Universität Karlsruhe (TH)
Fakultät für Informatik
Institut für Programmstrukturen und Datenorganisation (IPD)

Hauptseminar im Wintersemester 2004/2005

Text Mining:
Wissensgewinnung aus natürlichsprachigen Dokumenten

Herausgegeben von
Dr. René Witte und Jutta Mülle

Mit Beiträgen von
Markus Bestehorn
Thomas Gitzinger
Benjamin Heitmann
Thomas Kappler
Ralf Krestel
Tobias Lang
Johannes Leitner
Carsten Siegmund
Florian Wild

März 2006

Interner Bericht 2006-5

Inhaltsverzeichnis

Abbildungsverzeichnis

Vorwort

Das noch recht junge Forschungsgebiet *Text Mining* umfaßt eine Verbindung von Verfahren der Sprachverarbeitung mit Datenbank- und Informationssystemtechnologien. Es entstand aus der Beobachtung, dass ca. 85% aller Datenbankinhalte nur in unstrukturierter Form vorliegen, so dass sich die Techniken des klassischen *Data Mining* zur Wissensgewinnung nicht anwenden lassen. Beispiele für solche Daten sind Volltextdatenbanken mit Büchern, Unternehmenswebseiten, Archive mit Zeitungsartikeln oder wissenschaftlichen Publikationen, aber auch Ströme kontinuierlich auflaufender Emails oder Meldungen von Nachrichtenagenturen (Newswires).

Im Gegensatz zum *Information Retrieval* geht es beim Text Mining nicht darum, lediglich Dokumente anhand von Anfragen aufzufinden, sondern aus einem einzelnen oder einem Satz von Dokumenten neues Wissen zu gewinnen, etwa durch automatische Textzusammenfassungen, die Erkennung und Verfolgung benannter Objekte oder die Aufdeckung neuer Trends in Forschung und Industrie. Durch die ständig wachsende Zahl elektronisch verfügbarer Texte werden automatisch arbeitende Verfahren zur Bewältigung der Informationsflut immer dringender, was Text Mining zu einem sehr aktiven und auch kommerziell interessanten Forschungsgebiet macht.

Der vorliegende Bericht enthält eine Auswahl von Themen, die von Studierenden der Universität Karlsruhe im Rahmen eines Hauptseminars am IPD im Wintersemester 2004/2005 erarbeitet wurden. Sie reichen von den Grundlagen der Computerlinguistik über einzelne Algorithmen zur Sprachverarbeitung bis hin zu konkreten Anwendungen im Text Mining. Zahlreiche Literaturreferenzen zu jedem Kapitel sollen dem Leser eine weitergehende Studie der einzelnen Themen ermöglichen.

René Witte
Jutta Mülle

Karlsruhe,
im März 2006

Thomas Kappler

1 Einführung in die Computerlinguistik

Dieses Kapitel bietet einen Überlick über die Computerlinguistik mit den Themen Morphologie, Syntax und Grammatiken, kompositionelle und lexikalische Semantik. Die Gliederung entspricht dem logischen Aufbau der Linguistik von den Wörtern über die Satzstruktur zur Bedeutung.

Das Kapitel orientiert sich hauptsächlich an [JM00].

1.1 Einleitung

In diesem Kapitel wird umrissen, mit welchen Fragestellungen sich die Computerlinguistik befasst, gefolgt von einem kurzen geschichtlichen Abriss.

1.1.1 Wissen über Sprache

Das Wissen über natürliche Sprache wird in der Computerlinguistik typischerweise in sechs Kategorien eingeteilt. Mit zunehmender Abstraktion sind dies:

1. Phonetik und Phonologie
2. Morphologie
3. Syntax
4. lexikalische und kompositionelle Semantik
5. Pragmatik und Diskurs

Die *Phonetik* befasst sich mit den Lauten gesprochener Sprache und ist im Kontext des Text Mining kaum interessant.

Die *Morphologie* ist die Lehre von der Zusammensetzung und Formbildung der Wörter.

Unter dem Oberbegriff *Syntax* werden die von den Wörtern gebildeten Strukturen zusammengefasst. Hierzu zählen die Grammatiken.

Semantik ist der Überbegriff für die durch eine sprachliche Äußerung vermittelte Bedeutung. Die lexikalische Semantik befasst sich mit der Bedeutung auf Wortebene, die kompositionelle mit der Bedeutung von Sätzen oder längeren Abschnitten.

Die Themen *Pragmatik* und *Diskurs* gehen über den Rahmen dieses Kapitels hinaus und werden überblicksartig am Ende vorgestellt.

1.1.2 Geschichte der Computerlinguistik

Anfänge bis 1970 Im theoretisch-formalen Bereich wurden die Grundlagen für die Theorie endlicher Automaten gelegt. Chomsky entwickelte in seiner 1956 veröffentlichten Arbeit [Cho56] über die Modellierung von Grammatiken mit endlichen Automaten die Begriffe der *endlichen Sprache* und der *kontextfreien Grammatik* (siehe Abschnitt 1.4.1).

Klar davon getrennt entwickelten sich probabilistische und informationstheoretische Modelle. Entscheidend waren hier die von Shannon geprägten Begriffe des *störungsbehafteten Nachrichtenkanals* und der *Entropie* ([Sha48]).

Auf diesen zwei Strömungen aufbauend entwickelten sich zwei bis ca. 1970 deutlich voneinander getrennte Forschungsbereiche: Zum einen der symbolische mit den Kernbegriffen formale Sprachtheorie und erzeugende Syntax sowie speziell in der Informatik Parsing-Algorithmen und Reasoning, zum anderen der probabilistische Ansatz.

1970 bis heute Bis 1983 entwickelten sich aus diesen Ansätzen vier Paradigmen, zu denen hier nur die Kernbegriffe genannt werden:

- Das *stochastische*: Hidden Markov Modelle, Noisy Channel
- Das *logische*: Vorläufer von Prolog, funktionale Grammatiken
- *Verstehen natürlicher Sprache*: *Semantik* als neue Herausforderung
- *Diskurs-Modellierung*: Fokus, Struktur, Referenz-Auflösung

Mit den stochastischen Ansätzen wurde noch fast ausschließlich Spracherkennung betrieben.

Ab 1983 begann jedoch die Aufnahme probabilistischer Modelle in sämtliche Ansätze. Seit den 90er Jahren erlaubt die zunehmende Rechenleistung die tatsächliche Anwendung vieler bis dahin rein theoretischer Modelle auf großen Datenmengen. Einen zusätzlichen Schub erhielt das gesamte Forschungsgebiet der Computerlinguistik durch die nun bereitstehenden gewaltigen, aber unstrukturierten Datenmengen im Internet.

1.2 Morphologie

Morphologie
Morphem

Die *Morphologie* ist die Lehre von der Zusammensetzung von Wörtern aus Morphemen. *Morpheme* wiederum sind die kleinsten Einheiten mit Bedeutung in einer Sprache. Sie werden in zwei große Klassen aufgeteilt: *Stämme* und *Affixe*.

Abbildung 1.1 zeigt ein Beispiel aus dem Deutschen für eine solche Zusammensetzung nach Canoo [Can].

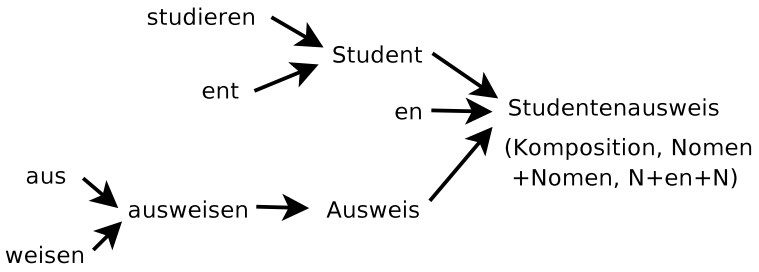


Abbildung 1.1: Die Zusammensetzung von „Studentenausweis“ aus Morphemen

1.2.1 Stemming und Lemmatisierung, Porter Stemmer

Stemming bezeichnet das Zurückführen eines Wortes auf seinen Wortstamm. Der Begriff „Wortstamm“ ist hier rein formal definiert, der beim Stemming entstehende Term muss kein tatsächliches Wort der Sprache sein. Wie der erzeugte Wortstamm konkret aussieht, hängt vom eingesetzten Algorithmus ab. Denkbar wäre zum Beispiel eine Reduzierung von „lachte“ auf „lach“. Beim Information Retrieval werden solche Verfahren verwendet, da es hier erwünscht ist, dass alle Formen eines Wortes auf denselben Stamm zurückgeführt werden, so dass für eine Anfrage möglichst viele Treffer gefunden werden können. Stemming

Im Unterschied dazu wird bei der *Lemmatisierung* das Wort auf eine in der Sprache vorhandene Grundform zurückgeführt, wie sie in Lexika verwendet wird, das sogenannte *Lemma*. Das Lemma des obigen Beispiels „lachte“ wäre „lachen“. Lemmatisierung

Die Begriffe „Lemma“ und „Stamm“ werden in der Literatur häufig nicht klar abgegrenzt.

Einer der bekanntesten Stemmer für Englisch ist der *Porter Stemmer*, den Martin F. Porter 1980 erstmals vorstellte ([Por97], [Por]). Er basiert auf kaskadierten, das heißt hintereinander ausgeführten Umschreiberegeln, zum Beispiel:

- `ATIONAL` → `ATE` (Beispiel: „relational“ → „relate“)
- `ING` → ϵ, wenn der Stamm einen Vokal enthält (Beispiel: „motoring“ → „motor“).

Lemmatisierung im Deutschen Im Englischen ist die Lemmatisierung im Wesentlichen das Streichen von Affixen unter Beachtung orthographischer Regeln. Im Deutschen wie auch in den meisten anderen Sprachen ist sie erheblich schwieriger, da die morphologischen Regeln komplexer sind und es mehr Ausnahmen gibt. Beispiele dafür sind:

- „Studenten“ → „Student“ ist richtig, „Enten“ → „Ent“ aber nicht.
- „Häuser“ → „Haus“: Der Wortstamm selbst verändert sich.

Daher wird normalerweise mit lexikon-basierten Ansätzen anstatt Umschreiberegeln gearbeitet. Es gibt zwei Varianten solcher Lexika. Bei der ersten besteht das Lexikon aus einer Liste aller möglichen Wortformen der Sprache. So kann in vielen Fällen ohne Anwendung von Regeln der korrekte Stamm zu einer konkreten Wortform einfach

nachgeschlagen werden. Es gibt jedoch auch gleiche Wortformen, die aus verschiedenen Stämmen gebildet werden, so dass zusätzliche Hilfsmittel benötigt werden. Der korrekte Stamm von „Buchten“ zum Beispiel kann je nach Kontext „die Bucht“ (am Meer, hier im Plural) oder „buchen“ (Verb) sein. Ein weiterer Nachteil dieses Ansatzes ist, dass das Lexikon sehr umfangreich wird.

Die andere Variante ist eine Kombination aus Regeln und einem Lexikon: Die Einträge des Lexikons sind Stammformen mit den zugehörigen Regeln, durch die aus der Stammform die möglichen Formen gebildet werden.

1.3 Syntax: Wortarten und Konstituenten

Im Bereich der formalen Beziehungen zwischen Wörtern werden vor allem untersucht:

- Klassen von Wörtern: *Wortarten*, engl. *part-of-speech (POS)*,
- Gruppierung in *Wortgruppen* oder *Phrasen*,
- Abhängigkeiten zwischen den Wörtern und Phrasen eines Satzes.

Abbildung 1.2 zeigt einen Beispielsatz, bei dem zu jedem Wort die Wortart bestimmt wurde (unterste Zeile, in Großbuchstaben) und die Wörter in Phrasen zusammengefasst wurden, was durch die Baumstruktur angedeutet wird. In den nächsten Abschnitten werden die erwähnten Begriffe definiert und näher erläutert.

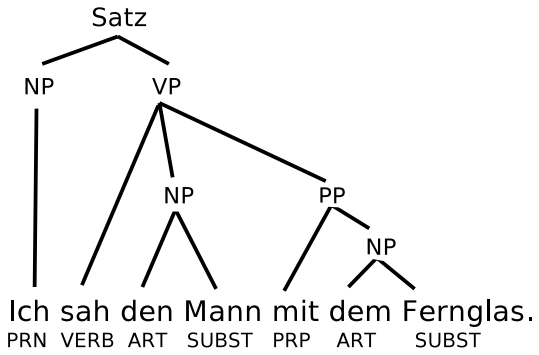


Abbildung 1.2: Ein mit den Wortarten annotierter Beispielsatz

1.3.1 Wortarten und Wortartbestimmung

POS tagging

Das Bestimmen von Wortarten wird im englischen als *part-of-speech tagging (POS tagging)* bezeichnet.

Die Wortart eines Wortes hat erheblichen Einfluß darauf, welche Wörter im Satz vorausgehen und folgen können. Die Kenntnis der Wortart hilft also bei der syntaktischen Analyse. Darüber hinaus ist das Wissen um die Wortarten beim Stemming hilfreich, da die Wortart bestimmt, welche Affixe für einen bestimmten Wortstamm möglich oder nicht möglich sind.

Wortartbestimmung ist mit formalen Ansätzen nicht immer verläßlich möglich, zum Beispiel beim Antreffen neuer Wörter oder bei sprachlichen Fehlern im vorliegenden Text. Sie sollte beim Text Mining daran aber nicht scheitern, da sonst keine linguistische Analyse durchgeführt werden kann. Darum werden die entsprechenden Systeme heute durch statistische Verfahren unterstützt.

Zur Klärung der Begriffsvielfalt ist zu erwähnen, dass folgende in der Literatur verwendeten englischen Bezeichnungen für „Wortart“ dasselbe bedeuten: part-of-speech, POS, word class, morphological class sowie lexical tag. Für die Wortartbestimmung werden POS tagging und tagging oft synonym verwendet.

Welche Wortarten gibt es nun? Die klassische Linguistik kennt acht Wortarten, die schriftlich zuerst von Dionysius Thrax von Alexandrien um 100 vor Christus genannt werden: Wortarten

- Nomen
- Verb
- Pronomen
- Präposition
- Adjektiv
- Adverb
- Konjunktion
- Artikel

Die Granularität dieser Unterscheidung ist jedoch für praktische Anwendungen der Computerlinguistik nicht fein genug. Für die syntaktische Analyse macht es beispielsweise einen deutlichen Unterschied, ob ein Verb im Aktiv oder Passiv vorkommt. Darum werden die oben genannten Wortarten noch einmal stark unterteilt, auch wenn es sich dann eventuell nicht mehr um eigentliche Wortarten im linguistischen Sinne handelt. Die drei in der Computerlinguistik meist eingesetzten Listen, sogenannte *Tagsets*, sind *Penn Treebank* ([MSM93], [oP]) mit 45, der *Brown corpus* [FK79] mit 87 und das vom CLAWS-Tagger [Uni] verwendete *C7 tagset* mit 146 Einträgen. Tabelle 1.1 zeigt einen Auszug aus dem Penn Treebank Tagset mit 45 Tags.

Definierend für eine Wortart ist nie semantische Kohärenz wie zum Beispiel die Tatsache, dass Substantive meist physische Gegenstände bezeichnen, sondern morphologische und syntaktische Eigenschaften, zum Beispiel ob Mehrzahlbildung möglich ist.

Tag	Beschreibung	Beispiel
CC	Conjunction	and, but
CD	Cardinal number	one, two, three
EX	Existential ‘there’	there
FW	Foreign word	mea culpa
IN	Preposition/sub-conj	of, in, by
JJ	Adjective	yellow
JJR	Adjective, comparative	bigger
JJS	Adjective, superlative	biggest

Tabelle 1.1: Ein Auszug aus dem Penn Treebank Tagset

In Kapitel 4 werden Algorithmen zur Wortartbestimmung vorgestellt.

1.3.2 Konstituenten

Konstituent
Chunking

Eine Gruppe von Wörtern kann sich in einem Satz grammatikalisch wie ein Wort verhalten, zum Beispiel „das alte Haus“. Eine solche Phrase heißt *Konstituent*, englisch *constituent*. Ihre Erkennung wird als *Chunking* bezeichnet.

Es gibt eine Vielzahl unterscheidbarer Konstituenten, von denen hier nur die wichtigsten vorgestellt werden können.

Nominalphrase

Die *Nominalphrase* (*NP*, englisch *Noun Phrase*) besteht aus einem Substantiv, das als *Kopf* (englisch *Head*) der NP bezeichnet wird, und sogenannten *Modifiern* und *Determinern*. Modifier ändern die Bedeutung des Substantivs im jeweiligen Zusammenhang oder spezifizieren sie. In der Regel sind Modifier Adjektive, im Englischen beispielsweise kann jedoch auch jedes Substantiv in der Adjektiv-Position verwendet werden. Im Beispiel

„cat – cat food – cat food can – …“

ist „cat“ in der ersten Nominalphrase, die nur aus einem Substantiv besteht, der Head und wird danach zum Modifier, der „food“ näher beschreibt. Determiner können mit der Wortart „Artikel“ identifiziert werden.

Verbalphrase

Die *Verbalphrase* (*VP*, engl. *Verb Phrase*) besteht analog zur Nominalphrase aus einem Verb, dem Head der Verbalphrase, und einer Reihe möglicher anderer Konstituenten. Diese können zum Beispiel eine Nominalphrase sein:

„Ich möchte einen Tee.“

„möchte“ ist der Head der Verbalphrase, „einen Tee“ ist eine zu dieser Verbalphrase gehörende Nominalphrase aus einem Determiner und dem Head „Tee“.

Es sind auch deutlich komplexere Konstituenten möglich, beispielsweise ein ganzer Nebensatz, der dann als *sentential complement* bezeichnet wird:

„Du sagtest, dass mehrere Flüge angeboten würden.“

„sagtest“ ist hier der Head der Verbalphrase, der folgende Nebensatz ihr Konstituent.

Eine zweite Verbalphrase ist ebenfalls als Konstituent einer Verbalphrase möglich:

„Ich möchte von Karlsruhe nach Berlin fliegen.“

In diesem Beispiel ist „möchte“ der Head der ersten Verbalphrase und „fliegen“ der Head der zweiten Verbalphrase, die ein Konstituent der ersten ist.

1.4 Syntax: Grammatiken und Sprachen

Nachdem die Wörter und ihre Zusammensetzung zu Phrasen untersucht wurden, ist der nächste Schritt zu einem formalen Modell menschlicher Sprache die Untersuchung ganzer Sätze.

1.4.1 Formale Grammatiken und die Chomsky-Hierarchie

Grammatik

Der Aufbau eines Satzes aus Phrasen folgt bestimmten Regeln, die unter dem Begriff *Grammatik* zusammengefasst werden. Abstrahiert man von der menschlichen Sprache, kann man Grammatiken allgemeiner definieren. Chomsky formalisierte den Begriff in [Cho56] und [Cho59].

Formale Grammatiken bestehen aus einer Menge von Ableitungsregeln, die Ersetzungen von Variablen (die sogenannten *Nichtterminale*) durch Symbole (*Terminale*) oder andere Variablen definieren. Allgemeiner können beliebige, also auch Terminale enthaltende Ausdrücke durch andere ersetzt werden. Mit einer bestimmten, als *Startsymbol* bezeichneten Variable beginnend, werden durch sukzessive Anwendung dieser Regeln Variablen durch Terminale ersetzt. Wenn der entstehende Ausdruck nur noch aus Terminalen besteht, wurde ein gültiges *Wort* der vorliegenden Sprache erzeugt.

Die Ableitungsregeln werden als

linke Seite → rechte Seite

notiert, wobei der Ausdruck auf der „linken Seite“ durch den auf der „rechten Seite“ ersetzt wird.

Chomsky charakterisierte in [Cho59] Grammatiken nach ihrer Komplexität und teilte sie so in verschiedene Klassen ein. Unter Komplexität wird dabei die Vielfalt der mit den Regeln erzeugbaren Wörter beziehungsweise Sätze verstanden. Einschränkungen der erlaubten Ableitungsregeln verringern die Komplexität einer Grammatik.

Dabei gelten folgende grundlegende Zusammenhänge: Je weniger Einschränkungen eine Klasse von Grammatiken hat

- umso mehr Sprachen kann sie erzeugen, aber
- umso komplexer werden formale Modellierung und Implementierung.

Die Grammatikklassen werden mit der jeweils hinzukommenden Einschränkung in ihrer Inklusionshierarchie, der *Chomsky-Hierarchie*, in Abbildung 1.3 gezeigt, aufsteigend nach Eingeschränktheit geordnet. Dabei inkludiert jede Klasse die spezifischeren Klassen.

Chomsky-Hierarchie

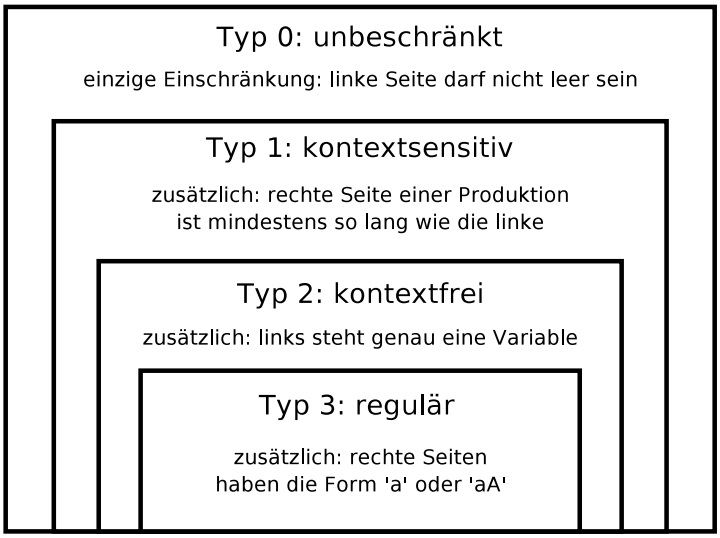


Abbildung 1.3: Die Chomsky-Hierarchie mit Inklusionen

1.4.2 Reguläre Ausdrücke

Reguläre Ausdrücke (*regular expressions*) sind Ausdrücke, die Klassen von Zeichenfolgen angeben. Formal handelt es sich um eine algebraische Notation für Mengen von Zeichenfolgen.

