konzept
pos_fa_name_part_in_quotation_marks	Definition wie ein Teil eines Firmennamens, der in Anführungszeichen steht, aussehen könnte	„Microsoft, „AP	Konzept
triggerword_in_front	Liste mit Wörtern, denen ein Firmenname folgen kann, die aber nicht Teil der Entität sind	Company, provider	Konzept
triggeracr_in_front	Definition, dass Akronyme am Anfang eines Firmennamens als Trigger fungieren	ITAR, AP	Konzept
triggerword_behind	Liste mit Wörtern (ein und zweistellig), die am Ende eines Firmennamens stehen und Teil der Entität sind	Holding, Cos.	Konzept
triggeracr_behind	Definition, dass Akronyme am Ende eines Firmennamens als Trigger fungieren	DRI, INC	Konzept
stop_words_for_fa_name_follows	Liste mit Wörtern, welche einen Firmennamen begrenzen, der auf Grund eines Triggers vor dem Firmennamen gefunden wurde	AG, Co.	Konzept
stop_words_for_fa_name_in_front	Liste mit Wörtern (ein- und mehrteilig), welche einen Firmennamen begrenzen, der auf Grund eines Triggers am Ende des Firmennamens gefunden wurde	The	Konzept
filter	**Ressource, welche ein Englisch-Deutsch-Lexikon enthält, das zum Filtrieren von einteiligen Firmennamen dient**		**Modul**
filterI	Liste mit englischen und deutschen Wörtern aus dem Allgemeinwortschatz in Grossbuchstaben, gängigen Abkürzungen und spezifisch zusammengestellten Wörtern	YOUTH, SHELL etc., WWW	Konzept

filter	Alle Einträge aus filterI und filterII ohne Wörter aus der Liste *stopwords_for_filter*.	YOUTH, etc., abrechnen	Konzept
stopwords_for_filter	Liste mit Wörtern, welche im Filter stehen, aber trotzdem häufige Firmennamen sind	Sun, Shell	Konzept
filterII	**Ressource, welche ein Englisch-Deutsch-Lexikon enthält, das zum Filtrieren von einteiligen Firmennamen dient (*filterII* hat dieselbe Funktion wie filter, musste allerdings abgetrennt werden, da sonst das Modul zu gross wird und nicht mehr funktioniert)**		**Modul**
filterII	Liste mit englischen und deutschen Wörtern aus dem Allgemeinwortschatz in gestemmter Form	abrechnen, youth, sun	Konzept

Tabelle 17

Das Konzept *triggeracr_in_front* entspricht im Deutschen *trigger_part_of_fa* (vgl. Kapitel III 3.3. „Ressourcen zur Erkennung von Firmennamen (De)“). Im Englischen habe ich den Versuch, auch Organisationsnamen zu erkennen, unterlassen, weil der Erfolg mit den deutschen Konzepten gering war. Somit sind nur Akronyme, die am Anfang einer Entität stehen, Auslöser für einen Firmennamen.

Die Ressource *triggerword_behind_acr* im Deutschen entspricht *triggeracr_behind*. Durch die Veränderung des Namens ist eine einheitlichere Benennung erreicht worden. Die Ressourcen *triggeracr_in_front* und *triggeracr_behind* erfüllen die gleiche Funktion innerhalb der Eigennamenerkennung (ein Akronym, das als Trigger fungiert) und sollten somit durch eine ähnliche Namengebung miteinander verbunden werden.

Sowohl die Liste der Stoppwörter vor (*stop_words_for_fa_name_in_front*) als auch hinter einem Firmennamen (*stop_words_for_fa_name_follows*) ist verglichen mit dem Deutschen kleiner. Dies ist mit der markanteren Bedeutung der Grossschreibung im Englischen zu begründen. Im Englischen werden ausser Eigennamen nur Wörter am Satzanfang sowie das Pronomen *I* grossgeschrieben. Somit muss man der Problematik, dass Wörter, die nicht zu Firmennamen gehören, grossgeschrieben werden, nur an Satzanfängen Beachtung schenken. Ich habe festgestellt, dass vor einem Firmennamen, der an zweiter Stelle eines Satzes steht, nur das Pronomen *The* auftaucht und somit der Stoppwortliste hinzugefügt werden muss. Um sicher zu stellen, dass Firmennamen, die sowohl einen Trigger davor als auch einen Trigger am Ende besitzen, nur einmal analysiert werden, muss eine Reihenfolge erstellt werden, die besagt, welche Muster zuerst, welche danach erkannt werden. In diesem Fall findet das System die Firmennamen in erster Linie über den Trigger am Ende. Nur wenn keiner vorhanden ist, wird nach einem Trigger vor dem Firmennamen gesucht. Deshalb enthält die Stoppwort-

liste (*stop_words_for_fa_name_follows*) Triggerwörter, die am Ende eines Firmennamens stehen (*AG*, *Company* etc.). Um die Reihenfolge für den Ablauf der Konzepte weiter zu gewährleisten, muss bei der Erkennung von Firmennamen durch einen Trigger am Anfang beachtet werden, dass kein Triggerwort am Ende eines Firmennamens auftaucht.

6. Implementierung der Eigennamenerkennung in englischen Texten

An den Methoden zur Eigennamenerkennung sind im Vergleich zur Implementierung ins Deutsche keine Veränderungen vorgenommen worden. Die Strategien *Anfügen* bzw. *Anwenden, Vergessen* bzw. *Filtern* werden gleich verwendet. Die Lernalgorithmen funktionieren unverändert, da bezüglich Grossschreibung von Eigennamen das Englische sich gleich wie das Deutsche verhält.

Um die Verständlichkeit der Konzepte zu erhöhen, habe ich im Englischen versucht, Konzepte, die dieselbe Funktion ausüben, gleich zu benennen. Bei Konzepten, denen ich denselben Namen gegeben habe, muss beachtet werden, dass nicht nur die längste Übereinstimmung einer Zeichenfolge als erfolgreich, sondern eine kürzere und eine längere Version derselben Zeichenfolge zurückgeliefert wird. Dies hat zur Folge, dass weitere Einschränkungsregeln im Bedingungsteil (*CONDITION*) erstellt werden müssen.

Ein Personenname wie z.B. *O'Harra Sr.* wird über ein Konzept *suffix_name* erkannt (vgl. Abbildung 33 und Abbildung 34). Damit der Personennamen allerdings gleichzeitig nicht ohne Titel (*Harra Sr.*) erkannt wird, muss das eine Konzept *suffix_name*, welches einen Nachnamen ohne Titel über ein Suffix erkennt, mit einer zusätzliche Einschränkung, dass vor dem Nachnamen kein Titel stehen darf, ergänzt werden (vgl. Abbildung 33).

Ich habe festgestellt, dass der Versuch, gleiche Funktionen mit gleichen Namen zusammenzufassen, mehr Aufwand mit sich gebracht hat und einem Benutzer das Verständnis solcher Konzepte vermutlich nur wenig erleichtert.

```
/*Bsp. Bush Jr.*/
        CONCEPT "suffix_name"
        IF
                ("pos_last_name":CONCEPT AND $?:POSLASTNAME);
                ("suffix_names":CONCEPT AND $?:LASTNAMEPART)
        CONDITION
                NOT find("sal_pers_name":CONCEPT, -3);
                NOT find("initial_of_name":CONCEPT, -2);
                NOT find("pre_pers_name":CONCEPT, -2);
                NOT find("pre_name":CONCEPT, -2);
                NOT find("pre_pers_double_name":CONCEPT, -2);
                NOT find("hyphenated_pre_pers_name":CONCEPT, -2);
                NOT find("last_name_part":CONCEPT, -1);
                NOT overlaps_with("part_of_fa":CONCEPT);
                create("$0 $1":PERS);
                create("$0 $1":LASTNAME);
                create("$0":LASTNAME)
        END
```

Abbildung 33

```
        /*Bsp. O'Harra Sr.*/
        CONCEPT "suffix_name"
        IF
                ("last_name_part":CONCEPT AND $?:LASTNAMEPART);
                ("pos_last_name":CONCEPT AND $?:POSLASTNAME);
                ("suffix_names":CONCEPT AND $?:LASTNAMEPART)
        CONDITION
                NOT find("sal_pers_name":CONCEPT, -3);
                NOT find("initial_of_name":CONCEPT, -2);
                NOT find("pre_pers_name":CONCEPT, -2);
                NOT find("pre_name":CONCEPT, -2);
                NOT find("pre_pers_double_name":CONCEPT, -2);
                NOT find("hyphenated_pre_pers_name":CONCEPT, -2);
                NOT overlaps_with("part_of_fa":CONCEPT);
                create("$0 $1 $2":PERS);
                create("$0 $1 $2":LASTNAME);
                create("$0 $1":LASTNAME);
                create("$1 $2":LASTNAME)
        END
```

Abbildung 34

6.1. Erkennung von Personennamen (En)

Die Schemata, welche ich zur Personennamenerkennung verwendet habe, weichen nur leicht von denjenigen in deutschen Texten ab. Die Namen *full_pers_name* und *last_name* gehören zu mehreren Konzepten, deren Funktion dieselbe ist. Optionalität innerhalb der Funktionen muss mit einem eigenen Konzept realisiert werden. Die Konzepte, welche die gleichen, obligatorischen Grundelemente zur Erkennung verwenden (z.B. eine Anrede), erhalten denselben

Namen. Die Konzepte entsprechen den deutschen Konzepten *full_pers_name_n*[115] und *last_name_n*[116]. Die Konzepte *abbr_pers_name_n*[117] habe ich in den englischen Konzepten vollständig zugunsten der beiden Kategorien *full_pers_name* und *last_name* aufgegeben, da ein Name mit Initial immer der Schablone *vollständiger Name* oder *Nachname* entspricht. Ein Initial ist gleichwertig mit einem Vornamen und kann den Vornamen durch eine Abkürzung (das Initial) ersetzen. Eine Ausnahme bilden allein stehende Vornamen. Sie werden nie mit einem Initial abgekürzt, da eine klare Referenz zum Objekt nicht mehr gegeben ist. Wenn Initial und Vorname in mehrteiligen Personennamen gleich behandelt werden, benötigt man die Schablone *abbr_pers_name* nicht mehr.

Um längere Personenamen zu erkennen, die in englischen Texten relativ häufig vorkommen, habe ich bei der Erkennung neu bis zu vier Vornamen[118] zugelassen. Im Deutschen waren es bisher zwei, die mit einem Initial erweitert werden konnten.

Die Konzepte *suffix_name* sind von mir speziell fürs Englische neu hinzugefügt worden, da solche Suffixe, die einen Namen genauer beschreiben (*Jr.*, *Sr.*) weit häufiger im Englischen als im Deutschen angewendet werden. Die Erkennung eines Nachnamens über ein Suffix kommt nur dann zum Zuge, wenn weder eine Anrede noch ein Vorname oder Initial vor dem Nachnamen steht. Um Doppelanalysen[119] zu vermeiden, muss in den Konzepten ausdrücklich festgehalten werden, dass die obligatorischen Elemente eines andern Konzepts nicht vor oder hinter dem Eigennamen auftreten. Die Reihenfolge der Konzepte muss somit explizit definiert werden. Die Konzepte *full_pers_name* werden vor den Konzepten *last_name* und *suffix_name* angewendet (vgl. im Anahang: „Modul zur Personenamenerkennung in englischen Texten“).

Name	**Funktion**	**Kategorie**
pers_rec	**Erkennung von Personennamen**	**Modul**
full_name	Regel zur Erkennung von vollständigen Personennamen: Anrede; *[ein bis zwei Vor-, Bindestrich- oder Doppelname]; [Titel]; Nachname	Konzept
last_name	Regel zur Erkennung von Personennamen ohne Anrede: ein bis zwei Vor-, Bindestrich- oder Doppelnamen; [Titel]; Nachname	Konzept
find_pre_name	Regel zur Erkennung von allein stehenden Vor-, Bindestrich- oder Doppelnamen	Konzept

[115] n entspricht einer Zahl zwischen 0 und 4.
[116] n entspricht einer Zahl zwischen 0 und 3.
[117] Vgl. Fussnote 116.
[118] In der Implementierung entsprechen vier Vornamen zwei Doppelnamen.
[119] Unter Doppelanalyse ist zu verstehen, dass der Name *Minister Simeon Nyachae* einmal als *Simeon Nyachae* auf Grund des Triggers Minister erkannt wird (*full_pers_name*), das andere Mal auf Grund des Vornamens *Simeon* (*last_name*).

suffix_name	Regel zum Erkennen eines Nachnamens auf Grund des Suffixes: [Titel]; Nachname; Suffix	Konzept
part_of_fa	Liste der Regeln, durch welche eine Firmennamen erkannt wird (Ein Personenname darf nicht erkannt werden, falls er Teil eines Firmennamens ist)	Konzept
learn_pers_name	**Anwendung von bereits gelernten Nachnamen**	**Modul**
find_last_name	Regel zur Wiedererkennung von Nachnamen innerhalb eines Dokuments, falls sie über ein Konzept erkannt wurden und nun in einem neuen Kontext stehen (ohne Vorname, Initial oder Anrede)	Konzept

* [] markieren optionale Elemente

Tabelle 18

Das Konzept *part_of_fa* wurde von mir neu implementiert. So kann man vermeiden, dass Teile von Personennamen innerhalb von Firmennamen als Personennamen statt Firmennamen analysiert werden. Zuvor war es nötig, eine Überlappung der Konzepte zur Firmennamenerkennung mit den Konzepten zur Personennamenerkennung im Bedingungsteil der Konzepte zur Personennamenerkennung auszuschliessen. Um nicht jedes Konzept zur Firmennamenerkennung einzeln aufzulisten, sind sie im Konzept *part_of_fa* zusammengefasst (vgl. Abbildung 35) und müssen in den Konzepten zur Personennamenerkennung nur einmal aufgerufen werden (vgl. Abbildung 34, Zeile *NOT overlaps_with("part_of_fa":CONCEPT);*).

```
CONCEPT "part_of_fa"
IF
        ("one_fa_name_follows":CONCEPT)
        OR ("one_fa_name_follows_in_quotation_marks":CONCEPT)
        OR ("two_fa_names_follow":CONCEPT)
        OR ("two_fa_names_follow_in_quotation_marks":CONCEPT)
        OR ("three_fa_names_follow":CONCEPT)
        OR ("three_fa_names_follow_in_quotation_marks":CONCEPT)
        OR ("four_fa_names_follow":CONCEPT)
        OR ("four_fa_names_follow_in_quotation_marks":CONCEPT)
        OR ("five_fa_names_follow":CONCEPT)
        OR ("five_fa_names_follow_in_quotation_marks":CONCEPT)
        OR ("six_fa_names_follow":CONCEPT)
        OR ("six_fa_names_follow_in_quotation_marks":CONCEPT)
        OR ("seven_fa_names_follow":CONCEPT)
        OR ("seven_fa_names_follow_in_quotation_marks":CONCEPT)
        OR ("one_fa_name_in_front":CONCEPT)
        OR ("two_fa_names_in_front":CONCEPT)
        OR ("three_fa_names_in_front":CONCEPT)
        OR ("four_fa_names_in_front":CONCEPT)
        OR ("five_fa_names_in_front":CONCEPT)
        OR ("six_fa_names_in_front":CONCEPT)
        OR ("seven_fa_names_in_front":CONCEPT)
END
```

Abbildung 35

Im Gegensatz zum Deutschen (Konzept *find_last_name*) ist das Wiederanwenden gefundener Nachnamen in einem eigenen Modul (*learn_pers_name*) implementiert, weil es sich hierbei um einen Schritt abgesondert von der Erkennung durch Muster handelt. Zuerst müssten Nachnamen mit den Konzepten *full_pers_name*, *last_name* und *suffix_name* gefunden und der Kategorie *LASTNAME* zugeordnet werden, damit sie innerhalb des Dokuments, falls sie ohne einen Trigger vorkommen, erneut gesucht werden können. Um die Ausbeute zu erhöhen, wird ein Nachname sowohl mit als auch ohne Titel abgespeichert (z.B. *van Haagen* und *Haagen*). Meist handelt es sich zwar nicht mehr um dieselbe Person. Weil jedoch in diesem Projekt Eigennamenerkennung und nicht Identifikation betrieben wird, ist es irrelevant, dass zwei verschieden Personen mit dem entsprechenden Namen bezeichnete werden. Es entsteht kein Fehler. Mit dem Konzept *learn_pers_name* überprüft man, ob ein Element der Kategorie *Lastname*, die in den Konzepten *full_pers_name*, *last_name* oder *suffix_name* gebildet worden ist, hinter bzw. vor der ersten Fundstelle nochmals auftritt (vgl. Abbildung 36).

```
CONCEPT "find_last_name"
IF
        ($?:POSLASTNAME)
CONDITION
        find("$0":LASTNAME, +10000);
        (find("full_name":CONCEPT)
        OR find("last_name":CONCEPT)
        OR find("suffix_name":CONCEPT));
        find("pos_last_name":CONCEPT);
        NOT overlaps_with("full_name":CONCEPT);
        NOT overlaps_with("last_name":CONCEPT);
        NOT overlaps_with("suffix_name":CONCEPT);
        create("$0":PERS)
END
ENDMODULE
```

Abbildung 36

Beim gesuchten Element muss es sich um einen möglichen Nachnamen (*POSLASTNAME*) handeln. Der Suchraum (in diesem Fall ist von mir + 10'000 empirisch ermittelt worden) muss der Dokumentengrösse angepasst werden, damit gewährleistet ist, dass Nachnamen nicht über Dokumentgrenzen hinweg gesucht werden.

Wie bei der Implementierung im Deutschen ist es notwendig, dass Nachnamen im ersten Schritt, also durch die Konzepte *full_pers_name*, *last_name* und *suffix_name*, korrekt erkannt werden. Eine Fehlanalyse eines Nachnamens hätte zur Folge, dass durch die Konzepte *learn_pers_name* der inkorrekte Nachname an jeder Stelle innerhalb des Dokuments erneut als Personennamen analysiert würde und es somit zu vielen Folgefehler kommt. Es handelt sich um dieselbe Problematik wie beim Wiedererlernen der Firmennamen in Kapitel III 4.3. „Erkennung von Firmennamen (De)".

6.2. Erkennung von geographischen Namen (En)

In der Implementierung zur Erkennung von geographischen Eigennamen werden wie im Deutschen nur Namen erkannt, die in den Ressourcen aufgelistet sind. Es bestehen keine Regeln, die auf Grund von Trigger Namen erlernen.

Um zyklische Abhängigkeiten innerhalb der Konzepte zu vermeiden, mussten die Erkennungsregeln in eigene Module getrennt von den Ressourcen implementiert werden. Die Erkennungsregel besagt, dass in den Ressourcen nachgeschaut werden muss, ob ein bestimmter Name aufgeführt ist. Weil in englischen Texten häufig Vornamen bzw. Nachnamen und einteilige geographische Eigennamen gleich lauten (z.B. *Dallas*), muss bei der Implementierung von einzelnen geographischen Namen (*find_single_geonamen*) berücksichtigt werden, dass ein Name nur dann eine geographische Entität ist, wenn in seiner direkten Umgebung weder Triggerwörter für Personenamen auftauchen (Anrede, Titel oder eine zweiter Vorname) noch

ein grossgeschriebenes Wort (Nachname) folgt und der Name innerhalb des Dokuments nicht als Nachname gelernt worden ist (*learn_pers_name*). Im Deutschen bestand dieses Problem hauptsächlich mit geographischen Adjektiven, die auf *er* endeten (z.B. *Schweizer*). Die spezifischere Unterscheidung zwischen Personennamen und einteiligen geographischen Eigennamen hat mit sich gebracht, dass die Erkennung von einzelnen geographischen Eigennamen in einem eigenen Modul implementiert werden muss (*geo_rec_II*), weil sonst eine zirkuläre Abhängigkeit zwischen Personennamenerkennung und Erkennung von geographischen Namen erzeugt würde. Personennamen sind nämlich in dem Sinn von mehrteiligen geographischen Eigennamen abhängig, dass mehrteilige geographische Namen, die gleich lauten wie Personennamen (z.B. *St. Petersbug*), vorzugsweise als geographische Eigennamen analysiert werden, falls vor dem Nachnamen weder Titel noch Anrede gefunden wird (vgl. Kapitel III 4.2. „Erkennung von geographischen Namen“, Beispiel 34, S. III-104). Abhängigkeiten, die zwischen Konzepten zur Personennamenerkennung und zur Erkennung von geographischen Eigennamen in englischen Texten bestehen, sind in Abbildung 37 und Abbildung 38 aufgezeigt.

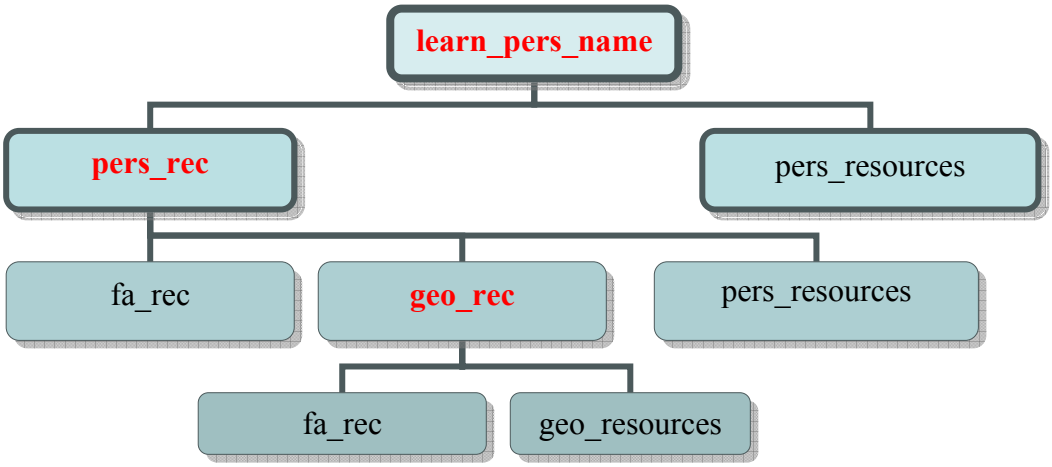


Abbildung 37

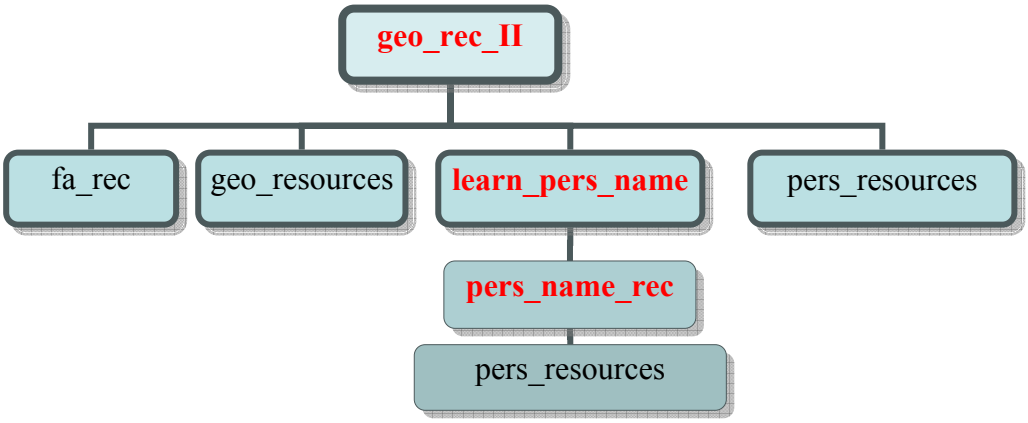


Abbildung 38

Wenn man *geo_rec_II* und *geo_rec* in einem Modul vereinen würde, wäre *pers_rec* von *geo_rec* abhängig und *geo_rec* wiederum von *pers_rec* und die Eigennamenerkennung scheitert auf Grund Fehlverhaltens der Konzepte.

Die Konzepte und Module, welche zur Erkennung von geographischen Eigennamen in Englischen Texten implementiert worden sind, weichen in den wenigsten Punkten von denjenigen zur Erkennung in deutschen Texten ab (vgl. Tabelle 19).

Name	**Funktion**	**Kategorie**
geo_rec	**Regeln zum Erkennen von geographischen Entitäten**	**Modul**
find_hyphenated_geonames	Regel zum Finden eines geographischen Namens, der einen Bindestrich besitzt	Konzept
find_double_names	Regel zum Finden eines zweiteiligen geographischen Namens	Konzept
find_triple_names	Regel zum Finden eines dreiteiligen geographischen Namens	Konzept
find_quadruple_names	Regel zum Finden eines vierteiligen geographischen Namens	Konzept
find_quintuple_names	Regel zum Finden eines fünfteiligen geographischen Namens	Konzept
find_slash_citiesI	Regel zum Finden eines Städtenamens, der einen Schrägstrich als Abkürzung verwendet	Konzept
find_slash_citiesII	Regel zum Finden eines dreiteiligen Städtenamens, der mit Schrägstrich geschrieben wird	Konzept
find_slash_citiesIII	Regel zum Finden eines zweiteiligen Städtenamens, der mit Schrägstrich geschrieben wird	Konzept
find_st_names	Regel zum Finden eines Städtenamens, der mit *St.* beginnt	Konzept
single_geoname_with_abbr	Regel zum Finden eines einteiligen geographischen Namens, dem eine geographische Abkürzung (als Präzision) folgt	Konzept
hyphenated_geonames_with_abbr	Regel zum Finden eines geographischen Namens, der einen Bindestrich besitzt und dem eine geographische Abkürzung (als Präzision) folgt	Konzept
double_names_with_abbr	Regel zum Finden eines zweiteiligen geographischen Namens, dem eine geographische Abkürzung (als Präzision) folgt	Konzept
triple_names_with_abbr	Regel zum Finden eines dreiteiligen geographischen Namens, dem eine geographische Abkürzung (als Präzision) folgt	Konzept
quadruple_names_with_abbr	Regel zum Finden eines vierteiligen geographischen Namens, dem eine geographische Abkürzung (als Präzision) folgt	Konzept
quintuple_names_with_abbr	Regel zum Finden eines fünfteiligen geographischen Namens, dem eine geographische Abkürzung (als Präzision) folgt	Konzept
slash_citiesI_with_abbr	Regel zum Finden eines Städtenamens, der einen Schrägstrich als Abkürzung verwendet und dem eine geographische Abkürzung (als Präzision) folgt	Konzept
slash_citiesII_with_abbr	Regel zum Finden eines dreiteiligen Städtenamens, der mit Schrägstrich geschrieben wird und dem eine geographische Abkürzung (als Präzision) folgt	Konzept

st_names_abbr	Regel zum Finden eines Städtenamens, der mit *St.* Beginnt und dem eine geographische Abkürzung (als Präzision) folgt	Konzept
find_abbrivation	Regel, um geographische Abkürzungen zu erkennen: geographischer Namen; , ; Abkürzung	Konzept
part_of_fa	Liste der Regeln, durch welche ein Firmennamen erkannt wird (ein geographischer Name darf nicht erkannt werden, falls er Teil eines Firmennamens ist)	Konzept
geo_rec_II[120]	**Regeln zum Erkennen von geographischen Entitäten**	**Modul**
find_single_geoname	Regel zum Finden eines einteiligen geographischen Namens	Konzept
part_of_fa	Liste der Regeln, durch welche ein Firmennamen erkannt wird (ein geographischer Name darf nicht erkannt werden, falls er Teil eines Firmennamens ist)	Konzept

Tabelle 19

Im englischen Testkorpus (vgl. Kapitel IV 4. „Testkorpus für die Evaluation in englischen Texten“) kommt es relativ häufig vor, dass ein geographischer Name durch eine Abkürzung des Staates genauer spezifiziert wird. Dies ist damit zu begründen, dass in den USA gleiche Städtenamen in verschiedenen Staaten vorkommen oder im Zuge der Kolonisation Städtenamen aus Europa übernommen worden sind. In einem Text ist es somit nicht immer möglich, ohne eine Zusatzangabe festzustellen, ob es sich z.B. um *Cambridge* in den USA handelt oder in Grossbritannien. Deshalb kommen solche Zusatzinformationen in Texten, die den englischen Sprachraum betreffen, öfter vor als in Texten aus dem deutschen Sprachraum. Im Testkorpus, der für diese Evaluation verwendet worden ist, wurden Abkürzungen als eigenständige, geographische Entitäten markiert, sofern sie durch Klammern oder Kommata als Zusatzinformation zu einem Städtenamen mitgegeben worden sind (vgl. Beispiel 42). Diese Abkürzungen besitzen den gleichen Status wie Ländernamen, die einen Städtename genauerer positionieren (z.B. *NAIROBI, Kenya*) und sollten somit nicht als Teil eines Städtenamens analysiert werden, sondern als eigene Entität. Deshalb habe ich bei der Implementierung mit dem Konzept *find_abbrivation* definiert, dass allein stehende, geographische Abkürzungen als Eigennamen erkannt werden.

Beispiel 42:

…San Mateo, Calif. …. →
Geographischer Eigennamen: *San Mateo*
Geographischer Eigennamen: *Calif.*

[120] Um Zirkuläre Beziehungen zu vermeiden, muss das Konzept find_single_geoname in ein eigenes Modul geschrieben werden.

Im deutschen Testkorpus gehören die Abkürzungen immer zu den Städtenamen und kommen nie abgetrennt von diesen vor. Deshalb war es nicht nötig, Abkürzungen, die allein stehen, zu analysieren.

Das Konzept *part_of_fa* entspricht demjenigen bei der Personennamenerkennung und übernimmt die gleiche Funktion (vgl. Kapitel III 6.1. „Erkennung von Personennamen (En)“). Die Integration des Konzepts im Bedingungsteil eines Konzepts zur Erkennung von geographischen Eigennamen verhindert, dass ein geographischer Eigenname innerhalb eines Firmennamens als geographischer Eigennamen analysiert wird.

6.3. Erkennung von Firmennamen (En)

Die Strategie zum Erkennen von Firmennamen konnte vom Deutschen bis auf die Erkennung über Bindestrichsuffixe übernommen werden. Bindestrichsuffixe existieren im Englischen nicht und sind deshalb bei der Erkennung sowie bei den Ressourcen ersatzlos entfallen.

Diejenigen Konzepte, welche besagen, dass ein Firmennamen nach einem Trigger folgt oder der Trigger am Anfang eines Firmennamens steht (*n_fa_names_follow* und *n_fa_names_folllow_in_quotationmarks*[121]) sind jeweils doppelt unter dem gleichen Namen implementiert. Im einen Konzept ist der Trigger Teil der Entität (vgl. Abbildung 40) im andern Konzept nicht (vgl. Abbildung 39).

```
CONCEPT "one_fa_name_follows"
IF
    ("triggerword_in_front":CONCEPT);
    ("pos_fa_name_part":CONCEPT AND $?:POSFANAMEPART)
CONDITION
    NOT overlaps_with("stop_words_for_fa_name_follows":CONCEPT);
    NOT overlaps_with("two_fa_names_follow":CONCEPT);
    NOT overlaps_with("two_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("three_fa_names_follow":CONCEPT);
    NOT overlaps_with("three_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("four_fa_names_follow":CONCEPT);
    NOT overlaps_with("four_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("five_fa_names_follow":CONCEPT);
    NOT overlaps_with("five_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("six_fa_names_follow":CONCEPT);
    NOT overlaps_with("six_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("seven_fa_names_follow":CONCEPT);
    NOT overlaps_with("seven_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("one_fa_name_follows_in_quotation_marks":CONCEPT);
    create("$0":FA);
    create("$0":ACR)
END
```

Abbildung 39

[121] n entspricht einer Zahl zwischen 1 und 7.

```
CONCEPT "one_fa_name_follows"
IF
    ("triggeraccr_in_front":CONCEPT AND $?:FANAMEPART);
    ("pos_fa_name_part":CONCEPT AND $?:POSFANAMEPART)
CONDITION
    NOT overlaps_with("stop_words_for_fa_name_follows":CONCEPT);
    NOT overlaps_with("two_fa_names_follow":CONCEPT);
    NOT overlaps_with("two_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("three_fa_names_follow":CONCEPT);
    NOT overlaps_with("three_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("four_fa_names_follow":CONCEPT);
    NOT overlaps_with("four_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("five_fa_names_follow":CONCEPT);
    NOT overlaps_with("five_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("six_fa_names_follow":CONCEPT);
    NOT overlaps_with("six_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("seven_fa_names_follow":CONCEPT);
    NOT overlaps_with("seven_fa_names_follow_in_quotation_marks":CONCEPT);
    NOT overlaps_with("one_fa_name_follows_in_quotation_marks":CONCEPT);
    NOT find("trigger_in_front":CONCEPT, -1);
    create("$0 $1":FA);
    create("$0":ACR);
    create("$1":ACR)
END
```

Abbildung 40

Um zu verhindern, dass ein Firmenname sowohl über ein Akronym am Anfang der Entität als auch über einen Trigger vor der Entität erkannt wird, muss das Konzept, welches den Firmennamen über ein Akronym erkennt, um die Bedingung erweitert werden, dass kein Firmentrigger vor dem Konzept gefunden werden darf. Die Anwendung der Methoden, mit welchen ein Firmennamen erkannt wird, ist somit über eine Reihenfolge festgelegt.

Die folgende Liste enthält alle implementierten Methoden zur Erkennung von Firmennamen in englischen Texten.

Name	**Funktion**	**Kategorie**
fa_rec	**Regeln zum Erkennen von Firmennamen**	**Modul**
one_fa_name_follows	Regel, um einen einteiligen[122] Firmennamen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
one_fa_name_follows_in_quotation_marks	Regel, um einen einteiligen Firmennamen in Anführungszeichen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
two_fa_names_follow	Regel, um einen zweiteiligen Firmennamen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept

[122] Teile von Firmennamen sind im Konzept *pos_fa_name_part* in den Ressourcen definiert. Ein Teil eines Firmennamens muss nicht einem Wort entsprechen.

two_fa_names_follow_in_quotation_marks	Regel, um einen zweiteiligen Firmennamen in Anführungszeichen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
three_fa_names_follow	Regel, um einen dreiteiligen Firmennamen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
three_fa_names_follow_in_quotation_marks	Regel, um einen dreiteiligen Firmennamen in Anführungszeichen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
four_fa_names_follow	Regel, um einen vierteiligen Firmennamen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
four_fa_names_follow_in_quotation_marks	Regel, um einen vierteiligen Firmennamen in Anführungszeichen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
five_fa_names_follow	Regel, um einen fünfteiligen Firmennamen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
five_fa_names_follow_in_quotation_marks	Regel, um einen fünfteiligen Firmennamen in Anführungszeichen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
six_fa_names_follow	Regel, um einen sechsteiligen Firmennamen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
six_fa_names_follow_in_quotation_marks	Regel, um einen sechsteiligen Firmennamen in Anführungszeichen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
seven_fa_names_follow	Regel, um einen siebenteiligen Firmennamen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
seven_fa_names_follow_in_quotation_marks	Regel, um einen siebenteiligen Firmennamen in Anführungszeichen zu erkennen, der einem Trigger folgt (der Trigger ist einmal Teil der Entität, das andere Mal nicht)	Konzept
one_fa_name_in_front	Regel, um einen einteiligen Firmennamen zu erkennen, der auf Grund eines Triggers am Ende erkannt wird	Konzept
two_fa_names_in_front	Regel, um einen zweiteiligen Firmennamen zu erkennen, der auf Grund eines Triggers am Ende erkannt wird	Konzept
three_fa_names_in_front	Regel, um einen dreiteiligen Firmennamen zu erkennen, der auf Grund eines Triggers am Ende erkannt wird	Konzept

four_fa_names_in_front	Regel, um einen vierteiligen Firmennamen zu erkennen, der auf Grund eines Triggers am Ende erkannt wird	Konzept
five_fa_names_in_front	Regel, um einen fünfteiligen Firmennamen zu erkennen, der auf Grund eines Triggers am Ende erkannt wird	Konzept
six_fa_names_in_front	Regel, um einen sechsteiligen Firmennamen zu erkennen, der auf Grund eines Triggers am Ende erkannt wird	Konzept
seven_fa_names_in_front	Regel, um einen siebenteiligen Firmennamen zu erkennen, der auf Grund eines Triggers am Ende erkannt wird	Konzept
learn_fa	**Anwendung von Teilen von bereits gelernten Firmennamen**	**Modul**
learn_fa_name	Regel zum wieder Erkennen von Teil-Firmennamen, welche bereits in einem anderen Konzept vollständig erkannt worden sind (Firmennamen ohne Triggerwörter)	Konzept

Tabelle 20

Über das Modul *learn_fa* ist das erneute Suchen nach Teilen von bereits gelernten Firmennamen (Akronyme) implementiert worden. Gleich wie bei den Nachnamen werden in den Konzepten des Moduls *fa_rec* Firmennamen (z.B. *Merill Lynche & Co*) erkannt und als einzelne Teile (z.B. *Merill*) sowie Abfolge von Teilen (z.B. *Merill Lynch*) der Kategorie Akronym (ACR) zugeordnet. Die Ableitung von Akronymen (vgl. Beispiel 40, S. III-113) habe ich nur teilweise implementiert (Ableitung eines Akronym aus Anfangsbuchstaben ist unmöglich). In den Konzepten des Moduls *learn_fa* wird überprüft, ob ein möglicher Firmenname (*pos_fa_name_part*) innerhalb eines vordefinierten Suchraums (z.B. -750[123]) bereits an einer andern Stelle über ein Konzept des Moduls *fa_rec* erkannt worden ist (vgl. Abbildung 41). Wenn dies zutrifft, so handelt es sich beim möglichen Firmennamen ebenfalls um einen Eigennamen, der zuvor nicht erkannt worden ist, weil ein entsprechender Trigger gefehlt hat.

[123] Diese Zahl ist vom mir empirisch ermittelt worden.

```
CONCEPT "learn_fa_name"
IF
        $?:POSFANAMEPART AND NOT "all_filter":CONCEPT
CONDITION
        find("pos_fa_name_part":CONCEPT)
        find("$0":ACR, -750);
        (find("one_fa_name_follows":CONCEPT)
        OR find("two_fa_names_follow":CONCEPT)
        OR find("three_fa_names_follow":CONCEPT)
        OR find("four_fa_names_follow":CONCEPT)
        OR find("five_fa_names_follow":CONCEPT)
        OR find("six_fa_names_follow":CONCEPT)
        OR find("seven_fa_names_follow":CONCEPT)
        OR find("one_fa_name_follows_in_quotation_marks":CONCEPT)
        OR find("two_fa_names_follow_in_quotation_marks":CONCEPT)
        OR find("three_fa_names_follow_in_quotation_marks":CONCEPT)
        OR find("four_fa_names_follow_in_quotation_marks":CONCEPT)
        OR find("five_fa_names_follow_in_quotation_marks":CONCEPT)
        OR find("six_fa_names_follow_in_quotation_marks":CONCEPT)
        OR find("seven_fa_names_follow_in_quotation_marks":CONCEPT);
        NOT overlaps_with("part_of_fa":CONCEPT);
        NOT find("pos_fa_name_part":CONCEPT, -1);
        NOT find("pos_fa_name_part":CONCEPT, +1);
        NOT find
        create("$0":FA)
END
```

Abbildung 41

Weil allerdings zu wenig Firmennamen über die Konzepte des Moduls *fa_rec* erkannt werden, verbessert das Modul *learn_fa_name* die Ausbeutewerte nur gering. Eine beträchtliche Fehlerquelle besteht in den unzureichenden Filtermodulen *filter* und *filterI* (vgl. Kapitel III 4.3. „Erkennung von Firmennamen (De)“). Einige, einteilige Firmennamen, die über *learn_fa_name* gefunden werden, sind keine Firmennamen, werden jedoch über den Filter nicht eliminiert, weil ein entsprechender Eintrag fehlt. Die Ableitungsregeln der Akronyme sind einerseits zu mächtig (das bedeutet, dass Akronyme abgeleitet werden, welche keine Firmenakronyme sind), der Filter andererseits weniger funktionstüchtig als erwartet. So kommt es zu vielen falsch gelernten Firmennamen, die den Präzisionswert beträchtlich drücken. In der Evaluation der Konzepte zur Firmennamenerkennung wurden auf des Grund Versagens des Filters oder der Regeln zur Generierung von Akronymen 31 von 61 verschiedene Entitäten falsch gelernt (vgl. Kapitel IV „Evaluation“).

Den Versuch, Firmennamen mittels Auflzählung in einem Text zu erkennen (vgl. *listing_of_fa_in_front* und *listing_of_fa_behind* in Kapitel III 4.3. „Erkennung von Firmennamen (De)“), habe ich bei der Implementierung im Englischen nicht weiter berücksichtigt, da diese Konzepte im Deutschen die Ausbeute nur geringfügig erhöht haben und dabei Präzisionswerte gefallen sind.

Gleich wie bei der Implementierung der Firmennamenerkennung in deutschen Texten gilt in diesem Fall, dass für gute Präzisionswerte bei hoher Ausbeute eine ‚Zwei-Weg-Lösung' unumgänglich ist. Der Versuch, den Wiederanwendungsschritt in einem eigenen Modul zu implementieren, ist nicht vergleichbar mit den Resultaten, welche eine ‚Zwei-Weg-Lösung' liefern würde. Dass die Strategie zum Erkennen von Firmennamen, welche auch in diesem Projekt grössten Teils angewendet worden ist, gute Präzisionswerte bei hoher Ausbeute liefern kann, zeigen die Evaluationswerte, welche Volk und Clematide in ihrem Projekten erreicht haben (vgl. [Volk und Clematide, 2001]).

Kapitel IV Evaluation

1. Evaluationsskript

Die Evaluation der Module zur Eigennamenerkennung in deutschen und englischen Texten erfolgte mit Hilfe eines Perlskripts von Martin Volk, das Präzision und Ausbeute für jede Kategorie (*Personenname*, *geographischer Name* und *Firmenname*) berechnet. Test und Evaluation erfolgten über dasselbe Korpus. Die verwendeten Korpora werden in Kapitel IV 2. „Testkorpus für die Evaluation in deutschsprachigen Texten“ und Kapitel IV 4. „Testkorpus für die Evaluation in englischen Texten“ detailliert vorgestellt.

Ein Perlskript[124] berechnet Präzision und Ausbeute (in Prozenten), indem die von den Konzepten erkannten Eigennamen mit den manuell annotierten verglichen werden. Als Eingabedatei für das Evaluationsskript dient neben dem annotierten Text die Ausgabe des Indexers (vgl. Abbildung 42).

TYPE=1, POS=1896, VALUE='ab_zu_bau_ende'
TYPE=1, POS=13308,21101, VALUE='baby'
TYPE=1, POS=9288,29792,73987, VALUE='zyklus'
TYPE=10, POS=3020, VALUE='39jaehrige'
TYPE=10, POS=58495, VALUE='suchbegriffe'
TYPE=10, POS=42485, VALUE='zwoelffache'
TYPE=95, POS=1964,19648,21031,35909,45367,52972,VALUE='Washington'
TYPE=95, POS=76277, VALUE='tuerkisches'
TYPE=95, POS=13798, VALUE='westeuropaeischen'
TYPE=96, POS=3702,18096, VALUE='A. C. Markkula'
TYPE=96, POS=76416, VALUE='Adele'
TYPE=96, POS=74862, VALUE='Adolf Klingler'
TYPE=97, POS=52881, VALUE='3M Deutschland'
TYPE=97, POS=38399, VALUE='ADE Angewandte Digital - Elektronik GmbH'
TYPE=97, POS=64316, VALUE='Albert Hartz GmbH'

Abbildung 42

Die Ausgaben *TYPE=1* und *TYPE=10* enthalten die Grundform und die flektierte Form eines Wortes mit der jeweiligen Positionsangabe (*POS*). Diese Informationen sind für die Evaluation unwichtig. Man benötigt lediglich die Typen 95, 96 und 97[125]. *TYPE=95* entspricht geographischen Eigennamen, *TYPE=96* Personennamen und *TYPE=97* den Firmennamen. Die Eingabedatei wird mit den manuell annotierten Eigennamen verglichen. Zur Berechnung von Präzision und Ausbeute werden Positionsangaben der Eingabedatei nicht weiter berücksichtigt. Für die Berechnung ist folglich nur ausschlaggebend, ob ein von den Konzepten erkann-

[124] Das Perlskript hat Martin Volk mir zur Verfügung gestellt. Ich musste es teilweise anpassen.
[125] Die Zahlen sind willkürlich aus dem noch freien Bereich gewählt worden. Gewisse Zahlen wie 1 und 10 sind vom System bereits mit Ausgaben anderer Funktionen besetzt und könne nicht mehr für die Ausgabe von neuen Funktionen verwendet werden.

ter Name tatsächlich im Testkorpus als zur entsprechenden Kategorie zugehörig markiert worden ist oder nicht. Es wird nicht überprüft, ob das markierte und das gefundene Wort an derselben Stelle innerhalb des Textes stehen und somit identisch sind. Folglich wird ebenfalls nicht überprüft, ob jedes gefundene Vorkommen eines Wortes tatsächlich an der entsprechenden Position im Text als Eigenname annotiert wurde. Aus einem korrekten gefundenen Namen wird der Rückschluss gezogen, dass weitere, gleichlautende Eigennamen als Entität richtig erkannt worden sind (vgl. Beispiel 43).

Beispiel 43:

„**Washington** (kg) - Das Programm Surfwatch schützt Kinder vor Pornographie im World Wide Web."
„Die neue Gewinnungstechnik wird von Chemieprofessor William Buhro an der **University of Washington** in St. Louis, Missouri, entwickelt und verspricht beträchtliche Kostenverringerungen bei sehr hoher Werkstoffqualität:…"
[Konradin-Verlag]

Das Wort *Washington* wurde im deutschen Testkorpus (vgl. Beispiel 43) sechsmal als geographischer Eigenname gefunden (vgl. Abbildung 42). Allerdings handelt es sich nur bei fünf Stellen um einen solchen. An der sechsten Stelle ist es Teil eines Organisationsnamens (*University of Washington*) und somit fälschlicherweise den geographischen Eigennamen zugeordnet. Bei der Berechnung der Ausbeute entsteht kein Fehler. Bei den falsch gefundenen Eigennamen erscheint *Washington* einmal und verschlechtert die Präzision leicht. Es genügt, dass *Washington* einmal als geographische Entität korrekt erkannt worden ist, damit die Ausbeute nicht sinkt.

Die Präzision wird im Perlskript gemäss dem in Kapitel I 2.3. „Bewertung von Eigennamen" eingeführten Mass (vgl. Formel 7, S. I-22) berechnet. Die Summe aller korrekt gefundenen Wörter wird durch die Summe aller gefundenen Wörter geteilt. Um den Wert in Prozenten zu erhalten, wird das Resultat mit dem Faktor 100 multipliziert. Korrekt gefundene Wörter sind Wörter, deren manuelle Annotation mit der automatische Kategoriezuordnung durch die Konzeptsensorsprache übereinstimmt, wobei Position und Anzahl nicht weiter berücksichtigt wird. Die Summe aller gefundenen Wörter umfasst alle Wörter, die über die Konzepte bzw. Module einer der drei Kategorien (*Personenname*, *geographische Name* oder *Firmenname*) zugeteilt worden sind.

Die Berechnung der Ausbeute entspricht der Formel 6, S. I-22. Die Summe aller korrekt gefundenen Namen wird durch die Summe aller tatsächlich markierten Namen einer entsprechenden Kategorie geteilt. Um eine Prozentangabe zu erhalten, wird das Resultat mit dem Faktor 100 multipliziert. Unter einem tatsächlich vorhandenen Namen ist zu verstehen, dass

ein Wort manuell einer bestimmten Kategorie zugeordnet und entsprechend markiert worden ist.

Die von mir gewählte Evaluationsart ist weniger präzise, weil sie einerseits Position und Anzahl der Wörter nicht berücksichtigt und ich andererseits in der Entwicklungsphase sowie in der Testphase dasselbe Testkorpus verwendet habe. Trotz der Ungenauigkeit erhält man über die Evaluation einen genügenden Eindruck, wie gut die von mir implementierte Eigennamenerkennung funktioniert und wo die Grenzen liegen.

Die Entscheidung, dasselbe Test- und Evaluationskorpus zu verwenden, ist damit zu begründen, dass es relativ schwierig ist, ein kostengünstiges, geeignetes, genug grosses und manuell annotiertes Testkorpus sowohl in Englisch als auch in Deutsch zu finden.

Die Problematik, dass die Konzepte zu stark auf einen spezifischen Text (in diesem Fall das entsprechende Test- und Evaluationskorpus) ausgerichtet sind (*overtuning*), habe ich zu vermeiden versucht, indem Änderungen der Konzepte möglichst grosse Schwankungen in den Präzisions- und Ausbeutewerten auslösen mussten. Zusätzlich war die Zeit der Evaluation jeweils auf ca. einen Monat beschränkt, um zu spezifische Anpassungen der Konzepte zu vermeiden. Denn je länger Konzepte getestet werden und je spezifischer die Fehleranalysen ausfallen, umso stärker sind die Korrekturen auf das Testkorpus angepasst. Die veränderten Konzepte erreichen auf anderen Korpora keine besseren Resultate mehr, weil die entsprechenden Fälle, die in den Konzepten korrigiert wurden, gar nicht auftreten. Es kann möglicherweise sogar zu einer Verschlechterung der Präzision führen, weil zu spezifische Muster entwickelt wurden, die in anderen Texten selten Eigennamen auslösen. Die bestehenden Regeln, wie z.B. dass ein Wort, das einem spezifischen Trigger folgt und grossgeschrieben ist, einer Kategorie von Eigenname angehört, kann bei falsch gewählten Triggern zu fehlerhafter Eigennamenerkennung führen. Es kann vorkommen, dass Trigger in einem Textkorpus häufig Eigennamen auslösen, in anderen Texten jedoch häufiger mit beliebigen Nomen auftreten und somit eine Fehlanalyse auslösen. In diesem Sinn können zu spezifische Anpassungen der Konzepte in anderen Textkorpora nicht nur die Eigennamenerkennung nicht verbessern, sondern sogar zu einer Verschlechterung der Präzision führen.

2. Testkorpus für die Evaluation in deutschsprachigen Texten

Bei der Textkollektion, die ich zum Testen und Evaluieren der Konzepte zur Eigennamenerkennung in deutschen Texten verwendet habe, handelt es sich um vier Ausgaben der Computer-Zeitung aus dem Jahr 1996: Computer-Zeitung Nr. 4 vom 25. Januar 1996, *Computer* Zeier-Zeitung Nr. 6 vom 8. Februar 1996, Computer-Zeitung Nr. 7 vom 15. Februar 1996 und Computer-Zeitung Nr. 11 vom 11. März 1996. Die Computer-Zeitung ist eine deutsche Wo-

chenzeitschrift, die Nachrichten im Bereich Informatik publiziert. Sie ist im Web unter der Adresse http://www.computer-zeitung.de zugänglich. Der Konradin Verlag (Herausgeber) hat Nachrichten aus den Jahren 1993 bis 1997 auf CD-Rom zur Volltextsuche archiviert (vgl. [Konradin-Verlag]). Die zur Evaluation verwendeten Texte stammen von dieser CD-Rom.

Die vier Ausgaben der Computer-Zeitung wurden von Marc Rössler an der Universität Duisburg manuell nach eigenen Richtlinien annotiert. Sie umfassen ca. 100'000 Wörter. In den Artikeln wurden Personennamen, Ortsnamen, geographische Adjektive und Nominalisierungen sowie Organisationsnamen markiert. Als Personennamen gelten Kombinationen aus Vornamen, Initialen und Nachnamen sowie Nachnamen allein. Die Anrede zählt nicht zum Personennamen (z.B. *Dr.* <per>***M. Michael König***</per>). Für Ortsnamen, geographische Adjektive und Nominalisierungen ist dieselbe Markierung verwendet worden (z.B. **<loc>***Frankfurt/M***</loc>** oder **<loc>***Amerikaner***</loc>**). Zu Ortsnamen zählen Namen von Dörfern, Städten, Kantonen bzw. Bundesländern oder Staaten, Nationen und Kontinenten. Zu geographischen Adjektiven und Nominalisierungen zählen Adjektive, die von Ortsnamen abstammen (z.B. *Zürcher*, *deutsche*) sowie von Ortschaften abgeleitete Nominalisierungen, die als Bezeichnung für Personen verwendet werden (z.B. *Kalifornier, (der) Kölner*). Die Kategorie *Organisationen* umfasst Namen von Firmen (z.B. *Microsoft*), Konzernen (*Kirch-Gruppe*), Lehranstalten (z.B. *Universität Karlsruhe*), Instituten (z.B. *Fraunhofer-Institut für Mikroelektronische Schaltungen und Systeme*), amtliche Organisationen (z.B. *Bundesamt für Sicherheit in der Informationstechnik*), Fernsehprogrammanbieter (z.B. *ARD*) und anderen Zusammenschliessungen oder Vereinigungen (z.B. *Bund demokratischer Wissenschaftlerinnen und Wissenschaftler*).