Reguläre Ausdrücke

Mit regulären Ausdrücken ist es beispielsweise möglich, ein Wort unabhängig von einem eventuellen Mehrzahl-s auszudrücken, oder nach allen Wörtern mit einem bestimmten Teilwort wie beispielsweise dem Wortstamm zu suchen. Ihre praktische Bedeutung für das Verarbeiten von Texten ist sehr hoch:

- Sie sind leicht zu modellieren.
- Das Durchsuchen von Texten anhand regulärer Ausdrücke läßt sich effizient implementieren.
- Sie decken zwar nicht alle, aber einen Großteil der interessanten Muster ab.

Reguläre Ausdrücke werden unmittelbar durch Grammatiken des Chomsky-Typs 3 (reguläre Grammatiken) definiert, und auch umgekehrt existiert zu jedem regulären Ausdruck eine Typ-3-Grammatik. Gleiches gilt für endliche Automaten. Diese drei formalen Modelle sind also in ihrer Ausdrucksmächtigkeit äquivalent.

Äquivalente Modelle

In Tabelle 1.2 sind einige Elemente regulärer Ausdrücke mit einer Erklärung und der durch diese Ausdrücke charakterisierten Zeichenmenge (zum Teil auszugsweise) angegeben.

Beispiel	Erklärung	charakterisierte Zeichenmenge
/Linguistik/	Zeichenfolge	{„*Linguistik*“}
/[lL]inguistik/	„[]“: „oder“	{„Computer*linguistik*“, ...}
/[a-z]/	„-“: Bereiche	Kleinbuchstaben
/[^a-z]/	„^“: „nicht“	nicht Kleinbuchstaben
/H.llo!/	„.“: bel. Zeichen	{„*Hallo!*“, „*Hello!*“, ...}
/.*glas/	„*“: bel. Anzahl	{„*Schnapsglas*“, „*Bierglas*“, ...}

Tabelle 1.2: Beispiele für reguläre Ausdrücke

Finite State Transducers Ein *Finite State Transducer (FST)* ist ein endlicher Automat, der äquivalent zu einem Mealy-Automaten ist. Im Unterschied zu einfachen endlichen Automaten (engl. *finite state automata, FSAs*) arbeitet er auf Paaren von Zeichenfolgen.

FST

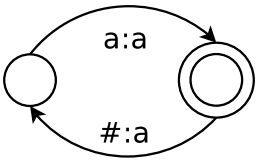


Abbildung 1.4: Ein einfacher Finite State Transducer

Die Interpretation eines FST ist auf vier Arten möglich, hier jeweils mit Anwendung auf das Beispiel in Abbildung 1.4:

1. Als Erkenner: akzeptiert, wenn das zweite Band doppelt so viele „a“s hat wie das erste.
2. Als Erzeuger: schreibt doppelt so viele „a“s auf das zweite Band wie auf das erste.
3. Als Übersetzer zwischen „a*“ und „(aa)*“.
4. Als Übersetzer zwischen „(aa)*“ und „a*“.

Beispiel für FSTs: NP-Chunking FSTs werden in der Computerlinguistik für verschiedene Aufgaben eingesetzt. Hier soll ihre Anwendung am Beispiel des NP-Chunking, also des Erkennens von Nominalphrasen in einem Satz (siehe 1.3.2), gezeigt werden.

Als Eingabe dienen die bereits bestimmten Wortarten. FSTs werden folgendermaßen eingesetzt:

- Zuerst eine Reihe von FSTs, die die Wortarten anhand ihrer Abfolge als Elemente einer NP einordnen (Modifier, Head, usw.).
- Dann ein FST, der diese Einordnungen als Eingabe erhält und komplette Nominalphrasen erkennt.

Die FSTs zur Einordnung der Wortarten als NP-Elemente werden anhand einer regulären Grammatik modelliert, die die drei NP-Elemente Head, Modifier und Determiner auf die jeweils möglichen Wortarten abbildet:

```
DET  →  DT
MOD  →  ADJ | [...|...]
HEAD →  NN | PRN
```

Die letzte Ableitungsregel sagt zum Beispiel aus, dass der Head entweder ein Substantiv (`NN`) oder ein (Personal-)Pronomen (`PRN`) ist.

Der danach geschaltete FST erkennt auf gleiche Weise anhand des regulären Ausdrucks

```
(DET)? (MOD)* (HEAD)
```

eine vollständige Nominalphrase: optional ein Determiner (`DET`), dann eine beliebige Zahl von Modifiern (`MOD`) und schließlich (zwingend) ein Head (`HEAD`). Ein komplettes Beispiel für den beschriebenen Prozess:

Eingabe:	*a*	*cat*	*food*	*can*
1. Wortarten:	*a*	*cat*	*food*	*can*
	DT	NN	NN	NN
2. NP-Elemente:	*a*	*cat*	*food*	*can*
	DT	NN	NN	NN
	DET	MOD	MOD	HEAD

3. Erkennung von „DET-MOD-MOD-HEAD“ als gültige Nominalphrase

Bei der Suche nach Nominalphrasen ist ein Vorgriff auf die nächsten Wortarten in der Eingabe nötig, da beispielsweise im Englischen wie im obigen Fall erst das letzte Substantiv der Head ist, die vorigen sind Modifier. Hätte das System zu einem

bestimmten Zeitpunkt „cat“ als Eingabe vorliegen und würde die Wortart des nachfolgenden Wortes nicht kennen, wäre nicht entscheidbar, ob „cat“ der Head einer NP oder ein Modifier wäre.

Falls das nachfolgende Wort ein Substantiv ist, ist auch mit Kenntnis dieser Tatsache nicht immer eine Entscheidung bezüglich einer Einteilung in Nominalphrasen möglich, wie folgendes Beispiel zeigt:

„In February, George Bush visited Germany“

Die Erkennung von „In February George Bush“ als (längstmögliche) Nominalphrase analog zum obigen Beispiel wäre hier falsch, da „February“ und „George Bush“ zwei aufeinanderfolgende Nominalphrasen sind. Die korrekte Erkennung solcher Ausdrücke ist nur mit Hilfe zusätzlicher Information über den vorliegenden Satz möglich.

Grenzen regulärer Grammatiken Reguläre Grammatiken sind aufgrund ihrer restriktiven Regeln und ihrer Äquivalenz zu Automaten und regulären Ausdrücken leicht zu handhaben, sind aber für die Modellierung natürlicher Sprache oft zu restriktiv.

Nicht modellieren kann man zum Beispiel eingebettete Relativsätze im Deutschen: „Der Mann, der die Frau, die das Kind, das die Katze füttert, sieht, liebt, schläft.“

Darum müssen oft auch ausdrucksmächtigere Grammatiken, also solche geringeren Typs in der Chomsky-Hierarchie, eingesetzt werden, insbesondere kontexfreie Grammatiken.

1.4.3 Syntaktisches Parsen

Syntaktisches Parsen oder *Syntaxanalyse* bezeichnet die Zuweisung einer grammatikalischen Struktur an eine Eingabe. Diese Struktur wird gewöhnlich als Baum dargestellt, dessen Blätter aneinandergereiht den analysierten Satz bilden. Dieser Baum wird als *Syntaxbaum*, engl. *parsetree* bezeichnet. Die Abbildungen 1.5 und 1.6 zeigen solche Syntaxbäume.

Syntaxanalyse

Syntaktisches Parsen und Mehrdeutigkeit Mehrdeutigkeit bedeutet in diesem Zusammenhang, dass für einen Satz mehrere grammatikalisch korrekte Syntaxbäume aufgebaut werden können. Je nach Kontext ist entweder nur einer inhaltlich korrekt, oder die Mehrdeutigkeit ist gewollt. Ein Beispiel zeigen die Abbildungen 1.5 und 1.6. Ohne Wissen um den inhaltlichen Zusammenhang können wir nicht entscheiden, ob hier ein Mann mit Hilfe eines Fernglases gesehen wird (Abbildung 1.5) oder ob ein Mann, der ein Fernglas bei sich hat (Abbildung 1.6), gesehen wird.

Das hier vorliegende durch Mehrdeutigkeit verursachte Problem wird als *PP (prepositional phrase)-attachment* bezeichnet, da das Problem darin besteht, die Präpositionalphrase (hier „mit dem Fernglas“) der richtigen, im Syntaxbaum höher gelegenen Phrase zuzuordnen. Auch andere Phrasen als Präpositionalphrasen können Gegenstand einer solchen sogenannten *attachment ambiguity* sein. Eine weitere Art von Mehrdeutigkeit ist die *coordination ambiguity*, bei der das Problem in der richtigen Klammerung von durch „und“ verbundenen Satzteilen besteht: „Alte (Frauen und Männer)“ oder „(Alte Frauen) und Männer“. Schließlich gibt es noch die *noun-phrase bracketing ambiguity*: sind in „booking Lufthansa flights“ Flüge der Lufthansa oder Flüge für die Lufthansa gemeint? Im Deutschen tritt diese Mehrdeutigkeit kaum auf, da Substantive, die zusammen einen Ausdruck bilden, gewöhnlich zusammen geschrieben werden („Lufthansa-Flüge“).

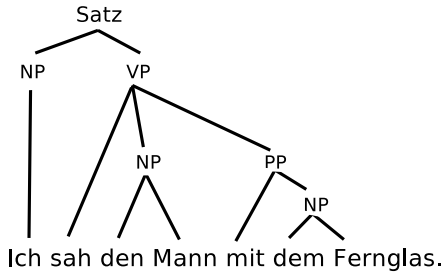


Abbildung 1.5: Ein möglicher Syntaxbaum für den Beispielsatz

Parsing-Algorithmen Die Algorithmen zum syntaktischen Parsen lassen sich in drei Typen aufteilen:

top-down Die Wurzel des Syntaxbaums wird als Startsymbol interpretiert, Ziel ist es, einen Pfad zu den Wörtern finden.

bottom-up Ziel ist das Finden eines Pfades von den Wörtern zu einer gemeinsamen Wurzel.

chart-basiert Ein Ansatz, der Techniken des dynamischen Programmierens verwendet.

Die top-down- und bottom-up-Ansätze haben mehrere Probleme: Sie sind ineffizient, da dieselben Teilbäume oft mehrfach durchgegangen werden, und Rekursion führt möglicherweise zu Endlosschleifen. Ihre Laufzeit ist mit $O(k^n)$ für eine Eingabe mit n Wörtern exponentiell.

Der chart-basierte Ansatz weist diese Probleme nicht auf und wird darum in der Praxis überwiegend eingesetzt. Dabei wird für jeden potentiellen Konstituenten der Eingabe eine Tabelle angelegt, in der der Parser die möglichen von diesem Konstituenten ausgehenden Teilbäume eines Syntaxbaums speichert. Am Ende wird versucht, die gespeicherten Teilbäume zu einem einzigen Syntaxbaum zusammenzufügen, der den Eingabesatz aufspannt. So wird zum einen mehrfaches Parsen verhindert, zum anderen kann zur Vermeidung von Schleifen nachgeschaut werden, ob der aktuelle Teilbaum schon einmal aufgebaut wurde. Der wichtigste Vertreter ist der *Earley*-Algorithmus [Ear70], der die Laufzeiten $O(n)$ für LR(k)-Grammatiken (nach Knuth), $O(n^2)$ für nicht mehrdeutige nicht-LR(k)-Grammatiken und $O(n^3)$ sonst besitzt.

chart-basiertes Parsen

Earley

1.5 Semantik

Das Ziel in der Computerlinguistik ist letztlich immer das inhaltliche Verständnis menschlicher Sprache. Morphologie und Syntax sind Hilfsmittel, die es ermöglichen, aus der formalen Struktur von Sätzen inhaltliche Aussagen abzuleiten.

Formaler ist das Ziel bei dieser Aufgabe das Verbinden linguistischer Elemente mit nicht-linguistischem Wissen „über die Welt“. Dazu muss die Bedeutung einer Äußerung in einer formalen Struktur repräsentiert werden. Diese Bedeutungsrepräsentation (engl. *meaning representation*.) muss der linguistischen Struktur zugeordnet wer-

semantische Analyse

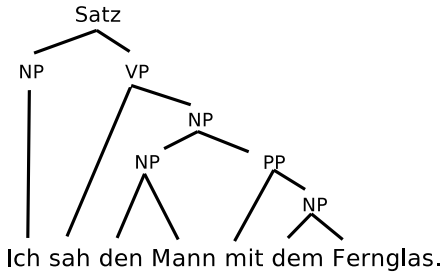


Abbildung 1.6: Ein anderer möglicher Syntaxbaum für denselben Satz

den. Dieser Prozess heißt *semantische Analyse*.

Die oben erwähnte Bedeutungsrepräsentation sollte drei Kriterien erfüllen:

1. Verifizierbarkeit
2. Eindeutigkeit
3. Kanonische Form

Verifizierbarkeit (engl. *verifiability*) bezeichnet die Möglichkeit des Vergleichs einer Aussage über einen Sachverhalt mit der Modellierung dieses Sachverhalts in der Wissensbasis. Die einfachste Möglichkeit zur Umsetzung von Verifizierbarkeit ist der Vergleich von in Anfragen enthaltenen Aussagen mit Aussagen in der Wissensbasis. Beispielsweise bezieht sich die Frage „Bietet dieses Lokal vegetarisches Essen an?“ und die Feststellung „Dieses Lokal bietet vegetarische Gerichte an“ auf denselben Sachverhalt. Diese Übereinstimmung soll erkannt und für Anfragen an die Wissensbasis genutzt werden können.

Eindeutigkeit bedeutet, dass eine Repräsentation eine einzige, eindeutige Interpretation besitzt. Mehrdeutigkeiten der natürlichen Sprache wie in dem bereits bekannten Beispiel „Ich sah den Mann mit dem Fernglas“ (siehe Abschnitt 1.4.3) müssen also aufgelöst werden, so dass aus der Repräsentation ersichtlich ist, welche Bedeutung hier gemeint ist.

Unterschiedliche Repräsentationen der gleichen Aussage werden durch die semantische Analyse in dieselbe Form, die sogenannte *kanonische Form* überführt. Zum Beispiel sollen die Sätze „In diesem Lokal gibt es vegetarisches Essen“ und „Dieses Lokal bietet vegetarische Gerichte an“ die gleiche kanonische Form haben.

1.5.1 Ein klassischer Ansatz: Prädikatenlogik

Der Idealfall einer Wissensbasis wäre, dass aus den vorhandenen, bekannten Fakten durch automatisches logisches Schließen auch die daraus ableitbaren Fakten erkannt würden. Fragen könnten dann mittels entsprechender Schlußfolgerungstechniken beantwortet werden. Eine Technik, die automatisches Schließen und Beweisen ermöglicht, ist die Prädikatenlogik.

Prädikatenlogik

Die *Prädikatenlogik* (engl. *first-order predicate logic, FOPC*) besteht aus drei aufeinander aufbauenden Teilen:

- Atomare Elemente

- Prädikate zur Modellierung von Beziehungen zwischen Elementen
- Logische Operatoren und Quantoren zur Verbindung von Prädikaten und Modellierung von Anfragen

Die atomaren Elemente sind Zeiger auf Objekte und Variablen, zum Beispiel `einLokal`. Aus den einzelnen Elementen können komplexere Ausdrücke, sogenannte *Terme* zusammengesetzt werden. Beziehungen zwischen Elementen werden mit Prädikaten ausgedrückt, zum Beispiel

```
bietetAn(einLokal, vegetarischesEssen).
```

Prädikate können wiederum mit den logischen Operatoren $\Rightarrow, \wedge, \vee$ zu Aussagen verbunden werden. Mit Variablen und den Quantoren $\forall$ und $\exists$ können schließlich Anfragen formuliert werden. Mit Hilfe von obigem Prädikat könnte man beispielsweise nach einem vegetarischen Restaurant suchen:

```
istLokal(x) ∧ bietetAn(x, vegetarischesEssen).
```

Im Idealfall kann man nun durch automatisches Beweisen Schlüsse ziehen. Dieser Vorgang heißt *Inferenz*. Der einfachste Fall ist der Modus Ponens: ist α gegeben und $\alpha \Rightarrow \beta$ bekannt, so kann daraus β gefolgert werden. α und β sind dabei Formeln der Prädikatenlogik. Inferenz

Schwierigkeiten beim Einsatz der Prädikatenlogik Der praktische Einsatz der Prädikatenlogik weist mehrere Probleme auf.

Zunächst ist es schwierig, die Prädikate zu gewinnen. Die gesamte Vorverarbeitungskette von Wortartbestimmung und syntaktischer Analyse muss dazu fehlerfrei funktionieren, da ihre Ergebnisse relevant für die Einstufung von Satzelementen als Prädikate sind.

Um ein inhaltlich ausreichend großes Gebiet abzudecken, ist eine große Zahl von Prädikaten erforderlich. Da die Algorithmen zum Schließen und Beweisen nicht sehr performant sind, ergibt sich in der Praxis oft eine zu geringe Verarbeitungsgeschwindigkeit.

Die Prädikatenlogik kann als „scharfe Logik“ nur schwer mit unscharfen Informationen wie geäußerten Vermutungen oder Meinungen umgehen.

Schließlich ist es fraglich, auf welcher Abstraktionsebene man die Kanonisierung der Prädikate durchführen sollte. Eine solche Entscheidung wäre beispielsweise, ob man zwischen Restaurants und Kneipen unterscheiden sollte oder sie gemeinsam als Lokale behandeln sollte. Wird die Abstraktionsebene zu grob gewählt, geht wertvolles Wissen verloren, wird sie zu fein gewählt, ist die Einstufung eines in einem Text gefundenen Begriffes als Prädikat sehr schwierig und fehleranfällig.

Darum wird die Prädikatenlogik oft nur unterstützend neben anderen Verfahren, meist statistischen, eingesetzt.

Im *Semantic Web* [Conb] wird nicht die Prädikatenlogik erster Ordnung eingesetzt, sondern die sogenannte Description Logic (DL) [B+02], meist in Kombination mit Ontologien (zum Beispiel in OWL-DL [Cona]), auf der im Gegensatz zur Prädikatenlogik automatisches Beweisen immer—in endlicher Zeit—möglich ist. Das Problem der Wissensakquisition bleibt jedoch bestehen.

1.5.2 Prädikat-Argument-Strukturen

Prädikat-Argument-Strukturen

Prädikat-Argument-Strukturen (engl. *predicate-argument structures*) legen Beziehungen zwischen den Konstituenten eines Satzes fest. Den formalen Rahmen dieser Beziehungen bildet die Grammatik.

Diese Strukturen sind der Kern der semantischen Struktur menschlicher Sprache, indem sie die Verbindung zwischen Syntax und Semantik herstellen. Dies geschieht auf zwei Ebenen, zur näheren Erläuterung sei auf das folgende Beispiel verwiesen.

Linking

Zum einen nehmen bestimmte Konstituenten an bestimmten Stellen im Satz auf der inhaltlichen Ebene bestimmte Rollen ein. Die Analyse solcher Zusammenhänge bezeichnet man als *Linking*. Besonders für die Informationsextraktion im Rahmen des Information Retrieval oder Text Mining ist Linking interessant, weil nach der Erkennung von semantischem Subjekt und Objekt Fragen der Art „Wer hat was mit welchem Objekt getan?“ beantwortet werden können.

Zum anderen existieren *semantische Einschränkungen* (englisch *semantic restrictions*) für die Argumente eines Ausdrucks. Hier kann die semantische Analyse Hilfestellung für die syntaktische Analyse leisten.

Beispiel Im Satz „I like Chinese food“ ist das Verb „like“ das Prädikat, „I“ und „Chinese food“ sind Argumente. Der Satz hat bezüglich der Argumente der Prädikate einen syntaktischen Rahmen, den sogenannten *syntactic argument frame*: „NP like NP“ (NP: Nominalphrase, siehe Kapitel 1.3.2). Durch den Rahmen werden folgende Fakten festgelegt:

- Das Prädikat hat zwei Argumente.
- Beide müssen Nominalphrasen sein.
- Die erste Nominalphrase kommt vor dem Verb, ihre semantische *Rolle* ist die des Subjekts: *Linking*.
- Nicht jedes Objekt kann etwas mögen: eine *semantische Einschränkung*.
- Die zweite Nominalphrase kommt nach dem Verb, ihre Rolle ist die des Objekts (Linking).

Die formale Notation dafür ist schließlich `like(I, Chinese food)`.

1.5.3 Lexikalische Semantik

Die semantische Analyse auf Satzebene scheitert oft an

- falschen Syntaxbäumen, die zu falschen Prädikat-Argument-Strukturen führen,
- unvollständigen Sätzen,
- mangelnder Information über syntaktische Rahmen und semantische Einschränkungen.

lexikalische Semantik

Daher ist in der Praxis die Semantik auf Wortebene, die *lexikalische Semantik*, wichtiger. Ihre Analyse ist robuster, da die Extraktion von lokaler Information oft auch aus insgesamt unverständlichen Sätzen möglich ist. Umgekehrt hilft die semantische Einordnung eines Wortes bei der Syntaxanalyse, da sie die Möglichkeiten für vorausgehende und folgende Wörter einschränkt.

The verb "bank" has 8 senses in WordNet.

1. bank – (tip laterally; "the pilot had to bank the aircraft")

2. bank – (enclose with a bank; "bank roads")

3. bank – (do business with a bank or keep an account at a bank; "Where do you bank in this town?")

[...]

The noun "bank" has 10 senses in WordNet.

1. depository financial institution, bank, banking concern, banking company – (a financial institution that accepts deposits and channels the money into lending activities; "he cashed a check at the bank"; [...]

[...]

Abbildung 1.7: Ein Auszug aus WordNet

Die zentralen Begriffe der lexikalischen Semantik sind das Lexem und das Lexikon.

Als *Lexem* bezeichnet man eine Kombination aus Form (orthographisch und phonologisch) und Bedeutung auf Wortebene, sozusagen ein „lexikalisches Morphem“. Diese in der Computerlinguistik gebräuchliche Definition weicht von der in der klassischen Linguistik üblichen ab, wonach der Wortstamm als Lexem bezeichnet wird. Der Bedeutungs-Teil eines Lexems heißt *Sinn* (Sense). Lexem

Darauf aufbauend ist ein *Lexikon* eine Liste von Lexemen. Lexikon

Als Beispiel eines Lexikons zeigt Abbildung 1.7 einen Auszug aus WordNet, welches im nächsten Abschnitt vorgestellt wird.

Dieser Lexikon-Auszug geht noch nicht über ein klassisches Wörterbuch hinaus. Entscheidend ist, dass die Einträge des Lexikons zueinander in Beziehung gesetzt werden, um so inhaltliches Wissen folgern zu können.