Die Entitäten werden im Text mit folgenden drei Markierungen annotiert:

- <per>…</per> (Person)
- <loc>…</loc> (geographische Entität)
- <org>…</org > (Organisation)

Ein annotierter Text sieht folgendermassen aus (vgl. Abbildung 43):

CZ 4/1996, S. 1

Kündigung

<org>Novell</org> bald ohne <per>Laube</per>

<loc>Provo</loc> (cz) - Novells Chief Technology Officer <per>Sheldon Laube</per> wirft das Handtuch. <per>Laube</per> gilt als einer der Planer der Novell-Strategie, sich in Zukunft als reine Netzwerk-Company zu positionieren.
Seine Entscheidung fiel nur wenige Tage nach der Ankündigung, daß die erst vor kurzem gebildeten drei Novell-Produktgruppen wieder unter einem Dach zusammengefaßt werden. Zwar habe CEO <per>Robert Frankenberg</per> versucht, <per>Laube</per> zu halten, dieser jedoch aufgrund der inkonsequenten Strategieumsetzung abgelehnt, berichtet die New York Times.
[Konradin-Verlag]

Abbildung 43

Die Annotation von Rössler ist nicht fehlerfrei, da sie in relativ kurzer Zeit von nur einer einzigen Person erstellt wurde. Fehler (z.B. fehlende Annotierung bei Personennamen) habe ich jedoch nicht korrigiert. Sie beeinflusst entsprechend Präzision und Ausbeute (vgl. „*Sonstige Fehler*“ bei den folgenden Analysen).
In gewissen Grenzfällen ist eine Diskussion der Richtlinie angebracht, welche Rössler zur Annotation festlegt. Es handelt sich dabei um die folgenden beiden Punkte:

1. Nominalisierungen, welche auf *er* enden und von Ländernamen abgeleitet worden sind, werden zu geographischen Entitäten gerechnet (vgl. Beispiel 44)

   Beispiel 44:
   „Mit dem Spielcomputer Pippin haben die **Kalifornier** ihr Verständnis …“
   [Konradin-Verlag]

   Es ist nicht offensichtlich, dass es sich bei diesen Nominalisierungen um Entitäten handelt, da sowohl propriale als auch appellative Eigenschaften vorhanden sind (vgl. Kapitel III 4.1. „Erkennung von Personennamen“). Es handelt sich einerseits um eine Gruppe Menschen, die entsprechend ihrem Herkunftsland mit einem Namen zusammengefasst werden (appellativ), andererseits besitzt ein solches Nomen die Funktion, auf eine existierende Ortschaft zu verweisen (proprial)[126]. Es hängt von der Gewichtung der Eigenschaften ab, ob diese Nomen zu den Eigennamen zählen sollen oder nicht. In der Onomastik wird sogar die Ansicht vertreten, diese Nomen zu Personen-

[126] Die Begriffe *appellativ* und *proprial* sind im Kapitel III 4.1. „Erkennung von Personennamen“ erklärt.

namen zu zählen (vgl. [Blanár, 2001], S. 40f.). Ich bin entgegen Rösslers Annotation der Ansicht, dass es sich bei diesen Nominalisierungen um keine Entitäten handelt, deshalb entstehen bei der Evaluation Fehler, und sowohl Präzision als auch Ausbeute fallen bei geographischen Eigennamen niedriger aus.

2. Geographische Adjektive, welche am Anfang eines Organisationsnamens auftreten, werden getrennt vom Organisationsnamen als geographische Entität markiert (vgl. Beispiel 45).

Beispiel 45:

„<loc>Mannheimer</loc> <org>Topware AG</org>“
[Konradin-Verlag]

Ob das geographische Adjektiv separat annotiert werden soll oder nicht, ist allein die Entscheidung des Annotators. Diese Entscheidung habe ich für das Deutsche übernommen. Im Englischen fallen geographische Adjektive (z.B. ***Swiss*** *chees* = Schweizer Käse) und Nominalisierungen (*a Swiss* = ein Schweizer). Zu Unterscheiden, ob es sich folglich um ein Eigenname oder ein Appellativ handelt[127], ist somit nicht möglich. Deshalb habe ich im Englischen beschlossen keine geographischen Adjektive zu annotieren, falls sie nicht gleich wie der Städte- bzw. Regionsnamen lauten.

Zur Evaluation und zum Testen wurden alle Artikel der Computer-Zeitung in einem einzigen Dokument abgespeichert. Für die Testphase der Konzepte musste jegliche Annotation ausser Markierung von Dokumentanfang und -ende entfernt werden. Ansonsten ist nichts am Ursprungstext geändert worden.

3. Evaluation der Eigennamenerkennung in deutschsprachigen Texten

Die Evaluation der Konzepte ist in mehreren Teilschritten verlaufen. Jeder Teilschritt hat eine Veränderung der Ausbeute und Präzision zur Folge gehabt. Anfangs betrafen die Änderungen hauptsächlich die Erkennungsregeln, da grundsätzliche Fehler vorhanden waren (z.B. vom geographischen Eigennamen *New York* wurde der Teil *New* erkannt), später musste ich hauptsächlich die Ressourcen anpassen, um bessere Resultate zu erhalten (z.B. Erweitern der Liste im Konzept *special_single_city_names*).

Die Endresultate der Evaluation sind in folgender Tabelle zusammengefasst (vgl. Tabelle 21)

[127] Ich habe im Deutschen lediglich Adjektive, nicht aber Nominalisierungen, zu den geographischen Eigennamen zu zählen. Diese Entscheidung ist allerdings willkürlich gefällt worden.

	DEUTSCH	
	Ausbeute in Prozent	*Präzision in Prozent*
alle	**44.68**	**73.80**
Personenname	**78.46**	**84.14**
geographischer Name	**84.34**	**72.61**
Firmenname	**14.39**	**58.62**

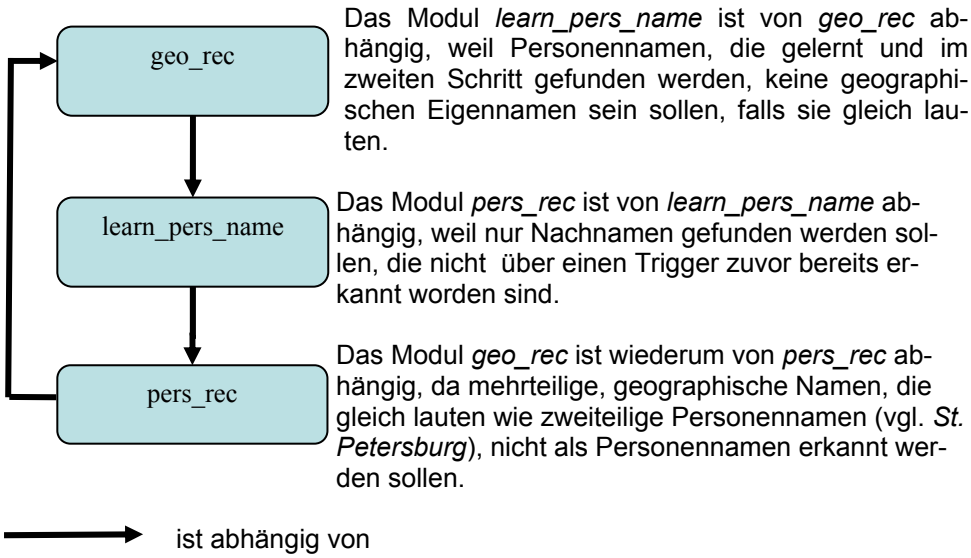
Tabelle 21

Die schwache Firmennamenerkennung beeinflusst die Gesamtresultate beträchtlich. Ausbeute und Präzision aller Eigennamen sind bezüglich der Güte der Konzepte nicht aussagekräftig. Die Resultate sollten in jeder Kategorie einzeln betrachtet werden. Die folgenden Unterkapitel enthalten eine genaue Analyse.

3.1. Evaluation von Personennamen (De)

Während der Evaluationsphase habe ich die Konzepte zur Personennamenerkennung in folgenden drei Punkten verändert.

1. Die Ausgabe des Indexers und die Notation der Wörter innerhalb des Testkorpus mussten im Evaluationsskript (Perlskript zur Evaluation) einander angepasst werden. Das bedeutet, dass Abstände vor und nach einem Bindestrich oder Abstände zwischen Buchstabe und Punkt in einem Initial entfernt oder diakritische Zeichen in eine Normform umgewandelt werden mussten, damit zwei Namen tatsächlich als identisch erkannt wurden (z.B. *Pia Grun**d** - **L**udwig* → *Pia Grun**d-L**udwig* oder *Peter Pag**é*** → *Peter Pag**e***).
2. Die Ressource der Anreden wurde um Wörter wie *Herr* und *Frau* ergänzt.
3. Zirkuläre Abhängigkeiten zwischen Firmennamen, geographischen Namen und Personennamen mussten behoben werden. Während der Evaluationsphase wurde das bestehende *Relevancy* durch eine neuere Version ersetzt, die schnellere Indexierungszeiten besass, das lokale *find* mit einer Variable unterstützte (vgl. Kapitel III 2.3. „Erweiterungen in der Konzeptsensorsprache zur Erkennung von Eigennamen“) und zirkuläre Abhängigkeiten von Modulen beim Kompilieren erkannte. Dadurch wurden diverse Abhängigkeiten festgestellt (vgl. Abbildung 44)



Die Module sind nicht funktionstüchtig, weil kein Startmodul existiert.

Abbildung 44

Indem ich Regeln zur Erkennung von Eigennamen und Ressourcen konsequent getrennt habe und Einschränkungen von Erkennungsregeln über die Ressourcen implementiert habe, konnten alle zirkulären Abhängigkeiten gelöst werden, so dass die Funktion der Konzepte nicht mehr beeinträchtigt wurde.

Eine höhere Ausbeute der Firmennamen hat eine höhere Präzision der Personennamen mit sich gebracht. Personennamen hängen von Firmennamen insofern ab, dass Personennamen, die Teil eines Firmennamens sind, nicht der Kategorie *Personennamen* zugeordnet werden dürfen. Sie sind hierarchisch miteinander verknüpft und beeinflussen einander bezüglich Präzision und Ausbeute.

Während der gesamten Evaluationsphase ist die Ausbeute von 64,04% auf 78.46% gestiegen, die Präzision von 80.24% auf 84.14%. Eine genaue Fehleranalyse der letzten Evaluationsdaten hat mehrere problematische Punkte ergeben. Die Fehlerkorrekturen, falls der Fehler behoben werden konnte, sind in neue Konzepte eingegangen, die ich jedoch nicht mehr evaluieren konnte, weil die Projektzeit bei der Firma *Eurospider* abgelaufen war. Der **Verlust der Ausbeute** ist auf Grund einer genauen Analyse der letzten Evaluationswerte hauptsächlich in folgenden Fehlverhalten der Konzepte zu begründen:

1. Fehlanalyse von Vornamen und doppelten Vornamen

Vornamen, die alleine in einem Text auftauchen (d.h. ohne Nachnamen, Titel oder Anrede), werden den Personennamen zugeordnet. Wenn jedoch der Nachname gleich lautet wie ein Vorname, kann eine Anrede davor stehen, weil es sich nicht um einen allein stehenden Vor-

namen handelt. Ein Nachname, der ebenfalls in der Ressource der Vornamen aufgeführt ist, wird immer als Vorname analysiert

Beispiel 46:

Professor Volker Claus …
Professor **Claus** …*

Die Namen *Volker Claus* bzw. *Claus* werden als Vornamen erkannt. Das Konzept *pre_name*, welches allein stehende Vornamen zu erkennen erlaubt, schlägt jedoch fehl, weil eine Anrede vor dem Namen steht. Somit werden solche Namen nicht als Personennamen erkannt, was eine niedrigere Ausbeute zur Folge hat. 8[128] Namen von 535 wurden auf Grund des oben beschriebenen Fehlers nicht gefunden.

Eine weitere Problematik, die im Konzept *pre_name* besteht und Fehler verursacht, ist die Einschränkung, dass kein grossgeschriebenes Wort folgen darf, da es sich sonst um einen Personennamen handelt, der im Konzept *last_name* erkannt würde. Um Konzepte robuster zu gestalten, wurde die Einschränkung auf die zwei folgenden Wörter ausgedehnt. Der Name in Beispiel 47 wurde nicht erkannt, weil *Roche* auch ein Vorname sein kann[129] und die Kombination als doppelter Vorname analysiert wurde, nicht aber als Vor- und Nachname. Beim zweiten folgenden Wort (das Komma wird wie ein Wort gezählt) handelt es sich um ein grossgeschriebenes, so dass die Erkennung fehlschlägt.

Beispiel 47:

„…, so **Klaus Roche**, Geschäftsführer …“
[Konradin-Verlag]

Die grössere Robustheit, die durch die Erweiterung auf zwei Wörter erreicht werden soll, hat jedoch eine tiefere Ausbeute zur Folge, so dass ich den Abstand, in dem kein grossgeschriebenes Wort folgen darf, in den neuen, ungetesteten Konzepten auf 1 verringert habe.

17 Namen von 535 wurden wegen zu gross gewählten Einschränkungsraums nicht gefunden. Dies entspricht 4,67%. Es ist nicht möglich, in einem Konzept zwischen Vor- und Nachnamen, die gleich lauten, zu unterscheiden, auch wenn so einige Fehler verhindert werden könnten. Es wäre möglich, Vornamen, die öfter als Nachnamen auftreten, aus der Vornamensliste zu streichen. Dies hat allerdings zur Folge, dass gewisse Namen, in denen der Vornamen tatsächlich als solcher vorkommt, nicht erkannt werden.

[128] Bei diesen Zahlen handelt es sich um verschiede Namen.
[129] Weil die Vornamensliste sehr gross ist, sind seltene Vornamen wie *Roche* ebenfalls enthalten.

2. Festlegung einer nicht zirkulären Erkennungsreihenfolge

Personennamen, die innerhalb eines Firmennamens stehen, gehören nicht zur Kategorie *Personenname*. Wenn der Firmenname nur aus einem möglichen Personennamen besteht (vgl. Beispiel 48) und weder vorher noch nachher im Dokument mit einem Trigger vorkommt, kann er nicht als Firmenname erkannt werden. Der gefundene Ausdruck wird fälschlicherweise den Personennamen zugeordnet und verschlechtert somit die Präzision (mehr dazu weiter unten im Kapitel).

Beispiel 48:

„Als Beispiel nennt Lünendonk die Übernahme von **A.T. Kearny** durch den Outsourcing-Riesen EDS.“
[Konradin-Verlag]

Um zu verhindern, dass Namen, die Teil eines Firmennamens sind, systematisch falsch zugeordnet werden, besteht die Einschränkung, dass in einem gewissen Bereich kein Firmentrigger gefunden werden darf (vgl. Abbildung 45). Damit auch Fälle mitberücksichtigt werden, bei denen der Personennamen innerhalb eines Firmennamens steht und somit der Firmentrigger nicht direkt dem Namen folgt (z.B. ***Booz Allen*** *und Hamilton* ***Inc.***), habe ich den Bereich, in dem kein Firmentrigger stehen darf, auf +8 gesetzt. Ein Firmenname besteht nach eigener Definition aus höchstens sieben Teilen. Im Extremfall könnte der Personennamen an erster Stelle des Firmennamens stehen und der Firmentrigger würde erst an 7. oder 8. Stelle folgen. Bei der Fehleranalyse hat sich dieser Bereich (8) jedoch als zu gross erwiesen.

```
/*Christof Schulte*/

CONCEPT "last_name_type1"
IF
    (("pre_pers_name":CONCEPT AND $?:NAMEPART)
    OR ("pre_pers_double_name":CONCEPT AND $?:NAMEPART)
    OR ("hyphenated_pre_pers_name":CONCEPT AND $?:HNAMEPART));
    ("pos_last_name":CONCEPT AND $?:POSLASTNAME)
CONDITION
    /*längere Namen werden vor kürzeren erkannt*/
    NOT find( "sal_pers_name":CONCEPT, -3);
    NOT find( "initial_of_name":CONCEPT, -2);
    /*Das folgende, grossgeschriebene Wort darf kein Firmen-Schlüsselwort sein*/
    NOT find( "triggerword_behind":CONCEPT, +8);
    /*Das vorherige Wort darf kein Firmen-Triggerwort (Firma,...) sein.*/
    NOT find( "triggerword_in_front":CONCEPT, -1);
    /*Innerhalb des Konzeptes darf kein Firmen-Triggerwort stehen. Z.B. Morris AG
    soll als FA erkannt werden.*/
    NOT overlaps_with("triggerword_behind":CONCEPT);
    /*Wenn es gleichzeitig ein GEO ist, dann soll es als GEO analysiert werden.*/
    NOT overlaps_with("find_double_names":CONCEPT);
    create("$0 $1":PERS);
    create("$1":LASTNAME)
END
```

Abbildung 45

In 24 Fällen führte die Einschränkung dazu, dass ein Personenname nicht erkannt wurde, weil ein Firmentrigger im Bereich von acht Wörtern danach vorkam. Der gefundene Trigger gehört jedoch zu einem neuen Firmennamen (vgl. Beispiel 49).

Beispiel 49:

„Christof Schulte wurde in den Vorstand der **Escom AG** berufen.“
[Konradin-Verlag]

Die Ausbeute ist somit um 4,49% verschlechtert worden. Die Begrenzung des Bereichs auf weniger Wörter nach einem Personennamen würde die Ausbeute erhöhen. Allerdings könnte die Präzision darunter leicht leiden, da das System in gewissen Fällen den Name fälschlicherweise der Kategorie *Personenname* zuordnet, weil der entsprechende Firmentrigger zu weit entfernt steht. In der Computer-Zeitung sind diese Fälle jedoch nicht vorhanden.

3. Unkorrekte Grossschreibung innerhalb eines Dokuments

Grossschreibung innerhalb des Testkorpus ist bei Personennamen nicht immer eingehalten worden. Wenn Vor- und Nachnamen nicht mit einem Grossbuchstaben beginnen, ist es unmöglich, den Personennamen über die Konzepte zu erkennen. Die Eigennamenerkennung ist auf eine korrekte Grossschreibung innerhalb des Dokuments angewiesen. Wenn die Eigennamenerkennung in Bezug auf Grossschreibung robuster gestaltet würde, wäre mehr externe Information nötig, um Ambiguitäten aufzulösen (vgl. Beispiel 50).

Beispiel 50:

Lachen (Ortschaft)
lachen (Verb)

Deutsch sowie Englisch gehören zu den Sprachen, die die Grossschreibung von Eigennamen unterstützen. Die optische Markierung erlaubt einem Rezipienten, die Entität leichter zu erkennen[130]. Die Grosschreibung jedoch ist weder im Deutschen noch im Englischen interferenzfrei angewendet. Neben Eigennamen werden auch Wörter an Satzanfängen und Nomen (De) grossgeschrieben. Grossschreibung ist folglich für die Eigennamenerkennung ein zentrales, wenn auch nicht genügendes Merkmal, und wurde von mir beim Erstellen der Regeln mitberücksichtigt. Die Ausbeute wird bei Kleinschreibung von Eigennamen verschlechtert, ohne dass die Konzepte eine Fehlanalyse erzeugen. Bei der Evaluation der Konzepte mit dem deutschen Testkorpus trat dieser Fehler nur in vier Fällen auf. Wenn in einem Text jedoch keine Grossschreibung eingehalten wird, ist das Erkennen von Eigennamen über die in diesem Projekt erstellten Konzepte unmöglich.

4. Keine Analyse von Nachnamen ohne Triggerwörter

Wenn ein Verfasser eines Textes der Meinung ist, dass seine Rezipienten eine gewisse Person gut kennen, muss der Personenname nicht über einen Vornamen oder sonstigen Trigger genauer spezifiziert werden. Solche Nachnamen, die uneingeleitet im Text vorkommen, können von den Konzepten nicht erkannt werden (vgl. Beispiel 51). Sie müssten, falls möglich, in einer eigenen Ressource aufgeführt werden (vgl. Beispiel 51 a). Wenn eine Art Anrede vorhanden ist, diese jedoch noch nicht in der entsprechenden Ressource steht, könnte der Name durch Hinzufügen der Anrede erkannt werden (vgl. Beispiel 51 b). Es muss jedoch überprüft werden, ob der Anrede in jedem Fall ein Personenname folgt. Falls andere Nomen dem Trigger öfter folgen, würde die Präzision der Konzepte verschlechtert werden.

Beispiel 51:

a) „Die MMBG-Runde, der kurzfristig die Kirch-Gruppe und Vebacom angehörten, ist kleiner geworden. **Kirch** puscht wieder seine d-Box, die MMBG ihren Seca-Decoder."
b) „Auch daß *Kanzler* **Kohl** - der die Datenautobahn einst zur Ländersache erklärte, …"

[Konradin-Verlag]

In 7 Fällen wurden Nachnamen wegen Fehlens einer Anrede in der Triggerliste oder Fehlens jeglicher Trigger im Text nicht gefunden. Die Ausbeute wird folglich nur gering gedrückt, da

[130] Mehr dazu in [Harweg] S. 195ff.

Nachnamen selten alleine auftreten und die vorhandene Liste mit Anreden alle typischen Anreden enthält.

5. Fehlende Vornamen in der Ressource

20 von 535 Personennamen wurden nicht erkannt, weil der Vorname in der entsprechenden Ressource fehlt. Es handelt sich meist um asiatische oder sehr seltene europäische Vornamen (vgl. Beispiel 52).

Beispiel 52:

Ina Hönicke
Masashi Kojima

Die Ausbeute wurde durch das Fehlen der entsprechenden Vornamen um 3,74% verschlechtert. Die Vornamen können leicht der Liste hinzugefügt werden. Die Verbesserung der Evaluationswerte würde in diesem Fall nur wenig aussagen, da die Erweiterung der Vornamensliste zu spezifisch für das Testkorpus ist und somit in einem anderen Text vermutlich keine Auswirkungen zeigen würde.

6. Nachnamen im Genitiv

Nachnamen, die im Genitiv stehen, werden oft ohne Trigger verwendet, weil sie bereits in längerer Form (mit Vornamen oder Titel) früher eingeführt worden sind. Solche Nachnamen werden von den Konzepten nicht erkannt, weil sie meist im Nominativ gelernt worden sind und somit das Konzept *find_last_name* sie in veränderter Form (mit einem angefügten *s*) nicht wieder erkennt (vgl. Beispiel 25, S. III-79).

9 von 535 Personennamen sind auf Grund dieses Fehlers nicht erkannt worden, weshalb sich die Ausbeute verschlechterte. Mit der ausschliesslichen Verwendung der Konzeptsensorsprache kann dieses Problem nicht gelöst werden. Im Testkorpus ist das Problem mit 1,68% jedoch relativ selten aufgetreten.

7. Folgefehler

Das Konzept *find_last_name* bedingt, dass ein Nachname zuvor in einem Konzept erkannt worden ist, welches den Nachnamen über einen Trigger analysiert hat. Wenn dies nicht der Fall ist, wird ein Nachname, wenn er ohne Trigger vorkommt, nicht gefunden. Ich bezeichne dies als Folgefehler (vgl. Kapitel III 4.1. „Erkennung von Personennamen“).

15 von 535 Namen sind wegen eines Folgefehlers nicht erkannt worden. Die Ausbeute wurde somit um 2,8% gedrückt. Weil diese Zahl relativ gering ist, kann daraus geschlossen werden, dass die Ausbeute, die die Konzepte zur Eigennamenerkennung mit Hilfe von Triggern erreichen, relativ hoch ist.

8. Sonstige Fehler

12 Namen wurden wegen diverser kleiner Fehler nicht erkannt. Dabei handelt es sich unter anderem um fehlenden Abgleich zwischen der Form[131], wie ein Name in der Ausgabe des Indexers steht, und der Form, wie er im annotierten Testkorpus steht (vgl. Punkt 1, S. IV-142). Bei dieser Fehlerart wurden die Namen von den Konzepten korrekt erkannt, konnten aber im Originaltext nicht gefunden werden, weil einer der Personennamen in einer anderen Schreibweise vorlag.

Beispiel 53:

Output Indexer: Tim O ' Donnell
Annotierte Version: Tim O'Donnell

In zwei Fällen wurden fehlende Analysen durch einen zu klein gewählten Suchraum im Konzept *find_last_name* verursacht. Wenn der Suchraum vergrössert wird, sinkt jedoch der Präzisionswert.

Gewisse Personennamen wurden nur in gekürzter Form gefunden (*Hans Dieter* statt *Hans Dieter-Wiedig*), bei anderen umfasste der Personenname zu viele Worte (*Axel Kuhn vom Frauenhofer Institut* statt *Axel Kuhn)*. Alle Analysen, auch wenn sie aus einem Teil einer korrekten Entität bestanden, wurden als Fehler gewertet und in die Liste der fehlenden Namen eingetragen. Zu korrekte Personennamen zählten nur vollständig gefundene Entitäten.

Fehler, die die **Präzisionswerte gedrückt** haben, sind hauptsächlich durch falsche Analysen von Personennamen entstanden, die durch anderen Kontext ihren Eigennamenstatus abgegeben haben (vgl. Punkt 1, S. IV-145). Personennamen, die zur Benennung eines Objektes verwendet werden (z.B. Restaurant, Bar oder Organisation), verlieren ihrer appellative Funktion und zählen somit nicht mehr zu der Kategorie *Personennamen*. Bei der Analyse wurden Personennamen falsch erkannt, die Teil des Namens einer Organisation (16), eines Produkts (7), eines Projekts (1) und eines Restaurants oder einer Bar (3) sind. Solange jedoch diese Namen nicht von neuen Konzepten erkannt werden können, ist es schwierig, solche Fehler zu vermeiden. Allerdings birgt die Erkennung von Organisationsnamen oder gar Produktenamen noch mehr Probleme als Personennamenerkennung, geographische Eigennamenerkennung oder Firmennamenerkennung. Geographische Eigennamen wurden nur dann fälschlicherweise als Personennamen analysiert, wenn sie keinen Eintrag in den Ressourcen besassen (z.B. *Ann Arbor*). Eine gute Entitätenerkennung in den verschiedenen Kategorien bewirkt folglich eine

[131] Fehler, die durch das Evaluationsskript verursacht worden sind, verschlechtern die Evaluationswerte ungerechter Weise, da eine korrekte Analyse des Eigennamens erfolgte. Weil die Evaluationswerte dadurch nur wenig gedrückt werden, habe ich sie nicht speziell berücksichtigt.

höhere Präzision, weil die Eigennamen voneinander abhängen und je nach Kontext verschiedenen Kategorien angehören können.

Insgesamt könnte die Ausbeute der Konzepte mit geringem Aufwand (d.h. Hinzufügen fehlender Vornamen sowie weiterer Anreden wie *Kanzler* oder *Regisseur* und Anpassung des Raums, in dem ein Firmentrigger erscheinen darf, auf +2) erhöht werden. Höhere Präzision kann nur mit einer verbesserten Firmennamenerkennung (oder einer Ausweitung auf Organisationsnamen) erreicht werden.

3.2. Evaluation von geographischen Eigennamen (De)

Während der Evaluationsphase haben die folgenden drei Anpassungen eine Verbesserung der Präzision und Ausbeute bewirkt, so dass die oben angeführten Werte (vgl. Tabelle 21) erreicht worden sind.

1. In der Ausgabe des Indexers werden geographische Adjektive, die auf *isch* enden, immer in Kleinbuchstaben ausgegeben unabhängig davon, ob das Wort im Originaltext gross oder kleingeschrieben ist (vgl. Beispiel 54).

   Beispiel 54:

   Annotierter Text:
   „<loc>**Südamerikanisch**</loc> ist jedoch nicht nur das Musikangebot, auf der Speise karte findet man viele <loc>brasilianische</loc> Gaumenfreuden.“
   [Konradin-Verlag]

   Indexerausgabe:
   TYPE=95, POS=76623, VALUE='**suedamerikanisch**'[132]

   Um markierte Wörter aus dem annotierten Text mit den Ausgaben aus dem Indexer vergleichen zu können, wurde im Evaluationsskript festgelegt, dass entsprechend markierte Wörter aus dem annotierten Text, die auf *isch* enden, immer kleingeschrieben sein müssen. Das Wort *Deutsch* wurde mit einer eigenen Regel umgewandelt, weil es bezüglich der Endung (*sch*) eine Ausnahme bildet. Sieben weitere Adjektive, die ebenfalls nicht auf *isch* enden, zählen zu nicht erkannten geographischen Eigennamen und haben die Ausbeute um 1,55% gedrückt. Sie müssten in eigenen Regeln normalisiert werden, um die Ergebnisse zu verbessern.

   In den Konzepten kann nicht zwischen Nomen und Adjektiven unterschieden werden, weil die vermeintlichen Adjektive in Grundform gesucht werden und somit jegliche

[132] Die Normierung der Umlaute von traditioneller (ä, ö, ü) auf ausgeschriebene Form (ae, oe, ue) ist bereits vor der Evaluationsphase als Notwendigkeit erkannt und im Perlskript entsprechend eingefügt worden (vgl. Punkt 1, S. IV-142).

Grossschreibung aufgehoben wird. Über das Konzept *isch_adj* werden somit neben geographischen Adjektiven *(kalifornisch*, *englisch*) auch Nominalisierungen (*Englisch*, *Französisch*) gefunden. Nach meiner Definition gehören Sprachen nicht zu geographischen Eigennamen und sollten somit nicht als Entität markiert werden. Im annotierten Text jedoch werden sie markiert. Folglich löst die Übergenerierung der Konzepte keinen Fehler aus, falls eine entsprechende Normalisierung stattgefunden hat.

2. Zirkuläre Abhängigkeiten zwischen Firmennamen, geographischen Namen und Personennamen mussten behoben werden. Gleich wie bei den Personennamen führte dieser Schritt zu einer Verbesserung von Präzision und Ausbeute, weil davor die Konzepte durch zirkuläre Abhängigkeiten blockiert waren und nicht erwartungsgemäss funktionierten (vgl. Punkt 3, S. IV-142).
3. Durch Erweiterung der Ressourcen ist die Ausbeute leicht gestiegen.

Während der Evaluation habe ich festgestellt, dass Abkürzungen von geographischen Namen sehr selten allein vorkommen. Wenn die Abkürzungen alleine gefunden werden können, steigt die Präzision, während die Ausbeute fällt. Weil die Ausbeute möglichst hoch sein sollte, werden geographische Abkürzungen nur in Kombination mit einem weiteren Eigennamen gefunden. Dies hat jedoch zur Folge, dass gewisse wichtige Abkürzungen wie USA oder US nicht gefunden werden. Sie müssen zusätzlich in der Länderliste aufgeführt werden.

Während der gesamten Evaluationsphase ist die Ausbeute von 84,82% auf 84,34% gefallen, weil mehr Firmennamen erkannt worden sind und geographische Eigennamen fälschlicherweise zu den Firmennamen gezählt wurden. Die Präzisionswerte konnten von 71,97% auf 72.61% durch die oben aufgezählten Punkte gesteigert werden.

Eine Analyse der letzten Evaluation hat die folgenden Fehler ergeben, die teilweise in neuen Konzepten verbessert worden sind. Die neuen Konzepte konnten allerdings nicht mehr getestet werden. Die **Ausbeute** wurde durch folgende, problematische Punkte **gedrückt**:

1. verschiedene Annotationsrichtlinien

Im Testkorpus sind neben geographischen Adjektiven auch Nominalisierungen als geographische Eigennamen annotiert worden. Meiner Ansicht nach zählen jedoch Nominalisierungen nicht mehr zu geographischen Entitäten (vgl. Kapitel IV 2. „Testkorpus für die Evaluation in deutschsprachigen Texten“, Beispiel 44, S. IV-140). Deshalb werden sie von meinen Konzepten nicht erkannt. Bei 29 fehlenden Wörtern handelt es sich um Nominalisierungen. 6,43% aller Fehler sind somit auf verschiedene Annotationsansichten und nicht auf Fehlerhaftigkeit der Konzepte zurückzuführen.

2. Fehlverhalten des Stemmers

Der Stemmer erzeugt nur dann eine Grundform eines Wortes, wenn es in einem Lexikon steht und entsprechende Regeln angewendet werden können (vgl. Kapitel III 2.4. „Probleme innerhalb der Konzeptsensorsprache und ihre Grenzen"). Bei 16 Adjektiven hat der Stemmer keine Grundform erzeugen können, und flektierte Formen des geographischen Adjektivs wurden nicht gefunden (vgl. Beispiel 55).

Beispiel 55:

finnische
finnischen

Das Versagen des Stemmers führte in 3,55% zu fehlenden Entitäten, die über die Konzepte erkannt werden müssten.

3. in den Ressourcen fehlend

13 geographische Eigennamen wurden nicht gefunden, weil sie in den Ressourcen fehlen. 6 davon sind geographische Adjektive, die nur über eine Inhaltsanalyse als geographischer Eigenname erkannt werden können (vgl. Beispiel 56). In einem anderen Kontext sind solche Ableitungen von Städtenamen Nominalisierungen.

Beispiel 56:

„Die neue Version mit grafischer Benutzeroberfläche wird von der Aschheimer Articon Informationssysteme GmbH vertrieben."
[Konradin-Verlag]

Nominalisierungen gehören nach meinen Richtlinien nicht zu den Eigennamen (vgl. Beispiel 57) und zählen bei der Fehleranalyse zu Punkt 1 „verschiedene Annotationsrichtlinien".

Beispiel 57:

Die **Dielheimer** haben sich auf computerintegrierte Lösungen für den gesamten Geschäftsprozeß von Fertigteilebauern spezialisiert.
[Konradin-Verlag]

Bei den restlichen sieben geographischen Eigennamen handelt es sich um relativ seltene Schreibweisen von Gebietsnamen (z.B. *Rhein-Main-Gebiet*) sowie geographische Adjektive mit Bindestrich (z.B. *baden-württembergische*).

Durch Hinzufügen der Namen in die Ressourcen können diese Fehler (2,88%) behoben werden. Die Eigennamenerkennung in einem beliebigen Text wird vermutlich nur wenig von den Veränderungen profitieren, weil es sich um seltene Formen handelt, die nicht zwingend im entsprechenden Text vorkommen müssen.

4. Sonstige Fehler

1,77% aller fehlenden Eigennamen sind durch diverse Fehler entstanden. Es handelt sich um aussergewöhnliche Grossschreibung (z.B. *DEUTSCHLAND*), Schreibfehler innerhalb des Testkorpus (z.B. *Frankurter* statt *Frankfurter*), fremdsprachige Schreibweisen (z.B. *American*) oder diskussionsbedürftige Annotation im Testkorpus (z.B. *Frankfurter Flughafen* oder *EU*) oder fehlende Abkürzungen, die ausnahmsweise alleine im Text vorkommen (z.B. *USA* oder *NRW*). Durch Hinzufügen der Abkürzungen in die Ländernamen-Ressourcen können die Konzepte verbessert werden.

Die **Präzision** der Konzepte wurde durch die folgenden Fehler **gedrückt**:

1. Ambiguität

In 39 Fällen wurden Namen gefunden, die sowohl ein geographischer Eigennamen als auch ein grossgeschriebenes Verb oder Adjektiv, flektierte Nomen oder Produktenamen sein können. In diesen Fällen ist es abhängig vom Kontext, ob es sich um eine Entität handelt (vgl. Beispiel 58).

Beispiel 58:

„**Kamen**[133] 1993 in <loc>Deutschland</loc> auf 100 Einwohner nur etwa zwölf PCs, werden es nach Schätzungen des <org>Fachverbands Informationstechnik im VDMA und ZVEI</org> Ende 1996 schon doppelt so viele sein …“ [Konradin-Verlag]

Indexerausgabe: **TYPE=95**, POS=13710, VALUE=**'Kamen'***[134]

8,69% Verlust der Präzision ist auf eine Fehlanalyse, bedingt durch Ambiguität, zurückzuführen. Um die Präzision zu steigern, müssen die ambigen Wörter der Liste *special_single_city_names* hinzugefügt werden. Beim Abgleich vom Englisch-Deutsch-Lexikon mit der Städtenamensliste sind sie nicht entdeckt worden, weil die Wörter im Lexikon in der Grundform eingetragen sind und diese Grundform, weil sie kleingeschrieben ist, keine Ambiguität in sich birgt (vgl. Kapitel III 3.2. „Ressourcen zur Erkennung von geographischen Namen“). Die Erweiterung der Liste *special_single_city_names* kann eine Verschlechterung der Ausbeute mit sich bringen, weil die Eigennamen in einem sehr spezifischen Kontext stehen müssen und dies nicht immer tun.

2. Teil eines Firmennamens

Geographische Eigennamen werden, wenn sie innerhalb eines Firmennamens vorkommen, nicht annotiert. Bei fehlendem Erkennen des Firmennamens wird der geographische Eigenname markiert (vgl. Beispiel 59).

---

[133] Die Ortschaft *Kamen* liegt in Ruhrgebiet von Deutschland in der Nähe von *Dortmund*.
[134] Die mit * und Farbe markierten Ausgaben repräsentier eine Fehlanalyse.

Beispiel 59:

„EDO virtuell: Ein neues Speichermodul für PCs namens Virtual EDO-Memory hat <org>***Kingston*** *Technology*</org> entwickelt.“
[Konradin-Verlag]

Indexerausgabe: **TYPE=95**, POS=45892,64697, VALUE=**'Kingston'***

Weil vor Firmennamen oft geographische Adjektive stehen und diese nicht zum Firmennamen zählen, habe ich daraus geschlossen, dass geographische Entitäten allgemein nicht Teil eines Firmennamens bilden. Dies ist korrekt, solange keine Organisationsnamen bei der Analyse berücksichtigt werden. Oft sind geographische Eigennamen Teil eines Organisationsnamens und werden nicht als geographische Entität markiert. Weil die Konzepte jedoch keine Organisationsnamen erkennen, führt dies zu gleichen Fehlanalysen, wie wenn der Firmenname nicht erkannt wurde. Somit wurden 49 von 451 Namen fälschlicherweise als geographische Eigennamen erkannt. Die Präzision sinkt um 10.86%.

Die Verbesserung der Firmennamenerkennung würde die Erweiterung auf Organisationsnamen zwangsläufig zu einer Verbesserung der geographischen Eigennamenerkennung führen. Dies ist jedoch nur über eine ‚Zwei-Weg-Lösung’ zu erreichen (vgl. Kapitel III 2.4. „Probleme innerhalb der Konzeptsensorsprache und ihre Grenzen“).

3. Überschneidung von Personennamen und geographischen Namen

In 28 Fällen wurde ein geographischer Eigenname falsch erkannt, weil es sich um einen Personennamen handelte (vgl. Beispiel 60).

Beispiel 60:

„Als erstes Beispiel für den Umbruch durch Multimedia nennt <per>**Berger**</per> die steigenden Zahlen von Online-Buchungen bei Luftfahrtgesellschaften,…“
[Konradin-Verlag]

Indexerausgabe: TYPE=95, POS=29223,29242,29272, VALUE='Berger'*
TYPE=96, POS=29223,29242,29272, VALUE=**'Berger'**

Um zu verhindern, dass einstellige geographische Eigennamen nicht in falschem Kontext als solche analysiert werden, müssten zusätzliche Einschränkungsregeln (wie z.B. dass kein Vorname vor dem geographischen Eigennamen stehen darf) im Bedingungsteil der Erkennungskonzepte geschrieben werden. Im Englischen habe ich die Reihenfolge verändert, die den Ablauf der Konzepte zur Erkennung von geographischen Eigennamen und Personennamen festlegt, so dass es zu weniger Fehlanalysen kommt.

Mit 6,21% hält sich der Fehler in Grenzen. Eine Veränderung der Konzepte, wie oben beschrieben, hätte jedoch zur Folge, dass die Ausbeute sinken würde, weil in Spezialfällen ein Vorname, gefolgt von einem geographischen Namen, nicht mehr erkannt würde.

4. Sonstige Fehler

Die Präzision wird um 3,1% durch diverse andere Fehler gesenkt. Es handelt sich um 7 Annotationsfehler (z.B. *Würzburg*), 3 Normierungsprobleme[135] zwischen der Indexerausgabe und dem annotierten Text (z.B. *sueddeutschen* vs. *Süddeutschen*), 3 falsche Analysen, ausgelöst durch den Stemmer (z.B. *Deutschlandkarte*) und einen fehlenden Eintrag in den Ressourcen (*Kova/Nakhon Ratchsima*).

Durch die Evaluation ist klar geworden, dass die geographischen Ressourcen eine zufriedenstellende Grösse erreicht haben. Durch Erweiterung der Liste *special_single_city_names* und durch eine verfeinerte Reihenfolge, die den Ablauf der Konzepte zur Erkennung von Personennamen und geographischen Namen festgelegt, könnte die Präzision zusätzlich leicht gehoben werden.

3.3. Evaluation von Firmennamen (De)

Die Evaluation der Firmennamenerkennung erreicht relativ tiefe Werte. Die Hauptprobleme, welche ich nur ansatzmässig lösen konnte, waren Definition eines Firmennamenteils, Implementierung geeigneter Stoppwörter, Definition von Firmenakronymen, Implementierung des Schritts zum Wiederfinden von bereits gelernten Firmennamen sowie Erstellung eines geeigneten Filters.

Während der Evaluationsphase habe ich verschiedene Veränderungen vorgenommen, um hauptsächlich die Ausbeute zu erhöhen. Meist hatte eine höhere Ausbeute das Sinken der Präzision zur Folge und umgekehrt. Die wichtigsten Veränderungsschritte sowie deren Auswirkungen sind die folgenden:

1. Erweiterung der Ressourcen

   Der Liste von Triggern, die vor einem Firmennamen stehen, wurden zusätzliche Wörter hinzugefügt. Die bestehenden Trigger habe ich teilweise in die Grundform umgewandelt (z.B. *"firma":STEM*), um Firmennamen zu erkennen, die einer flektierten oder zusammengesetzten Form eines Triggers folgen. Allerdings gilt erneut, dass der Stemmer nur einen Teil aller Wörter auf ihre Grundform reduzieren kann. Fremdwörter wie *Konzern* oder seltene Formen wie *Dienstleister* bilden ein Problem.

[135] Normierungsprobleme sind nicht durch die Konzepte verursacht worden, sondern durch das Evaluationsskript. Sie dürfen bei der Berechnung entsprechend nicht als Fehler gerechnet werden, da die Eigennamen von den Konzepten korrekt erkannt wurden.

Neue Stoppwörter sind der Liste *stop_words_for_fa_name_in_front* in Grundform hinzugefügt worden. Eine Reduktion aller vorhandenen Stoppwörter auf die Grundform hat eine Verschlechterung der Präzision und nur eine geringfügige Steigerung der Ausbeute mit sich gebracht, so dass diesbezüglich keine Veränderungen von mir vorgenommen wurden.

2. Entfernen zirkulärer Schleifen

   Gleich wie bei der Evaluation von Personennamen und geographischen Namen hat die Entfernung zirkulärer Schleifen die Funktionstüchtigkeit der Konzepte erhöht (vgl. Punkt 3, S. IV-142). Präzision sowie Ausbeute sind leicht gestiegen.

3. Erweiterung der Erkennung von Firmennamen auf Organisationsnamen

   Weil im annotierten Text Organisationsnamen und nicht nur Firmennamen markiert worden sind, fällt die Ausbeute bei der Evaluation tiefer aus als erwartet. Um die Werte zu steigern, habe ich versuchsweise Trigger für Organisationsnamen implementiert (z.B. *Universität*). Die Erkennung von Organisationsnamen bringt allerdings Probleme wie unkonstante Grossschreibung mit sich, die nicht so leicht gelöst werden können. Weil mein Ansatz die Präzision gleichermassen gesenkt (24.2%) wie die Ausbeute erhöht (12.87%) hat, habe ich keine weiteren Trigger implementiert.

4. Definition von Akronymen

   Um die Ausbeute zu erhöhen, habe ich Akronyme definiert, die als Trigger am Anfang sowie am Ende eines Firmennamens fungieren. Die Ausbeute ist auf 29,89% gesteigert worden. Die Präzision ist dabei allerdings auf 40,53% gefallen. Ein besserer Filter könnte dies verhindern.

5. Hinzufügen eines Filters

   Mit dem Hinzufügen eines Filters sollten die Präzisionswerte bei Beibehaltung der Ausbeute gesteigert werden. Der Filter enthält Einträge aus einem Deutsch-Englisch-Lexikon. Einteilige Firmennamen, die über ein Konzept gelernt, bzw. aus einem gelernten Firmennamen abgeleitet wurden und im Filter stehen, sind keine Firmennamen, weil sie dem deutschen oder englischen Allgemeinwortschatz angehören und es sich somit mit grosser Wahrscheinlichkeit um keine Eigennamen handelt (vgl. Kapitel III 3.3. „Ressourcen zur “). Trotz des einfach gestalteten Filters, wird eine Steigerung von 3,03% erreicht.

6. Integration von Aufzählungen

   Weil Firmennamen im Textkorpus relativ häufig in Aufzählungsform aneinandergelistet stehen, habe ich mir zu Nutze gemacht, dass weitere Firmennamen vor oder hinter

einem bereits bekannten Eigennamen ebenfalls analysiert werden können. Wenn bei einer zweiteiligen Aufzählung ein Firmenname wegen eines Triggers erkannt worden ist, wird der folgende bzw. führende Firmenname ohne Trigger ebenfalls als Entität erkannt.

Die Verwendung des Aufzählungsschemas hat gleichermassen eine Erhöhung der Ausbeute wie auch eine Verschlechterung der Präzision mit sich gebracht, weil nicht nur Firmennamen durch Kommata voneinander getrennt werden (z.B. „*Debitel, Deutschlands …*“, [Konradin-Verlag]). Die letzten Evaluationswerte liegen bei 34,55% Präzision und 33,78% Ausbeute.

Weil in Zukunft eine ‚Zwei-Weg-Lösung' (vgl. Kapitel III 2.4. „Probleme innerhalb der Konzeptsensorsprache und ihre Grenzen“) zur Firmennamenerkennung unumgänglich ist, habe ich keine weitere Zeit verwendet, um die Definition von Akronymen zu verfeinern und den Filter zu erweitern. Die Evaluationsergebnisse der Tabelle 21 sind ohne Wiederfinden bereits gelernter Firmennamen entstanden. Das Wiederfinden hat in der Implementierung schlecht funktioniert, weil sehr wenige Firmennamen im ersten Schritt gelernt werden konnten und davon zusätzlich einige falsch waren. Die tieferen Werte in der Ausbeute sind mit dem geänderten Gebrauch des Filters zu begründen (er wurde nicht nur bei einteiligen Firmennamen eingesetzt) sowie durch den ausschliesslichen Gebrauch der Akronyme als Trigger (allein stehende Akronyme werden nicht erkannt).

Eine genaue Fehleranalyse hat verschiedene problematische Punkte ergeben, die Präzision und die Ausbeute drücken. Zur Analyse wurde diejenige Ausgabedatei verwendet, die die Evaluationsergebnisse der Tabelle 21 geliefert hat. Die kritischen Punkte sind teilweise in den englischen Konzepten von mir verbessert worden. Für die Eigennamenerkennung in deutschen Texten habe ich neue Konzepte geschrieben, die ich allerdings nicht mehr evaluieren konnte.

Von 290 wurden 120 Firmennamen falsch erkannt, was die **Präzisionswerte** entsprechend **gesenkt** hat. Die Hauptfehler sind folgende:

1. Fehlen der Definition des Firmennamenteils

Firmennamenteile können diverse Formen annehmen wie Aufzählungen durch Kommata getrennt (z.B. *Sears**,** Röbuck & Co.*), Zahlenkombinationen (z.B. ***3D** Systems*), Buchstabenkombinationen (z.B. *C & W*), Wortkombinationen (z.B. *Q-Labs **Software Engineering** GmbH*) und Zeichen (z.B. *GFU Cyrus + Rölk mbH*) etc. Gewisse Formen fehlen in meinen Konzepten. Dies hat zur Folge, dass ein Firmenname nur teilweise erkannt wird und somit als Fehler zählt (vgl. Beispiel 61).

Beispiel 61:

„Gemeinsam mit der <loc>Wuppertaler</loc> <org>***B&K* Computersysteme GmbH**</org> hat …“
[Konradin-Verlag]

Indexerausgabe:
TYPE=97, POS=61784, VALUE=**'Computersysteme GmbH'***

In 31 Fällen führte eine ungenaue oder fehlende Definition eines Firmennamenteils zu einer Fehlanalyse. Um eine Verbesserung zu erreichen, können die Konzepte *pos_fa_name_part* teilweise angepasst werden. Eine entsprechende Anpassung bringt allerdings immer das Risiko mit sich, die Präzision zu verringern, weil es kaum spezifische Formen von Firmennamenteilen gibt, die ausschliesslich in Firmennamen auftreten.

2. Erkennungsablauf

Firmennamen können auf Grund eines Triggers vor der Entität oder am Ende der Entität erkannt werden. Wenn allerdings sowohl ein Trigger davor als auch ein Trigger am Ende steht, wird der Firmenname fälschlicherweise zweimal erkannt. Da allerdings der Trigger am Ende als Stoppwort in der Erkennung des Firmennamens über einen Trigger davor fungiert, wird die Entität einmal falsch und einmal korrekt erkannt. Die falschen Analysen verschlechtern die Präzision um 6,9%.

Beispiel 62:

„Der IT-Dienstleister <org>Tele Daten Service</org> (<org>TDS</org>) hat das im Bereich DV für Finanzdienstleister tätige *Unternehmen* <org>**DAT Systemhaus *GmbH***</org> mehrheitlich übernommen.“

Indexerausgabe:
TYPE=97, **POS=49845**, VALUE='DAT Systemhaus'*
TYPE=97, **POS=49846**,49862, VALUE='DAT Systemhaus GmbH'[136]

In den englischen Konzepten sowie in der verbesserten Version der deutschen Konzepte ist der Fehler behoben worden. Ich habe eine Reihenfolge festgelegt, die sicherstellt, dass die Regel, welche den Firmennamen auf Grund eines Triggers am Ende findet, immer vor der andern zur Anwendung kommt. Eine Doppelanalyse ist so ausgeschlossen.

3. Fehler durch Akronymdefinition

Die bestehende Definition von Akronymen hat diverse Fehlanalysen mit sich gebracht. Die Definition, dass ein Akronym aus einer Zahl, gefolgt von einem grossgeschriebenen Wort oder gefolgt von Grossbuchstaben bestehen kann, ist problematisch, weil die Länge der Wör-

[136] Die Positionsangabe bezieht sich immer auf das erste Wort der Analyse. Weil das erste Muster abhängig vom Wort *Unternehmen* erkannt wird, ist die Positionsangabe im Vergleich zur zweiten Analyse um ein Wort verschoben.

ter innerhalb der Konzepte nicht beeinflusst werden kann. Wörter, die zum Markieren absichtlich in Grossbuchstaben geschrieben werden, erzeugen automatisch eine Fehlanalyse (vgl. Beispiel 63).

Beispiel 63:

„KURZ NOTIERT“
[Konradin-Verlag]

Indexerausgabe: TYPE=97, POS=31723, VALUE='KURZ NOTIERT'*

Unter Einbeziehung des Filters, der die Wörter sowohl in Grundform als auch in grossgeschriebenen Buchstaben enthält, verringert sich die Anzahl der Fehler. Allerdings sinkt entsprechend auch die Anzahl der möglichen Akronyme, so dass weniger Firmennamen gefunden werden. Die Anpassung der Akronyme sowie die Einbeziehung des gesamten Filters (in Grundform und grossgeschriebene Buchstaben) sind in den englischen Konzepten realisiert und in einer späteren Phase den deutschen Konzepten hinzugefügt worden. Die Präzision und Ausbeute sind leicht gestiegen (vgl. Kapitel IV 5.3. „Evaluation von Firmennamen“).

4. Falsche Trigger durch Grundformbildung

18 falsche Analysen sind dadurch entstanden, dass die Triggerwörter vor einem Firmennamen in der Grundform stehen (STEM) und somit auch in zusammengesetzten Formen entsprechend fungieren. In gewissen Fällen kann es vorkommen, dass so ein Nomen, das kein Firmenname ist, als Entität kategorisiert wird.

Beispiel 64:

„Die großen ***firmen****eigenen* Datenmengen…“
[Konradin-Verlag]

Indexerausgabe: TYPE=97, POS=59764, VALUE='Datenmengen'*

Um diese Fehler zu beheben, müssten alle flektierten Formen sowie die möglichen Kompositionen einzeln aufgelistet werden. Wie das Beispiel *Firma* bereits zeigt (*Firmen, Beratungsfirma, -Firma* etc.) ist dies sehr aufwendig und verursacht einen grossen Anwachs der Ressourcen.

5. Fehlender Eintrag in den Ressourcen

Lediglich 6 Fehlanalysen sind wegen fehlender Einträge in den Ressourcen entstanden (1 Trigger vor dem Firmennamen und 5 Stoppwörter hinter dem Firmennamen). Dies bedeutet, dass Trigger- und Stoppwortlisten zufriedenstellend gross und genau sind. Allerdings ist zu vermuten, dass eine grössere Ausbeute mehr fehlende Stoppwörter zur Folge hätte.

6. Sonstige Fehler

Die restlichen Fehlanalysen sind hauptsächlich bedingt durch Normalisierungsprobleme[137] innerhalb des Evaluationsskripts (*CAD&Art* statt *CAD & Art*). Bei einer anderen Evaluationsmethode ergäben diese Wörter keinen Fehler, weil sie durch die Konzepte der richtigen Kategorie zugeordnet worden sind.

Wie ich bereits schon bei der Evaluationsphase feststellen konnte, ist die Erkennung von Firmennamen sehr schwierig, weil die Namen nur wenige spezifische Merkmale besitzen, die als Trigger verwendet werden können. Der Kontext, in dem sie erscheinen, ist ebenfalls sehr variabel. Der Versuch, eine höhere Ausbeute zu erreichen, hatte zur Folge, dass die Präzision entsprechend gesunken ist. Die **tiefe Ausbeute** ist hauptsächlich in den folgenden Punkten zu begründen:

1. drücken der Werte durch Annotierung von Organisations- statt Firmennamen

203 aller fehlenden Entitäten sind Organisationsnamen, die entsprechend meiner Aufgabe nicht erkannt werden müssen. Somit ist bei der Evaluation im Prinzip eine 19,36% höhere Ausbeute erreicht worden.