Beziehungen zwischen Lexemen Zwischen Lexemen existieren eine Vielzahl von unterscheidbaren Beziehungen. Hier können nur die wichtigsten beispielhaft vorgestellt werden.

Homonymie Wörter haben dieselbe Form, aber unterschiedliche Bedeutungen, zum Beispiel „Bank“.

Polysemie Unterschiedliche, aber verwandte oder auf einen gemeinsamen Ursprung zurückzuführende Bedeutungen: „Horn“ als Berg, Instrument oder Gebäck, wobei der Name jeweils von der Form stammt.

Synonymie Verschiedene Lexeme mit gleicher Bedeutung: „Streichholz“, „Zündholz“.

Antonymie Lexeme mit gegensätzlicher Bedeutung: „groß“, „klein“.

Hyponymie Klassen- und Unterklassenbildung: „Auto“ ↔ „Fahrzeug“.

1.5.4 WordNet

Das wichtigste in der Praxis eingesetzte System für lexikalische semantische Analyse des Englischen ist *WordNet* ([Lab], [Fel98]). Es wird an der Universität Princeton seit

1985 entwickelt. Die Verwendung, auch kommerziell, ist unentgeltlich möglich. Die Entwickler beschreiben es auf der Webseite so:

> „WordNet is an online lexical reference system whose design is inspired by current psycholinguistic theories of human lexical memory. English nouns, verbs, adjectives and adverbs are organized into synonym sets, each representing one underlying lexical concept. Different relations link the synonym sets."

Einen Überblick über den quantitativen Umfang von WordNet gibt Tabelle 1.3. Bemerkenswert ist die Anzahl der insgesamt eingetragenen Wörter, 152.059, und das Verhältnis von Worten mit nur einer Bedeutung zu Worten mit mehreren Bedeutungen von ungefähr eins zu fünf. Dabei haben die 26.275 Einträge mit mehreren Bedeutungen insgesamt 77.739 registrierte Bedeutungen.

POS	Unique strings	Synsets	Total word-sense pairs
Nouns	114648	79689	141690
Verbs	11306	13508	24632
Adjectives	21436	18563	31015
Adverbs	4669	3664	5808
Total	152059	115424	203145

POS	Monosemous Entries	Polysemous Words	Polysemous Senses
Nouns	99524	15124	42325
Verbs	6256	5050	18522
Adjectives	16103	5333	14979
Adverbs	3901	768	1913
Total	125784	26275	77739

Tabelle 1.3: Der Umfang von WordNet

Die wesentliche Erweiterung zu einem Lexikon liegt in den Relationen zwischen Einträgen, die in WordNet registriert sind. Hier sind vier der wichtigsten Relationen am Beispiel von Verben aufgeführt:

Relationen

Relation	Definition	Beispiel
Hypernym	Aktivität –> übergeordnete Aktivität	fly –> travel
Troponym	Aktivität –> untergeordnete Aktivität	walk –> stroll
Entails	Aktivität –> eingeschlossene (bedingte) Aktivität	snore –> sleep
Antonym	Gegenteil	increase <–> decrease

Beispiel für die Relation „Hyponymie" bei Substantiven Abbildung 1.8 zeigt einen Auszug aus WordNet. Zwei Bedeutungen des Substantives „bass" werden anhand der Hyponymie-Relation (siehe 1.5.3) in der Bedeutungshierarchie eingeordnet. Ein längerer Pfeil verweist dabei auf eine höhere Einordnung, was einer allgemeineren Bedeutung entspricht.

bass (an adult male singer with the lowest voice)
-> singer, musician
—-> musician, instrumentalist, player
———-> performer, performing artist
———-> person, individual, someone,
———-> life form, organism, being
———-> entity, something
———-> causal agent, cause, causal agency
———-> entity, something
bass ([instrument])
-> musical instrument
—-> device
———-> instrumentality, instrumentation
———-> artifact, artefact
———-> object, physical object
———-> entity, something

Abbildung 1.8: Ein Beispiel für Hyponymie bei Substantiven in WordNet

1.6 Zusammenfassung und Ausblick

In der Geschichte der Computerlinguistik ist der Unterschied, aber auch die Wechselwirkung zwischen syntaktischen und statistischen Verfahren das Hauptcharakteristikum der Entwicklung.

Die Felder der Computerlinguistik können „von unten her“ beschrieben und aufeinander aufgebaut werden.

Zunächst befasst sich die *Morphologie* mit Gestalt und Aufbau einzelner Wörter. Die relevanten Begriffe hierbei sind *Morphem* als kleinste Einheit und *Stemming*, das Extrahieren des Wortstammes.

Dann folgt die Unterscheidung von Wortarten und die Untersuchung, wie aus Wörtern Phrasen zusammengesetzt werden. Dies zählt zum Feld der *Syntax*. *Part-of-speech tagging* ist der englische Fachbegriff für die Wortartbestimmung, Phrasen werden als *Konstituenten* bezeichnet und die Bestimmung des Typs eines Konstituenten heißt *Chunking*.

Der nächste Schritt führt zur Zusammensetzung vollständiger Sätze aus Konstituenten. Der entscheidende Begriff hier ist die *Grammatik*, wobei wir uns noch im Feld der Syntax bewegen. Die *Chomsky-Hierachie* klassifiziert Grammatiken nach ihrer Ausdrucksmächtigkeit. Vorgestellt wurden vor allem die *regulären* Grammatiken, die äquivalent zu den in der Computerlinguistik relevanten *regulären Ausdrücken* sind.

Bis zu diesem Punkt wurden allein äußere, formale Kriterien der Sprache untersucht. Die *Semantik* befasst sich nun mit der inhaltlichen Bedeutung. Als klassischer Ansatz, Bedeutung zunächst zu repräsentieren, um dann inhaltliche Schlußfolgerungen ziehen zu können, wurde die *Prädikatenlogik* vorgestellt. Zur Verbindung von Syntax und Semantik werden *Prädikat-Argument-Strukturen* verwendet. Beschränkt sich die semantische Analyse auf die Wortebene, die sogenannte *lexikalische Seman-*

tik, wird der Prozess robuster. Das bedeutendste praktisch eingesetzte System zur lexikalischen semantischen Analyse des Englischen ist *WordNet*.

Ausblick: Pragmatik und Diskurs Die *Pragmatik* beschreibt die korrekte Verwendung von Sprache in Abhängigkeit vom Kontext nach Kriterien wie zum Beispiel Höflichkeit.

Unter *Diskurs* versteht man den inhaltlichen Verlauf eines Textes und Referenzen innerhalb des Textes. Relevante Fragen sind hier

- wie auf Personen und Objekte Bezug genommen wird,
- wie der Kontext eines Gespräches entsteht,
- wie Äußerungen eines Konversationspartners richtig zu interpretieren sind.

Literaturverzeichnis

[B+02] Franz Baader et al. *The Description Logic Handbook: Theory, Implementation and Application*. Cambridge University Press, 2002.

[Can] Canoo.net. Deutsche Grammatik, Online Wörterbuch zur Rechtschreibung, Flexion und Wortbildung für die Sprache Deutsch. http://www.canoo.net.

[Cho56] Noam Chomsky. Three models for the description of language. *IRE Transactions on Information Theory*, 3:113–124, 1956.

[Cho59] Noam Chomsky. On certain formal properties of grammars. *Information and Control*, 2:137–167, 1959.

[Cona] World Wide Web Consortium. W3C OWL. http://www.w3.org/TR/2004/REC-owl-features-20040210/.

[Conb] World Wide Web Consortium. W3C Semantic Web. http://www.w3.org/2001/sw/.

[Ear70] Jay Earley. An Efficient Context-Free Parsing Algorithm. In *Communications of the ACM, 6 (8)*, pages 451–455, 1970. http://portal.acm.org/citation.cfm?id=362035.

[Fel98] Christiane Fellbaum, editor. *WordNet. An Electronic Lexical Database*. The MIT Press, 1998.

[FK79] W. N. Francis and H. Kucera. Brown corpus manual. http://helmer.aksis.uib.no/icame/brown/bcm.html, 1964, revised 1979.

[JM00] Daniel Jurafsky and James H. Martin. *Speech and Language Processing*. Prentice Hall, 2000. http://www.cs.colorado.edu/~martin/slp.html.

[Lab] Princeton University Cognitive Science Laboratory. Wordnet Webseite. http://wordnet.princeton.edu/.

[MSM93] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993. http://citeseer.ist.psu.edu/marcus93building.html.

[oP] University of Pennsylvania. Penn Treebank Webseite. http://www.cis.upenn.edu/~treebank/home.html.

[Por] M. F. Porter. Porter Stemmer Webseite. http://www.tartarus.org/~martin/PorterStemmer/.

[Por97] M. F. Porter. An algorithm for suffix stripping. In *Readings in information retrieval*, pages 313–316. Morgan Kaufmann Publishers Inc., 1997.

[Sha48] C.E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423 and 623–656, 1948.

[Uni] Lancaster University. Claws part-of-speech tagger for english. http://www.comp.lancs.ac.uk/computing/research/ucrel/claws/.

Tobias Lang

2 Grundlagen statistischer Sprachverarbeitung

In diesem Kapitel werden die Grundlagen der Sprachverarbeitung, auf denen die in den späteren Kapiteln vorgestellten Methoden und Werkzeuge aufbauen, aus statistischer Sicht erklärt.

2.1 Einleitung

Das Ziel der Computerlinguistik ist es, natürliche Sprache in maschinenverständliche Sprache zu überführen, um sie mit verschiedenen, automatisierten Anwendungen verarbeiten zu können. Es bieten sich dabei zahlreiche Möglichkeiten, wie maschinelle Übersetzung, automatische Textzusammenfassung, Extraktion von Information aus gegebenen Texten, etcetera Da sich natürliche Sprache an verschiedenen Orten unter unterschiedlichen Einflüssen entwickelt hat, ist bereits der Vergleich zweier Sprachen bzw. das Finden von Gemeinsamkeiten schwierig. Diese Gemeinsamkeiten sind jedoch unerlässlich, um allgemeingültige Modelle zu entwickeln, die für die entsprechenden Anwendungen wesentlich sind.

Für die Computerlinguistik entstehen dadurch eine Vielzahl von Problemen. Wie lassen sich grammatikalische Regeln modellieren? Wie kann mit Doppeldeutigkeiten von Wörtern umgegangen werden? Wie kann ein Text syntaktisch und semantisch klassifiziert werden?

Um diesen Problemen entgegen zu treten, wird versucht, allgemeingültige Muster im Sprachgebrauch zu finden. Grundlage hierfür sind statistische und wahrscheinlichkeitstheoretische Methoden, die auf Textsammlungen angewendet werden, die den entsprechenden Sprachgebrauch charakterisieren.

2.2 Voraussetzungen

Es ist also nötig, eine Sammlung digitalisierter Texte anzulegen, mit denen im weiteren Verlauf gearbeitet werden kann. Solch eine Textsammlung nennt man Korpus. Je nachdem welche Aufgabe bestritten werden soll, werden unterschiedliche Voraussetzungen an einen Korpus gestellt und ihre Quelltexte entsprechend ausgewählt.

Korpus

Der *Brown Corpus* der Brown University ist eine repräsentative Auswahl Amerikanischer Texte in den 1960ern und 70ern. Er besteht aus Zeitungsartikeln, religiöser Literatur, Romanen, etcetera. Es handelt sich dabei um einen sogenannten getaggten Korpus, das heißt, die einzelnen Wörter sind entsprechend ihrer Wortklasse mit einem Tag versehen beziehungsweise beschriftet. Der *Lancaster-Oslo-Bergen Corpus* stellt das britisch englische Pendant zum Brown Corpus dar. Beide sind balanzierte Korpora, da sie reale Stichproben ihrer Sprache darstellen. *Penn Treebank* vom *Linguistic Data Consortium* dagegen ist ein syntaktisch annotierter, das heißt, geparster, Korpus, dessen Sätze als Syntaxbäume dargestellt werden können. Die Texte sind aus dem *Wall Street Journal* entnommen. Weiterhin gibt es unter anderem bilinguale Korpora, wie den *Canadian Hansard* des kanadischen Parlaments, das seine Texte zwei oder mehrsprachig herausgibt.

Brown Corpus

Lancaster-Oslo-Bergen Corpus

Penn Treebank

Canadian Hansard

2.2.1 Zipfs Gesetz

Wenden wir uns jedoch erst einer Möglichkeit zu, einen digitalisierten Text auszuwerten. Als Beispiel möge uns Marc Twains *Tom Sawyer* (siehe [MS99]) dienen. Diesen Text interpretieren wir der Einfachheit halber als flache Liste von Wörtern. Es gibt zwei Möglichkeiten, die Länge dieser Liste zu bestimmen. Betrachten wir die absolute Länge, das heißt, alle vorhandenen Wörter, so sprechen wir von *Worttoken*, betrachten wir die Anzahl von unterschiedlichen Wörtern, so nennen wir diese *Worttypen*. *Tom Sawyer* besitzt 71370 Worttoken und 8018 Worttypen. Die Anzahl der Worttypen ist selbstverständlich abhängig vom benutzten Korpus. Ein Kinderbuch, wie *Tom Sawyer* es ist, besitzt nicht so viele verschiedene Wörter, wie eine wissenschaftliche Ausarbeitung.

Worttoken

Worttypen

Wenn wir die durchschnittliche Häufigkeit eines Wortes berechnen, so ergibt sich für *Tom Sawyer* $\frac{Anzahl\ Worttoken}{Anzahl\ Worttypen} = \frac{70370}{8018} \approx 8,9$. Das heißt, jedes Wort kommt durchschnittlich neun mal im Text vor. Natürlich spiegelt dieser Durchschnitt nicht die Realität wider, da die Wortverteilung natürlicher Sprache ungleichmäßig ist. Es gibt eine geringe Anzahl von allgemeinen Worten, deren Häufigkeit besonders hoch ist, wie zum Beispiel Artikel, und etliche individuelle Wörter, die nur sehr selten in einem Text vorkommen. Bei *Tom Sawyer* kommen knapp 50 Prozent der Wörter nur einmal vor. Diese Wörter nennt man *hapax legomena*.

hapax legomena

In Tabelle 2.1 sind Wörter mit ihrer Anzahl im Text der Reihe nach ihrer Häufigkeit eingetragen. Die Stelle, an der eines dieser Wörter steht, nennt man Rang. G. Zipf veröffentlichte 1932, was J. B. Estoup 1916 schon festgestellt hat, dass die Häufigkeit eines Worts indirekt proportional zu ihrem Rang ist (siehe [iCS02]), also

$$f \propto \frac{1}{r} \quad \text{bzw.} \quad f \cdot r = const \ . \tag{2.1}$$

In anderen Worten heißt das, dass das zehnthäufigste Wort fünfmal häufiger vorkommt als das 50st-häufigste Wort. Tabelle 2.1 zeigt dies am Beispiel von *Tom Sawyer*.

Wort	Hfgkt. (f)	Rang r	$f \cdot r$	Wort	Hfgkt. (f)	Rang (r)	$f \cdot r$
the	3332	1	3332	turned	51	200	10200
and	2972	2	5944	you'll	30	300	9000
a	1775	3	5235	name	21	400	8400
he	877	10	8770	comes	16	500	8000
but	410	20	8400	group	13	600	7800
be	294	30	8820	lead	11	700	7700
there	222	40	8880	friends	10	800	8000
one	172	50	8600	begin	9	900	8100
about	158	60	9480	family	8	1000	8000
more	138	70	9660	brushed	4	2000	8000
never	124	80	9920	sins	2	3000	6000
Oh	116	90	10440	Could	2	4000	6000
two	104	100	10400	Applausive	1	8000	8000

Tabelle 2.1: Empirische Veranschaulichung von Zipfs Gesetz an *Tom Sawyer*

Man erkennt allerdings, dass Zipfs Gesetz kein Gesetz an sich ist, sondern lediglich als gute Näherung dienen kann. Es bietet eine grobe Beschreibung der Häufigkeitsverteilung in menschlicher Sprache: es gibt eine geringe Anzahl allgemeiner dafür aber häufig genutzter Wörter, einige Wörter mit mittlerer Häufigkeit und viele individuelle und seltene Wörter. Aus diesem Grund versuchen sowohl *Sprecher* als auch *Zuhörer*, ihre Anstrengung zu verringern. Die Anstrengung des *Sprechers* wird geringer, indem er ein kleines Vokabular von allgemeinen Wörtern benutzt, während die Anstrengung des *Zuhörers* kleiner wird, wenn er einen großen Wortschatz an seltenen Wörtern hat, er somit allen Aussagen eines *Sprechers* folgen kann. Ein Kompromiss zwischen diesen beiden gegensätzlichen Ansichten lässt sich in der Beziehung von Häufigkeit eines Wortes und seinem Rang finden.

B. Mandelbrot beschäftigte sich in den 1950ern mit den Mängeln des zipfschen Gesetzes, nämlich die Ungenauigkeiten an den Rändern und die Wölbung in der Mitte. Diese Mängel versuchte er mit geeigneten Textparametern auszugleichen. Diese Parameter sind ein gemeinsames Maß für die Reichhaltigkeit von Wortnutzung eines Textes. Mandelbrot erreicht eine bessere Näherung der empirische Verteilung von Wörtern in einem Text mit

$$f = P\,(r+\rho)^{-B} \quad \text{oder} \quad \log f = \log P - B \log (r+\rho) \quad . \tag{2.2}$$

Für $B = 1$ und $\rho = 0$ entspricht Mandelbrots Erweiterung dem zipfschen Gesetz.

Zipfs Prinzip der geringsten Anstrengung lässt sich auch in vielen anderen Gebieten beobachten: ein kleiner Teil der Bevölkerung besitzt einen großen Teil des Reichtums, ein kleiner Teil von Produkten einer Firma ist für den größten Teil des Umsatzes verantwortlich, etc.

2.2.2 Kollokationen

Während bei Zipfs Gesetz nur einzelne Wörter betrachtet werden, besteht eine weitere Aufgabe der statistischen Sprachanalyse darin, sogenannte *Kollokationen* zu finden, also Ausdrücke bestehend aus zwei oder mehr Wörtern, die gemeinsam einen Sinn ergeben. Dabei lässt sich dieser Sinn nicht aus der Bedeutung der einzelnen Teile

schließen, wie es bei einfach zusammengesetzten Wörtern ist, sondern erst aus dem gemeinsamen Auftreten der Wörter. Das *Schwarze Brett* zum Beispiel muss weder unbedingt schwarz sein, noch ein Brett an sich. Erst die Kombination aus *schwarz* und *Brett* gibt den Sinn dieser Kollokation wieder. Es gibt drei Arten von Kollokationen, nämlich Nominalphrasen, wie *hellichter Tag* oder *maschinelle Übersetzung*, Verbalphrasen, wie *Kritik üben* oder *Abschied nehmen*, und feste Wendungen wie *hin und wieder* oder *an und für sich*.

Anhand der Beispiele sieht man, dass Kollokationen recht häufig sind. Für eine Vielzahl von Anwendungen sind sie deswegen zu beachten. Bei computergestützter Lexikographie, also der Erstellung von Wörterbüchern, können wichtige Kollokationen einen eigenen Eintrag erhalten. Diese Einträge sind dann bei der Erzeugung von natürlich klingender Sprache von Bedeutung, denn es macht einen (wohlklingenden) Unterschied, ob man *kräftigen* oder *starken Kaffee* trinkt. Die Kenntnis von Kollokationen kann zudem bei der automatisierten Satzbauanalyse die Arbeit erleichtern. Als weiteres Anwendungsgebiet ist die Sprachforschung zu nennen, da die Verwendung von bestimmten Ausdrücken von sozialen Phänomenen beeinflusst wird. Diese Einflüsse können durch das Finden der entsprechenen Kollokationen verfolgt werden. Es stellt sich an dieser Stelle die Frage, wie Kollokationen gefunden werden können.

Häufigkeit Die einfachste Möglichkeit ist das Zählen von häufig wiederkehrenden Wortkombinationen. Allerdings passiert es leicht, dass das Ergebnis aus Kombinationen von Funktionswörtern, also Präposition/Konjunktion und Artikel oder dergleichen, besteht. Die gefundenen Wortpaare müssen also gefiltert werden. Naheliegend sind Wortklassenmuster, also Adjektiv und Nomen, Nomen und Nomen, Adjektiv und Adjektiv und Nomen, drei Nomen hintereinander, etcetera. Diese Muster können als sogenannte *Part-of-Speech Tags* dargestellt werden, in diesem Fall zum Beispiel JJ NN, NN NN, JJ JJ NN, NN NN NN, etc. Diese Muster können beliebig erweitert und attributiert werden. Zum Beispiel ist es wünschenswert, dass drei Nomen als solche gefunden werden und nicht als jeweils zwei Nomen. *New York City* soll also nicht als *New York* und *York City* gefunden werden.

Part-of-Speech Tags

Ein weiteres Problem ist, dass auch einfach zusammengesetzte Wortpaare gefunden werden, die jedoch keine Kollokationen sind, wie zum Beispiel *letzte Woche*. Zudem versagt diese Methode bei durch eingeschobene Wörter getrennte Wortpaare. Im Satz *Er nahm am verregneten Sonntag von ihr Abschied* wird *nahm Abschied* nicht als Kollokation erkannt. Diese Methode ist also nur für feste und unzertrennliche Phrasen wie *Bundeskanzler Schröder* oder dergleichen sinnvoll.

Mittelwert und Streuung Eine andere Möglichkeit, die auch getrennte Wortpaare als Kollokationen identifizieren kann, berechnet den Mittelwert und die Streuung des Abstands zwischen den beiden Wörtern. Um diese Wortpaare zu finden, wird ein Fenster von zum Beispiel drei bis fünf Wörter auf jeder Seite eines Wortes festgelegt, in dem jedes Wortpaar als Kollokation erkannt werden kann. Die Berechnungen basieren also auf einer größeren Menge möglicher Wortpaare. An folgenden exemplarischen Sätzen zu *nahm Abschied* werden Mittelwert und Streuung berechnet:

a. er nahm von ihr Abschied

b. Opa nahm an der Haustür Abschied

c. seine Mutter nahm am kalten Bahnhof keinen Abschied von ihm

Der Mittelwert wird als durchschnittlicher Abstand der beiden Wörter bestimmt und beträgt hier:

$$\overline{d} = \frac{1}{3}(3+4+5) = 4,0$$

Wenn *Abschied* vor *nahm* auftreten sollte, wird der Wortabstand negativ angegeben, also -3 für *der Tag, an der sie Abschied von ihm nahm*. Die Fensterbreite ist entsprechend weit gewählt. Die Streuung gibt an, wie weit die einzelnen Abstände vom Mittelwert abweichen, und wird folgendermaßen bestimmt:

$$s^2 = \frac{\sum_{i=1}^{n} \left(d_i - \overline{d}\right)^2}{n-1} \tag{2.3}$$

Dabei beschreibt n, wie oft die beiden Wörter zusammen auftreten, d_i ist der Abstand für das gemeinsame Auftreten i und $\overline{d}$ ist der mittlere Abstand. Der Gebräuchlichkeit halber verwenden wir die Standardabweichung $s = \sqrt{s^2}$, die Wurzel der Streuung. Sie beträgt für das obige Beispiel:

$$s = \sqrt{\frac{1}{2}\left((3-4,0)^2 + (4-4,0)^2 + (5-4,0)^2\right)} \approx 1,41$$

Der Mittelwert und die Standardabweichung charakterisieren die Verteilung von Abständen zwischen zwei Wörtern in einem Korpus. Paare mit geringer Standardabweichung deuten auf eine Kollokation hin, da dies bedeutet, dass die beiden Wörter gewöhnlich mit einem ungefähr gleichen Abstand auftreten. Eine Standardabweichung von null heißt, dass die beiden Wörter stets mit genau dem gleichen Abstand auftreten.

Es lassen sich drei Arten von Wortpaaren beobachten: Paare mit sehr geringer Standardabweichung (kleiner 1) und einem Mittelwert größer 1, die schnell als Kollokation eingeordnet werden können, Paare mit hoher Standardabweichung und geringem Mittelwert, was darauf hindeutet, dass die beiden Wörter keine feste Bezieung haben, und Paare mit hohem Mittelwert und einer Standardabweichung größer 1. Paare aus der ersten Gruppe mit einem Mittelwert gleich 1 werden auch mit der Häufigkeitsmethode gefunden.

Hypothesen Testen Die Methode von Mittelwert und Streuung löst allerdings nicht alle Probleme. Wenn nämlich die Wörter eines Wortpaares an sich sehr häufig sind, jedoch eine niedrige Streuung haben, kann es sein, dass sie rein zufällig gemeinsam auftreten, obwohl es sich nicht um eine Kollokation handelt.

In diesem Fall findet Hypothesen Testen seinen Einsatz. Es wird eine Nullhypothese H_0, dass nur eine zufällige Beziehung zwischen den beiden Wörtern existiert und sie keine Kollokation bilden, formuliert. Anschließend wird die Wahrscheinlichkeit p des Ereignisses unter der Annahme, dass H_0 wahr ist, berechnet und die Nullhypothese verworfen, falls p kleiner als eine vorgegebene Signifikanz ist. Falls p größer als dieser Schwellwert ist, wird die Nullhypothese für möglich gehalten. Diese Methode wird zusätzlich zu einer musterbezogenen Suche angewendet.

Nullhypothese

Als Nullhypothese wird die Unabhängigkeit der beiden Wörter w_1 und w_2 angenommen, so dass die Wahrscheinlichkeit, dass die beiden Wörter in Kombination auftreten, mit

$$P(w_1 w_2) = P(w_1)(w_2)$$

berechnet wird.

Mit verschiedenen Verfahren wird die Wahrscheinlichkeit, dass die Nullhypothese zutrifft, berechnet und mit der Signifikanz verglichen. Solche Verfahren sind der *T-Test* oder der χ^2*-Test*. Details dazu siehe [MS99].

2.2.3 Bayes' Theorem

Bei der letzten Methode zum Finden von Kollokationen wurde davon ausgegangen, dass die beiden Wörter unabhängig voneinander sind und die Gesamtwahrscheinlichkeit durch das Produkt der Einzelwahrscheinlichkeiten berechnet werden kann. Wie wird nun mit abhängigen Ereignissen umgegangen, bzw. deren Wahrscheinlichkeit berechnet? Dazu ein kombinatorisches Beispiel:

Ein Urne ist gefüllt mit $0,7\%$ *grünen und* $99,3\%$ *roten Kugeln. Von den grünen haben* 95% *einen Punkt, von den roten* 3%*. Eine gezogene Kugel hat einen Punkt. Mit welcher Wahrscheinlichkeit ist sie grün?*

Diese Aufgabe lässt sich leicht mit einem Wahrscheinlichkeitsbaum (siehe Abbildung 2.1) visualisieren und berechnen. Das Ereignis $A1$ steht für eine rote Kugel, $A2$ für eine grüne und B für einen Punkt. Somit ist die Wahrscheinlichkeit, dass eine Kugel grün ist unter der Bedingung, dass sie einen Punkt hat,

$$P\left(\text{grün}|\text{Punkt}\right) = \frac{\text{grün und Punkt}}{\text{alle mit Punkt}} = \frac{665}{665 + 2979} \approx 0,18$$

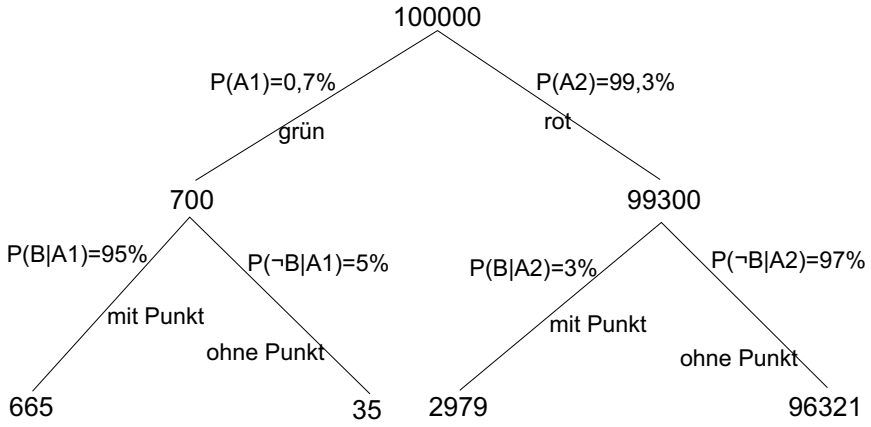


Abbildung 2.1: Wahrscheinlichkeitsbaum für bedingte Wahrscheinlichkeiten

Daraus lässt sich das Theorem von Bayes herleiten:

$$\begin{aligned} P\left(A_1|B\right) &= \tfrac{P(A_1 \cap B)}{P(B)} \\ P\left(A_1|B\right) P\left(B\right) &= P\left(A_1 \cap B\right) \\ &\Downarrow \\ P\left(B|A_1\right) P\left(A\right) &= P\left(B \cap A_1\right) \end{aligned}$$

$$\Rightarrow P\left(A_1|B\right) = \frac{P\left(B|A_1\right) P\left(A\right)}{P\left(B\right)} \tag{2.4}$$

Aus dem Satz von Bayes ergibt sich gewissermaßen die Umkehrung von Schlussfolgerungen. Oft ist die Wahrscheinlichkeit $P(\text{Ereignis}|\text{Ursache})$ bekannt, während die Wahrscheinlichkeit $P(\text{Ursache}|\text{Ereignis})$ gesucht ist. Als Beispiel ist an dieser Stelle die Vermeidung von Spam im E-Mail-Verkehr zu erwähnen. Von charakteristischen Wörtern in E-Mails (Ereignis) wird auf die Eigenschaft, Spam zu sein (Ursache), geschlossen.

Ein gutes Werkzeug ist dabei ein nach Bayes benannter E-Mail-Filter. Der Benutzer klassifiziert ankommende E-Mails in erwünschte und unerwünschte (Ham/Spam). Der Filter erstellt Listen mit Wörtern und ihren Häufigkeiten zusammen, die in E-Mails vorkommen, die als Spam eingeordnet wurden. Je häufiger ein Wort in der Spamliste katalogisiert wird, desto größer ist seine Spamwahrscheinlichkeit. Mails mit hohen solcher Wortvorkommnissen haben eine hohe Spamwahrscheinlichkeit. Der E-Mail-Client MozillaThunderbird benutzt einen Bayesfilter zur Spamerkennung. Ihm genügen schon circa 30 Spammails, um anschließend den größten Teil ankommender E-Mails richtig einzusortieren.

2.3 Modelle

Mit diesen Voraussetzungen wenden wir uns nun zwei Modellen der statistischen Sprachverarbeitung zu. Sprachverarbeitung schließt dabei nicht nur die Verwertung geschriebener Sprache, sondern auch die Bearbeitung und Manipulation gesprochener beziehungsweise entstehender Sprache mit ein. Das heißt, sie findet ebenso Einsatz in Spracherkennung, Handschriftenerkennung, Rechtschreibfehlerüberprüfung, etcetera.

2.3.1 n-Gramm Modell

Die Spracherkennung gestaltet sich dabei insofern schwierig, da der Eingabe verzerrt beziehungsweise durch Nebengeräusche gestört ist. Ähnlich verhält es sich auch bei der Handschriftenerkennung, bei der nicht jeder Buchstabe für sich automatisch erkannt werden kann. In einem entstehenden Text ist es in denjenigen Fällen leicht, Fehler zu finden, wenn dabei nicht-existierende Wörter entstehen. Schreibt man jedoch *Löcher* anstatt *Köcher*, ist dieser Fehler nicht per se zu erkennen. In diesem Fall wäre es sinnvoll, wenn der Computer durch die vorhergehenden Wörter erkennen könnte, was gemeint ist. Ein Mensch erkennt im Satz *Der Indianer hat viele Pfeile in seinem Löcher.* sehr schnell den Fehler, während ein Automatismus sich an dieser Stelle schwer tut.

Wenn vorher jedoch irgendwo gespeichert wäre, dass auf die Worte *Der Indianer hat viele Pfeile in seinem* das Wort *Köcher* folgt, würde auch eine automatische Rechtschreibprüfung den Fehler erkennen und das richtige Wort vorschlagen. Natürlich können auch andere Wörter folgen. Der Indianer kann seine Pfeile zum Beispiel auch im *Tipi* aufbewahren. Deswegen ist es sinnvoll, Wörter mit derselben Vorgeschichte, also mit denselben vorausgehenden Worten zu klassifizieren und in eine Äquivalenzklasse einzuordnen.

Diese Einordnung wird mit Trainingsdaten eines Trainingskorpus durchgeführt. Meist sind die Versuchsdaten, auf denen die Äquivalenzklassen anschließend arbeiten, unbekannt und es sind keine identischen Vorgeschichten vorhanden. Deswegen ist es sinnvoll, die Länge der Vorgeschichte auf ein vernünftiges Maß zu kürzen. Hierbei ist die sogenannte Markov Annahme, dass das nächste Wort nur durch wenige

Markov Annahme

vorhergehende Wörter beeinflusst wird, hilfreich.

n-Gramm

Wir verwenden folglich nur noch Wortfolgen mit n Wörtern. Diese werden *n-Gramme* genannt. Natürlich wäre es wünschenswert, n beliebig groß zu wählen, damit *der Indianer*, der intuitiv das gesuchte Wort charakterisiert, ebenfalls in der Äquivalenzklasse zu finden ist. Wenn wir jedoch einen Wortschatz von 20000 Wörtern voraussetzen und es würden nur Wortpaare, also eine Wortfolge von zwei Wörtern, sogenannte *Bigramme* gespeichert, gäbe es in diesem Fall bereits 20000×19999 also knapp 400 Millionen mögliche Kombinationen. Dementsprechend werden höchstens Folgen von vier Wörtern (*Tetragramme*) verwendet.

Bigramm

Tetragramm

Die Wahrscheinlichkeit für das n-te Wort mit den $n-1$ vorausgehenden Wörtern eines N-Gramm-Modells wird approximiert durch

$$P(w_n|w_1 \cdots w_{n-1}) \approx P(w_n|w_{n-N+1} \cdots w_{n-1}) \tag{2.5}$$

Es werden lediglich die letzten N Wörter betrachtet.

Erstellen von n-Gramm Modellen Bevor n-Gramm Modelle erstellt werden können, müssen die Trainingsdaten entsprechend vorverarbeitet werden. Meist liegt Plain Text vor, in dem je nach Aufgabe die Interpunktionszeichen gelöscht und die einzelnen Sätze dafür mit speziellen Tags markiert werden. Wenn davon ausgegangen wird, dass Abhängigkeiten über Satzgrenzen hinweg möglich sind, brauchen die Sätze nicht einzeln getaggt werden. Die Großschreibung am Satzanfang kann ebenfalls bearbeitet werden, falls die Äquivalenzklassen Groß- und Kleinschreibung unterscheiden. N-Gramm Modelle arbeiten am besten, wenn sie auf einer enormen Menge Trainigsdaten trainiert werden. Allerdings wird für die Verarbeitung der Daten viel CPU-Zeit und viel Plattenplatz benötigt.

Nachdem der Text vorverarbeitet worden ist, werden die Äquivalenzklassen gebildet, so dass gewissermaßen Behälter mit einer bestimmten Anzahl von Trainingsinstanzen gefüllt werden. Beispielsweise kommen zehn Instanzen des Wortpaares *come across* (englisch *to come across sth = auf etwas stoßen*) (siehe [MS99]) vor. Acht mal folgt dabei das Wort *as* (engl. *to come across as = auf jemanden wirken*), einmal das Wort *more* und einmal das Wort *a*.

Als Wahrscheinlichkeitsschätzung für das nächste Wort nach *come across* ist die relative Häufigkeit naheliegend:

$$\begin{aligned} P(as) &= 0,8 \\ P(more) &= 0,1 \\ P(a) &= 0,1 \\ P(x) &= 0,0 \text{ das heißt, keines der drei oberen Wörter folgt} \end{aligned}$$

Maximum-Likelihood-Schätzung

Diese Schätzung nennt man *Maximum-Likelihood-Schätzung* (engl. *maximum likelihood estimation*) (MLE):

$$P_{\text{MLE}}(w_1 \cdots w_n) = \frac{C(w_1 \cdots w_n)}{N} \tag{2.6}$$

$$P_{\text{MLE}}(w_n|w_1 \cdots w_{n-1}) = \frac{C(w_1 \cdots w_n)}{C(w_1 \cdots w_{n-1})} \tag{2.7}$$

$C(w_1 \cdots w_n)$ ist dabei die Häufigkeit der n-Gramme in Trainingstext, N ist die Zahl der Trainingsinstanzen.

Die Maximum-Likelihood-Schätzung hat dabei einen gravierenden Nachteil. Wie wir bereits in Kapitel 2.2.1 gesehen haben, kommt die Mehrzahl der Wörter in einem Text sehr selten vor, so dass längere n-Gramme, die diese enthalten, noch seltener werden. Da diese n-Gramme jedoch auf einen anderen Text angewendet werden, passiert es schnell, dass unbekannte Wortfolgen eine Wahrscheinlichkeit von 0 erhalten (im obigen Beispiel $P(x)$). Diese Null-Wahrscheinlichkeit werden an die Satzwahrscheinlichkeiten weitergegeben, da die Wahrscheinlichkeit eines langen Strings durch Multiplizieren der Einzelwahrscheinlichkeiten berechnet wird.

Smoothing Einen Lösungsansatz bietet das Smoothing. Dabei wird von den Wahrscheinlichkeiten der bekannten Ereignisse ein gewisses Maß abgezogen, das für das unbekannte Ereignis verwendet werden kann.

Add-One Smoothing

Das *Add-One Smoothing* ist ein einfacher Algorithmus, der zwar nicht sonderlich gut arbeitet, aber einen Einblick in das Konzept der Glättung gibt und für kompliziertere Algorithmen eine nützliche Grundlage bietet. Der Name ergibt sich aus der Tatsache, dass zu den einzelnen n-Gramm-Häufigkeiten eines Textes 1 dazugezählt wird. Ein Beispiel hilft, den Algorithmus leichter zu verstehen.

Unigramm

Wir nehmen einen Text an, der aus ca. 10000 Sätzen besteht und insgesamt 1616 Worttypen, also unterschiedliche Wörter, besitzt. Die Gesamtheit der Worttypen bilden das Vokabular V. Aus diesem Text wählen wir einen exemplarischen Satz *Ich gehe nach Hause*. Die einzelnen Wörter (*Unigramme*) dieses Satzes haben folgende Häufigkeiten im Text:

Ich	3734
gehe	1215
nach	3228
Hause	213

Tabelle 2.2: Absolute Häufigkeiten der vier Beispielwörter

	Ich	gehe	nach	Hause
Ich	8	1060	32	0
gehe	3	0	827	0
nach	4	9	0	325
Hause	1	0	3	0

Tabelle 2.3: Bigrammhäufigkeiten der vier Beispielwörter

Tabelle 2.3 zeigt eine Matrix einzelner Bigrammhäufigkeiten der vier Wörter unseres Beispielsatzes. Bevor wir diese zu Wahrscheinlichkeiten normalisieren, addieren wir 1 zu jeder Anzahl von Bigrammen.

Tabelle 2.4 zeigt die geglätteten Bigrammhäufigkeiten. Wir erinnern uns, dass die Bigrammwahrscheinlichkeit durch Normalisieren jeder Zeile mit der entsprechende Unigrammhäufigkeit berechnet wird:

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})}$$

	Ich	gehe	nach	Hause
Ich	9	1061	33	1
gehe	4	1	828	1
nach	5	10	1	326
Hause	2	1	4	1

Tabelle 2.4: Geglättete Bigrammhäufigkeiten der vier Beispielwörter

Wenn zu der Anzahl jedes Bigramms 1 addiert wird, jedes Bigramm also einmal mehr vorkommt, dann kommt jedes Unigramm soviel öfter vor, wie es Wordtypen gibt, da jedes Bigramm von einem Worttyp charakterisiert wird. Es muss also an dieser Stelle mit der Summe von Unigrammhäufigkeit und Anzahl der Worttypen normalisiert werden. Die Wahrscheinlichkeit p^* eines geglätteten Bigramms wird also folgendermaßen berechnet:

$$p^*\left(w_n|w_{n-1}\right) = \frac{C\left(w_{n-1}w_n\right)+1}{C\left(w_{n-1}\right)+V} \tag{2.8}$$

Zu jeder Unigrammhäufigkeit wird V $(= 1616)$ addiert:

Ich	$3734 + 1616$	$=$	5350
gehe	$1215 + 1616$	$=$	2831
nach	$3228 + 1616$	$=$	4844
Hause	$213 + 1616$	$=$	1829

	Ich	gehe	nach	Hause
Ich	0,0017	0,20000	0,00620	0,00019
gehe	0,0014	0,00035	0,29000	0,06700
nach	0,0010	0,00210	0,00021	0,06700
Hause	0,0010	0,00055	0,00220	0,00055

Tabelle 2.5: Geglättete Bigrammwahrscheinlichkeiten der vier Beispielwörter

Das Ergebnis sind die geglätteten Bigrammwahrscheinlichkeiten in Tabelle 2.5.

Witten-Bell Discounting
Good-Turing Discounting

Andere Möglichkeiten des Smoothings sind das *Witten-Bell Discounting* und das *Good-Turing Discounting*, wie sie in [JM00] gezeigt werden.

2.3.2 Hidden Markov Modell

Das zweite Modell, das gerade in der Spracherkennung eingesetzt wird, ist das Hidden Markov Modell (HMM). Das Markov Modell beruht auf sogenannten Markovketten. Eine Markovkette ist ein stochastischer Prozess, der aus einer Folge von Zufallsvariablen $X = (X_1, \ldots, X_T)$ besteht, die die Werte des Zustandsraumes $S = \{s_1, \ldots, s_N\}$ annehmen, und folgende Markoveigenschaften erfüllt:

Begrenzter Horizont:

$$P\left(X_{t+1} = s_k|X_1, \ldots, X_t\right) = P\left(X_{t+1} = s_k|X_t\right) \tag{2.9}$$

und **Zeitinvarianz**:

$$P\left(X_{t+1} = s_k | X_1, \ldots, X_t\right) = P\left(X_2 = s_k | X_1\right) \tag{2.10}$$

Anders ausgedrückt ist die Wahrscheinlichkeit eines Zustandes zum Zeitpunkt $t+1$ nur abhängig vom Zustand zum Zeitpunkt t und nicht von der vorherigen Zustandssequenz. Dies nennt man Gedächtnislosigkeit. Zweitens ist die Wahrscheinlichkeit eines Zustandes zum Zeitpunkt $t+1$ die gleiche wie an jedem anderen Zeitpunkt, zum Beispiel $t = 2$, die Zustandswahrscheinlichkeiten ändern sich also nicht. Die Übergangswahrscheinlichkeiten können in einer Übergangsmatrix A dargestellt werden, wobei

$$a_{ij} = P\left(X_{t+1} = s_j | X_t = s_i\right) \tag{2.11}$$

mit $a_{ij} \geq 0 \; \forall \, i, j$ und $\sum_{j=1}^{N} a_{ij} = 1 \; \forall \, i$. An der Stelle a_{ij} steht also die Übergangswahrscheinlichkeit des Zustandes s_i nach s_j.

Zusätzlich benötigen wir die Startwahrscheinlichkeiten, also die Wahrscheinlichkeit π_i mit der die Markovkette im ersten Zeitabschnitt $t = 1$ in Zustand s_i ist:

$$\pi_i = P\left(X_1 = s_i\right) \tag{2.12}$$

Für die Summe der Startwahrscheinlichkeiten gilt natürlich $\sum_{i=1}^{N} \pi_i = 1$.

Das Hidden Markov Modell unterscheidet sich insofern vom Markov Modell, dass die Zustände der Markovkette nicht bekannt sind, sie sind sozusagen versteckt (engl. *hidden*). Dafür wird das Markov Modell um einen weiteren Zufallsprozess erweitert. Dieser liefert für jeden Zeitpunkt t_i eine beobachtbare Ausgabe mit einer gewissen Ausgabewahrscheinlichkeit. Formal gesprochen ist das Hidden Markov Modell also ein 5-Tupel (S, K, Π, A, B) mit folgenden Eigenschaften:

Zustandsraum	$S = \{s_1, \ldots, s_N\}$
Ausgabealphabet	$K = \{k_1, \ldots, k_M\}$
Startwahrscheinlichkeiten	$\Pi = \{\pi_i\}, \; i \in S$
Zustandsübergangswahrscheinlichkeiten	$A = \{a_{ij}\}, \; i, j \in S$
Symbolausgabewahrscheinlichkeiten	$B = \{b_{ijk}\}, \; i, j \in S, \; k \in K$

Daneben gibt es noch die Zustandsfolge $X = (X_1, \ldots, X_{T+1})$ mit $X_t : S \mapsto \{1, \ldots, N\}$ und die Folge von Ausgabesymbolen $O = (o_1, \ldots, o_T)$ mit $o_t \in K$. Anschaulich gesprochen ist das Hidden Markov Modell ein endlicher Automat, dessen Zustandsübergänge und Ausgaben nicht durch Eingaben, sondern probabilistisch festgelegt sind.