2. Übergenerierender Filter

Wenn die Wörter des Lexikons, welches ich für den Filter verwendet habe, alle in Grossbuchstaben geschrieben werden, werden ebenfalls korrekte Akronyme ausfiltriert (z.B. *ARM*[138]). 183 Firmennamen sind nicht erkannt worden, weil die Akronyme ausfiltriert oder von den Konzepten nicht erkannt worden sind. Um höhere Präzision bei der Firmennamenerkennung zu erhalten, ist ein besserer Filter, so wie ihn Volk (vgl. [Volk und Clematide, 2001]) verwendet hat, unumgänglich. Wenn die Akronyme von tatsächlich bestehenden Firmennamen abgeleitet und nicht selbst generiert würden, wäre die Zuverlässigkeit höher, und der Filter erhält weniger Gewicht. Eine Ableitung wäre jedoch nur in einer ‚Zwei-Weg-Lösung' umsetzbar.

3. Fehlende Trigger

Viele Firmennamen stehen ohne jeglichen Trigger in Texten. Es ist deshalb sehr schwierig sie zu erkennen. Etwa 600 aller nicht erkannten Wörter gehören zu dieser Kategorie. Mit einem entsprechend guten Algorithmus kann ein Teil der Namen durch Ableitungen von über einen Trigger bereits gelernten Firmennamen erkannt werden. Die Ableitungen bzw. Akronyme sollten über Dokumentgrenzen hinweg angewendet werden. Die Ableitung sowie die Suche weiterer Namen konnten nur ansatzweise in die Konzepte integriert werden (vgl. Modul *learn_fa_name*). Sie haben kaum Auswirkungen gezeigt (und wenn eher negative), da

[137] Vgl. Fussnote 135, S. 155.
[138] *ARM* ist das Akronym der britischen Firma *Advanced Risc Machines.*

einerseits nur wenige Firmennamen überhaupt gelernt wurden, von denen abgeleitet werden konnte, und andererseits ist davon ein Teil zusätzlich falsch.

Um eine Firmennamenerkennung mit hoher Ausbeute bei hoher Präzision zu erreichen, sind Regeln allein ungenügend, weil Firmennamen stark in Form und Kontext variieren. Genauere Regeln könnten unter Berücksichtigung von Wortarten erstellt werden. Wörter und Grossschreibung allein sind zu wenig aussagekräftig. Eine Liste häufiger Firmenakronyme und Namen in Kombination mit den Regeln würde eine bessere Firmennamenerkennung ermöglichen. Geläufige Firmennamen werden in Texten häufiger abgekürzt und treten deshalb ohne Trigger auf, während seltene oder neue Firmennamen in voller Länge ausgeschrieben werden. Sie können über die Regeln erkannt werden, während geläufige Namen in einer Liste aufgeführt werden. Dies wäre eine Lösung, um Firmennamen relativ unabhängig von Text und Zeit zu erkennen.

4. Testkorpus für die Evaluation in englischen Texten

Bei der Textkollektion, die ich zum Testen der Konzepte zur Eigennamenerkennung in englischen Texten verwendet habe, handelt es sich um Nachrichten der *New York Times* und der *Associated Press* sowie *Broadcast News* aus den Monaten März bis April 1998. Bei den Artikeln aus der *New York Times* und *Associated Press* handelt es sich um gedruckte Nachrichten, die bezüglich Satzzeichensetzung und Grossschreibung dem Schreibstandard entsprechen. Die *Broadcast News* sind gesprochene Nachrichten, die transkribiert wurden. Transkribierung bedeutet in diesem Fall, dass jedes gesprochene Wort, auch Fragmente, in Buchstaben (nicht in phonetischer Schrift) auf Papier festgehalten wurden. Korrekte Interpunktion und Grossschreibung sind jedoch nicht garantiert. In den Texten können folglich Wortwiederholungen, Wörter aus dem umgangssprachlichen Wortschatz sowie unterschiedliche Interpunktion und Grossschreibung vorkommen. Dieses heterogene Korpus ist für das Projekt ***I**nformation **E**xtraction - **E**ntity **R**ecognition Evaluation* (IE-ER)[139] im Jahre 1999 zusammengestellt worden. Das Projekt hatte das Ziel, den durchschnittlichen Stand der Forschung im Bereich der Informationsextraktion aus möglichst verschiedenen Textquellen voranzutreiben. Neben Informationsextraktionssystemen für englische Texte wurden in diesem Projekt auch Systeme für Texte in Mandarin miteinander verglichen. Die annotierte Textkollektion, die zum Training und zur Entwicklung der Informationsextraktionssysteme für Englisch verwendet und von DARPA-TIDES finanziert wurde, kann unter [IE-ER-Evaluation, http://www.nist.gov/speech/tests/ie-er/er_99/er_99.htm] unentgeltlich heruntergeladen werden. Sie hat mir als englisches Testkorpus in diesem Projekt gedient.

[139] Weitere Informationen unter [IE-ER-Evaluation, http://www.nist.gov/speech/tests/ie-er/er_99/er_99.htm].

Die Annotationsrichtlinien, welche für das Testkorpus verwendet worden sind, entsprechen den Richtlinien von HUB-98[140]. Der ursprünglich annotierte Text enthält neben Markierungen von Entitäten auch Markierungen weiterer Informationen wie Sprecher, Dokumententyp, etc. (vgl. Abbildung 46).

"<IEER_DOC type="NEWSWIRE" fileid="" collect_date="" collect_src="
" src_lang="" content_lang="" proc_remarks="IEER document translation">
<DOC>
<DOCNO> APW19980314.0391 </DOCNO>
<DOCTYPE> NEWS STORY </DOCTYPE>
<DATE_TIME> 03/14/1998 10:36:00 </DATE_TIME>
<BODY>
<HEADLINE>
Kenyans protest tax hikes
</HEADLINE>
<TEXT>
<b_enamex type="LOCATION">NAIROBI<e_enamex>,
<b_enamex type="LOCATION">Kenya<e_enamex>
(<b_enamex type="ORGANIZATION">AP<e_enamex>) _
<b_numex type="CARDINAL">Thousands<e_numex> of laborers, students and opposition politicians on <b_timex type="DATE">Saturday<e_timex> protested tax hikes imposed by their cash-strapped government, which they accused of failing to provide basic services."
[IE-ER-Evaluation, 1998]

Abbildung 46

Martin Braschler von der Firma *Eurospider* hat die Texte von allen für dieses Projekt nicht benötigten Annotierungen gereinigt (vgl. Abbildung 47). Die folgenden Annotationen sind geblieben:

- **<person>… </person>** **Personenname**
- **<location>…</location>** **geographischer Eigenname**
- **<organization>…</organization>** **Organisationsname**
- <cardinal>…</cardinal> Zahl
- <date>…</date> Datum
- <time> </time> Zeitangabe
- <percent>…</percent> Prozentangabe
- <money>…</money> Geldbetrag
- <duration>…</duration> Dauer

Davon sind für die Evaluation lediglich Personennamen, geographische Entitäten und Organisationen relevant. Die restlichen Markierungen werden bei der Evaluation vom Perlskript nicht weiter beachtet.

[140] Vgl. [IE-ER-Evaluation, http://www.nist.gov/speech/tests/ie-er/er_99/er_99.htm].

<location>NAIROBI</location>, <location>Kenya</location> (<organization>AP</organization>) <cardinal>Thousands</cardinal> of laborers, students and opposition politicians on <date>Saturday</date> protested tax hikes imposed by their cash-strapped government, which they accused of failing to provide basic services. [IE-ER-Evaluation, 1998]

Abbildung 47

Die gesamte Testkollektion besteht aus ca. 80 Texten (351576 Wörter). Davon sind 6770 Wörter als Personenname, 5307 als geographische Entität und 5181 als Organisation annotiert. Die Kategorie *geographische Eigennamen* enthält neben Ortschafts-, Regions- und Inselnamen auch Gewässer- (z.B. *Black Sea*), Berg- (z.B. *Rocky Mountains*), Quartier- (z.B. *French Quarter*) und Strassennamen (z.B. *209 West Houston St.*) sowie sonstige geographische Ausdrücke (z.B. *milky way, Arctic Circle* oder *earth*). Geographische Entitäten umfassen somit mehr Unterkategorien, als in meinen Konzepten implementiert sind. Diese Unterkategorien könnten als weitere Ressourcen hinzugefügt werden. Dies habe ich jedoch nicht gemacht. Die Kategorie *Organisationsname* umfasst entsprechend der Annotation im deutschen Testkorpus neben Firmennamen auch Parteinamen (*Communist Party*), Institutsnamen (z.B. *Brookings Institution*), Universitätsnamen (z.B. *Boston University School of Medicine*), Namen von Behörden (z.B. *Housing and Local Government*), Restaurationsbetriebsnamen (z.B. *Maison de Ville hotel*) und Namen sonstiger Vereinigungen (z.B. *British Museum* oder *Freedom Forum*).

Aus dem Testkorpus sind für die Evaluation zwei Versionen erstellt worden. Die eine Version enthält Eigennamenmarkierungen und dient im Perlskript als Vergleichsdokument zur Ausgabedatei des Indexers. Die Version ohne Markierungen dient als Eingabedatei für den Indexer. Über die Konzeptsensoren werden die Eigennamen innerhalb des Dokuments erkannt. Um realistischere Ausbeute- und Präzisionswerte zu erhalten, habe ich das vorhandene Testkorpus um Texte ohne Grossschreibung gekürzt und Ortsangaben, die am Anfang der Nachrichten mit Grossbuchstaben hervorgehoben wurden (z.B. *NAIROBI*), in die übliche Grossschreibung umgewandelt (z.B. *Nairobi*).

5. Evaluation der Eigennamenerkennung in englischen Texten

Die Abänderungen der englischen Konzepte während der Evaluationsphase bestanden hauptsächlich in Anpassungen der Bedingungsteile, um doppelte Analysen zu vermeiden und Reihenfolgen im Erkennungsablauf der Konzepte zu garantieren. Weil ich versucht habe, die englischen Konzepte einfacher als die deutschen zu gestalten, wurden diese Abänderungen notwendig. Während der Evaluation mussten diverse Ressourcen (Stoppwort- und Triggerliste für Firmennamenerkennung, Stoppwortliste für allein stehende Vornamen etc.) angepasst

werden. In der Liste *special_single_city_names* fehlten einige Einträge, weil die grösseren Listen im Bereich der geographischen Eigennamen mehr Ambiguität mit sich gebracht haben, die beim Abgleich von Lexikon und Städtenamen nicht immer erkannt worden ist. Die Liste musste während der Evaluationsphase manuell erweitert werden. Eine Ergänzung der Stoppwort- und Triggerliste während der Evaluation war notwendig.

Die Endwerte, welche ich bei der Evaluation der englischen Konzepte erhalten habe, sind in Tabelle 22 aufgelistet. Im Vergleich zu den deutschen Konzepten schneiden die Werte leicht besser ab. Dies war zu erwarten, weil ich die Erfahrungen von den deutschen Konzepten nutzen konnte und die Grossschreibung der Entitäten in englischen Texten ein markanteres Merkmal ist und weniger Überschneidungen mit sich bringt (englische Nomen werden nicht grossgeschrieben).

	ENGLISCH	
	Ausbeute (in Prozent)	*Präzision (in Prozent)*
alle	**61.75**	**83.15**
Personen	**80.31**	**88.95**
geographische Namen	**78.22**	**80.44**
Firmennamen	**18.95**	**60.84**

Tabelle 22

Die Gesamtwerte werden nach wie vor hauptsächlich durch die schwache Firmennamenerkennung gedrückt. Die Problematik bleibt bei den englischen Konzepten dieselbe wie bei den deutschen. Eine genaue Fehleranalyse folgt in den folgenden Unterkapiteln.

5.1. Evaluation von Personennamen (En)

Während der Evaluationsphase hat die Personennamenerkennung hauptsächlich durch die Präzisierung der geographischen Eigennamenerkennung sowie der besseren Firmennamenerkennung profitiert. Die Problematik, welche in Kapitel IV 3.2. „Evaluation von geographischen Eigennamen“ unter Punkt 3, S. IV-154, besprochen worden ist, wurde bei der Evaluationsphase mit berücksichtigt. Um zwischen Personennamen und geographischen Namen zu unterscheiden, wurden spezifischere Schemata erstellt. Wenn beispielsweise ein allein stehender geographischer Eigenname vorkommt, der gleich lautet wie ein Personenname, dürfen in der unmittelbaren Umgebung keine Trigger für Personennamen stehen, da sonst das Wort der Kategorie *Personenname* zugeordnet werden muss (z.B. *Dieter Rombach* als Personenname vs. *Rombach* als geographischer Eigenname). Eine höhere Ausbeute der Firmennamen

hatte eine Verbesserung der Personennamenerkennung zur Folge, da Personennamen, die in Firmennamen vorkommen, nicht der Kategorie *Personennamen* angehören (vgl. Punkt 3, S.IV-154). An den Konzepten habe ich lediglich die Ressourcen verändert. Das Hinzufügen weiterer Vornamen in die Liste sowie die Erweiterung der Stoppwortliste von allein stehenden Vornamen hat eine Verbesserung von Präzision und Ausbeute mit sich gebracht. Das Hinzufügen eines einzelnen Vornamens hatte zur Folge, dass weitere drei Nachnamen durch das erneute Suchen des gelernten Nachnamens (*learn_pers_name*) gefunden wurden.

Insgesamt ist während der Evaluationsphase die Präzision durch die Veränderungen der Konzepte von 67,18% auf 88,95% gestiegen, die Ausbeute von 71,99% auf 88,31%. Eine genaue Analyse der letzten Evaluationsergebnisse hat gezeigt, dass 8 Hauptfehler bestehen, die die Eigennamenerkennung verschlechtern. Die Konzepte sind teilweise entsprechend verbessert worden, konnten aber nicht mehr an Texten getestet werden, da die Projektzeit bei *Eurospider* abgelaufen war. Die **Präzision** der Personennamenerkennung ist durch folgende problematische Punkte **gesenkt** worden:

1. Kategorisierungsprobleme

Wenn Personennamen innerhalb von Titeln, Produkte-, Organisationsnamen und geographischen Eigennamen stehen, gehören sie gemäss der vorhandenen Annotation nicht mehr der Kategorie Personennamen an. Sie haben ihre appellative Funktion verloren. Weil aber weder Titel noch Produktenamen von Konzepten erkannt werden können, kommt es zu Fehlanalysen (vgl. Beispiel 65).

Beispiel 65:

"<person>DiCaprio</person> earned an Academy Award nomination for best supporting actor at <measure>19</measure> for his portrayal of a mentally impaired youngster in ``What's Eating **Gilbert Grape**."
[IE-ER-Evaluation, 1998]

Incorrectly found: person **Gilbert Grape**

Unerkannte Organisationsnamen und geographische Eigennamen bringen dieselbe Problematik mit sich. In 14 Fällen handelte es sich um Personennamen in Titeln, in 15 Fällen um Personennamen in Organisationen und in 7 Fällen um Personennamen in geographischen Entitäten.

Die 35 falsch gefundenen Namen können nur durch eine entsprechende Verbesserung der vorhandenen Konzepte bzw. eine Integration neuer Konzepte eliminiert werden. Diese Fehler würde ich jedoch nicht als gravierend bezeichnen, da sie kein Fehlverhalten der vorhandenen Konzepte zum Ausdruck bringen, sondern lediglich die Problematik der klaren Abgrenzung von Eigennamen sowie deren Definition thematisieren.

2. Fehlanalysen und Folgefehler

Es hat sich herausgestellt, dass Erkennungsregeln in gewissen Fällen zu Fehlanalysen führen können. Das fehlende Erkennen hat zur Folge, dass einzelne Buchstaben am Ende eines Satzes als Initial analysiert werden und entsprechend als Trigger für einen Personennamen fungieren. Eine Fehlanalyse entsteht (vgl. Beispiel 66).

Beispiel 66:

"… he said adding that the water rationing exercise was being implemented as planned in all affected areas in Zone A **and B.**
Meanwhile employees at <organization>Sun Metal Works Sdn Bhd</organization> when interviewed said that the factory had been deprived of water supply for the past <duration>two days</duration>."
[IE-ER-Evaluation, 1998]

Ausgabedatei aus dem Evaluationsskript:
Incorrectly found: person B. Meanwhile*

Durch das erneute Suchen der Nachnamen kann die Fehlanalyse weitere Fehler auslösen. Die Folgefehler führen zu einer weiteren Verringerung der Präzision.

Beispiel 67:

"**Meanwhile**, in another big sale, <organization>Freddie Mac</organization> priced …"
[IE-ER-Evaluation, 1998]

Folgefehler können jedoch ebenfalls bei korrekt gelernten Nachnamen (z.B. *Mary Jo* ***White***) entstehen, weil der Name in einem anderen Kontext (z.B. ***White*** *House*) kein Personenname mehr ist.

Insgesamt handelt es sich um 5 durch die Erkennungsregeln ausgelöste Fehlanalysen und 10 Folgefehler, die sowohl durch die falsch erkannten Nachnamen als auch durch ambige Nachnamen ausgelöst wurden.

Wenn der Indexer Satzgrenzen erkennen könnte, was momentan nicht möglich ist, würde das Problem der Fehlanalysen durch Erkennungsregeln sowie deren Folgefehler gelöst sein. Um Folgefehler zu vermeiden, die durch ambige Nachnamen ausgelöst werden, müsste das Wiederfinden lediglich auf die Dokumentengrenze beschränkt werden (vgl. Abbildung 21, S. III-80). Die Dokumentgrösse war im Testkorpus jedoch variabel. Eine vollkommene Auflösung der Ambiguität wird jedoch nicht möglich sein, da dies eine Eigennamenidentifikation (vgl. Kapitel III 4.1. „Erkennung von Personennamen") voraussetzt.

3. Ressourcen

6 Namen sind falsch erkannt worden, weil die Vornamen ambig sind (z.B. *Sen* oder *Verginia*) und allein stehend seltener der Kategorie *Personenname* angehören als einer andern. Die

Namen müssen der Stoppwortliste für allein stehende Vornamen angefügt werden. In lediglich zwei Fällen war der Vorname nicht Teil der Ressource und der gesamte Personenname ist nur halb erkannt worden. Teilanalysen zählen ebenfalls zu falsch erkannten Personennamen und verschlechtern somit die Evaluationswerte. Fehler in den Ressourcen können durch Ergänzung behoben werden. Wie die Evaluationswerte gezeigt haben, sind zufriedenstellend grosse Ressourcen zusammengetragen worden, da fehlende Einträge in den Listen keine Hauptquelle für Fehlanalysen bilden.

4. Normierungsprobleme und sonstige Fehler

Die Normierungsprobleme sind dieselben, wie sie in Beispiel 48 auf S. IV-145 erläutert werden. Um diese 7 Fehler zu vermeiden, müssten die Vergleichstexte ebenfalls durch den Indexer analysiert oder das Perlskript entsprechend angepasst werden. Bei Fehlerangaben, verursacht durch die Normierung, handelt es sich um keine fehlerhaften Konzepte. Dies wird jedoch bei der Evaluation nicht berücksichtigt.

Andere Fehler sind inkorrekte Zuordnung von Wörtern, die durch einen sehr seltenen, wenn nicht einmaligen Fehler verursacht wurde. Dazu zählen Dinge wie „unglückliche Reihenfolge“ von Firmen- und Personenname (z.B. *Chicago* ***Bear Walter Payton***) oder zu lange Personennamen (z.B. *Selangor Menteri Besar Datuk Abu Hassan Omar*). Eine Korrektur dieser Fehler wäre sinnlos, da sie zu spezifisch in den Texten des Testkorpus vorkommen.

Insgesamt ist die Präzision durch Anpassung der Konzepte zur Erkennung von Personennamen nur noch leicht zu verbessern. Die Präzision kann hauptsächlich durch die umfangreichere Erkennung anderer Kategorien von Entitäten verbessert werden.

Der **Verlust** von 19,96% bei der **Ausbeute** wird durch 152 fehlende Personennamen verursacht. Der Verlust ist durch folgende Punkte entstanden:

1. Fehlende Vornamen in Ressourcen

Trotz der grossen Vornamensliste fehlen 63 Einträge. Wenn kein Trigger, wie z.B. ein Vorname, vorhanden ist, können die Entitäten nicht erkannt werden (vgl. Punkt 5, S. IV-148). Bei den fehlenden Vornamen handelt es sich hauptsächlich um Kurzformen (z.B. *Pam*), ungewöhnliche Schreibweisen (z.B. *Ahron*) und asiatische Namen (z.B. *Badja*). Das Hinzufügen der Kurzformen und asiatischen Namen würde eine höhere Ausbeute mit sich bringen. Das Ergänzen der Liste um seltene Schreibweisen wäre vermutlich eine zu spezifische Anpassung und würde bei einem fremden Text (nicht beim Testkorpus) keine höhere Ausbeute zur Folge haben.

2. Nachnamen ohne Trigger und Spitznamen

Wenn Personen allgemein bekannt sind (z.B. *Wagner*) oder momentan in den Medien oft beredet werden (z.B. *Bush*), verwenden die Journalisten lediglich den Nachnamen oder Spitznamen. In diesen Fällen ist die Erkennung über die Konzepte unmöglich, ausser die Namen würden in einer Liste aufgeführt. Dies ist in den bestehenden Konzepten jedoch nicht der Fall, so dass 37 Namen wegen Fehlen eines Triggers von den bestehenden Konzepten nicht erkannt worden sind.

3. Folgefehler

8 Nachnamen sind nicht gefunden worden, weil der Vorname nicht in der Liste stand, der Nachname somit nicht gelernt werden konnte und im Wiederfindungsprozess nicht zur Verfügung steht (vgl. Punkt 7, S. IV-148).

4. Sonstige Fehler

Das Fehlen von Personennamen ist in 31 Fällen auf verschiedene kleinere Fehler zurückzuführen. Dazu zählen Normierungsprobleme[141] (*McCourts* vs. *Mc Courts*), Überschneidung von Nachname und Anrede (z.B. *Mark Bishop*), Überschneidung von geographischem Namen und Personennamen (z.B. *St. Louis*) und Länge der Namen[142] (z.B. *Selangor Menteri Besar Datuk Abu Hassan Omar*).

Der Versuch, die Fehler zu beheben, hätte vermutlich nur einen geringen Einfluss auf die Ausbeute, da es sich um sehr spezifische Fehler handelt. Die Normalisierungsprobleme treten bei einer anderen Evaluationsmethode nicht auf. Die Personennamen werden von den Konzepten korrekt erkannt und sollten deshalb nicht zu den Fehlern gerechnet werden.

Die Ausbeute der bestehenden Konzepte ist hauptsächlich durch Erweiterung der Ressourcen und Hinzufügen neuer Ressourcen (z.B. bekannte Personennamen und Spitznamen) leicht zu erhöhen. Die verwendeten Regeln zum Erkennen von Personennamen sind korrekt und erfassen die meisten bestehenden Muster. Eine Veränderung hätte keine Auswirkung auf die Ausbeute.

5.2. Evaluation von geographischen Namen (En)

Die Evaluationswerte der geographischen Eigennamen liegen relativ hoch. Da die Erkennung lediglich auf Ressourcen basiert, bedeutet dies, dass die erstellten Listen zufriedenstellend viele Einträge für die entsprechenden Sprachgebiete enthalten. Während der Evaluationsphase wurden folgende Fehler in den Konzepten behoben:

---

[141] Vgl. Fussnote 135, S. 155.

[142] Nach meiner Definition können Personennamen aus höchstens vier Vornamen, gefolgt von einem einfachen oder doppelten Nachnamen, bestehen.

1. Erweiterung der Liste *special_single_city_names* hat zur Erhöhung der Präzision beigetragen.
2. Die Bildung einer Reihenfolge, die den Ablauf von Konzepten zur Personennamenerkennung und zur Erkennung von geographischen Namen in eine Abhängigkeit zueinander setzt, hat eine Erhöhung der Präzision zur Folge. Mehrteilige Namen, die sowohl Personenname als auch geographische Entität sein können, werden den geographischen Eigennamen zugeordnet, falls keine Anrede davor steht. Einteilige Namen, die sowohl Personenname als auch geographische Entität sein können, werden den geographischen Eigennamen zugeordnet, falls kein Personentrigger in direkter Umgebung steht.
3. Die Erweiterung der Ressourcen und Berücksichtigung von Abkürzungen hat zur Erhöhung der Ausbeute beigetragen.
4. Die verbesserte Firmennamenerkennung hat zu einer höheren Präzision der geographischen Eigennamen geführt.

Die oben aufgezählten Veränderungen der Konzepte haben zu einem Anstieg der Präzision von 57,86% auf 80.44% und zu einem Anstieg der Ausbeute von 54,47% auf 78,22% geführt. Eine genaue Analyse der letzten Evaluationsergebnisse folgt im nächsten Abschnitt und gibt Auskunft, wie weit die Konzepte in Präzision und Ausbeute noch verbessert werden könnten. Geographische Eigennamen wurden hauptsächlich nicht erkannt, weil die bestehenden Ressourcen nur einen Teil aller möglichen Kategorien von geographischen Eigennamen enthalten. Im annotierten Text allerdings bestehen weitere Unterkategorien, die annotiert wurden (vgl. Kapitel IV 4. „Testkorpus für die Evaluation in englischen Texten“). Weil diese Unterkategorien in den Ressourcen fehlen, werden die entsprechenden Eigennamen nicht erkannt und die Ausbeute sinkt. 39 Eigennamen fehlen bei der Evaluation, weil sie als Bezeichnungen von Seen, Bergen, Bezirken, Quartieren etc. nicht in die Ressourcenliste aufgenommen worden sind. Wenn man diese fehlenden 39 Namen, welche von den Konzepten bewusst nicht erkannt werden sollen, bei der Berechnung berücksichtigt, steigt die Ausbeute um 10,63%. Die restlichen fehlenden Namen sind nicht in den Ressourcen aufgeführt und konnten deshalb nicht erkannt werden. Eine Erweiterung der Ressourcen ist leicht möglich, bringt jedoch das Risiko mit sich, dass wegen Ambiguität auch Fehlanalysen entstehen. Die Ausbeute würde dann zu Gunsten der Präzision steigen. Bei den fehlenden 31 Namen handelt es sich hauptsächlich um Ortsnamen aus dem asiatischen Bereich. Die Erweiterung der Ressourcen um diese asiatischen Namen würde wenig Ambiguität mit sich bringen, weil die Namen entsprechend den Regeln einer anderen Sprache wie Deutsch oder Englisch gebildet

worden sind. Die Überschneidung des Allgemeinwortschatzes des Englischen mit den geographischen Eigennamen aus dem asiatischen Raum wird vermutlich gering sein. Die verschiedenen Schreibweisen (z.B. *San'a* vs. *Sana*) der asiatischen Namen erschweren das Sammeln. Eine Ausweitung der Ressourcen auf den asiatischen Raum würde viel Zeitaufwand mit sich bringen. Die bestehenden geographischen Ressourcen sind, wie die Evaluation gezeigt hat, für den englischen und deutschen Sprachraum zur Erkennung der Eigennamen zufriedenstellend gross.

Falsch gefundene Eigennamen haben die **Präzision verschlechtert**. Es bestehen zwei Hauptgründe für Fehlanalysen:

1. Teil einer anderen Entität

Gleich wie bei der Evaluation der deutschen Konzepte werden geographische Namen, die Teil einer nicht erkannten Entität sind, fälschlicherweise den geographischen Eigennamen zugeordnet (vgl. Punkt 2 und 3, S. IV-154). In 15 Fällen war der Name Teil eines Organisationsnamens (z.B. *Berkeley Center for Law and Technology*), in 18 Fällen handelte es sich um einen Personennamen (z.B. *Douglas*) und in 5 Analysen bildete der Name Teil eines geographischen Eigennamens (z.B. *Riverview Street*), der aber keiner von mir erkannten Unterkategorie angehört.

Die Fehler können durch eine bessere Erkennung der Firmennamen und eine genauere Erkennung der Personennamen verbessert werden. Firmennamen müssten auf Organisationsnamen ausgedehnt werden.

2. Ambiguität

Die Ambiguität ist, verglichen mit den deutschen Konzepten (vgl. Punkt 1, S. IV-153), zwar geringer, jedoch immer noch vorhanden.17 Namen wurden fälschlicherweise gefunden, weil es sich um gleichlautende Nomen (z.B. *Commerce*), Adjektive oder Verben (z.B. *Reading*) am Satzanfang handelte. Um die Fehler zu verhindern, müssen die Namen in der Liste *special_single_city_names* aufgenommen werden. Die Namen werden dort nur in einem gewissen Kontext als geographische Eigennamen erkannt (vgl. Kapitel III 4.2. „Erkennung von geographischen Namen"). Der definierte Kontext (vgl. Abbildung 48) hat in 3 Fällen nicht genügt und eine Fehlanalyse (z.B. „Another Day *in* ***Paradise***") ist ausgelöst worden.

```
CONCEPT "special_single_city_names"
IF
        ($"March":CAPWORD)
        OR ($"Hall":CAPWORD)
…
CONDITION
        find( "capitalized_word":CONCEPT);
        (find("in":WORD, -1)
        OR find("of":WORD, -1)
        OR find("from":WORD, -1)
        OR (find("section":WORD, -3) AND find("on":WORD, -1))
        OR find(",":SIGN, -1) AND (find(",":SIGN, +1)));
        create("$0":GEONAME)
END
```

Abbildung 48

Insgesamt entstanden 20 Fehler, weil die Namen in der Liste *special_single_city_names* fehlten oder die Einschränkungsregeln fehlschlugen. Dass jedoch nur wenige Fehler durch ungenügende Kontextregeln entstanden sind, zeigt, dass es sich um sehr spezifische und seltene Fälle handelt, die vernachlässigt werden können. Durch Hinzufügen weiterer Namen zur Liste *special_single_city_names* wird zwar die Präzision gesteigert, die Ausbeute könnte jedoch fallen, weil Eigennamen, die in einem anderen Kontext als dem definierten stehen, nicht mehr gefunden werden. Im Fall meiner Konzepte ist dies jedoch nie geschehen.

Insgesamt können die Konzepte zur Erkennung von geographischen Eigennamen hauptsächlich durch Erweiterung der Ressourcen verbessert werden. Bei den Erweiterungen sollte jedoch Vorsicht walten, da Überschneidungen mit dem Allgemeinwortschatz oder mit anderen Entitäten durch Sonderregeln berücksichtigt werden müssen. Weil eine Reihenfolge beim Erkennen der verschiedenen Eigennamen besteht, ist die Güte jeder einzelnen Kategorie (*Personenname*, *geographischer Name* und *Firmenname*) von der Güte der vorherigen abhängig. Je höher Präzision und Ausbeute bei einer bestimmten Kategorie sind, umso stärker steigen die Werte in den abhängigen Kategorien. Die Konzepte sind so konzipiert, dass die oberste Kategorie die Firmennamenerkennung ist, die leider eher schlecht bei der Evaluation abschneidet. Eine Verbesserung hätte gewisse Auswirkungen auf die geographische Eigennamenerkennung.

5.3. Evaluation von Firmennamen (En)

Die Evaluationswerte der Firmennamenerkennung sind gleich wie in den deutschen Texten sehr tief, da sehr wenige Firmennamen durch einen Trigger markiert sind, der Schritt des Wiederfindens von Teilen bereits gelernter Firmennamen nur ansatzweise implementiert werden konnte, der Filter zur Elimination von falsch gelernten oder abgeleiteten, einteiligen

Firmennamen (vgl. Kapitel III 5.3. „Ressourcen zur Erkennung von Firmennamen (En)“) sich als ungenügend erwiesen hat und neben Firmennamen auch Organisationsnamen im Testkorpus annotiert wurden. In der Evaluationsphase habe ich hauptsächlich mit dem Filter experimentiert, um Präzision und Ausbeute zu erhöhen. Indem der Filter nur noch bei Akronymen und einstelligen Firmennamen zur Reduktion falsch gelernter Entitäten eingesetzt worden ist und neben grossgeschriebenen Formen auch Grundformen enthält (*filter* und *filterII*), steigt die Ausbeute von 6,40% auf 18,95% und die Präzision von 27,43% auf 69,84%.

Eine genaue Analyse der falsch gelernten Namen (**Senkung der Präzision**) aus der letzten Evaluation hat folgende problematische Punkte ergeben:

1. Akronyme und Filter

31 von 61 Firmennamen wurden falsch gelernt, weil falsche Akronyme generiert worden sind und im Filter kein entsprechender Eintrag zur Elimination stand. Zu den Fehlern zählen Zeilen in Grossbuchstaben (vgl. Beispiel 68), die als Akronyme interpretiert werden, Kombinationen zwischen Zahlen und Buchstaben (z.B. *30K*), einzelne Wörter, die nicht im Filter stehen (z.B. *DIETER*), oder Namen, die durch eine römische Zahl, ein Akronym, gelernt werden (z.B. *Louis XIV*).

Beispiel 68:

“COPYRIGHT <date>1998</date> BY <organization>WORLDSOURCES, INC.</organization>, A JOINT VENTURE OF <organization>FDCH, INC.</organization> AND <organization>WORLD TIMES, INC.</organization> **NO PORTION OF THE MATERIALS CONTAINED HEREIN MAY BE USED IN ANY MEDIA WITHOUT ATTRIBUTION TO** <organization>WORLDSOURCES, INC.</organization>”
[IE-ER-Evaluation, 1998]

Um die Fehler zu verhindern, müssten Zeilen, die in Grossbuchstaben stehen, in herkömmliche Grossschreibung umgewandelt werden. Zusätzlich müsste die Definition der Akronyme verändert werden. Dies bringt allerdings eine Verschlechterung der Ausbeute mit sich, weil durch eine engere Definition (z.B. ohne Zahlen) Firmennamen verloren gehen. Die Erweiterung des Filters würde eine Verringerung der Ausbeute zur Folge haben. Die beste Lösung wäre, die Akronyme nicht zu definieren, sondern von gefundenen Firmennamen abzuleiten, so wie es in der ‚Zwei-Weg-Lösung’ vorgesehen ist.

2. Sonstige Fehler

Die restlichen 30 Namen wurden auf Grund verschiedener Fehlverhalten falsch erkannt. In 15 Analysen ist nur ein Teil des Firmennamens gefunden worden (z.B. *Sach & Co* statt *Sachs & Co.*, *Sony* statt *Sony Corp.* oder *TBWA Chiat* statt *TBWA Chiat/Day*). 3 Normalisierungsfeh-

ler[143] (*A & E* statt *A&E*) haben falsche Firmennamen erzeugt. 9 Fehler sind entstanden, weil es sich bei den erkannten Eigennamen um Organisationsnamen, nicht aber um Firmennamen handelt. Auf die restlichen 3 Fälle gehe ich nicht weiter ein, weil es sich um sehr spezifische Fehler handelt.

Teilanalysen sind keine gravierenden Fehler, da der Firmenname oft in vollständiger Extension (z.B. *Sony Corp.*) ein zweites Mal gefunden wurde. Es handelt sich in diesem Fall um Doppelanalysen, die auf Grund der Definitionen der Firmennamenteile, die sich überschneiden, entstanden sind. Sie können behoben werden. Gewisse Teilanalysen sind durch fehlende Definition des Firmennamenteils entstanden (z.B. *TBWA Chiat* statt *TBWA Chiat/Day).* Durch Hinzufügen weiterer Definitionen von Firmennamenteilen können auch diese Fehlanalysen vermieden werden. Die Erweiterung der Definition muss allerdings vorsichtig gehandhabt werden, da durch eine breitere Definition auch mehr falsche Firmennamen gelernt werden.

Die Zahl der fehlenden Firmennamen ist bei der Evaluation sehr hoch, weil im Text Organisationsnamen und nicht nur Firmennamen annotiert wurden. Von 355 nicht gefundenen Namen sind 214 Organisationen und keine Firmen. Dies ergibt eine Ausbeute von 42,45%. Die restlichen fehlenden Namen (**Senkung der Ausbeute**) sind auf Grund folgender problematischer Punkte entstanden:

1. Fehlen eines Triggers

100 Firmennamen konnten nicht erkannt werden, weil keine Trigger in ihrer Umgebung standen. Weitere 7 sind nicht erkannt worden, weil es sich um gekürzte Formen handelt, die von einem zuvor nicht gefundenen Firmennamen abgeleitet werden.

Um mehr Firmennamen zu finden, müssten weitere Trigger definiert werden, die allerdings sehr spezifisch sind. (z.B. *Bank* zur Erkennung von Banknamen → *Nation* ***Bank***). Bei Wörtern, die oft in einem Text vorkommen und nicht nur als Trigger bei Firmennamen stehen, kann es zu Fehlanalysen kommen und somit zu einer Schwächung der Präzision (vgl. Beispiel 69).

Beispiel 69:

<organization>*Finance*</*organization*> **Minister** and *Prime* **Minister***

Eine Erweiterung der Trigger brächte zusätzlich die Problematik mit sich, dass eine Unterscheidung von Firmen- und Organisationsnamen nicht mehr möglich ist. Organisationsnamen unterliegen bezüglich der Grossschreibung den gleichen Gesetzen wie Firmennamen und in ihrer Umgebung stehen ähnliche Wörter. Eine Ausweitung der Konzepte auf Organisations-

[143] Vgl. 135, S. 155.

namen wäre sinnvoll, wenn auch nicht so einfach (vgl. Kapitel III 4.3. „Erkennung von Firmennamen“).

2. Fehlende Definition bei Firmennamenteil und Akronym

16 Namen sind nicht erkannt worden, weil sie aus einem Firmennamenteil bestehen, der nicht definiert ist (z.B. *Standard & **Poor's** DRI* oder *Courriere della Sella*). Im Fall des Genitiv-*s* ist eine Erweiterung der Definition bedenkenlos auszuführen. Ansonsten kann eine Erweiterung die Präzisionswerte schwächen, da andere falsche Namen ebenfalls gelernt werden. Bei der Definition der Akronyme hat sich dieses Zusammenspiel zwischen Erhöhung der Ausbeute durch eine möglichst allgemeine Definition und Sinken der Präzision gezeigt (vgl. Punkt 4, S. IV-156). Die Definition der Akronyme scheint genug allgemein zu sein, da lediglich 2 Firmennamen auf Grund einer nicht definierten Form eines Akronyms fehlen.

3. Sonstige Fehler

Bei den restlichen Fehlern handelt es sich hauptsächlich um fehlende Analysen wegen Normierungsproblemen, die durch das Evaluationsskript entstanden sind (z.B. *A & E* statt *A&E*) und unübliche Schreibweisen (z.B. WORDL TIMES, INC.), die nicht erkannt worden sind, weil sie in einer nicht normierten Form vorliegen. Diese Fehler treten bei einer anderen Evaluationsmethode, bzw. bei Texten ohne normierte Schreibweisen nicht auf.

Wie bei den deutschen Konzepten gilt auch hier, dass die Firmennamenerkennung nur durch eine ‚Zwei-Weg-Lösung' bedeutend verbessert werden kann, weil die Akronymbildung präzisiert würde. Die Verwendung eines besseren Filters, z.B. *Gertwol* ([Volk und Clematide, 2001]), würde zusätzlich eine Steigerung der Präzision bewirken. Nur so lassen sich sowohl Präzision als auch Ausbeute erhöhen.

Kapitel V Ausblick

1. Resultate im Vergleich zu anderen Systemen

Mit einem F-Wert von insgesamt 55,66% im Deutschen und 70,87% im Englischen bei gleicher Gewichtung von Ausbeute und Präzision liegen die in diesem Projekt entwickelten Ressourcen und Regeln, verglichen mit den in Kapitel I 2. „Eigennamenerkennung“ vorgestellten Systemen, unter dem Durchschnitt. Die verwendeten Methoden mussten wegen der Konzeptsensorsprache einfach gestaltet werden, so dass Werte in höherem Bereich nicht erwartet werden konnten. Die Zeitbeschränkung des Projekts hatte zusätzlich zur Folge, dass ich die Konzepte nicht vollständig ausgereift entwickeln konnte und deshalb die Eigennamenerkennung keine optimalen Evaluationswerte erreicht. Mit Ausnahme der Firmennamenerkennung, die wegen der fehlenden Mächtigkeit der Konzeptsensorsprache nicht korrekt implementiert werden konnte und deshalb sehr schlechte Evaluationswerte geliefert hat, ist in den beiden anderen Kategorien ein befriedigender Standard erreicht worden. Im Vergleich mit Rösslers Werte, die er mit seinem Ansatz ([Rössler, 2002]) erreicht hat, zeigt sich, dass die Resultate der Erkennung von Personennamen und geographische Namen sehr nahe zusammen liegen (vgl. Tabelle 23). Rössler hat im Gegensatz zu diesem Projekt einen statistischen Ansatz zur Eigennamenerkennung in deutschen Texten gewählt, der ohne Ressourcen auskommt (vgl. Kapitel I 2. „Eigennamenerkennung“). Markov Modelle, welche in Rösslers Ansatz zur Anwendung kommen, sind jeweils von einem Korpus abhängig, auf dem sie trainiert werden müssen. Abgesehen von der Abhängigkeit, die durch den Lernschritt bedingt ist, findet eine präzisere, wenn auch spezifischere in Bezug auf die Eigennamenkategorie, auf die Textart und auf die Sprache, Regelbildung im Vergleich zu diesem Projekt statt. Eine Übertragung auf andere Texte, Sprachen oder Eigennamenkategorien ist somit vom Trainingstext abhängig.

Der von mir gewählte Ansatz hat sich dagegen als relativ leicht übertragbar auf andere Sprachen erwiesen. Regeln sind von mir möglichst allgemein formuliert. Eine Anwendung der Konzepte auf andere Textgattungen als Nachrichten würde vermutlich nur wenig an den Evaluationsergebnissen ändern, weil die Konzepte möglichst allgemeine Ressourcen und Regeln enthalten. Dadurch können allerdings auch keine Spitzenwerte erreicht werden.

	Rössler		**Konzeptsensoren (Deutsch)**	
	Ausbeute	*Präzision*	*Ausbeute*	*Präzision*
Personennamen	78%	71%	78%	84%
geographische Namen	77%	86%	84%	73%
Organisations- bzw. Firmennamen	62%	85%	14%	59%

Tabelle 23

Die Personennamenerkennung durch Konzepte schneidet besser ab als Rösslers Ansatz. Durch den Einsatz von Vornamenslisten und sonstigen Ressourcen kann eine höhere Präzision erreicht werden. Die geographischen Namen schneiden in der Präzision schlechter ab, weil grosse Listen mehr Ambiguität mit sich bringen. Andererseits ist es schwierig, Regeln und spezifische Wörter, die eine geographische Entität auslösen, zu finden, weshalb Rösslers Ausbeutewerte niedriger ausfallen. Die Ergebnisse der Firmennamenerkennung sind nicht repräsentativ und sollten deshalb auch nicht weiter mit anderen Extraktionsansätzen verglichen werden (vgl. Kapitel IV 3.3. und Kapitel IV 5.3. „Evaluation von Firmennamen").

Wie der Vergleich zwischen den Evaluationswerten, welche Rössler erzielt hat, und den Werten, die in diesem Projekt erreicht wurden, zeigt, führen sowohl statistische Ansätze als auch einfache, manuell erstellte Regeln mit Verwendung von Ressourcen zu befriedigenden Ergebnissen. Systeme, die auf ausgereifte Sprachverarbeitungsprozesse aufbauen, können höhere Evaluationswerte als in diesem Projekt erreichen. MUSE (vgl. Kapitel I 2. „Eigennamenerkennung“) beispielsweise, das auf die GATE-Architektur ([GATE, http://gate.ac.uk/]) aufbaut, erreicht eine Ausbeute zwischen 85-95%[144] bei gleich hoher Präzision ([Maynard et al., 2002b]). GATE (*a **G**eneral **A**rchitektur for **T**ext **E**ngineering*), die man an der Universität Sheffield entwickelt hat, ist im Jahre 1996 erstmals eingesetzt worden und hat sich während der folgenden Jahre immer weiter entwickelt. In diversen Informations-Extraktions-Projekten wurde die Architektur eingesetzt und erprobt, um bessere Werte zu erreichen. GATE verwendet natürlichsprachliche Prozesse, um Texte fundiert linguistisch zu analysieren (vgl. Abbildung 3, S. I-14). Während in meinem Projekt ausser der Aufteilung des Dokuments in einzelne Wörter und der Grossschreibung keine linguistischen Methoden verwendet werden konnten, sind in MUSE durch die GATE-Architektur diverse natürlichsprachliche Prozesse wie Satzabtrennung oder Wortartenbestimmung integriert. Diese natürlichsprachlichen Prozesse erleichtern die Eigennamenerkennung und erlauben, eine höhere Präzi-

[144] Diese Werte wurden auf Grund einer unabhängigen Evaluation (unabhängig von einer Konferenz oder einem sonstigen Kontest) erreicht. Beim ACE-Wettbewerb lagen die Werte zwischen 82% und 86% (vgl. Kapitel I 2. Eigennamenerkennung).

sion bei einer grösseren Ausbeute zu erreichen. Ausserdem profitiert MUSE von der langjährigen Erfahrung, die man mit GATE sammeln konnte.

MUSE ist regel- und ressourcenbasiert und dient zur Eigennamenerkennung in englischen Texten. Weil das System, wie oben erläutert, sowohl auf eine fundierte linguistische Analyse des Textes aufbauen kann als auch ein präzises und ausgereiftes Regelinventar besitzt, sind Präzisions- und Ausbeutewerte höher als bei der in diesem Projekt implementierten Eigennamenerkennung in englischen Texten (vgl. Tabelle 24).

	MUSE		**Konzeptsensoren (Englisch)**	
	Ausbeute	*Präzision*	*Ausbeute*	*Präzision*
Personennamen	82%-86% (ACE-Evaluation) 85%-95% (bei eigenen Testen)		80,5%	89%
geographische Namen			78%	80,5%
Organisations- bzw. Firmennamen			19%	61%

Tabelle 24

In der Firma *Eurospider* ist die Konzeptsensorsprache erstmals in einem grösseren Projekt getestet worden und brachte somit einige Probleme mit sich, die gelöst werden mussten. Bis anhin sind nur wenige, kleine Konzepte in *Relevancy* integriert worden. Somit musste zuerst herausgefunden werden, wie das System auf grosse Konzepte reagiert und wie Regeln in die Konzeptsensorsprache umgesetzt werden können. Erfahrung bestand keine, was bei der Implementierung Zeit kostete. Die Regeln fielen entsprechend „einfach“ aus. Um die bestehenden Ergebnisse dieses Projekts zu erhöhen, wäre es nötig, linguistische Prozesse wie Satzabtrennung, Wortartenbestimmung, linguistische Grundformbildung[145] etc. in der bestehenden Architektur von *Relevancy* zu integrieren. Dann wäre es möglich, in der Konzeptsensorsprache weitere, verfeinerte Regeln zu bilden, die eine genauere und bessere Eigennamenerkennung erlauben.

2. Verbesserungsmöglichkeiten

Die Firmennamenerkennung ist in diesem Projekt am schwächsten ausgefallen und enthält somit das grösste Verbesserungspotential. Es ist zu überlegen, ob Firmennamen auf Organisationsnamen ausgedehnt werden sollten, da eine klare Unterscheidung dieser beiden Kategorien schwierig ist. Allerdings würde die Erweiterung der Firmennamen auf Organisationsnamen eine Ergänzung der Ressourcen mit sich bringen, was einige Stunden Arbeitsaufwand bedeutet. Es ist dagegen zu erhoffen, dass Organisationsnamen sicherer erkannt werden kön-

[145] In der Konzeptsensorsprache kann mit Grundformen (gestemmte Formen) gearbeitet werden. Es handelt sich jedoch dabei nicht um linguistische Grundformen, sondern lediglich nach gewissen Regeln reduzierte Formen → Trunkieren (vgl. Kapitel III 2.4. „Probleme innerhalb der Konzeptsensorsprache und ihre Grenzen“).

nen, weil sie eine stärkere innere Struktur als Firmennamen aufweisen, und man somit präziser und umfassender Regeln erstellt kann. Die Problematik der Firmennamenerkennung bleibt allerdings bestehen. Um Präzisions- und Ausbeutewerte bei der Firmennamenerkennung zwischen 60% -80% zu bekommen, muss eine ‚Zwei-Weg-Lösung' (vgl. Kapitel III 2.4. „Probleme innerhalb der Konzeptsensorsprache und ihre Grenzen") realisiert und zusätzlich der Filter präzisiert werden. Der Filter enthält einteilige Wörter, die, falls sie als einteilige Firmennamen gelernt oder abgeleitet wurden, wieder aus der Firmennamenliste entfernt werden müssen (vgl. Kapitel III 3.3. „Ressourcen zur "). Die Korrekturen der Konzepte, wie sie in Kapitel IV 3.3. und Kapitel IV 5.3. „Evaluation von Firmennamen" besprochen wurden, würden ebenfalls zu einer Erhöhung der Evaluationswerte führen und könnten entsprechend effizient ausgeführt werden.

Personennamenerkennung sowie geographische Namenserkennung könnten durch die Anpassung, bzw. Erweiterung, der Ressourcen an bestimmte Themenkreise profitieren. Wenn *Relevancy* beispielsweise in einem spezifischen Themenkreis eingesetzt würde, wie z.B. im universitären Bereich, könnten die Ressourcen zur Erkennung von Personennamen mit Triggern wie *Privatdozent* oder *PHIL* ergänzt werden. Die geographischen Ressourcen können Mit Namen aus andern Regionen wie den bestehenden (der englische und deutsche Sprachraum) ergänzt werden. Asiatische Städte oder Dörfer sind beispielsweise kaum vorhanden. Ich habe die geographischen Ressourcen bewusst auf den deutschen und englischen Sprachraum eingeschränkt. Städte-, Dorf- und Regionsnamen anderer Gebiete (z.B. Italien, Spanien etc.) müssten entsprechend erweitert werden.

3. Weiterverwendung

Um die Eigennamenerkennung in das bestehende Retrievalsystem *Relevancy* zu integrieren, muss die Firma *Eurospider* die Firmennamenerkennung entsprechend überarbeiten. Die Personennamenerkennung und die geographische Eigennamenerkennung sind auf Grund der erreichten Resultate zufrieden stellend, wenn auch weiter entwicklungsfähig. Weitere Evaluationen sind jedoch nötig, um herauszufinden, wie die Eigennamenerkennung in grösseren Textmengen abschneidet. Um genauere Fehleranalysen der Konzepte aufstellen zu können, muss eine Evaluation durchgeführt werden, die die Position der gefundenen Eigennamen mitberücksichtigt (vgl. Kapitel IV 1. „Evaluationsskript"). Eine Benutzeroberfläche, welche erlaubt, die Eigennamenkategorien bei der Suche zu integrieren, muss neu geschaffen oder die Bestehende entsprechend angepasst werden. Wann die Eigennamenerkennung tatsächlich in *Relevancy* implementiert werden soll, ist unbekannt. Dass die Eigennamenerkennung jedoch eine notwendige Erweiterung des bestehenden Retrievalsystems ist und von den Benutzern

des Systems gewünscht wird, hat sich bereits vor diesem Projekt abgezeichnet. Deshalb zeigte *Eurospider* Interesse an meiner Lizentiatsarbeit. Sie liessen mich das Projekt in ihrer Firma durchführen, um so erste Erfahrungen in diesem Bereich zu sammeln.

4. Zusammenfassung

Dieses Projekt ist in Zusammenarbeit mit der Firma *Eurospider* entstanden. Das Ziel, Personennamen, geographische Eigennamen und Firmennamen innerhalb eines Texts über Konzeptsensoren zu erkennen, ist von mir erreicht worden. Auf den Grundlagen von [Volk, 2002] aufbauend, habe ich bestehende Regeln von Volk übernommen, teilweise erweitert sowie neue Regeln entwickelt Um das Verständnis zu erhöhen habe ich die Regeln zuerst in einen Pseudocode umgewandelt. Dieser Pseudocode bildete wiederum die Grundlage, um entsprechende Regeln und Ressourcen in der Konzeptsensorsprache zu implementieren. Die Konzeptsensorsprache ist eine von der Firma *Eurospider* entwickelte Mustererkennungssprache. Sie ist in den Indexer, das Herzstück des IR-Systems *Relevancy* von *Eurospider*, eingebunden. In der Konzeptsensorsprache lassen sich Regeln (Muster) definieren, die zu einer Erkennung gewisser Komponenten (Wörter), z.B. Eigennamen, in einem Text dienen. Folglich basiert die Eigennamenerkennung in diesem Projekt auf Regeln und Ressourcen. Die Ressourcen habe ich von [Volk, 2002] übernehmen können, musste sie aber über diverse Quellen, wie das Internet, stark erweitern. Insbesondere geographische Eigennamen aus dem englischensprachigen Bereich, sowie Triggerlisten, welche Eigennamen in englischen Texten auslösen, mussten ergänzt werden. Die Regeln und Ressourcen sind zuerst für die Erkennung in deutschen Texten konzipiert worden. Nach einer Evaluationsphase sind die Regeln und Ressourcen von mir für die Erkennung von Eigennamen in englischen Texten angepasst worden. Die Ressourcen insbesondere mussten an die neue Sprache angepasst werden. Regeln konnten meist gleich übernommen werden, da das markanteste Merkmal, welches ich zur Erkennung von Eigennamen verwendet habe, nämlich die Grossschreibung, sowohl im Deutschen, als auch im Englischen besteht.