Hidden Markov Modelle finden ihren Einsatz gerade im Gebiet der Spracherkennung, denn damit lässt sich die zeitliche Auflösung von Sprache, zum Beispiel verschiedene Aussprachen eines Wortes, modellieren. In Abbildung 2.2 ist ein Automat dargestellt, der die Aussprache des Wortes *heben* modelliert (siehe [Wen04]). Die zeitliche Dehnung der einzelnen Laute wird durch das Verbleiben im momentanen Zustand generiert. Die entsprechend hohe Übergangswahrscheinlichkeit des ersten Zustandes *e* auf sich selbst zeigt, dass die erste Silbe wahrscheinlich lang ausgesprochen wird. Der direkte Übergang des dritten in den fünften Zustand modelliert das Verschlucken des zweiten *e*. Die Ausgabewahrscheinlichkeit des letzten Zustandes regelt, ob das *n* als *n* oder *m* ausgesprochen wird. Formen wie *hheeeebeenn*, *hebm* oder *heeben* sind modellierbar.

Für HMMs lassen sich drei grundlegende Fragen formulieren:

1. Wie kann effizient berechnet werden, wie wahrscheinlich eine bestimmte Ausgabe bei einem gegebenem Modell $\mu = (A, B, \Pi)$, das heißt, wie groß $P\left(O|\mu\right)$ ist?

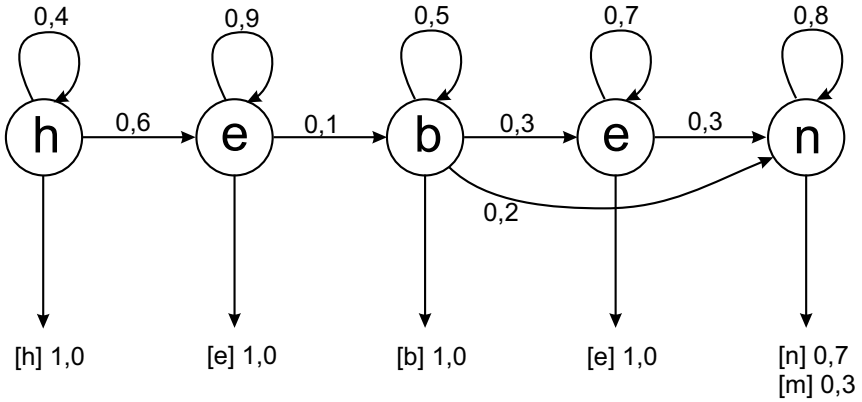


Abbildung 2.2: Automat zur Erzeugung des Wortes *heben*

2. Wie kann eine Zustandsfolge $(X_1, \ldots, X_{T+1})$, die eine gegebene Ausgabesequenz O bei einem gegebenem Model μ beschreibt, ausgewählt werden?

3. Zu einer gegebene Ausgabenfolge O wird eine Menge von möglichen Modellen durch Variieren der Modellparameter $\mu = (A, B, \Pi)$ gefunden. Wie kann dasjenige Modell ausgewählt werden, das die Ausgabedaten am besten beschreibt?

Die erste Frage beschäftigt sich damit, wie die Wahrscheinlichkeit einer bestimmten Ausgabe berechnet werden kann. Die zweite Frage zielt darauf ab, welcher Pfad durch die Markovkette genommen wurde. Dieser versteckte Pfad kann anschließend für Klassifikationsprozesse verwendet werden. Die dritte Frage ist, wie wir aus den Ausgabedaten die Parameter folgern können. Für jede dieser Fragen gibt es einen entsprechenden Algorithmus.

Finden der Wahrscheinlichkeit einer Ausgabe Da die direkte Berechnung mittels Addition und Multiplikation von Zustandsübergangs- und Ausgabewahrscheinlichkeiten zu aufwändig ist (es sind im allgemeinen $(2T+1) \cdot N^{T+1}$ Multiplikationen nötig), geschieht diese Berechnung heute unter anderem mit dem sogenannten *Vorwärts-Algorithmus*. Die Grundidee ist, dass Teilergebnisse zwischengespeichert werden, anstatt sie noch einmal zu berechnen. Dazu wird eine Vorwärtsvariable folgendermaßen festgelegt:

Vorwärts-Algorithmus

$$\alpha_i(t) = P(o_1 o_2 \cdots o_{t-1}, X_t = i|\mu) \tag{2.13}$$

Diese Variable wird fortwährend in folgender Reihenfolge berechnet:

1. Initialisierung
$$\alpha_i(1) = \pi_i, \quad 1 \leq i \leq N$$

2. Induktion
$$\alpha_j(t+1) = \sum_{i=1}^{N} \alpha_i(t)\, a_{ij} b_{ijo_t}, \quad 1 \leq t \leq T,\ 1 \leq j \leq N$$

3. Ergebnis

$$P(O|\mu) = \sum_{i=1}^{N} \alpha_i (T+1)$$

Dieser Algorithmus benötigt nur $2N^2T$ Multiplikationen.

Finden der besten Zustandsfolge Für das Finden der besten Zustandsfolge wird ebenfalls ein Algorithmus aus dem Gebiet der dynamischen Programmierung verwendet. Das heißt, Teilergebnisse werden zwischengespeichert und weiterverwendet. Der eingesetzte *Viterbi-Algorithmus* definiert dafür zwei Variablen $\delta_j(t)$ und $\psi_j(t+1)$, die diese Teilergebenisse rekursiv berechnen. Die genaue Funktionsweise des Viterbialgorithmus findet der geneigte Leser in der einschlägigen Literatur, zum Beispiel [MS99, S. 332*ff*].

Viterbi-Algorithmus

Parameter Abschätzen Mit einer gegebenen Ausgabesequenz sollen die Modellparameter gefunden werden, die diese Ausgabe am wahrscheinlichsten erzeugt hat. Auf dieses Problem wird der *Baum-Welch-Algorithmus* angewendet: Durch ein beliebig ausgewähltes Modell wird die Wahrscheinlichkeit für die Ausgabesequenz berechnet. Dabei wird darauf geachtet, welche Zustandsübergänge und Symbolausgaben wahrscheinlich am häufigsten genutzt werden. Anschließend kann ein verbessertes Modell ausgewählt werden, indem die entsprechenden Wahrscheinlichkeiten erhöht werden. Diese Modell erzeugt eine höhere Wahrscheinlichkeit für die Ausgabesequenz. Dieser Maximierungsprozess trainiert gewissermaßen das Modell. Für weitere Details wird auch hier auf die entsprechende Literatur verwiesen [MS99, S. 333*ff*].

Baum-Welch-Algorithmus

2.4 Anwendungsbeispiele

Die vorgestellten Modelle werden in verschiedenen Anwendungen zur Hilfe genommen. Während wir uns in den letzten Kapiteln eher mit unstrukturierten Daten befasst haben, versuchen wir in diesem Kapitel, die Struktur von Sätzen zu erfassen und zu verarbeiten.

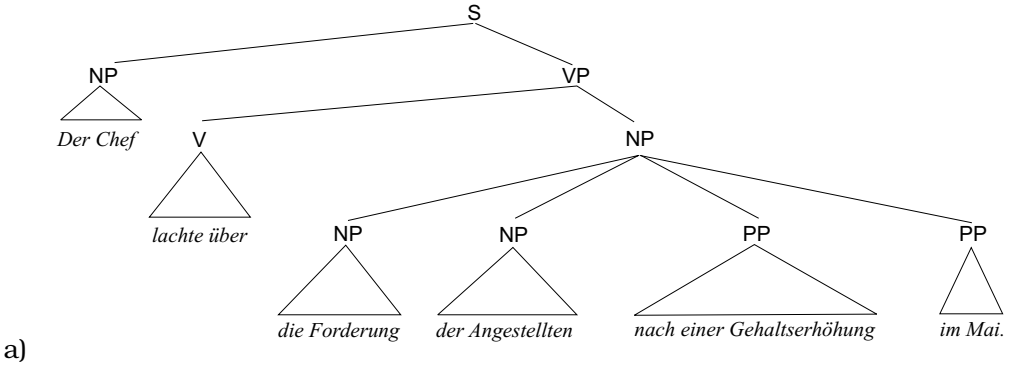
2.4.1 Probabilistisches Parsen

Die Struktur eines Satzes lässt sich mittels eines *Syntaxbaums* erfassen. Ausgehend vom gesamten Satz als Wurzel werden die einzelnen Satzteile in immer detailiertere Struktureinheiten bis zu den Wortklassen eingeteilt. Wir unterscheiden bei den Struktureinheiten unter anderem Nominal-, Verbal- und Präpositionalphrasen. Selbstverständlich gibt es je nach Bedarf weitere Einheiten.

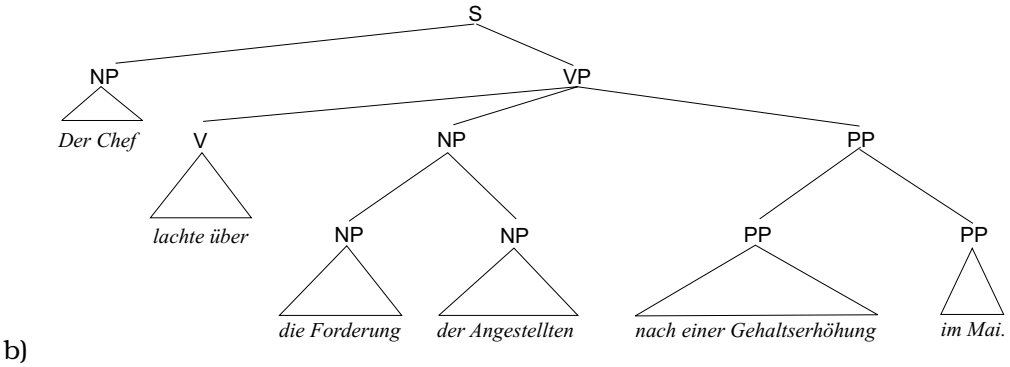
Syntaxbaum

Beim Parsen, also der Syntaxanalyse, eines Satzes kann es vorkommen, dass es mehrere, mögliche Syntaxbäume gibt. Ein einfaches Beispiel ist *Der Chef lachte über die Forderung der Angestellten nach einer Gehaltserhöhung im Mai.* Einige, mögliche Syntaxbäume stellt Abbildung 2.3 dar.

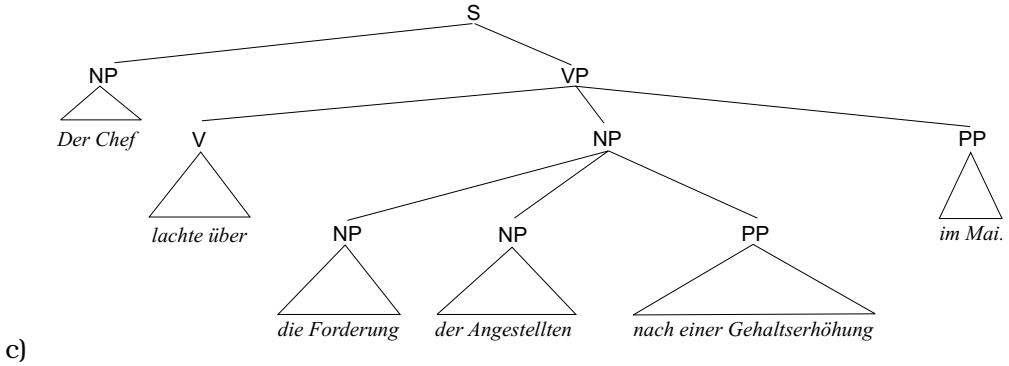
Die Aufgabe des probabilistischen Parsers ist jetzt, diese Doppeldeutigkeiten zu erkennen und den wahrscheinlichsten Syntaxbaum $\hat{t}$ unter den möglichen Syntaxbäumen t für den Satz s zu identifizieren, also den besten *Parse* zu finden:



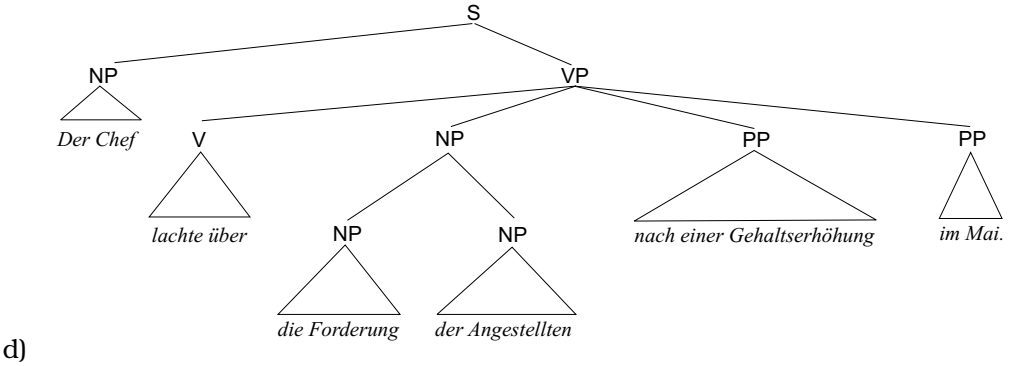
a)



b)



c)



d)

Abbildung 2.3: Verschiedene Syntaxbäume zu *Der Chef lachte über die Forderung der Angestellten nach einer Gehaltserhöhung im Mai.*

$$\hat{t} = \underset{t}{\operatorname{argmax}} P(t|s) = \underset{t}{\operatorname{argmax}} \frac{P(t,s)}{P(s)} = \underset{t}{\operatorname{argmax}} P(t,s) \tag{2.14}$$

Dabei gilt, dass die Wahrscheinlichkeit $P(s)$ des Satzes s für alle Syntaxbäume gleich ist, also nicht beeinflusst, welcher am wahrscheinlichsten ist. Deswegen kann $P(s)$ eleminiert werden.

Für einen Satz und seinen Syntaxbaum gilt

$$P(t,s) = P(T)\,P(s|t)$$

Da ein Syntaxbaum eines Satzes alle Wörter des Satzes enthält, ist die Wahrscheinlichkeit des Satzes in Abhängigkeit des Syntaxbaumes $P(s|t)$ gleich 1. Also gilt

$$P(t,s) = P(t)$$

Also folgt

$$\hat{t} = \underset{t}{\operatorname{argmax}} P(t,s) = \underset{t}{\operatorname{argmax}} P(t) \tag{2.15}$$

Hier stellt sich die Frage, wie die Wahrscheinlichkeit eines Syntaxbaumes berechnet werden kann. Die Antwort sind Probabilistische Grammatiken. Das heißt, jeder Produktion einer Grammatik wird eine Wahrscheinlichkeit für diese Produktion zugeordnet. Es entsteht also eine *Probabilistische Kontextfreie Grammatik*, kurz *PCFG*, die als 5-Tupel definiert: $G = (N, \Sigma, P, S, D)$. N ist eine Menge von Nichtterminalen, Σ ist eine Menge von Terminalen, P ist eine Menge von Produktionen der Form $A \to \beta$, wobei A ein Nichtterminal und β eine Symbolkette aus der Menge $\Sigma \cup N$ ist, S ist das Startsymbol und D ist eine Funktion, die jeder Produktion aus P eine Wahrscheinlichkeit $P(A \to \beta)$ zuordnet. PCFG

Die Summe über alle Wahrscheinlichkeiten von Produktionen mit demselben Nichtterminal auf der linken Seite ergibt 1. Tabelle 2.6 zeigt exemplarisch eine minimale PCFG. Erweitert um Produktionen der einzelnen Terminale, also der einzelnen Wörter, würde sie den obigen Beispielsatz *Der Chef lachte über die Forderung der Angestellten nach einer Gehaltserhöhung im Mai* gemäß des ersten Syntaxbaums aus Abbildung 2.3 parsen.

$S \to NP\ VP$	$[1,0]$
$NP \to Art\ Nom$	$[0,2]$
$NP \to NP\ NP\ PP\ PP$	$[0,4]$
$NP \to NP\ NP$	$[0,3]$
$NP \to NP\ NP\ PP$	$[0,1]$
$VP \to V\ NP$	$[0,8]$
$VP \to V\ NP\ PP$	$[0,1]$
$VP \to V\ NP\ PP\ PP$	$[0,1]$
$PP \to Präp\ NP$	$[0,6]$
$PP \to PP\ PP$	$[0,4]$

Tabelle 2.6: Exemplarische PCFG

Die Wahrscheinlichkeiten für die Produktionen sind nicht per se bekannt, sondern werden an bereits geparsten Sätzen eines entsprechenden Korpus, wie es Penn Tree-

bank einer ist, trainiert. Wenn man solch einen Treebank hat, kann die Wahrscheinlichkeit einer Produktion berechnet werden, indem man zählt, wie oft diese Produktion auftritt, und dann mit der Anzahl der auftretenden linken Seite normalisiert.

$$P(\alpha \rightarrow \beta|\alpha) = \frac{C(\alpha \rightarrow \beta)}{C(\alpha)} \tag{2.16}$$

Probleme von PCFGs Probabilistische kontextfreie Grammatiken haben mit zwei Problemen zu kämpfen, nämlich strukturellen und lexikalischen Abhängigkeiten. Diese Probleme rühren von der Annahme her, dass die einzelnen Produktionen unabhängig sind. Dies ist jedoch nicht der Fall. Die strukturellen Abhängigkeiten sind darin begründet, dass Subjekte eines Satzes größtenteils Pronomen sind, die sich auf vorherige Sätze beziehen. Sie repräsentieren sozusagen die alte Information des letzten Satzes. Andersherum kommen Pronomen seltener als Objekte in Sätzen vor. Diese Abhängigkeiten können umgangen werden, indem die Produktionen einer Nominalphrase in ein Pronomen oder einen Artikel und ein Nomen abhängig davon gemacht werden, ob es sich um das Subjekt oder das Objekt eines Satzes handelt.

Das zweite und wichtigere Problem beruht auf dem Fehlen semantischer Information. Das heißt, die Wahrscheinlichkeit einzelner Produktionen müsste abhängig von der semantischen Bedeutung der Terminale sein. Nehmen wir obiges Beispiel *Der Chef lacht über die Forderung der Angestellten nach einer Gehaltserhöhung im Mai.* Wir erkennen sofort, dass sich die Präposition *nach* auf die *Forderung* bezieht und keine zeitliche Präposition ist, die vom Verb *lachen* abhängt. Dies betrifft insbesondere Verben, die meist mit bestimmten Orts- oder Zeitpräpositionen auftreten. Dies muss bei den Wahrscheinlichkeiten entsprechender Produktionen beachtet werden. Ein Lösungsweg ist das Mitführen einer separaten Statistik über lexikalische Abhängigkeiten.

2.4.2 Statistische Zuordnung

Die Syntaxbäume, die im letzten Kapitel gewonnen worden sind, sind für die statistische Zuordnung wichtig. Diese befasst sich mit dem Problem, zwei unterschiedlichsprachliche, aber inhaltlich gleiche Texte einander zuzuordnen, was die Grundlage für maschinelle Übersetzer ist. Maschinenübersetzung hat das Ziel, eine fehlerfreie und natürlichklingende Übersetzung eines Textes in eine andere Sprache zu erzeugen. Dabei gibt es vier Möglichkeiten eine Übersetzung durchzuführen.

Wort für Wort

Die einfachste Variante ist die *Wort für Wort-Übersetzung*. Diese liefert jedoch nur unzureichende Ergebnisse, da zwischen verschiedenen Sprachen keine 1-zu-1 Beziehung herrscht und die auftretenden Doppeldeutigkeiten nicht aufgelöst werden können. Ein kleines Beispiel liefert das englische Wort *to miss*, auf Deutsch *vermissen, verpassen*. *I missed you – Ich vermisste Dich*, aber *I missed the bus – Ich verpasste den Bus*! Der Übersetzer müsste also auf den Kontext, in dem das Wort steht, achten, um die richtige Bedeutung des Wortes zu erkennen. Ein Wort für Wort-Übersetzer hat zudem mit dem weiteren Problem zu kämpfen, dass in verschiedenen Sprachen verschiedene Wortreihenfolgen vorkommen. *Yesterday I walked – Gestern wanderte ich*.

syntaktische Überführung

Dieses Problem kann jedoch mit einer anderen Art der Übersetzungsart gelöst werden, der *syntaktischen Überführung*. Der Originaltext wird geparst, der entsprechende Syntaxbaum erstellt und dieser dann in die andere Sprache übersetzt. Natürlich ist darauf zu achten, dass syntaktische Doppeldeutigkeiten, wie sie in Kapitel 2.4.1 be-

schrieben worden sind, aufgelöst werden. Allerdings kann es vorkommen, dass eine syntaktisch korrekte Übersetzung semantisch falsch ist. *Ich esse gern* wird im Englischen mit *I like to eat* und nicht mit *I eat willingly* übersetzt. Die richtige Überführung von Adverbkonstruktion zu eine Verbkonstruktion kann die syntaktische Überführung nicht leisten.

semantische Überführung

Eine *semantische Überführung* löst diese syntaktischen Ungleichheiten. Dabei wird die Bedeutung eines jeden Satzes geparst und nur diese in die andere Sprache übersetzt. Doch auch diese Variante ist nicht frei von Fehlern. Während nämlich die wörtliche Bedeutung einer Übersetzung korrekt sein kann, kann sie unnatürlich klingen. Ein klassisches Beispiel ist, wie das Englische und das Spanische die Richtung und die Art einer Bewegung ausdrücken. Das Englische benutzt für die Art der Bewegung das Verb und für die Richtung eine Präposition oder ein Adverb, also *The bottle floated out* (*Die Flasche schwamm hinaus*). Das Spanische dagegen verwendet für die Richtung das Verb und für die Art der Bewegung das Gerundiv, also *La botella salió flotando* (wörtlich etwa *die Flasche ging schwimmend hinaus*).