Mit der Implementierung in die Konzeptsensorsprache ist von mir gleichzeitig erforscht worden, inwiefern sich diese firmeneigene Sprache für Eigennamenerkennung eignet und wo die Grenzen liegen. Bis anhin hat man die Konzeptsensorsprache lediglich mit einfachen und kleinen Konzepten getestet und sie nicht für ein grösseres Projekt als Mustererkennungssprache eingesetzt. Während der Projektzeit hat sich herausgestellt, dass für die Firmennamenerkennung die Konzeptsensorsprache zu wenig mächtig ist (Schleifenstrukturen können nicht umgesetzt werden). Grosse Ressourcen lassen sich problemlos integriert, so dass die geographische Eigennamenerkennung befriedigend abgeschnitten hat. Dass keine Wortbearbeitung

(Stringbearbeitung) möglich ist, hat dazu geführt, dass ich grössere Ressourcen anlegen musste. Es wäre wünschenswert, dass linguistische Analysen wie Wortartenbestimmung oder Lemmatisierung im Indexer ermöglicht werden, so dass man sie in der Konzeptsensorsprache verwendet verwenden kann.

Bei der Evaluation der Konzepte zur Erkennung von Personennamen, geographischen Eigennamen und Firmennamen in deutschen sowie englischen Texten sind Werte zwischen 14,39% und 84,34% Ausbeute und 58,62% und 88,95% Präzision erreicht worden. Die Werte divergieren so stark, weil die Firmennamenerkennung, welche wegen der eingeschränkten Funktionsfähigkeit der Konzeptsensorsprache nur ansatzweise implementiert werden konnte, sehr tiefe Werte ergeben hat. Personennamen und geographische Eigennamen haben Werte zwischen 78,22% und 84,34% Ausbeute bei einer Präzision zwischen 72,61% und 88,95% erreicht. Diese Zahlen sind mit der Güte anderer Eigennamenerkennungssysteme durchaus vergleichbar. Weil die Konzeptsensorsprache in ihren Möglichkeiten sehr beschränkt ist, mussten die Regeln entsprechend einfach gehalten werden, so dass es schwierig war hohe Evaluationswerte zu erreichen. Weder die Abtrennung von Sätzen, noch Wortartenbestimmung, noch eine Reduktion der Wörter auf ihre linguistische Grundform (engl. stemming) standen zur Verfügung. Die Definition von grossgeschriebenen Wörtern, grossgeschriebenen Buchstaben sowie Initialen musste in der Konzeptsensorsprache über Konzepte definiert werden.

Das Ziel des Projektes, verwendbare Konzepte zur Erkennung von Personennamen, geographischen Namen und Firmennamen zu entwickeln, die möglichst unabhängig von der Textart eingesetzt werden können, ist erreicht worden. Durch die Umsetzung der Konzepte, welche Eigennamen in deutschen Texten erkennen, in Konzepte, welche Eigennamen in englischen Texten erkennen, ist deutlich geworden, mit welchem Aufwand die bestehende Eigennamenerkennung sich in andere Sprachen umsetzten lässt. Solange es sich um indogermanische Sprachen handelt, die strukturelle Gemeinsamkeiten aufweisen und ähnlichen grammatischen Gesetzten unterliegen, ist die Anpassung hauptsächlich auf eine Veränderung und Erweiterung der Ressourcen beschränkt. Dies ist mit angemessenem Arbeitsaufwand möglich. Die meisten entwickelten Regeln können gleich übernommen werden.

Um die Eigennamenerkennung ins Retrival-System *Relevancy* einzubinden, muss das System bezüglich Benutzeroberfläche und sonstigen Zugriffsmöglichkeiten erweitert werden. Die Erkennung von Personennamen und geographische Eigennamen haben bei der Evaluation Werte ergeben, die zu keinen, grundsätzlichen Veränderung Anlass geben. Sie könnten bereits ohne grössere Verbesserungen integriert werden. Die Firmennamenerkennung dagegen erfor-

dert weitere Entwicklungs- und Evaluationsschritte, um als funktionstüchtiges Modul im System eingesetzt zu werden.

Anhang

Pseudocode

Durch die Erstellung dieses Pseudocodes habe ich die Algorithmen von Volk, welche in *Perl* implementiert waren, in eine allgemeine Form gebracht, um sie weiter in die Konzeptsensorsprache umzusetzen. Der Zwischenschritt hat einerseits zu einem präziseren Verständnis der Algorithmen geführt und andererseits die bestehenden Probleme aufgezeichnet. Die Pseudocodeversion enthält bereits erste Abänderungen der ursprünglich von Volk verwendeten Regeln und Ressourcen.

Ressourcen

Die folgenden Ressourcen sind im folgenden Pseudocode verwendet worden.

- Liste mit Vornamen
  List_of_person_First_Name
  Die Liste besteht aus europäischen sowie einigen asiatischen Vornamen.
- Liste mit Städte- und Dorfnamen
  External_List_of_Cities
  Die Liste enthält Städtenamen aus Deutschland, der Schweiz und Österreich.
- Liste mit Namen von Nationen
  List_of_Nations
  Die Liste enthält die Ländernamen der Welt, wobei mit Ländern Nationen gemeint sind. Die Bundesländer sind in einer eigenen Liste aufgeführt.
- Liste mit Namen von Kantonen und Ländern
  List_of_Kantonen_and_Laender
  Die Liste enthält Namen von Schweizer Kantonen, deutschen und österreichischen Bundesländern und englischen und kanadischen Staaten.
- Liste mit Regions- und Inselnamen
  List_of_Regions_and_Islands
  Die Liste enthält regionale Bezeichnungen aus dem deutschsprachigen Raum sowie Inselnamen, welche in den andern Listen mit geographischen Eigennamen nicht aufgeführt sind.
- Liste mit Namen von Kontinenten
  List_of_Continents
  Die Liste enthält die Namen der Kontinente der Welt.
- Listen mit geographische Adjektiven
  List_of_er_Adjectives
  External_List_of_isch_Adjectives
  Diese Listen enthalten Einträge mit geographischen Adjektiven, welche entweder auf *–isch* oder auf *–er* enden. Die Adjektive wurden von den bestehenden geographischen Ressourcen abgeleitet und stammen somit hauptsächlich aus dem deutschsprachigen Raum (entspricht Deutschland, Österreich und der Schweiz).
- Lexikon Englisch – Deutsch
  Lexikon
  Das Lexikon ist ein kleines Wörterbuch, das einen Standardwortschatz in Deutsch und Englisch enthält.

- Abkürzungsverzeichnis
  External_List_of_Abbrivation
  Enthält die gängigen Abkürzungen aus dem deutschen Sprachgebrauch.

Regeln

Personennamen

```
*************************************************************************

## Ressourcen

Personen_Namen_Trigger(W) {
  IF  (W in [Prof., Dr., Sir, Professor, Ing., Paesident, Chef, Senator,
             Baron, Herr, Frau])
  THEN TRUE
}

First_Name (W) {
  IF  (W in External_List_of_Person_First_Name)
  THEN TRUE
}

Salutation_Word(W) {
  if      (W in [von, van, de, vom, della, del, di, d', O', von zur,
                 van den])
  THEN TRUE
}

*************************************************************************

## Regeln

Hyphaneted_Name (W) {
  IF  (W == hyphenated)
  THEN      {split (Firstpart, Secondpart, W)}
      IF    (Firstname(Firstpart))
            AND (Fristname(Secondpart))
      THEN TRUE
      ELSIF (Initial(Firstpart))
            AND (Firstname(Secondpart))
      THEN TRUE
      ELSIF (Firstname(Firstpart))
            AND (Initial(Secondpart))
      THEN TRUE
}

Pos_Lastname (W) {
  IF  (Capitalized_Word (W) OR Capitalized_Word(W),-,Capitalized_Word(W))
  THEN TRUE
}
```

```
Learn_Person_Name {
  IF  (sequence_of_tokens(W0, W1, W2, W3, W4))
  THEN      IF    (Person_Name_Trigger(W0))
      THEN  (Full_Person_Name(W1, W2, W3, W4))
      ELSE  (Abbriviated_Person_Name(W1, W2, W3, W4))
  ELSIF     (sequence_of_tokens(W0, W1, W2, W3)
  THEN      IF    (Person_NAME_Trigger(W0))
      THEN  (Full_Person_Name(W1, W2, W3))
        ELSE      (Abbriviated_Person_Name(W1, W2, W3))
}

## ,save Familyname' bedeuted, dass der Name abgespeichert wird und im
## gleichen Dokument erneut gesucht werden kann.
## ,mark Personname' bedeuted, dass die entsprechende Sequenz mit einem
## speziellen Tag markiert wird.

Full_Person_Name (W1, W2, W3, W4) {
  IF  (First_Name(W1)) OR (Hyphaneted_Name(W1))
      AND (Initial(W2)) OR (Salutation_Word(W2))
      AND (Pos_Lastname (W3))
      AND NOT (Company_Trigger_Word(W4))
  THEN {save Familiyname(W3);
        mark Personname(W1, W2, W3)}
}

Full_Person_Name (W1, W2, W3) {
  IF  (First_Name(W1)) OR (Hyphaneted_Name(W1))
      AND (Pos_Lastname (W2))
      AND NOT (Company_Trigger_Word(W3))
  THEN {save Familiyname(W3);
        mark Personname(W1, W2)}
}

Abbrivated_Person_Name (W1, W2, W3, W4) {
  IF  (Initial(W1))
      AND (Initial(W2)) OR ( Salutation_Word(W2))
      AND (Pos_Lastname (W3))
      AND NOT (Company_Trigger_Word(W4))
  THEN {save Familiyname(W3);
        mark Personname(W1, W2, W3)}
}

Abbrivated_Person_Name (W1, W2, W3) {
  IF  (Initial(W1))
      AND (Initial(W2)) OR ( Salutation_Word(W2))
      AND (Pos_Lastname (W2))
      AND NOT (Company_Trigger_Word(W4))
  THEN {save Familyname(W2);
        mark Personname(W1, W2)}
}

## ,Familyname' enthaelt alle zuvor gelernten Familiennamen, welche mit
## ,save Familiyname' abgespeichert wurden.

learn_Familyname (W) {
  IF  (Familyname(W)
  THEN {mark Personname(W)}
}
```

****

Geographische Namen

```
*************************************************************************

## Ressourcen

## Mehrteilige Geo-Namen (bis zu vierteilige Geo-Namen) werden jeweils in
## separaten Listen abgespreichert. Alle geographischen Listen beinhalten
## somit bis zu vierstellige Namen.

City_Name (W) {
  IF  (W in External_List_of_cities)
  THEN TRUE
}

Nation_Name (W) {
  IF  (W in External_List_of_nations)
  THEN TRUE
}

Kanton_Laender_Name (W) {
  IF  (W in External_List_of_Kantone_and_Laender)
  THEN TRUE
}

Region_and_Island_Name (W) {
  IF  (W in External_List_of_regions_and_islands)
  THEN TRUE
}

Continent_Name (W) {
  IF  (W in External_List_of_continents)
  THEN TRUE
}

Er_Adjetive (W) {
  IF  (W in External_List_of_er_adjectives)
  THEN TRUE
}

Isch_Adjective (W) {
  IF  (W in External_List_of_isch_adjectives)
  THEN TRUE
}

*************************************************************************

## Regeln

Hyphenated_er_Adjective (W) {
  IF  (W == hyphenated)
  THEN {split (Firstpart, Secondpart, W)}
      IF    (Er_Adjective(Firstpart))
            AND (Er_Adjective(Secondpart))
  THEN TRUE
}

## Isch-Adjektive werden gestemmt gesucht.
## Er-Adjektive sind prizipiell nicht flektiert.
```

```
Find_geoname (W) {
  IF  (City_Name (W))
      OR (Nation_Name (W))
      OR (Kantone_Laender_Name (W))
      OR (Region_and_Island_Name (W))
      OR (Continent_Name (W))
      OR (Er-Adjetive (W))
      OR (Isch_Adjective (W))
      OR (Hyphenated_er_Adjective (W))
  THEN {mark Geoname(W)}
}

*************************************************************************

## Jeder Name sollte auch im Genitiv vorhanden sein. Dies bedeuted, dass
## der Buchstabe s dem Namen angehaengt wird.
## Der Algorithmus wird nur die Listen City_Name, Kanton_Laender_Name,
## Continenten_Namen angewendet.


Save_Genitive     {
  IF  (sequence_of_tokens(W0, W1))
  THEN {save_as_genitive(W1)
       AND  mark Geoname(W0, W1)}
  ELSE {save_as_genitiv(W)
       AND mark Geoname(W)}
}

*************************************************************************
```

Firmennamen

```
*************************************************************************

##Ressourcen

Trigger_in_Front (W) {
  IF  (W in [([-?][Ff]])irma, Technologie von, Uebernahme von, Unternehmen,
            Beratung, Veranstalter, Konzern,Treuhandfirma,
            Consultingfirma, Gemeinschaftsunternehmen, [Kk]onzern,
            Netzwerkspezialist, Anbieter]
  THEN TRUE
}

Trigger_Behind (W) {
  IF  (W in [GmbH, mbH, Ltd[.?], Inc[.?], SA, AG, AB, KG, Oy, B.V., Co[.?],
            Corp., Groupe, Corporation, Holding, Institute, Associates,
            Agentur])
  THEN TRUE
}

Hyphanted_Affix (W) {
  IF  (W in [-Tochter, -Chef, -Praesident])
  THEN TRUE
}

Stopword_for_Trigger_in_Front (W) {
  IF  ((W in [Software, Trade, Investitionen]) OR (Trigger_Behind(W)))
  THEN TRUE
}
```

```
Stopword_for_Trigger_Behind (W) {
  IF  (W in [Die, Der, DAS [Tt]ochter, [Uu]unternehmer[s?],[Gg]ruppe,
            [Ff]uehrer[s?], [Hh]erhesteller[s?], [Aa]nbieter[s?],
            [Ee]ntwickler[s?] [Kk]onzer[s?], [Ll]iferant[s?], [Hh]aus,
            [Gg]essellschaft, [Ss]pezialist])
  THEN TRUE
}

Sign (W) {
  IF          (W in [-,,,',[0-9]])
  THEN TRUE
}

Filter (W) {
  IF  (W in External_List_Lexikon)
      OR W in External_List_of_Abbrivation)
      OR ((W in [Buerger, Internet[s?], GMbH, PC[s?], [L?]Cd[s?], WWW, EG,
                EU, Unix, IT-, konferenz, scene])
      OR (Initial(W)) OR (City_Name (W)) OR (Nation_Name (W)) OR
         (Katon_Laender_Name (W)) OR (Region_and_Island_Name (W)) OR
(Continent_Name (W)))
  THEN TRUE
}

*************************************************************************

## Regeln

Possible_FA  {
  IF  (Capitalized Word (W) OR Initial (W))
  THEN TRUE
  ELSIF     (Sequence_of_Tokens(W0, W1, W2))
  THEN      IF (Capitalized Word (W0) OR Initial (W0))
          AND (Sign (W1))
          AND (Capitalized_Word (W0) OR Initial (W0))
      THEN TRUE
}

## W0 ist das erste Token einer Sequenz (S) von W0 - WN
## Bei der Methode save FANAME wird der Name gespeichert, damit er ueber
## alle Dokumente erneut gesucht werden kann.

Searche_Forward {
  IF  (Sequence_of_Tokens(S))
  THEN      IF (Trigger_in_Front (W0))
      THEN
            WHILE (NOT Possible_FA (W)) OR (Stopword_for_Trigger_in_Front
(W))
                  DO {(check (W) == Possible_FA (W)}
            IF (Possible_FA (W))
              THEN {save FANAME(W);
                  mark FANAME(W)}
}
```

```
## WL ist das letzte Token einer Sequenz (S)  von W0 bis WN

Searche_backwards {
  IF  (Sequence_of_Tokens(S))
  THEN      IF    (Trigger_Behind(WL))
      THEN
            WHILE ((W != Possible_FA (W))
                   OR (W = Stoword_for_Trigger_Behind (W)))
                  DO {check (W) == Posible_FA (W)}
            IF    (Posible_FA (W))
            THEN {save FANAME(W);
                  mark FANAME(S)}
}

Acronym (W) {
  IF  (W == (([A-Z]+) OR ([A-Z] AND &)))
  THEN TRUE
}

Find_Acronym {
  IF  ((FANAME(W)) AND (Sequence_of_Tokens(W)))
  THEN
      WHILE (NOT Sequence_of_Tokens(Rest))
            DO {split(First,Rest,W)}
       IF   (First(Acronym))
       THEN {save FANAME(Acronym)}
}

Find_Faname {
  IF  ((FANAME(W)) AND NOT (Filter(W))
  THEN {mark FANAME(W)}
}
```

****

Segmentierungen

****

```
## Zusaetzliche Segmentierung

## ‚Capital_letter' ist ein Grossbuchstabe oder eine Sequenz von
## Grossbuchstaben

Initial (W) {
 IF        (W == Initial)
 THEN {W = (Capital_Letter,.)}
}

## Ein am Anfang grossgeschriebenes Wort

Capitalized_Word (W) {
  IF  (W == Capitalized_Word)
  THEN {W = (Capital_Letter,Letters)}
}
```