Interlingua

Eine Übersetzung zweier Sprachen via einer Zwischensprache, einer sogenannten *Interlingua*, würde auch diese Probleme lösen. Eine Interlingua ist ein Formalismus zur Wissensrepräsentation, der unabhängig von der Art ist, wie verschiedene Sprachen einen Sinngehalt ausdrücken. Ein weiterer Vorteil ist die effiziente Übersetzung in einem System. Anstatt jede Sprache in jede andere Sprache mit einem Aufwand von $O\left(n^2\right)$ zu übersetzen, wird ein System generiert, in dem die Übersetzung nur zwischen einer Sprache und der Interlingua stattfindet, was einen Aufwand von $O\left(n\right)$ bedeuten würde. Leider ist dieser Formalismus schwer zu erstellen, da die Vielzahl von Doppeldeutigkeiten bei der Übersetzung von natürlicher Sprache in eine wissensbeschreibende Sprache aufgelöst werden müssen.

Die Grundlage jeder maschinellen Übersetzung sind zweisprachige Wörterbücher und parallele Grammatiken. Diese können mittels Textzuordnung gewonnen werden. Zwei- oder mehrsprachige Korpora, also Texte mit gleichem Inhalt in unterschiedlichen Sprachen, werden verarbeitet, um Wörterbücher zu trainieren. Diese Texte nennt man auch *parallele Texte*. Diese stammen meist von Parlamenten mehrsprachiger Länder, wie es beispielsweise Kanada und die Schweiz sind, denn diese sind erstens leicht zu bekommen und bestehen zweitens meist aus konsistenten und wörtlichen Übersetzungen, da für solch offizellen Texte eine akkurate Übersetzung erforderlich ist.

parallele Texte

Auf diesen Texten werden Abschnitts-, Satz und Wortzuordnungen ausgeführt, um zu erkennen, welches Wort der einen Sprache mit welchem Wort der anderen Sprache übersetzt wurde. Damit lassen sich anschließend die Wörterbücher füllen. Die Satzzuordnung gestaltet sich nicht immer sehr einfach, da Sätze nicht in jedem Fall eins zu eins übersetzt werden. Ziel ist es, diejenige Satzgruppe der einen Sprache zu finden, die dem Inhalt nach zur Satzgruppe der anderen Sprache gehört. Diese zugeordneten Satzgruppen, die natürlich auch aus nur jeweils einem Satz bestehen können, nennt man *Beads*. Großteils entspricht jeder Satz der einen Sprache einem Satz der anderen Sprache, es handelt sich also um eine $1:1$-Zuordnung, doch kommen auch andere Variationen vor. Sätze können zusammengefasst oder geteilt werden, also zum Beispiel $1:2$- oder $2:1$-Zuordnungen, wahlweise auch aus der Übersetzung rausfallen, also $1:0$-Zuordnungen, oder zu der Übersetzung hinzugefügt werden, $0:1$-Zuordnungen. Teilweise werden auch einzelne Satzteile in den nächsten Satz mitgenommen, oder es können überkreuzte Abhängigkeiten auftreten, das heißt, die Reihenfolge der Sätze ist verändert. Daraus entstehen dann komplexere Zuordnungen, wie $2:2$, $2:3$, $3:3$, etcetera. Um die verschiedenen Zuordnungen zu finden, gibt es unterschiedliche Methoden.

Bead

Längenbasierte Methode Eine naheliegende Möglichkeit, Satzzuordnungen zu finden, geht von dem Prinzip aus, dass kurze Sätze in kurze Sätze und lange Sätze in lange Sätze übersetzt werden. Die Länge kann dabei als Anzahl der Wörter oder Anzahl der Buchstaben in einem Satz definiert werden. Es soll also die höchstwahrscheinliche Zuordnung A von parallelen Texten S und T gefunden werden:

$$\underset{A}{\operatorname{argmax}} P(A|S,T) = \underset{A}{\operatorname{argmax}} P(A,S,T) \tag{2.17}$$

Der Einfachheit halber wird der zugeordnete Text in Folgen von Beads $(B_1,\ldots,B_K)$ eingeteilt, deren Wahrscheinlichkeiten unabhängig voneinander sind, so dass

$$P(A,S,T) \approx \prod_{k=1}^{K} P(b_k) \tag{2.18}$$

Bevor mit der eigentlichen Satzzuordnung begonnen wird, wird eine Abschnittszuordnung durchgeführt, damit nicht ein Satz des einen Abschnitts einem Satz eines völlig anderen Abschnitts zugeordnet werden kann. Diese Abschnittszuordnung ist zudem leicht, da die Abschnittsstruktur klar markiert ist.

Bei der Satzzuordnung bestimmt die Satzlänge, wie wahrscheinlich die Zuordnung einer Satzgruppe des Textes L_1 zu eine Satzgruppe der Textes L_2 ist. Der Einfachheit halber können die möglichen Zuordnungen auf $\{0:1, 1:0, 1:1, \ldots, 2:2\}$ begrenzt werden. Anschließend wird der kleinstmögliche Abstand $D(i,j)$ zwischen zwei Satzgruppen rekursiv bestimmt. Dabei wird die Annahme gemacht, dass jeder Buchstabe einer Sprache eine zufällige Anzahl von Buchstaben einer anderen Sprache ergibt. Weiterhin wird angenommen, dass diese Zufallsvariablen unabhängig und gleichverteilt sind, so dass diese Zufälligkeit durch eine Normalverteilung mit dem Mittelwert μ und der Varianz s^2 modelliert werden kann. Diese Parameter werden mit Hilfe der Korpusdaten geschätzt, nämlich μ durch das Verhältnis der Länge beider Texte, bei einer Deutsch/Englisch-Übersetzung zum Beispiel $1,1$, und s^2 durch das Quadrat der Längenunterschiede der Abschnitte. Daraus lassen sich Wahrscheinlichkeiten für bestimmte Zuordnungstypen ($1:1$, $2:1$, etcetera) berechnen, die in die Berechnung des kleinstmöglichen Abstands $D(i,j)$ mit einfließen. Der Algorithmus von Gale und Church wird in [MS99] detailiert beschrieben. Andere Algorithmen verwenden als Satzlänge nicht die Anzahl der Buchstaben, sondern die Anzahl der Wörter des Satzes.

Lexikalische Methode Bei der lexikalischen Methode von Satzzuordnungen geht man ähnlich vor wie bei der längenbasierten Methode. Hinzu kommt, dass lexikalische Information dazu benutzt wird, den Zuordnungsprozess zu führen. Diese Information liefert eine Bestätigung für eine Zuordnung, was die Methode robuster gegenüber der längenbasierten Methode macht. Es gibt verschiedene Algorithmen, wie die lexikalische Information verwendet werden kann, Kay und Röscheisen (siehe [MS99]) gehen folgendermaßen vor:

Zu Beginn nehmen sie den ersten und den letzten Satz einer Zuordnung. Diese beiden Sätze bilden die anfänglichen Anker. Daraufhin wird eine Hülle von möglichen Zuordnungen durch das kartesische Produkt über die Liste der Sätze der einen Sprache und der Liste der Sätze der anderen Sprache gebildet. Diejenigen Zuordnungen, die über die Ankersätze hinausgehen oder deren entsprechender Abstand zu einem Anker zu sehr abweicht, werden ausgeschlossen. Anschließend werden Wortpaare, die oft gemeinsam in diesen möglichen Teilzuordnungen auftreten, und Wörter, deren Verteilung ähnlich ist, das heißt, Sätze, in denen das eine Wort auftaucht, sind Sätze, in

denen das andere auftaucht, zuordenbar. Diese Zuordnung darf jedoch nicht zufällig sein. Am Ende werden Paare von Quell- und Zielsätzen gesucht, die möglichst viele lexikalische Übereinstimmungen haben. Das sicherste dieser Paare führt eine Menge von Teilzuordnungen an, die zum Endergebnis und zur Liste der Anker hinzugefügt werden. Anschließend wird von neuem begonnen.

Je mehr Paare man in jeder Wiederholung als sicher ansieht, desto weniger Wiederholungen werden benötigt. Allerdings leidet das Ergebnis darunter. Gewöhnlich reichen fünf Wiederholungen, um ein vernünftiges Ergebnis zu erhalten.

Um daraus zweisprachige Wörterbücher zu generieren, muss für jeden Satz noch eine Wortzuordnung stattfinden. Wenn entsprechende Zuordnungen gefunden worden sind, müssen bestimmte Kriterien darüber entscheiden, ob sie ins Wörterbuch aufgenommen werden können. Solch ein Kriterium kann zum Beispiel die Häufigkeit sein, mit der die Zuordnung im Korpus vorkommt. Wenn eine Zuordnung beispielsweise nur einmal vorkommt, wird sie nicht ins Wörterbuch aufgenommen werden, da die Übersetzung aus dem Zusammenhang heraus anders als gewöhnlich lautet.

2.5 Zusammenfassung

In diesem Kapitel sind einige Methoden vorgestellt worden, wie stochastische Mittel eingesetzt werden können, um Texte und Sprache maschinell verarbeiten zu können. Dabei haben wir statistische Eigenschaften natürlicher Sprache kennengelernt, die es uns erleichtern, Sprachmodelle zu entwerfen. Diese Sprachmodelle sind für verschiedene Anwendungen notwendig, wie wir auch in den folgenden Kapiteln sehen werden.

Literaturverzeichnis

[iCS02] Ramon Ferrer i Cancho and Ricard V. Solé. Zipf's Law and random Texts. *Advances in Complex Systems*, 5(1):1–6, 2002.

[JM00] Daniel Jurafsky and James H. Martin. *Speech and Language Processing*. Prentice Hall, 2000.

[MS99] Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, 1999.

[Wen04] Andreas Wendemuth. *Grundlagen der stochastischen Sprachverarbeitung*. Oldenbourg, 2004.

Carsten Siegmund

3 Einführung in Text Mining

Vor dem Hintergrund einer täglich zunehmenden Informationsflut sind Analysewerkzeuge zur Bearbeitung von natürlichsprachlichen Texten ein wichtiger Erfolgsfaktor für Unternehmen geworden. Bislang wurden jedoch überwiegend Datenverarbeitungstechniken für den Umgang mit strukturierten Informationen wie sie in relationalen Datenbanken vorliegen, angewandt. In diesem Kapitel soll im Folgenden eine Einführung in das recht junge Feld des Text Mining gegeben werden, welches nun auch die Verarbeitung großer Mengen unstrukturierter Texte ermöglichen wird.

3.1 Einleitung

Die tägliche Nutzung und der Umgang mit Informationsressourcen werden in der heutigen Zeit als Selbstverständlichkeit angesehen. Zusätzlich ermöglichen Internettechnologien scheinbar die Allgegenwärtigkeit von Informationen und damit unkomplizierte Nutzung dieses Wissens. Mit der erhöhten Zugänglichkeit von Informationen steigen aber auch die Anforderungen an Informationssysteme, von denen eine automatische Generierung und Aufbereitung von Wissen erwartet wird. Vor diesem Hintergrund gewinnen Hilfsmittel zum Umgang mit Informationsquellen zunehmend an Bedeutung (Martin [Mar98, S. 418]).

Sowohl Entwickler als auch Nutzer von Informationssystemen werden dabei mit grundlegenden Problemen konfrontiert. Zum einen stellt sie der Reichtum an zugänglichen und damit nutzbaren Informationsquellen vor ein Auswahlproblem. Insbesondere das Internet als weltweit größte verteilte Textdatenbank wurde bereits im Jahr 2001 konservativ auf ein Volumen von circa einer Milliarde statischer Webseiten und circa 500 Milliarden dynamisch erzeugter Webseiten geschätzt (Sullivan [Sul01, S. 6]). Jedoch ist das Internet nicht die einzige an Volumen stark wachsende Datenbank. Unternehmensinterne Datenbanken und Dokumentenmanagementsysteme wachsen genauso schnell und unaufhaltsam, da insbesondere elektronische Medien immer stärker an Bedeutung gewinnen, was sowohl Unternehmensberichte als auch interne und externe Korrespondenz per Email einschließt. So wurde das gespeicherte Da-

tenvolumen in Unternehmen und Regierungen, wovon circa 80% in Textdokumenten enthalten ist (Tan [Tan98, Tan99]), schon im Jahr 2000 auf insgesamt 1000 Petabyte geschätzt mit deutlichen jährlichen Wachstumsraten (Lycyk [Lyc00]). Dieser Trend exponentiell wachsenden Datenaufkommens wird in der Literatur allgemein als Information Overload bezeichnet.

Information Overload

Eine weitere Problematik stellt die in den meisten Informationsquellen vorliegende natürliche Sprache dar. Liegen Daten strukturiert in einer Datenbank vor, so sind sie für einen Computer leicht lesbar und können durch geeignete Programme in eine Form gebracht werden, die es dem Menschen erlaubt, einen überblick zu gewinnen und Entscheidungen aufgrund des gewonnen Wissens zu fällen. Texte natürlicher Sprache sind jedoch nicht für die automatisierte Bearbeitung durch Computer entworfen (Hearst [Hea03]) und weisen keine dieser typischen Strukturen auf (Sullivan [Sul01, S. 9]).

natürliche Sprache

Viele Unternehmen greifen daher bereits heute auf Werkzeuge zurück, die es erlauben ihre internen Datenbestände zu analysieren. So lassen sich beispielsweise Kundeninformationen aus Customer-Relationship-Management-Systemen und anderen internen Datenbanken zusammenführen und auswerten, um das Kaufverhalten zu untersuchen. Die als Data Mining benannten Anwendungen erlauben den Zugang und die Bearbeitung von strukturierten Informationen, sind jedoch nicht auf unstrukturierten Texten wie sie im Internet zumeist vorliegen sinnvoll einsetzbar (Hoffmann [Hof03]).

Data Mining

Eine Technik, die es erlauben soll, auch Textdatenbanken zu analysieren und Wissen zu extrahieren, nennt man Text Mining. In der folgenden Ausarbeitung wird der Begriff des Text Mining definiert, von anderen Techniken abgegrenzt und zugrunde liegende Methoden vorgestellt. Schließlich soll an Beispielen erläutert werden, wie Text Mining eingesetzt werden kann.

Text Mining

3.2 Definition

Text Mining ist eine relativ neue Disziplin und wie in einer Reihe anderer neuer Disziplinen ist es genauso schwer eine generell akzeptierte Definition für diesen Bereich zu finden als auch eine generell benutzte Bezeichnung einzuführen (Kodratoff [Kod99]). Grob läßt sich die große Menge unterschiedlicher Definitionen in zwei Gruppen aufspalten mit jeweils unterschiedlichen Ansichten und Ausprägungen, die aber auch untereinander eine Reihe von Variationen aufweisen.

Die erste Gruppe von Definitionen beschreibt eine sehr weite Umfassung des Themenbereichs und identifiziert Text Mining als jede Operation, die sich mit dem Sammeln und der Analyse von Text auseinandersetzt. Ein Beispiel für eine Definition aus dieser Gruppe wird von Richard D. Hackathorn [Hac99] gegeben, der Text Mining unter dem Begriff Web Farming verwendet und damit alle solche Techniken zusammenfat, die es dem Benutzer ermöglichen, interessante Informationen aus externen Ressourcen zu extrahieren, was explizit auch das Internet einschließt. Damit wird jedoch weder die Art der Ressourcen eingegrenzt, also beispielsweise zwischen strukturierten Daten und unstrukturierten Texten unterschieden, noch zwischen eingesetzten Techniken unterschieden und auch nicht vorgeschrieben, dass die gefundenen Zusammenhänge zuvor unbekannten sein müssen.

Eine zweite Gruppe von Definitionen setzt genau bei den zuvor nur sehr weit gefassten Grenzen an und definiert Text Mining damit deutlich enger. Eine Definition aus dieser Gruppe wird von Marti Hearst [Hea03] genannt, die Text Mining als Menge von

Techniken zum Entdecken und automatischen Extrahieren von neuen, zuvor unbekannten Informationen aus Texten definiert. Durch diese Definition wurden somit die Art der zu entdeckenden Informationen, nämlich neue und zuvor unbekannte Informationen und auch die Art der Informationsquellen, nämlich unstrukturierte Texte, eingegrenzt. Dadurch kann man Text Mining als eine Extension des Data Mining oder auch der Knowledge Discovery verstehen, die sich mit der Wissensendeckung in Datenbanken beschäftigt (Hearst [Hea99]). Die tatsächlich eingesetzten Technologien und in welchem Zusammenhang sie zueinander stehen wurde damit jedoch noch nicht genannt. Auch wurde weder durch Definitionen aus der ersten noch durch Definitionen der zweiten vorgestellten Gruppe nähere Angaben zu den genauen Aufgaben gemacht, die durch Text Mining übernommen werden sollen. Lediglich aus der gesamten Motivation des Bereiches wurde klar, dass Text Mining ein automatisiertes Werkzeug zur Verfügung stellen soll, um der Informationsflut in der heutigen Zeit gerecht werden zu können.

Das Problem mit der genauen Zuordnung von eingesetzten Techniken im Text Mining ist die Tatsache, dass Text Mining sich einer Reihe von Werkzeugen aus anderen Disziplinen bedient. Typische Vertreter dieser eingesetzten Techniken im Text Mining sind Computerlinguistik, statistische Sprachverarbeitung, Informationsextraktion, Information Retrieval, Mustererkennung, Klassifikation und Clusteranalyse. Da diese Aufzählung aber in keiner Weise Vollständigkeit garantiert sondern im Text Mining sowohl nur Untermengen eingesetzt werden oder aber auch hier nicht aufgeführte Werkzeuge einsetzt werden können, sehen einige Autoren die gemeinsame Verbindung aller Text Mining Werkzeuge in der Art ihrer Anwendung. So schreiben Franke, Nakhaeizadeh und Renz [FNR03] beispielsweise, dass typische Anwendungen in den Bereichen Textsuche, Informationsextraktion und Analyse von Textkollektionen zu suchen sind. Dabei schließt der Bereich Textsuche nicht nur die einfache Suche nach relevanten Texten ein (Information Retrieval) sondern bedient sich auch Verfahren der künstlichen Intelligenz und des Collaborative Filtering. Der Bereich Informationsextraktion soll bei dabei helfen, einzelne Informationen aus Texten herauszulösen und sie dem Benutzer zur Verfügung zu stellen. Das kann die einfache Information beinhalten, ob ein Text relevant für ein bestimmtes Thema ist oder nicht, aber auch die Angabe von Schlüsselbegriffen bis hin zu einer ganzen Zusammenfassung oder die Beantwortung von inhaltlichen Fragen sein. Schließlich umfasst die Analyse von Textkollektionen Techniken der Kategorisierung und des Clustering, um zu entscheiden, zu welcher Klasse ein gegebenes Dokument einzuordnen ist und zwischen welchen Dokumenten welche Art von Zusammenhängen bestehen.

Text Mining Prozess

Die Arten der Aufgaben, die durch Text Mining erfüllt werden sollen, lassen sich wiederum in einen Prozess überführen, wobei jeweils spätere Prozessschritte auf Zwischenergebnisse der vorherigen Prozessschritte zurückgreifen. Techniken können nun diesen Prozessschritten zugeordnet werden, wobei diese Festlegung jedoch aus den oben genannten Gründen in der Regel nicht vollständig zu sein braucht. Eine einfache Beschreibung dieses Text Mining Prozesses wird durch Ari Visa [Vis01] gegeben. Dabei werden die Schritte Datenvorverarbeitung, Zusammenfassung und Kodierung verwendet. Im ersten Schritt werden die Textdokumente in eine Form transferiert, die durch Computersysteme besser verarbeitet werden können als Text. Im nächsten Schritt werden wichtige Informationen extrahiert und Generalisierungen von sehr spezifischen Daten vorgenommen. Im letzten Schritt werden für die vorverarbeiteten und eventuell zusammengefassten Texte eine Kodierung gefunden, die es ermöglichen soll, Beziehungen zwischen den Dokumenten und den in ihnen verwendeten Konzepten erkennen zu können.

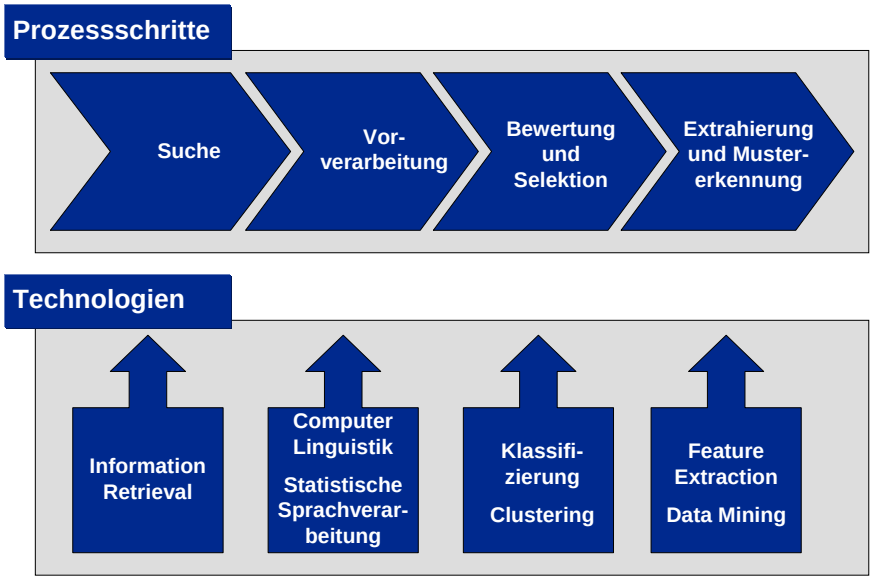


Abbildung 3.1: Text Mining Prozess

Eine etwas genauere Aufschlüsselung dieses Prozesses wird durch Sullivan [Sul01, S. 324] gegeben, der insgesamt vier Prozessschritte einführt: Suche, Vorverabeitung, Bewertung und Extrahierung/Mustererkennung (siehe Abbildung 3.1). Im ersten Schritt werden durch Techniken des Information Retrieval möglicherweise passende bezüglich einer Anfrage gestellte Dokumente gesucht. Diese werden im nächsten Schritt linguistisch so vorverarbeitet, dass die in Sprache verwendete generische Struktur aus Sätzen, Phrasen und Worten durch automatisierte Datenverarbeitung erkannt und verarbeitet werden können. Im dritten Schritt werden die so vorverarbeiteten Dokumente miteinander in Beziehung gesetzt bevor im letzten Schritt gezielt Informationen extrahiert werden, auf denen dann Mustererkennungsalgorithmen eingesetzt und Kausalketten erstellt werden können.

Im Folgenden werde ich mich an die enge Definition von Hearst [Hea03] halten und die am gebräuchlichsten eingesetzten Techniken im Text Mining am Beispiel des Prozesses von Sullivan [Sul01, S. 324] erläutern.

3.3 Suche

Im ersten Prozessschritt der Suche sollen aus einer großen Menge von Dokumenten, die gegeben einer Benutzeranfrage eine Teilmenge möglichst relevanter Dokumente zurückliefert. Eine typischerweise eingesetzte Technologie ist das in Suchmaschinen oft verwendete Information Retrieval. Ziel eines solchen Systems ist es, möglichst viele relevante und möglichst wenig irrelevante Dokumente als Antwort auf eine Suchanfrage zu geben. Dabei liegt der Fokus auf natürlichsprachlichen Texten, die meist keine definierte Struktur besitzen (Schwarz [Sch04b]).

3.3.1 Information Retrieval

Der Bereich des Information Retrievals hat sich lange Jahre parallel zu der Entwicklung von Datenbanksystemen entwickelt (Han et al. [HK99, S. 428]). Im Gegensatz zu Datenbanksystemen, bei denen man sich auf Anfragen und Transaktionsdurchführung auf strukturierten Daten konzentriert, beschäftigt sich das Information Retrieval mit der Organisation und dem Auffinden von Informationen aus einer großen Ansammlung von textbasierten Dokumenten. Ein Information Retrieval System arbeitet dabei ähnlich wie ein Filter auf der Gesamtmenge von Dokumenten, die zum Beispiel in Unternehmensdatenbanken oder in Teilen des Internets verfügbar sind. Ziel dieses Prozesses ist es allerdings noch nicht die Menge von Dokumenten bereits soweit zu reduzieren, dass ein menschlicher Endbenutzer das Ergebnis seiner Text Mining Anfrage bereits durch Lesen der verbliebenen Texte erhält, sondern lediglich das ausfiltern von allen von den Dokumenten, die wahrscheinlich bei der weiteren Bearbeitung der Anfrage nicht hilfreich sind (Sullivan [Sul01, S. 327]).

Im Information Retrieval werden verschiedene Modelle zur Darstellung großer Mengen von Text verwendet, die es erlauben, Dokumente zu bestimmten Themenbereichen zu finden. Im Folgenden wird genauer dargelegt werden, wie in diesem Bereich Dokumente repräsentiert werden und wie relevante Dokumente zu einem gegebenen Thema identifiziert werden. Dabei werden zwei Methoden im Detail vorgestellt werden: das Vektorraum-Modell und das Latent Semantic Indexing. Während die erste Methode ein Grundrepräsentationsschema für viele Information Retrieval Techniken darstellt, ist die zweite Technik eine Erweiterung, die aufgrund der Limitierungen des Vektorraummodells entstanden ist und insbesondere den Problemen der Synonyme und der Polyseme vorbeugen soll.

Vektorraummodell

Das Vektorraummodell wurde Anfang der 60er-Jahre von Gerard Salton [Sal83] entwickelt und löst alle drei Hauptaufgaben des Information Retrievals: Die Repräsentation von Dokumenten, die Repräsentation von Anfragen und das Auffinden von Dokumenten, die einer gegebenen Anfrage genügen. Zur Lösung der ersten beiden Probleme werden geometrische Darstellungen verwendet, die Lösung des dritten Problems ergibt sich dann unmittelbar aus dieser Darstellungsart.

Darstellung von Dokumenten und Suchanfrage

Um eine Anfrage nach Dokumenten performant durchführen zu können, sollte sie in gleicher Weise wie die Dokumente dargestellt werden, da dann eine Anfrage nicht in einen Ausführungsplan kompiliert werden muss, wie es bei relationalen Datenbankanfragen der Fall wäre. Dabei wäre ein besonders effektiver Ansatz, wenn man nach Schlüsselbegriffe in Dokumenten nicht sequentiell sucht, sondern in einer Operation bearbeitet. Das kann natürlich nur dann erreicht werden, wenn die Repräsentation einer Anfrage durch ein einfaches Objekt erfolgen kann. Stellt man die Dokumente so dar, dass für jedes enthaltene Schlüsselwort eine Dimension im Vektorraum vorhanden ist, so lässt sich jedes Dokument durch einen Vektor repräsentieren, der in jeder Dimension entweder eine Eins für Schlüsselwert enthalten oder eine Null für Schlüsselwert nicht enthalten aufweist. In gleicher Weise können auch Anfragen repräsentiert werden unabhängig davon wie viele Dimensionen durch den Vektorraum aufgespannt werden. Das Gewicht der einzelnen Begriffe kann auf unterschiedliche Weise berechnet werden, jedoch basieren die meisten von diesen auf der Termfrequenz innerhalb des Dokumentes und der gesamten Frequenz dieses Begriffes in der Dokumentenkollektion.

Vektorrepräsentation

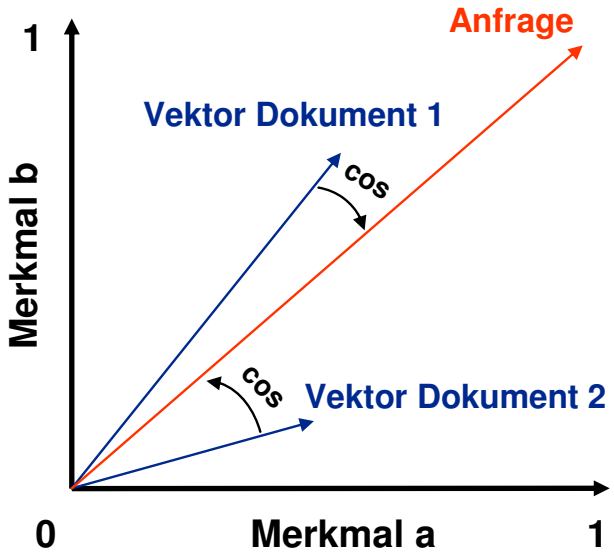


Abbildung 3.2: Vektorraummodell

geometrisches Maß

Da der Vektorraum eine geometrische Repräsentation von Dokumenten und Anfragen ist, ist auch das Maß, wie ähnlich ein Dokument einer Anfrage ist, geometrisch zu interpretieren. Eine gewöhnliche Technik, die ähnlichkeit eines Dokuments zu einer Anfrage festzustellen, ist die Messung des Winkels zwischen den Linien, der zwischen dem Nullpunkt des Vektorraums und dem Punkt des Dokuments bzw. der Anfrage gezogen werden kann (siehe Abbildung 3.2). üblichweise wird als ähnlichkeitsmaß der Kosinus des Winkels der beiden Vektoren genommen (Sullivan [Sul01, S. 333]).

Bei einer gegebenen Anfrage werden nun alle solche Dokumente als Ergebnis der Suchanfrage zurückgegeben, deren Winkel zwischen Dokumentenvektor und Anfragevektor unterhalb einer vorgegebenen Grenze liegen. In der Praxis ist es jedoch üblich, dass es extrem hohe Mengen an Termen gibt, die in den Dokumenten auftreten können, von denen viele natürlich nicht in jedem Dokument vorkommen. Das führt dazu, dass das Modell in einigen Bereichen sehr dünn besiedelt ist, da die Begriffe, die in diesen Dokumenten auftreten, selten auftreten, während Dokumente in anderen Bereichen mit oft verwendeten Termen Cluster bilden. Die Problematik, die sich ergibt, ist dass mit einer sehr hohen Anzahl von Dimensionen gearbeitet werden muss, wobei auch eine Ausdünnung durch das Entfernen von Stoppworten, also inhaltsleeren Worten wie Artikeln, Konjunktionen oder Hilfsverben, nicht problemlösend wirkt (Carstensen et al. [CEE+04, S. 484]). Zum anderen tritt das Problem auf, dass unterschiedliche Worte mit derselben Bedeutung (Synonyme) in unterschiedliche Bereiche des Vektorraums abgebildet werden und dass gleiche Worte mit unterschiedlichen Bedeutungen (Polyseme) auf denselben Bereich abgebildet werden (Sullivan [Sul01, S. 337]).

Synonyme

Polyseme

Latent Semantic Indexing

Ein Ansatz, der sich mit der Lösung dieser Probleme beschäftigt, wird Latent Semantic Indexing genannt und beruht auf der Erkenntnis von drei Schwächen des herkömmlichen Ansatzes (Deerwester [DDF+90]). Das ist zum einen das Vorliegen von

unvollständigen Indizes, die daraus resultieren, dass ein Index meist deutlich weniger Terme enthält als der Benutzer erwartet vorzufinden. Das kann daran liegen, dass die Terme entweder nicht in dem Dokument vorkommen oder aber durch bestimmte Filterkriterien des Information Retrieval Systems nicht berücksichtigt wurden.

Ein weiteres Problem liegt in dem bereits aufgeführten Problem mit Polysemen, welches auch auf verschieden einfache Arten angegangen werden könnte. Es könnten beispielsweise nur bestimmte Wörterbücher zugelassen werden, in denen Wörter mit mehreren Bedeutungen nicht zugelassen werden oder dem Benutzer die Möglichkeit gegeben werden durch den Zusatz weiterer Begriffe klarzustellen, welche Bedeutung gemeint ist. Beide Ansätze scheitern jedoch neben der Einschränkung auf einen bestimmen Wortschatz an einem zu großen Arbeitsaufwand, der bei der Erstellung der Wörterbücher oder der Spezifizierung der Bedeutung durch den Benutzer anfallen würde.

Schließlich liegt ein weiteres Problem in dem Umstand, dass Terme, die regelmäßig mit bestimmten anderen Termen gemeinsam auftreten nicht anders behandelt werden als solche Terme, die selten mit anderen Termen gemeinsam auftreten. So sollten gewöhnliche Kombinationen höher bewertet werden als sehr seltene, da eine zu starke Berücksichtigung der seltenen Kombinationen in den meisten Fällen auch beim Suchergebnis zu ungewollten Ergebnissen führen kann (Deerwester et al. [DDF+90]).

Latent Semantic Indexing setzt gleich bei allen drei Problemen an (Landauer [LD96]). Zwar wird bei diesem Verfahren nicht die inhaltliche Bedeutung eines Wortes oder eines Konzeptes berücksichtigt, dafür werden jedoch Verfahren der Statistik und der linearen Algebra eingesetzt, um Termcluster zu finden, die bestimmte Konzepte beschreiben. Die Cluster werden nicht vorgegeben sondern werden anhand der Wahrscheinlichkeit für das Auftreten eines Terms gegeben eines anderen Terms geschätzt. Dadurch können Konzepte approximiert werden, die regelmäßig gemeinsam auftretende Terme beschreiben. Während in einem einfachen Vektorraummodell alle Begriffe verschiedene Dimensionen benötigen, wird genau durch das Zusammenführen von Dimensionen zu einem gemeinsamen Konzept so viel Genauigkeit verloren, dass nun auch Dokumente, die nur einen Teil der Begriffe beinhalten eine Suchanfrage erfüllen.

Termcluster

Durch die Reduzierung der Dimensionen kann aber nicht nur der Einfluss eines fehlenden Terms reduziert werden, sondern gleichzeitig auch das Problem der Polysemie gelöst werden, da Gruppen aus statistisch zusammenhängenden Termen eine gemeinsame Dimension bilden und die übrigen zusammengefassten Terme den Sinn des Polysems erklären. Weiterhin kann durch die Reduzierung der Dimensionen der Platz für Indizes verringert werden (Han et al. [HK99], S. 431*f*).

3.4 Vorverarbeitung

Das Information Retrieval stellt dem Benutzer trotz einer Vorsortierung von eventuell relevanten Dokumenten immer noch eine zu große Menge von Dokumenten zur Verfügung, als dass dieser alle lesen und bearbeiten könnte. Daher werden Werkzeuge benötigt, die Strukturen innerhalb eines Textes erkennen um die Möglichkeit zu geben in späteren Schritten genau die gesuchten Informationen extrahieren können. Ansätze, die dazu im Weiteren näher betrachtet werden sollen, sind die Computerlinguistik, die statistische Sprachverarbeitung und die Ausnutzung von Makrostrukturen in Texten.

3.4.1 Computerlinguistik

generische Struktur

Natürlichsprachlicher Text wird meist als unstrukturiert angesehen, da er keine Struktur besitzt, wie es aus Datenbanken bekannt ist. Jedoch besitzt Text eine generische Struktur aus Worten, Phrasen und Sätzen und mit einem Verständnis darüber, wie Worte, Phrasen und Sätze strukturiert sind, kann auch Text automatisiert bearbeitet werden und effektiver Informationen extrahieren werden, als es durch einfache String Manipulation und Mustererkennungstechniken möglich wäre (Sullivan [Sul01, S. 31*f*]).

Natürliche Sprache ist aus Worten aufgebaut und es existieren Regeln, wie diese Worte angeordnet werden können. Aus Sicht des Text Minings ist es also notwendig Systeme zu entwickeln, die sowohl diese Regeln als auch Wortbedeutungen ausnutzen um Texte zu bearbeiten. Von den verschiedenen Bereichen der Computerlinguistik haben insbesondere die Bereiche Morphologie, Syntax und Semantik für das Text Mining eine hohe Bedeutung.

Morphologie

Wortstamm
Affixe
Flexione

Worte sind aufgebaut aus Wortstamm, Affixen (Präfixe und Suffixe) und Flexionen. Der Wortstamm ist der Kern eines jeden Wortes und meist selbst ein Wort. Die Bedeutung eines Wortstammes wird durch die Benutzung von Affixen verändert. Flexionen schließlich sind Beugungen eines Wortes und verändern Zeiten und Anzahl. Die morphologische Analyse eines Textes unterstützt das Text Mining dabei die Komplexität der Analyse zu verringern und die Wortbedeutungen zu repräsentieren (Williams [Wil00, S. 10]).

Damit Dokumente überhaupt informationstechnisch bearbeitet werden können, müssen sie so aufbereitet werden, dass aus dem Strom von Zeichen sprachlich relevante Einheiten (zum Beispiel Wörter, Phrasen und Sätze) extrahiert werden. Dazu werden die einzelnen Einheiten (Tokens) im Schritt der so genannten Tokenisierung zunächst aus dem Text herausgelöst. An diesen Schritt folgt zumeist eine Anreicherung der zuvor selektierten Token um grammatische Informationen. Diese Klassifizierung einer jeden Einheit des Textes gemäß seiner Wortart erfolgt durch einen Part-of-Speech Tagger (POS-Tagger), der die zusätzlichen Metainformationen in Tags an die Token anhängt (Williams [Wil00, S. 10]). Schließlich werden die so vorverarbeiteten Worte durch einen Chunk-Parser zu phrasalen Strukturen zusammengefasst. Dieses Verfahren liefert dabei keine vollständigen syntaktischen Strukturen, die hierarchische Beziehungen repräsentieren, sondern identifizieren nebengeordnete Teilstrukturen (chunks). Diese Informationen werden durch phrasale Tags ebenfalls dem Text angehängt und steht damit nachfolgenden Techniken zur Verfügung (Carstensen et al. [CEE+04, S. 218*ff*]).

Tokenisierung
Part-of-Speech-Tagger
Chunk-Parser

Bereits durch die Rückführung der Worte zu Wortstämmen kann die Komplexität einer Textanalyse deutlich verringert werden, da die Anzahl einzelner Wortvorkommen spezifiziert werden kann, was bereits ein guter Indikator dafür ist, wie wichtig ein Thema in einem Dokument ist. Weiterhin erlaubt die Benutzung der Morphologie die Implementierung von Werkzeugen wie Wörterbüchern und Lexika und das Erkennen zusammengehöriger Worte und Wortphrasen (zum Beispiel mehrwortige Eigennamen) (Sullivan [Sul01, S. 35]).

Eine morphologische Analyse kann jedoch noch weitere Informationen als den Stamm eines Wortes hervorbringen (Sullivan [Sul01, S. 342*ff*]). Ein Wort ist die kleinste linguistische Einheit, die im grammatikalischen Sinne allein stehen kann. Worte können

aus Morphemen aufgebaut sein, die die kleinste Einheit darstellen, die eine Bedeutung tragen können. Es lässt sich unterscheiden zwischen gebundenen und freien Morphemen, je nach dem ob sie alleine stehend als Wort im Text vorkommen können oder aber als Präfix oder Suffix an ein Wort gebunden werden müssen. Weiterhin können wir unterscheiden zwischen inhaltlichen und funktionalen Morphemen. Während inhaltliche Morpheme ihre Bedeutung unabhängig davon tragen wie die Grammatik ist und typischerweise Wortstämme sind, helfen funktionale Morpheme dabei, Worte einer Grammatik anzupassen. Eine weitere Unterscheidung zwischen Morphemen ist ob sie flektiert oder abgeleitet sind. Abgeleitete Morpheme bilden neue Worte und wandeln zum Beispiel ein Verb in ein Substantiv um. Flektierte Morpheme hingegen kreieren keine neuen Worte sondern erweitern den Wortstamm eines Wortes zur Anpassung an eine grammatikalische Anforderung.

Morphem

Im Text Mining ist es wichtig, Wortstämme zu identifizieren um eine bessere Frequenzanalyse vornehmen zu können sowie Worte nach ihrer grammatikalischen Funktion zu identifizieren. Inhaltliche Morpheme liefern die gesuchten Wortstämme zur Frequenzanalyse, die Kombination von inhaltlichen und funktionalen Morphemen erlaubt es, die grammatikalische Information eines Wortes zu finden.

Syntax

Während die Morphologie unmittelbare praktische Implikationen für die Analyse von Worten hat, kann der Syntax dieselbe Funktion für Phrasen und Sätze haben, da die Regeln der Linguistik beschreiben, wie Worte in Phrasen und Sätzen kombiniert werden können. Es können sowohl Substantiv-, Verb-, Präpositional- und Adjektivphrasen erkannt werden. Diese Phrasen können dann Sätze oder kompliziertere Phrasen bilden. Durch syntaktische Regeln können einzelne Phrasen dann in eine hierarchische Form gebracht werden, die symbolisieren, in welchem Verhältnis sie zueinander stehen und sich gegenseitig modifizieren (Carstensen et al. [CEE+04, S. 232*ff*]).

Phrasen

Durch Analyse des Verhältnisses zwischen Verb- und Substantivphrasen kann die Rolle von Substantiven in einem Satz festgestellt werden. ähnlich wie Wortstämme in Worten die Möglichkeiten der benutzbaren Affixe beeinflussen, limitieren Verben in Sätzen die Anzahl und Art von Substantiven, die in einem Satz benutzt werden können. Diese Informationen (case assignments) werden in Lexika gespeichert und werden dazu verwendet nach Mustern zu suchen. Unter Verwendung von morphologischen Informationen können syntaktische Regeln helfen, Strukturen in Form von Wort- und Phrasenmustern zu erkennen und bilden damit die Basis für die semantische Analyse (Sullivan [Sul01, S. 35*ff*]).

Case Assignments

Semantik

Die Semantik ist die Teildisziplin der Linguistik, die sich mit der Bedeutung natürlichsprachlicher Ausdrücke beschäftigt. In der Semantik geht es um die Bedeutung von Worten (lexikalische Semantik), Sätzen (Satzsemantik) und Texten (Diskurssemantik) (Carstensen et al. [CEE+04, S. 276*ff*]).

Eine sinnvolle Repräsentation der Bedeutung sollte sowohl platzeffizient sein als auch Programmen erlauben schnelle Entscheidungen treffen zu können. Ein möglicher Ansatz, der diese Vorgaben erfüllen kann, ist der Einsatz eines semantischen Netzwerks. Semantische Netzwerke benutzen Knoten und Pfeile um Objekte, Events und Konzepte und deren Beziehungen zueinander darzustellen. Diese Art Netzwerke sind nützlich für die Klassifizierung und Generalisierung von Themen, die notwendig

Semanisches Netzwerk

bei Suche nach Themen anstelle von Schlüsselbegriffen sind. Ein Problem ist jedoch, dass generalisierte semantische Netzwerke mit einem reichen Vokabelschatz von Beziehungen schwierig aufzubauen sind und nur in limitierenden Domänen arbeiten (Sullivan [Sul01, S. 37*ff*]).

3.4.2 Statistische Sprachverarbeitung

Textbearbeitungstechnologien basierend auf Morphologie, Syntax und Semantik sind mächtige Werkzeuge zur Extraktion von Informationen aus Texten. Sie erlauben das Finden von Dokumenten auf Basis von Themen oder Schlüsselbegriffe, Texte können nach sinnvollen Phrasenmustern gescannt werden und Schlüsselmerkmale und ihre Beziehungen extrahiert werden. Zudem können Dokumente auf einfache Weise so gespeichert werden, dass sie eine einfache Navigation weit über die Möglichkeiten von Information Retrieval Techniken ermöglichen und die weiterführende Extraktion von Informationen erlauben.

Der Einsatz dieser vorgestellten Techniken hat jedoch auch seine Grenzen. Zu den Problemen gehören unter anderen die korrekte Identifikation von Rollen identifizierter Substantivphrasen, die eine korrekte Extraktion von Informationen beeinflusst, und die Repräsentation von abstrakten Konzepten. Während sich semantische Netzwerke gut zur Darstellung von Komponenten- (Kompositionen und Aggregationen) und Teilmengenbeziehungen (Vererbung) eignen, ist es sehr viel schwieriger, Herleitungen darzustellen ohne einen zu hohen Grad an Komplexität zu überschreiten. Weiterhin problematisch sind Synonyme und spezialisierte Domänen, in denen viele verschiedene Begriffe sehr ähnliche Konzepte beschreiben. Schließlich benötigt ein generelles Klassifikationssystem zu viele Konzepte, um wirklich alle möglichen Themengebiete klassifizieren zu können. Dadurch steigt die Anzahl von Konzepten so sehr an, dass eine gleichzeitige Repräsentation nicht mehr möglich wird. Einige dieser Probleme der natürlichen Sprachverarbeitung können durch die Verwendung von statistischen Techniken begegnet werden, bei denen die Ergebnisse der linguistischen Analyse mit einfachen statistischen Maßen kombiniert werden (Williams [Wil00, S. 9]).