Modul zur Personenamenerkennung in englischen Texten

Das folgende Modul ist ein Auszug aus dem im Projekt erstellten Programmcode. Das Modul enthält die Regeln, welche ich zur Erkennung von Personennamen in der Konzeptsensorsprache implementiert habe. Die Ressourcen sind in einem eigenen anderen Modul aufgelistet.

```
MODULE "pers_rec" LANGUAGE "english"

IMPORT "capitalized_word"
IMPORT "initial_of_name"
IMPORT "capital_letters"

IMPORT "sal_pers_name" FROM "pers_resources"
IMPORT "suffix_names" FROM "pers_resources"
IMPORT "pre_pers_name" FROM "pers_resources"
IMPORT "pre_pers_double_name" FROM "pers_resources"
IMPORT "hyphenated_pre_pers_name" FROM "pers_resources"
IMPORT "last_name_part" FROM "pers_resources"
IMPORT "stop_word_list_for_single_names" FROM "pers_resources"
IMPORT "stop_word_list_for_initial_of_names" FROM "pers_resources"
IMPORT "pos_last_name" FROM "pers_resources"
IMPORT "pre_name" FROM "pers_resources"

IMPORT "find_hyphenated_geonames" FROM "geo_rec"
IMPORT "find_double_names" FROM "geo_rec"
IMPORT "find_triple_names" FROM "geo_rec"
IMPORT "find_quadruple_names" FROM "geo_rec"
IMPORT "find_quintuple_names" FROM "geo_rec"
IMPORT "find_slash_citiesI" FROM "geo_rec"
IMPORT "find_slash_citiesII" FROM "geo_rec"
IMPORT "find_slash_citiesIII" FROM "geo_rec"
IMPORT "find_st_names" FROM "geo_rec"
IMPORT "hyphenated_geonames_with_abbr" FROM "geo_rec"
IMPORT "double_names_with_abbr" FROM "geo_rec"
IMPORT "triple_names_with_abbr" FROM "geo_rec"
IMPORT "quadruple_names_with_abbr" FROM "geo_rec"
IMPORT "quintuple_names_with_abbr" FROM "geo_rec"
IMPORT "slash_citiesI_with_abbr" FROM "geo_rec"
IMPORT "slash_citiesII_with_abbr" FROM "geo_rec"
IMPORT "slash_citiesIII_with_abbr" FROM "geo_rec"
IMPORT "st_names_with_abbr" FROM "geo_rec"
IMPORT "find_abbrivation" FROM "geo_rec"

IMPORT "one_fa_name_follows" FROM "fa_rec"
IMPORT "one_fa_name_follows_in_quotation_marks" FROM "fa_rec"
IMPORT "two_fa_names_follow" FROM "fa_rec"
IMPORT "two_fa_names_follow_in_quotation_marks" FROM "fa_rec"
IMPORT "three_fa_names_follow" FROM "fa_rec"
IMPORT "three_fa_names_follow_in_quotation_marks" FROM "fa_rec"
```

```
IMPORT "four_fa_names_follow" FROM "fa_rec"
IMPORT "four_fa_names_follow_in_quotation_marks" FROM "fa_rec"
IMPORT "five_fa_names_follow" FROM "fa_rec"
IMPORT "five_fa_names_follow_in_quotation_marks" FROM "fa_rec"
IMPORT "six_fa_names_follow" FROM "fa_rec"
IMPORT "six_fa_names_follow_in_quotation_marks" FROM "fa_rec"
IMPORT "seven_fa_names_follow" FROM "fa_rec"
IMPORT "seven_fa_names_follow_in_quotation_marks" FROM "fa_rec"
IMPORT "one_fa_name_in_front" FROM "fa_rec"
IMPORT "two_fa_names_in_front" FROM "fa_rec"
IMPORT "three_fa_names_in_front" FROM "fa_rec"
IMPORT "four_fa_names_in_front" FROM "fa_rec"
IMPORT "five_fa_names_in_front" FROM "fa_rec"
IMPORT "six_fa_names_in_front" FROM "fa_rec"
IMPORT "seven_fa_names_in_front" FROM "fa_rec"

TYPEDEF CAPWORD
TYPEDEF CAPLET
TYPEDEF INITIAL
TYPEDEF LASTNAMEPART
TYPEDEF NAMEPART
TYPEDEF POSLASTNAME
TYPEDEF LASTNAME
TYPEDEF PERS
TYPEDEF HNAMEPART
TYPEDEF DNAMEPART

/*Bsp: Herr Hans Peter Maurer*/
CONCEPT "full_name"
IF
        ("sal_pers_name":CONCEPT);
            (("pre_name":CONCEPT AND $?:NAMEPART)
        OR ("pre_pers_double_name":CONCEPT AND $?:DNAMEPART)
        OR ("hyphenated_pre_pers_name":CONCEPT AND $?:HNAMEPART)
        OR ("initial_of_name":CONCEPT AND $?:INITIAL));
        ("pos_last_name":CONCEPT AND $?:POSLASTNAME)
CONDITION
        NOT find("initial_of_name":CONCEPT, -1);
        NOT find("pre_pers_name":CONCEPT, -1);
        NOT find("pos_last_name":CONCEPT, +1);
        create("$0 $1":PERS);
        create("$1":LASTNAME)
END
```

```
/*Frau A. Maria von Hausen*/
CONCEPT "full_name"
IF
                ("sal_pers_name":CONCEPT);
                    (("pre_name":CONCEPT AND $?:NAMEPART)
                OR ("pre_pers_double_name" AND $?:DNAMEPART)
                OR ("hyphenated_pre_pers_name" AND $?:HNAMEPART)
                OR ("initial_of_name":CONCEPT AND $?:INITIAL));
                ("last_name_part":CONCEPT AND $?:LASTNAMEPART);
                ("pos_last_name":CONCEPT AND $?:POSLASTNAME)
CONDITION
                NOT find("initial_of_name":CONCEPT, -1);
                NOT find("pre_pers_name":CONCEPT, -1);
                NOT find("pos_last_name":CONCEPT, +1);
                create("$0 $1 $2":PERS);
                create("$1 $2":LASTNAME);
                create("$2":LASTNAME)
END

/*Frau  Maria Anna-Barabara von Hausen*/
CONCEPT "full_name"
IF
                ("sal_pers_name":CONCEPT);
                    (("pre_name":CONCEPT AND $?:NAMEPART)
                OR ("pre_pers_double_name" AND $?:DNAMEPART)
                OR ("hyphenated_pre_pers_name" AND $?:HNAMEPART)
                OR ("initial_of_name":CONCEPT AND $?:INITIAL));
                    (("pre_name":CONCEPT AND $?:NAMEPART)
                OR ("pre_pers_double_name" AND $?:DNAMEPART)
                OR ("hyphenated_pre_pers_name" AND $?:HNAMEPART)
                OR ("initial_of_name":CONCEPT AND $?:INITIAL));
                ("last_name_part" AND $?:LASTNAMEPART);
                ("pos_last_name":CONCEPT AND $?:POSLASTNAME)
CONDITION
                create("$0 $1 $2 $3":PERS);
                create("$2 $3":LASTNAME);
                create("$3":LASTNAME)
END

/*Frau von Hausen*/
CONCEPT "full_name"
IF
                ("sal_pers_name":CONCEPT);
                ("last_name_part" AND $?:LASTNAMEPART);
                ("pos_last_name":CONCEPT AND $?:POSLASTNAME)
CONDITION
                NOT find("pos_last_name":CONCEPT, +1);
                create("$0 $1":PERS);
                create("$0 $1":LASTNAME);
                create("$1":LASTNAME)
END
```

```
/*Frau Hausen*/
CONCEPT "full_name"
IF
        ("sal_pers_name":CONCEPT);
        ("pos_last_name":CONCEPT AND $?:POSLASTNAME)
CONDITION
        NOT find("pos_last_name":CONCEPT, +1);
        create("$0":PERS);
        create("$0":LASTNAME)
END

/*Bsp. Jean-Pierre Maier*/
CONCEPT "last_name"
IF
            (("pre_name":CONCEPT AND $?:NAMEPART)
        OR ("pre_pers_double_name" AND $?:DNAMEPART)
        OR ("hyphenated_pre_pers_name" AND $?:HNAMEPART)
        OR ("initial_of_name":CONCEPT AND $?:INITIAL));
        ("pos_last_name":CONCEPT AND $?:POSLASTNAME)
CONDITION
        NOT find("sal_pers_name":CONCEPT, -3);
        NOT find("initial_of_name":CONCEPT, -2);
        NOT find("pre_pers_name":CONCEPT, -1);
        NOT find("pos_last_name":CONCEPT, +1);
        NOT overlaps_with("stop_word_list_for_initial_of_names":CONCEPT);
        NOT overlaps_with("full_name":CONCEPT);
/*Zusammengesetzte Geo-Namen dürfen keine Personennamen sein.*/
        NOT overlaps_with("find_hyphenated_geonames":CONCEPT);
        NOT overlaps_with("find_double_names":CONCEPT);
        NOT overlaps_with("find_triple_names":CONCEPT);
        NOT overlaps_with("find_quadruple_names":CONCEPT);
        NOT overlaps_with("find_quintuple_names":CONCEPT);
        NOT overlaps_with("find_slash_citiesI":CONCEPT);
        NOT overlaps_with("find_slash_citiesII":CONCEPT);
        NOT overlaps_with("find_slash_citiesIII":CONCEPT);
        NOT overlaps_with("find_st_names":CONCEPT);
        NOT overlaps_with("hyphenated_geonames_with_abbr":CONCEPT);
        NOT overlaps_with("double_names_with_abbr":CONCEPT);
        NOT overlaps_with("triple_names_with_abbr":CONCEPT);
        NOT overlaps_with("quadruple_names_with_abbr":CONCEPT);
        NOT overlaps_with("quintuple_names_with_abbr":CONCEPT);
        NOT overlaps_with("slash_citiesI_with_abbr":CONCEPT);
        NOT overlaps_with("slash_citiesII_with_abbr":CONCEPT);
        NOT overlaps_with("slash_citiesIII_with_abbr":CONCEPT);
        NOT overlaps_with("st_names_with_abbr":CONCEPT);
        NOT overlaps_with("sal_pers_name":CONCEPT);
        NOT overlaps_with("part_of_fa":CONCEPT);
        NOT overlaps_with("find_abbrivation":CONCEPT);
        create("$0 $1":PERS);
        create("$1":LASTNAME)
END
```

```
/*Bsp. H. Pierre Maier*/
CONCEPT "last_name"
IF
                    (("pre_name":CONCEPT AND $?:NAMEPART)
                OR ("pre_pers_double_name" AND $?:DNAMEPART)
                OR ("hyphenated_pre_pers_name" AND $?:HNAMEPART)
                OR ("initial_of_name":CONCEPT AND $?:INITIAL));
                    (("pre_pers_double_name" AND $?:DNAMEPART)
                OR ("hyphenated_pre_pers_name" AND $?:HNAMEPART)
                OR ("initial_of_name":CONCEPT AND $?:INITIAL));
                ("pos_last_name":CONCEPT AND $?:POSLASTNAME)
CONDITION
                NOT find("sal_pers_name":CONCEPT, -3);
                NOT overlaps_with("stop_word_list_for_initial_of_names":CONCEPT);
/*Zusammengesetzte Geo-Namen dürfen keine Personennamen sein.*/
                NOT overlaps_with("find_hyphenated_geonames":CONCEPT);
                NOT overlaps_with("find_double_names":CONCEPT);
                NOT overlaps_with("find_triple_names":CONCEPT);
                NOT overlaps_with("find_quadruple_names":CONCEPT);
                NOT overlaps_with("find_quintuple_names":CONCEPT);
                NOT overlaps_with("find_slash_citiesI":CONCEPT);
                NOT overlaps_with("find_slash_citiesII":CONCEPT);
                NOT overlaps_with("find_slash_citiesIII":CONCEPT);
                NOT overlaps_with("find_st_names":CONCEPT);
                NOT overlaps_with("hyphenated_geonames_with_abbr":CONCEPT);
                NOT overlaps_with("double_names_with_abbr":CONCEPT);
                NOT overlaps_with("triple_names_with_abbr":CONCEPT);
                NOT overlaps_with("quadruple_names_with_abbr":CONCEPT);
                NOT overlaps_with("quintuple_names_with_abbr":CONCEPT);
                NOT overlaps_with("slash_citiesI_with_abbr":CONCEPT);
                NOT overlaps_with("slash_citiesII_with_abbr":CONCEPT);
                NOT overlaps_with("slash_citiesIII_with_abbr":CONCEPT);
                NOT overlaps_with("st_names_with_abbr":CONCEPT);
                NOT overlaps_with("sal_pers_name":CONCEPT);
                NOT overlaps_with("part_of_fa":CONCEPT);
                NOT overlaps_with("find_abbrivation":CONCEPT);
                create("$0 $1 $2":PERS);
                create("$2":LASTNAME)
END
```

```
/*Bsp. Andra-Maria von der Flüh*/
CONCEPT "last_name"
IF
                    (("pre_name":CONCEPT AND $?:NAMEPART)
            OR ("pre_pers_double_name" AND $?:DNAMEPART)
            OR ("hyphenated_pre_pers_name" AND $?:HNAMEPART)
            OR ("initial_of_name":CONCEPT AND $?:INITIAL));
            ("last_name_part" AND $?:LASTNAMEPART);
            ("pos_last_name":CONCEPT AND $?:POSLASTNAME)
CONDITION
            NOT find("sal_pers_name":CONCEPT, -3);
            NOT find("initial_of_name":CONCEPT, -1);
            NOT find("pre_pers_name":CONCEPT, -1);
            NOT find("pos_last_name":CONCEPT, +1);
/*Zusammengesetzte Geo-Namen dürfen keine Personennamen sein.*/
            NOT overlaps_with("find_hyphenated_geonames":CONCEPT);
            NOT overlaps_with("find_double_names":CONCEPT);
            NOT overlaps_with("find_triple_names":CONCEPT);
            NOT overlaps_with("find_quadruple_names":CONCEPT);
            NOT overlaps_with("find_quintuple_names":CONCEPT);
            NOT overlaps_with("find_slash_citiesI":CONCEPT);
            NOT overlaps_with("find_slash_citiesII":CONCEPT);
            NOT overlaps_with("find_slash_citiesIII":CONCEPT);
            NOT overlaps_with("find_st_names":CONCEPT);
            NOT overlaps_with("hyphenated_geonames_with_abbr":CONCEPT);
            NOT overlaps_with("double_names_with_abbr":CONCEPT);
            NOT overlaps_with("triple_names_with_abbr":CONCEPT);
            NOT overlaps_with("quadruple_names_with_abbr":CONCEPT);
            NOT overlaps_with("quintuple_names_with_abbr":CONCEPT);
            NOT overlaps_with("slash_citiesI_with_abbr":CONCEPT);
            NOT overlaps_with("slash_citiesII_with_abbr":CONCEPT);
            NOT overlaps_with("slash_citiesIII_with_abbr":CONCEPT);
            NOT overlaps_with("st_names_with_abbr":CONCEPT);
            NOT overlaps_with("sal_pers_name":CONCEPT);
            NOT overlaps_with("part_of_fa":CONCEPT);
            NOT overlaps_with("find_abbrivation":CONCEPT);
            create("$0 $1 $2":PERS);
            create("$1 $2":LASTNAME);
            create("$2":LASTNAME)
END
```

```
/*Bsp. Anna Andra-Maria von der Flüh*/
CONCEPT "last_name"
IF
                    (("pre_name":CONCEPT AND $?:NAMEPART)
                OR ("pre_pers_double_name" AND $?:DNAMEPART)
                OR ("hyphenated_pre_pers_name" AND $?:HNAMEPART)
                OR ("initial_of_name":CONCEPT AND $?:INITIAL));
                    (("pre_name":CONCEPT AND $?:NAMEPART)
                OR ("pre_pers_double_name" AND $?:DNAMEPART)
                OR ("hyphenated_pre_pers_name" AND $?:HNAMEPART)
                OR ("initial_of_name":CONCEPT AND $?:INITIAL));
                ("last_name_part" AND $?:LASTNAMEPART);
                ("pos_last_name":CONCEPT AND $?:POSLASTNAME)
CONDITION
                NOT find("sal_pers_name":CONCEPT, -3);
/*Zusammengesetzte Geo-Namen dürfen keine Personennamen sein.*/
                NOT overlaps_with("find_hyphenated_geonames":CONCEPT);
                NOT overlaps_with("find_double_names":CONCEPT);
                NOT overlaps_with("find_triple_names":CONCEPT);
                NOT overlaps_with("find_quadruple_names":CONCEPT);
                NOT overlaps_with("find_quintuple_names":CONCEPT);
                NOT overlaps_with("find_slash_citiesI":CONCEPT);
                NOT overlaps_with("find_slash_citiesII":CONCEPT);
                NOT overlaps_with("find_slash_citiesIII":CONCEPT);
                NOT overlaps_with("find_st_names":CONCEPT);
                NOT overlaps_with("hyphenated_geonames_with_abbr":CONCEPT);
                NOT overlaps_with("double_names_with_abbr":CONCEPT);
                NOT overlaps_with("triple_names_with_abbr":CONCEPT);
                NOT overlaps_with("quadruple_names_with_abbr":CONCEPT);
                NOT overlaps_with("quintuple_names_with_abbr":CONCEPT);
                NOT overlaps_with("slash_citiesI_with_abbr":CONCEPT);
                NOT overlaps_with("slash_citiesII_with_abbr":CONCEPT);
                NOT overlaps_with("slash_citiesIII_with_abbr":CONCEPT);
                NOT overlaps_with("st_names_with_abbr":CONCEPT);
                NOT overlaps_with("sal_pers_name":CONCEPT);
                NOT overlaps_with("part_of_fa":CONCEPT);
                NOT overlaps_with("find_abbrivation":CONCEPT);
                create("$0 $1 $2 $3":PERS);
                create("$2 $3":LASTNAME);
                create("$3":LASTNAME)
END
```

```
CONCEPT "find_pre_names"
IF
                    (("pre_name":CONCEPT AND $?:NAMEPART)
                OR ("pre_pers_double_name" AND $?:DNAMEPART)
                OR ("hyphenated_pre_pers_name" AND $?:HNAMEPART))
CONDITION
                NOT find("sal_pers_name":CONCEPT, -3);
                NOT overlaps_with("full_name":CONCEPT);
                NOT overlaps_with("last_name":CONCEPT);
                NOT overlaps_with("suffix_name":CONCEPT);
                NOT overlaps_with("stop_word_list_for_single_names":CONCEPT);
                NOT overlaps_with("initial_of_name":CONCEPT);
/*Zusammengesetzte Geo-Namen dürfen keine Personennamen sein.*/
                NOT overlaps_with("find_hyphenated_geonames":CONCEPT);
                NOT overlaps_with("find_double_names":CONCEPT);
                NOT overlaps_with("find_triple_names":CONCEPT);
                NOT overlaps_with("find_quadruple_names":CONCEPT);
                NOT overlaps_with("find_quintuple_names":CONCEPT);
                NOT overlaps_with("find_slash_citiesI":CONCEPT);
                NOT overlaps_with("find_slash_citiesII":CONCEPT);
                NOT overlaps_with("find_slash_citiesIII":CONCEPT);
                NOT overlaps_with("find_st_names":CONCEPT);
                NOT overlaps_with("hyphenated_geonames_with_abbr":CONCEPT);
                NOT overlaps_with("double_names_with_abbr":CONCEPT);
                NOT overlaps_with("triple_names_with_abbr":CONCEPT);
                NOT overlaps_with("quadruple_names_with_abbr":CONCEPT);
                NOT overlaps_with("quintuple_names_with_abbr":CONCEPT);
                NOT overlaps_with("slash_citiesI_with_abbr":CONCEPT);
                NOT overlaps_with("slash_citiesII_with_abbr":CONCEPT);
                NOT overlaps_with("slash_citiesIII_with_abbr":CONCEPT);
                NOT overlaps_with("st_names_with_abbr":CONCEPT);
                NOT overlaps_with("part_of_fa":CONCEPT);
                NOT find("find_abbrivation":CONCEPT, +2);
                create("$0":PERS)
END
```

```
/*Bsp. Bush Jr.*/
CONCEPT "suffix_name"
IF
        ("pos_last_name":CONCEPT AND $?:POSLASTNAME);
        ("suffix_names":CONCEPT AND $?:LASTNAMEPART)
CONDITION
        NOT find("sal_pers_name":CONCEPT, -3);
        NOT find("initial_of_name":CONCEPT, -2);
        NOT find("pre_pers_name":CONCEPT, -2);
        NOT find("pre_name":CONCEPT, -2);
        NOT find("pre_pers_double_name":CONCEPT, -2);
        NOT find("hyphenated_pre_pers_name":CONCEPT, -2);
        NOT overlaps_with("part_of_fa":CONCEPT);
        create("$0 $1":PERS);
        create("$0 $1":LASTNAME);
        create("$0":LASTNAME)
END

/*Bsp. O'Harra Sr.*/
CONCEPT "suffix_name"
IF
        ("last_name_part":CONCEPT AND $?:LASTNAMEPART);
        ("pos_last_name":CONCEPT AND $?:POSLASTNAME);
        ("suffix_names":CONCEPT AND $?:LASTNAMEPART)
CONDITION
        NOT find("sal_pers_name":CONCEPT, -3);
        NOT find("initial_of_name":CONCEPT, -2);
        NOT find("pre_pers_name":CONCEPT, -2);
        NOT find("pre_name":CONCEPT, -2);
        NOT find("pre_pers_double_name":CONCEPT, -2);
        NOT find("hyphenated_pre_pers_name":CONCEPT, -2);
        NOT overlaps_with("part_of_fa":CONCEPT);
        create("$0 $1 $2":PERS);
        create("$0 $1 $2":LASTNAME);
        create("$0 $1":LASTNAME);
        create("$1 $2":LASTNAME)
END
```

```
CONCEPT "part_of_fa"
IF
                ("one_fa_name_follows":CONCEPT)
            OR ("one_fa_name_follows_in_quotation_marks":CONCEPT)
            OR ("two_fa_names_follow":CONCEPT)
            OR ("two_fa_names_follow_in_quotation_marks":CONCEPT)
            OR ("three_fa_names_follow":CONCEPT)
            OR ("three_fa_names_follow_in_quotation_marks":CONCEPT)
            OR ("four_fa_names_follow":CONCEPT)
            OR ("four_fa_names_follow_in_quotation_marks":CONCEPT)
            OR ("five_fa_names_follow":CONCEPT)
            OR ("five_fa_names_follow_in_quotation_marks":CONCEPT)
            OR ("six_fa_names_follow":CONCEPT)
            OR ("six_fa_names_follow_in_quotation_marks":CONCEPT)
            OR ("seven_fa_names_follow":CONCEPT)
            OR ("seven_fa_names_follow_in_quotation_marks":CONCEPT)
            OR ("one_fa_name_in_front":CONCEPT)
            OR ("two_fa_names_in_front":CONCEPT)
            OR ("three_fa_names_in_front":CONCEPT)
            OR ("four_fa_names_in_front":CONCEPT)
            OR ("five_fa_names_in_front":CONCEPT)
            OR ("six_fa_names_in_front":CONCEPT)
            OR ("seven_fa_names_in_front":CONCEPT)
END

ENDMODULE
```

Bibliographie

ACE, http://www.nist.gov/speech/tests/ace/index.htm. NIST (National Institute of Standards and Technology), Speech Group.Letztes Update am 14.07.03. Zuletzt besucht am 9.10. 2003.

ACL, http://www.aclweb.org/. ACL, Dragomir Radev und Sandra Carberry.Letztes Update am 29.9.2003. Zuletzt besucht am 8.10. 2003.

Appelt, Douglas E. und Martin, David. 1999. Named Entity Extraction from Speech: Approach and Results using the TextPro System. Abhandlung vorgestellt an *DARPA Boadcast News Workshop*, Virginia.

Baeza-Yates, Racardo und Ribeiro-Neto, Berthier. 1999. *Modern Information Retrieval*. New York: ACM-Press and Addison-Wesley.

Blanár, Vincent. 2001. *Theorie des Eigennamens. Status, Organisation und Funktionieren der Eigennamen in der Gesellschaftlichen Kommunikation.*: Germanistische Linguistik. Hildesheim, Zürich, New York: Georg Olms Verlag.

Borthwick, Andrew, Sterling, John, Agichtein, Eugen und Grisham, Ralph. 1998. Exploiting Diverse Knowledge Sources via Maximum Entropy in Named Entity Recognition, Computer Science Department, New York University: Bericht.

Borthwick, Andrew. 1999. A Maximum Entropy Approach to Named Entity Recognition, Informatik, New York University: PH. D.

Brants, Thorsten, Crocker, Matthew und Lieblang, Enrico. 2000. Mathematische Grundlagen der Computerlinguistik III. Statistische Methoden. Ms., *Computerlinguistics & Phonetics*. Saarbrücken.

Brüning, Barbara. 1996. *Über Sinn und Bedeutung von Eigennamen. Eine semantisch-erkenntnistheoretische Untersuchung.* Wien: Turia+Kant.

Bussmann, Hadumod ed. 2002. *Lexikon der Sprachwissenschaft*. Stuttgart: Alfred Kröner Verlag.

CoNLL, http://cnts.uia.ac.be/conll2002/ner/. CNTS Centrum voor Nederlandse Taal en Spraak, University of Antwerp.Letztes Update am 3.8.2003. Zuletzt besucht am 8.10 2003.

Eichler, Enst, Hilty, Gerold, Löffler, Heinrich, Steger, Hugo und Zgusta, Ladixlav. 1995. Namenforschung: ein internationales Handbuch zur Onomastik. *Handbücher zur Kommunikations- und Sprachwissenschaft* 1.

Eichler, Ernst. 1995. Namensforschung: Überblick, Geschichte, Richtungen, Institutionen. In *Namenfoschung. Ein Internationales Handbuch zur Onomastik*, eds. Ernst Eichler,

Gerold Hilty, Heinrich Löffler, Hugo Steger und Ladislav Zgusta, 1-7. Berlin: de Gruyter.

GATE, http://gate.ac.uk/. University of Sheffield, Natural Language Processing Group.Letztes Update am. Zuletzt besucht am 13.09. 2003.

Gaus, Wilhelm. 2000. *Dokumentations- und Ordnungslehre. Theorie und Praxis des Information Retrieval*. Berlin, Heidelberg, New York, Barcelona, Honkong, London, Mailand, Paris, Singapur, Tokio: Springer. [Neu aufgelegt in 3. Auflage].

Gertwol, http://www.lingsoft.fi/cgi-bin/gertwol. Lingsoft.Letztes Update am 1996. Zuletzt besucht am 13.09. 2003.

Grisham, Ralph und Sundheim, Beth. 1995. Message Understanding Conference - 6: A Brief History. Abhandlung vorgestellt an *Message Understandig Conference*, Columbia, Maryland.

Harweg, Roland. 1999. *Studien zu Eigennamen*: Bochumer Beiträge zur Semiotik. Bochum: Shaker Verlag GMBH.

HUB-4, http://www.nist.gov/speech/tests/bnr/hub4_98/hub4_98.htm. NIST / Speech Group.Letztes Update am 27.9.00. Zuletzt besucht am 29.08 2003.

IdentiFinder, BBN Technologies, http://www.bbn.com/speech/identifinder.html. BBN Technologies.Letztes Update am. Zuletzt besucht am 29.08. 2003.

IE-ER-Evaluation. 1998. Evaluationstexte englisch. In *New York Times, Associated Press, Broadcast News*.

IE-ER-Evaluation, http://www.nist.gov/speech/tests/ie-er/er_99/er_99.htm. NIST, Speech Group.Letztes Update am. Zuletzt besucht am 18.09. 2003.

Jurafsky, Daniel und Martin, James H. 2000. *Speech and Language Processings. An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recongition*: Artificial Intelligence. New Jersey: Prentice Hall.

Konradin-Verlag. 1998. Computer-Zeitung auf CD-Rom. Volltextrecherche aller Artikel der Jahrgänge 1993 bis 1996. In *Computer-Zeitung*. Leinfelden-Echterdingen.

Koß, Gerhard. 2002. *Namenforschung. Eine Einführung in die Onomastik*.vol. 34: Germanistische Arbeitshefte. Tübingen: Max Niemeyer Verlag. [Neu aufgelegt in 3. Auflage].

Krenn, Brigitte und Samuelsson, Christer. 1997. The Linguist's Guide to Statistic, Universität Saarbrücken: Guide.

Kubala, Francis, Schwartz, Richard, Stone, Rebecca und Weischedel, Ralph. 1998. Named Entity Extraction From Speech. Abhandlung vorgestellt an *DARPA Broadcast News Transcription and Understanding Workshop*, Lansdowne, VA.

Lötscher, Andreas. 1995. Der Name als lexikalische Einheit: Denotation und Konnotation. In *Namenfoschung. Ein Internationales Handbuch zur Onomastik*, eds. Ernst Eichler, Gerold Hilty, Heinrich Löffler, Hugo Steger und Ladislav Zgusta, 448-457. Berlin: de Gruyter.

Lyman, Peter und Varian, Hal R. 2000. How much Information, School of Information Management and Systems, University of California: Reportage.

Manning, Christopher D. und Schütze, Heinrich. 2002. *Foundations of statistical natzural language processing*. Cambridge (Ma): MIT Press.

Maynard, Diana, Tablan, Valentin, Ursu, Christian, Cunningham, Hamish und Wilks, Yorick. 2001. Named Entity Recognition from Diverse Text Types. Abhandlung vorgestellt an *Recent Advandes in Natural Language Processing*, Tzigov Chark, Bulgaria.

Maynard, Diana, Cunningham, Hamish, Bontcheva, Kalina und Dimitrov, Marin. 2002a. Adapting A Robust Multi-Genre NE System for Automatic Content Extraction. Abhandlung vorgestellt an *10th International Conference on Artificial Intelligence: Methodology, Stystems Application (AIMSA)*, Varna, Bulgaria.

Maynard, Diana, Tablan, Valentin, Cunnigham, Hamish, Ursu, Christian, Saggion, Horacio, Bontcheva, Kalina und Wilks, Yorick. 2002b. Architectural Elements of Language Engineering Robustness, Dept. of Computer Science, University of Sheffield: Report.

Mikeev, Andrei, Moens, Marc und Grover, Claire. 1999. Named Entity Recognition without Gazetteers. Abhandlung vorgestellt an *9th Conference of teh Eutopean Chapter of Association for Computational Linguistics (EALC)*, Bergen.

MUC-6, http://www.cs.nyu.edu/cs/faculty/grishman/muc6.html. New York University.Letztes Update am 25.4.1996. Zuletzt besucht am 8.10. 2003.

MUC, http://www.itl.nist.gov/iaui/894.02/related_projects/muc/index.html. NIST, Donna Harman.Letztes Update am 16.3.2001. Zuletzt besucht am 9.10. 2003.

NAACL'03-Workshop, http://www.sims.berkeley.edu/research/conferences/hlt-naacl03/. Andras Kornai (Metacarta Inc.), Berth Sundheim (US Navy SPAWAR Systems Center).Letztes Update am. Zuletzt besucht am 13.10. 2003.

Pazienza, Maria Teresa ed. 1997. *Information Extraction. A Mulidisciplinary Approach to an Emerging Information Technology. International Summer School, SCIE-97. Frascati, Italy, July 14-18, 1997.* vol. 1299. *Lecture Notes in Artificial Intelligence. Subseries of Lecture Notes in Computer Science*. Berlin, Heidelberg, New York, Barcelona, Budapest, Hong Kong, London, Milan, Paris, Santa Clara, Singapore, Tokyo: Springer.

Poetzsch, Eleonore. 2001. *Information Retrieval. Einführung in Grundlagen und Methoden.*vol. 5: Materialien zur Information und Dokumentation. Potsdam: Verlag für Berlin-Brandenburg. [Neu aufgelegt in 2. Auflage].

Quasthoff, Uwe, Biemann, Christian und Wolff, Christian. 2002. Named Entity Learning and Verification: Expectation Maximization in Large Corpora. Abhandlung vorgestellt an *6th Conference of Natural Language Learning (CoNLL)*, Taipei, Taiwan.

Robertson, S. E., Walkers, S., Jones, S., Hancock-Beaulieu, M. M. und Gatford, M. 1994. OKAPI at TRACK-3. Abhandlung vorgestellt an *3d Text REtrieval Conference*, Gaithersburg, Maryland.

Robinson, Patricia, Brown, Erica, Burger, John, Chinchor, Nancy, Aaron, Douthat, Ferro, Lisa und Hirschman, Lynette. 1999. Overview: Information Extraction form Broadcast News. Abhandlung vorgestellt an *DARPA Broadcast News Workshop*, Virginia.

Rössler, Marc. 2002. Using Markov Models for Named Entity recognition in German newspaper. Abhandlung vorgestellt an *14th European Summer School in Logic, Language and Information (ESSLLI)*, Trento, Italy.

Schäuble, Peter. 1997. *Multimedia Information Retrieval. Content-Based Information Retrieval from Large Text and Audio Databases.*vol. 397: Engineering and Computer Science. Boston, Dordrecht, London: Kluwer Academic Publishers.

Schäuble, Peter und Braschler, Martin. 2002. Information nutzbar machen, 13. Zürich: Eurospider.

SIGNLL, http://ilk.kub.nl/~signll/. ACL (Association for Computer Linguistics), Antal van den Bosch.Letztes Update am 3.3.2003. Zuletzt besucht am 8.10. 2003.

Singhal, Amis, Buckley, Chris und Mitra, Mandar. 1996. Pivoted Document Length Nomalization. Abhandlung vorgestellt an *ACM SIGIR (ACM Special Information Group on Information Retrieval) Conference on Researche and Development in Information Retrieval (19th)*, Zürich.

Sullivan, Dan. 2001. *Document Warehousing and Text Mining. Techniques for Improving Business Operations, Marketing, and Sales*. New York: John Wiley & Sons.

TextPro, http://www.ai.sri.com/~appelt/TextPro/.Letztes Update am 10.10.99. Zuletzt besucht am 29.08. 2003.

Thompson, Christopher und Dozier, Christopher. 1999. Name Recongnition and Retrieval Performence. In *Natural Language Information Retrieval*, ed. Tomek Strazalkowski, 384. Dordrecht: Kluwer Academic Publisher.

TnT, http://www.coli.uni-sb.de/~thorsten/tnt/. Universität des Saarlandes, Computational Linguistics, Thorsten Brants.Letztes Update am 26.1098. Zuletzt besucht am 06.09 2003.

TREC, Text REtriaval Conference, http://trec.nist.gov. NIST.Letztes Update am 02.06.03. Zuletzt besucht am 29.08 2003.

Volk, Martin und Clematide, Simon. 2001. Learn - Filter - Apply - Forget. Mixed Approaches to Named Entity Recognition. Abhandlung vorgestellt an *6th International Workshop on application of Natural Language for Information Systems (NLDB)*, Madrid, Spain.

Volk, Martin. 2002. The Automatic Resolution of Prepositional Phrase-Attachment Ambiguities in German, Institut für Informatik, University of Zurich: Habilitation.

Wimmer, Rainer. 1995. Eigennamen im Rahmen einer allgemeinen Sprach- und Zeichentheorie. In *Namenfoschung. Ein Internationales Handbuch zur Onomastik*, eds. Ernst Eichler, Gerold Hilty, Heinrich Löffler, Hugo Steger und Ladislav Zgusta, 372-379. Berlin: de Gruyter.

Yahoo, http://www.yahoo.com. Yahoo.Letztes Update am 2003. Zuletzt besucht am 13.09. 2003.

Curriculum Vitae

Personalien:

Name: Kühnlein
Vorname: Carola
E-Mail: carokue@swissonline.ch
Geburtsdatum: 24.01.1977
Geburtsort: Zürich

Ausbildung

Primarschule in Zürich-Witikon	1984 – 1987 3 Jahre
Primarschule in Küsnacht (ZH)	1987 – 1990 3 Jahre
Sekundarschule in Küsnacht (ZH)	1990 –1992 2 Jahre
Kantonsschule Küsnacht (ZH) Maturitätstypus: Lehramt	1992 – 1997 4,5 Jahre
Universität Zürich PHIL I	Herbst 1997 bis voraussichtlich Frühling 2004 6,5 Jahre

Hauptfach: Germanistik

1. Nebenfach: Computerlinguistik (Beginn WS 98/99)

2. Nebenfach: Informatik (Beginn WS 99/00)

(Vertiefungsgebiet Software Engineering)

Studentische Tätigkeiten

Sommer 2000 Hilfsassistentenjob an der Universität Zürich

3 Monate Korpusannotation

2001/2002 Hilfsassistentenjob an der Universität Zürich

Vorbereitung und Mitgestaltung eines Fortbildungsseminars des Instituts für Informatik zum Thema „Text Mining“

Ab 2000 Vorstandsmitglied des Fachvereins für Computerlinguistik

(Aufgaben: Kassier, Präsidentin)