Beziehungen

Eine übliche Aufgabe im Text Mining ist die automatisierte Erstellung von Textzusammenfassungen. Diese Aufgabe kann wie bereits dargestellt durch das Finden der signifikantesten Konzepte in einem semantischen Netz angegangen werden. Ein anderer Ansatz, der ohne semantische Netze und deren Limitierungen auskommt, benutzt Wortfrequenzen, um die wichtigsten Konzepte eines Textes zu finden. Die Wichtigkeit eines Wortes kann dann bereits durch das einfache Zählen gemeinsam verwendeter Wortstämme ermittelt werden und anschließend die Wichtigkeit eines Satzes durch die Wichtigkeit der darin vorkommenden Worte. Eine einfache, aber dennoch effektive Zusammenfassung eines Textes könnte dann bereits die Extraktion der so ermittelten wichtigsten Sätze sein.

Wortfrequenzen

Ein weiterer Ansatz, der linguistische mit statistischen Techniken kombiniert, vereinfacht die Konstruktion von semantischen Netzen. In einem gewöhnlichen semantischen Netz repräsentiert jeder Knoten ein Wort oder einen Term und die Pfeile beschreiben die Beziehung dieser Knoten. Diese Pfeile können nun dazu verwendet werden, ein Grad der Korrelation zwischen Knoten zu berechnen. Die Korrelation misst, wie oft Worte nebeneinander verwendet werden. Während bei dieser Methode noch nicht die volle Bedeutung eines Textes repräsentiert werden kann, so kann doch die Wichtigkeit von Themen zueinander identifiziert werden. Und anstelle einfach nur Worthäufigkeiten zu zählen, könnte eine Anwendung zusätzlich in Betracht ziehen, wie andere

Terme zu diesen im Verhältnis stehen.

Generell sind kleine Texte wie zum Beispiel einzelne Nachrichtentexte zugänglich zu linguistischen Ansätzen, statistische Verfahren eignen sich besonders bei großen Textansammlungen wie zum Beispiel bei Newsgroups oder Zeitungsarchiven.

3.4.3 Makrostrukturen

Bislang haben alle vorgestellten Techniken jeden Teil eines Textes gleichsam behandelt. Das kann dann problematisch sein, wenn ein großer Text aus verschiedenen Sektionen mit unterschiedlichen Inhalten und Schwerpunkten besteht. Würde von einem solchen Text eine Zusammenfassung erstellt werden und nur die Sätze mit den am häufigsten verwendeten Begriffen berücksichtigt werden, so würde die Zusammenfassung signifikant mehr Informationen über die längeren Sektionen enthalten. In vielen Texten sind jedoch einige Abschnitte wichtiger als andere unabhängig von ihrer Länge. Weiterhin unberücksichtigt blieb bislang, dass einige Arten von Texten Informationen für verschiedene Zielgruppen beinhalten können wie zum Beispiel wöchentliche Memos oder Reports. Die verschiedenen Inhalte haben daher auch für verschiedene Benutzer eine völlig unterschiedliche Wichtigkeit, so dass dann einfache statistische Maße bei der Textanalyse ungeeignet sind.

Im Gegensatz zur Sprachebene eines Textes, die eine Mikrostruktur darstellt, existiert meist auch eine künstlich erzeugte Makrostruktur, um große Mengen von Text besser zu strukturieren. Zu solchen Makrostrukturen zählen zum einen Unterteilungen wie zum Beispiel Kapitel und überschriften, aber auch Informationen zur Darstellung oder Bedeutung von Textelementen, die in Tags wie zum Beispiel bei XML oder HTML eingeschlossen sind. Zudem werden insbesondere im Internet Verweise zu anderen Dokumenten verwendet, die einer leichteren Navigation zwischen relevanten Texten dienen und ebenfalls zur Analyse der Wichtigkeit eines Dokuments verwendet werden können. Dabei wird gemessen, wie viele Verweise von einem Dokument (Hub) auf andere Dokumente gesetzt sind und wie vielen Verweise von anderen Dokumenten auf ein einzelnes Dokument (Authority) zeigen. Diese Technik wird beispielsweise in der Internetsuchmaschine Google unter dem Namen Pagerank eingesetzt, um Suchergebnisse zusätzlich zu der Gewichtung der Häufigkeit der Suchbegriffe auch nach Wichtigkeit, die sich aus der gegebenen Linkstruktur ergibt, zu sortieren (Han [HK99, S. 437*ff*]).

Hubs und Authorities

3.5 Bewertung und Selektion

Nachdem Dokumente identifiziert wurden und ihre sprachliche Struktur erkannt worden ist, können sie je nach Anforderungen von der Benutzerseite Themengebieten zugeordnet werden und mit ähnlichen Dokumenten gruppiert werden. Damit stehen sie auch späteren Suchoperationen weiterhin zur Verfügung.

3.5.1 Klassifikation

Durch die Untersuchung von Wortmustern und behandelten Themen in einem Text, können Dokumente in grobe Partitionen unterteilt und vordefinierten Gruppen zugeordnet werden. Generell gibt es zwei Arten der Klassifikation: Labeling und Multidimensionale Taxonomien.

Labeling

Labeling beschreibt den Prozess des Hinzufügens eines Themas oder einer Beschreibung (Label) zu einem gegebenen Dokument. Die gewählten Bezeichnungen können dabei domänenbezogen sein oder generelle Themengebiete beinhalten. Für eine feinere Einteilung können auch mehrere Bezeichner verwendet werden, deren Zugehörigkeitsgrad durch ein Gewicht ausgedrückt wird. Die Labels und zugehörigen Gewichte können bei Textanfragetools dazu verwendet werden um eine minimale Grenze bei der Suche nach Dokumenten zu beschreiben. Erfolgreiches automatisches Labeling hängt dabei von besonders von den Faktoren Wortfrequenzstatistiken, morphologisches Wissen und typenspezifische Terme ab (Sullivan [Sul01, S. 199]).

Wortfrequenzstatistiken

Bei den Wortfrequenzstatistiken unterscheidet man zwischen der relativen Frequenz und der absoluten Frequenz. Während die relative Frequenz die Anzahl bestimmt, wie oft ein Wort in einem Dokument vorkommt, misst die absolute Frequenz, wie oft ein Wort in einer Menge von Dokumenten auftritt. Morphologisches Wissen wird verwendet um Variationen zu eliminieren, die in einer Sprache durch die Verwendung von zum Beispiel Deklinationen und Konjugationen auftreten können. Daher spielt es auch keine Rolle, in welcher Form der Stamm eines Wortes verändert wird, da alle Variationen als der derselbe Term gewertet werden. Typenspezifische Terme sind zum Beispiel Städtenamen und Länder, übliche Abkürzungen oder Namen von Personen und Unternehmen und werden dazu verwendet, um Lexika und Thesauri zu erweitern.

Sobald die morphologische Analyse Worte in eine Standardform gebracht haben, können die relativen Frequenzen bestimmt werden. Das üblichste Gewicht zur Bestimmung des Gewichts eines Terms innerhalb eines Dokumentes ist die inverse Dokumentenfrequenz. Die Idee hinter diesem Maß ist, dass hohe Gewichte genau solchen Termen zugeordnet werden sollten, die nur in wenigen Dokumenten auftreten und sie damit sehr gut unterscheiden. Da die relative Frequenz misst, wie oft ein Term in einem Dokument auftritt, ist dieses Gewicht proportional zum relativen Gewicht. Terme die in vielen Dokumenten auftreten haben eine hohe absolute Frequenz und eine schwache Diskriminierung. In diesen Fällen ist das Gewicht invers proportional zur absoluten Frequenz (Williams [Wil00, S. 9]).

inverse Dokumentenfrequenz

Die Kombination von Termen und Gewichten hat sich als gute Technologie zur Klassifikation von Texten herausgestellt. Eine Limitierung dieses Ansatzes ist jedoch, dass es nicht generalisiert. So werden Dokumente, die zum Beispiel die Begriffe Bus, Bahn oder Auto zugeordnet sind, nicht gleichzeitig auch dem Begriff Bodentransport zugeordnet.

Multidimensionale Taxonomien

Multidimensionale Klassifikationsstrukturen erlauben es zunächst nach groben Strukturen zu suchen und dann Dokumentensets nach und nach weiter einzuschränken. Gegeben eine Taxonomie können Dokumente nach spezifischen Termen klassifiziert werden und nach hierarchischen Kategorien zugeordnet werden. Der Netzeffekt ist vergleichbar zu Drilling-Up-Hierarchien in einem multidimensionalen Data Warehouse (Sullivan [Sul01, S. 200f]).

Mit Taxonomien ist es sinnvoll zwischen der Intention und der Extension eines Terms zu unterscheiden. Die Intention eines Terms beschreibt den Term abstrakt und relativiert ihn zu anderen. So ist zum Beispiel ein Automobil eine Art von Bodentransport. Die Extension eines Terms ist die Menge von Dokumenten, der von diesem

Term handelt. Die Extension deutet also auf eine Menge von Dokumenten, die ein bestimmtes Konzept instanziieren.

Multidimensionale Hierarchien klassifizieren spezielle Dokumente (Extension) in mehrere Kategoriestufen einer Generalisierung (Intention) und liefern also ein reicheres Klassifikationsschema als Labeling allein.

Extension
Intension

3.5.2 Clustering

Das Clustern von Dokumenten kann zum schnellen Auffinden von ähnlichen Dokumenten verwendet werden und nutzt die Makrostrukturen einer großen Kollektion von Dokumenten. Clustering kann ferner dazu verwendet werden, Duplikate zu erkennen. Im Gegensatz zur Klassifikation verwendet das Clustering keine vordefinierte Menge von Termen oder Taxonomien, die zum Gruppieren der Dokumente verwendet werden. Stattdessen werden Gruppen auf Basis der Dokumentenmerkmale erstellt, die in einer zu clusternden Menge von Dokumenten vorkommen. Beim Clustering kommen hauptsächlich die Techniken des Binären Clustering (Binary Relational Clustering) und des Hierarchischen Clustering (Hierarchical Clustering) zum Einsatz (siehe Abbildung 3.3).

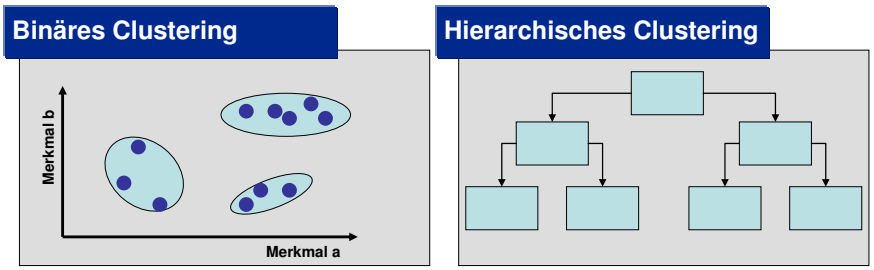


Abbildung 3.3: Binäres und Hierarchisches Clustering

Binäres Clustering

Binäres Clustering partitioniert eine Menge von Dokumenten in Gruppen, wobei jedes Dokument genau einer Gruppe zugeordnet ist. Dabei wird ein ähnlichkeitsmaß innerhalb eines Clusters maximiert und ein ähnlichkeitsmaß zwischen Dokumenten unterschiedlicher Cluster minimiert.

Typisch für das Binäre Clustering ist, dass die Cluster flach sind. ähnlich wie das Labeling genau eine Klassifikation für ein Dokument vornimmt, wird ein Dokument auch nur einem Cluster zugeordnet. Jedes Cluster beschreibt genau ein Thema, das zu einer Menge von Merkmalen gehört, die alle Dokumente in diesem Cluster gemeinsam haben. Meist werden die Dokumente einem aus einer festgelegten Anzahl von Clustern zugeordnet, die Verteilung muss jedoch nicht gleichmäßig sein (Carstensen et al. [CEE+04, S. 493]).

Hierarchisches Clustering

Hierarchisches Clustering gruppiert Dokumente gemäß eines ähnlichkeitsmaßes in einer Baumstruktur. So können Dokumente einer Vielzahl von Clustern in einer hier-

archischen Struktur angehören. Anstelle nur ein Cluster zu finden, dass zu einem Dokument am besten passt, gruppieren hierarchische Clusteralgorithmen Dokumente iterativ in größere Cluster (Carstensen et al. [CEE+04, S. 494]).

Clusterbaum

Im einfachsten Fall ist so jedes Dokument zu Beginn in einem eigenen Cluster, welches ein Blatt in einem Clusterbaum darstellt. Im nächsten Schritt werden jeweils zwei möglichst ähnliche Cluster zusammengefasst. Dieser Prozess geht fort bis schließlich alle kleinen Cluster einem einzigen großen Cluster aller Dokumente zugeordnet sind.

Drill-down und Roll-up

Der große Vorteil des hierarchischen Clusterns ist die Unterstützung von Browsing durch Drill-down und Roll-up-Operationen.

3.6 Mustererkennung und Informationsextraktion

Mustererkennung beschreibt den Prozess der Suche nach zuvor definierten Sequenzen im Text. Dabei werden sowohl Worte als auch morphologisch syntaktische Einheiten als Eingaben berücksichtigt und nicht regulären Ausdrücken, wie es bei vielen Programmiersprachen der Fall ist. Dabei werden im Folgenden die Techniken Word and Term Matching und Relevancy Signatures näher betrachtet. Während Word and Term Matching deutlich einfacher implementiert ist und eine manuelle Steuerung erforderlich macht, arbeitet der Ansatz mit Relevancy Signatures vollständig automatisch auf morphologischen und syntaktischen Informationen, die durch den Part-of-Speech-Tagger geliefert werden, und kann direkt zusammen mit Programmen zur Informationsextraktion verwendet werden (Riloff et al. [RL94]).

3.6.1 Word and Term Matching

Die einfache Suche nach Schlüsselbegriffen ist oft limitiert durch schlechte Precision und Recall, die unter anderem durch das Auftreten von Polysemen und Synonymen hervorgerufen werden. Daher stellt sich die Frage, warum überhaupt auf einfache Systeme zurückgegriffen wird, die sich der Technik des einfachen Vergleichs von Worten und Termen bedienen. In diesem Fall jedoch wird nicht nur einfach nach bestimmten Begriffen in einem einzelnen Dokument gesucht sondern es kann nach Korrelationen zwischen Termen innerhalb einer Gruppe von Dokumenten gesucht werden. Diese

Kausalketten

Korrelationen können dann wieder dazu verwendet werden, Kausalketten und andere Zusammenhänge zwischen Konzepten innerhalb einer Menge von Dokumenten zu entdecken (Hearst [Hea03]).

Sinnvoll ist ein solcher Ansatz zum Beispiel im Bereich der Forschung, wo Experten auf Grund der Masse von Publikationen nicht in der Lage sind zu verfolgen, welche Entwicklungen sich in verwandten Forschungsgebieten ergeben haben. In diesen Fällen sollte es möglich sein, nützliche Verweise zwischen Informationen in verwandter Literatur zu entdecken, auch wenn die Verfasser der Arbeiten keinen expliziten Verweis gesetzt haben. So konnten Swanson et al. [SS94, SS97] zeigen, wie Kausalketten innerhalb einer Menge medizinischer Literatur zu neuen Hypothesen für die Verursachung seltener Krankheiten dienen können.

Je nach Begriff unterscheidet sich jedoch die Anzahl von korrelierenden Termen in einem enormen Maß insbesondere dann, wenn sehr allgemeine Begriffe oder Bezeichnungen mit einer Vielzahl unterschiedliche Verbindungen aufgenommen werden. Um in einem solchen Fall einer kombinatorischen Explosion vorzubeugen können Schwellenwerte und Termeliminierungen eingesetzt werden.

Schwellenwert

Bei der Verwendung eines Schwellenwertes wird eine minimale Korrelation zwischen

zwei Termen festgelegt, so dass Terme erst bei einer überschreitung dieses Wertes in die Liste von verwandten Termen aufgenommen werden. Dies kann einfach durch statistische Techniken erreicht werden. Termeliminierung hingegen setzt einen etwas komplizierteren Mechanismus voraus. So werden Terme aufgrund anderer Kriterien als Korrelationsgewichten entfernt wie zum Beispiel Stopp-Worten, die sich in Versuchen nicht als sinnvoll erwiesen haben (Swanson et al. [SS99]). Die hier verwendete Liste beinhaltet unter anderem Präpositionen, Konjunktionen, Artikel und andere Worte, die keinen Inhalt tragen sondern eine grammatische Funktion erfüllen. Zudem kann das Ergebnis eines Part-of-Speech-Taggers verwendet werden, solche Wortgruppen zu eliminieren, die nicht als relevant betrachtet werden. In vielen Fällen werden Substantive verwendet um Korrelationen zu anderen Substantiven zu finden (zum Beispiel Diebstahl und Kreditkartenmissbrauch) und Verben um Beziehungen festzustellen (zum Beispiel Rauchen verursacht Krebs) (Beeferman [Bee98]). Diese Art des Text Minings kann sehr gut dazu verwendet werden, herauszufinden, welche Art von Zusammenhängen zwischen zwei Konzepten bestehen und kann auch auf sehr großen Mengen von Textdokumenten angewendet werden.

Termeleminierung

Ist bereits vorher bekannt nach was für einer Art von Zusammenhängen gesucht wird, zum Beispiel wenn es sich um immer wiederkehrende Anfragen handelt, die im Rahmen eines monatlichen Geschäftsberichtes erfolgen soll, so bietet sich auch der Einsatz von Templates an. In diesen existieren bereits definierte Felder, die das Text-Mining-Werkzeug durch Analyse einer Menge von Dokumenten zu füllen versucht. Diese Daten können dann bereits das gewünschte Endergebnis liefern und können aufgrund seiner Struktur die Quelle für spätere Data-Mining-Operationen sein (Grishman [Gri97]).

3.6.2 Relevancy Signatures

Da Terme oft nur in einem bestimmten Kontext gute Indikatoren sind und nur wenige Begriffe in jedem Zusammenhang gut verwandt werden können, wird in einem anderen Ansatz versucht den Kontext mit einzubeziehen. So haben Riloff et al. [RL94] erkannt, dass das zum Beispiel Wort „dead“ kein guter Indikator für einen terroristischen Akt ist, wenn man auf der Suche nach terroristischen Zwischenfällen ist. Stattdessen wäre „was found dead“ ein guter Indikator insbesondere wenn der Artikel aus einem Land stammt, der viel von Terrorismus betroffen ist.

Eine Signatur ist ein Paar bestehend aus einem Wort und einem Konzept mit dem das Wort assoziiert wird. Eine Relevanzsignatur ist nun eine Signatur, die oft vorkommt und stark korreliert mit einem bestimmten Thema auftritt. Relevanzsignaturen verbessern die Performanz eines Term Matching durch die Berücksichtigung von Morphologie und Syntax. Je mehr Kontext berücksichtigt wird, desto mehr kann das Ergebnis verbessert werden. Dieser Ansatz allein ist jedoch immer noch anfällig gegen metaphorische Sprache (zum Beispiel Börsencrash und Flugzeugcrash) oder wenn zusätzliche semantische Attribute benötigt werden (zum Beispiel ist der Angriff eines Rebellen ein terroristischer Akt, nicht jedoch der Angriff eines Hundes). Daher setzen Relevanzsignaturalgorithmen vermehrt auch semantische Merkmale ein (Han [HK99, S. 433]).

Signatur

Dieser letzte Ansatz verwendet den lokalen Kontext eines Schlüsselbegriffs und verbessert die Precision durch Verringerung von false hits. Relevancy Signatures beschreiben einen heuristischen Ansatz um Informationen aus einem Text zu extrahieren. Diese Heuristiken sind anpassbar auf unterschiedliche Muster in verschiedenen Domä-

nen.

3.7 Anwendungsszenarien

Text Mining ist eine Kollektion von Techniken und Prozessen und kann bei der Lösung einer Vielzahl von Problemen behilflich sein. So kann es im wissenschaftlichen Bereich eingesetzt werden Zusammenhänge zwischen verschiedenen Bereichen zu finden (zum Beispiel Ursachenforschung in der Medizin oder das Genomprojekt in der Biologie), zur Analyse gesellschaftlicher Problemfelder und des Wählerverhaltens (zum Beispiel Auswertung von Newsgroups), zur Optimierung von Informationsflüssen (zum Beispiel Spam-Filterung) oder zur Kommunikationsüberwachung bei der Verbrechensbekämpfung (Hearst [Hea03], Hoffmann [Hof03], Schüler [Sch04a]).

Mehr und mehr wird Text Mining jedoch sicherlich kommerziell genutzt werden. Insbesondere können Unternehmen Informationen nutzen, die bereits in ihrem eigenen Unternehmen vorliegen. Aufgrund der Informationsflut können Informationssysteme jedoch meist nur auf einen Bruchteil der Informationsquellen automatisch zugreifen, da die Informationen in Statusberichten, Projektdokumentationen und anderen schriftlichen Ausarbeitungen vorliegen und nicht ohne weiteres ausgewertet werden können.

Customer Relationship Management

Jedes Unternehmen existiert jedoch nicht in einem Vakuum sondern agiert in Märkten und tritt in Kontakt mit Kunden. Daher rückt das Customer Relationship Management auch immer mehr in den Fokus von Unternehmen. In der heutigen vernetzten Welt ist es Unternehmen immer leichter möglich, das Kaufverhalten von Kunden zu beobachten, Muster zu erkennen und diese gewinnbringend zu nutzen. So wie durch die Analyse von internen Berichten mehr Informationen über das eigene Geschäft genutzt werden können, ist es möglich, jede Kommunikation mit Kunden zu nutzen, die Art des geschäftlichen Kontakts besser zu verstehen. So können zum Beispiel Newsgroups und Emails analysiert werden, um Interessen und Kundenwünsche frühzeitig zu erkennen, Beschwerden automatisch klassifiziert und dem richtigen Ansprechpartner innerhalb des Unternehmens übergeben werden oder kommerzielle Kunden in deren Marktumfeld zu analysieren. Anstatt nur auf Bestellungen zu achten kann so vorausschauend geplant werden, Chancen genutzt und Risiken frühzeitig erkannt werden. Schließlich können die Märkte selbst näher untersucht werden, Konkurrenten erkannt werden, deren Umfeld analysiert werden, Gesetze und Patente beobachtet und neue Allianzen und Produkte berücksichtigt werden (Sullivan [Sul01, S. 365*ff*])).

3.8 Fazit

Moderne Text-Mining-Lösungen sind in der Lage, sowohl Dokumente aus dem Internet als auch unternehmensinterne und -externe Datenbestände zu analysieren. Durch die Möglichkeit auch große Datenbestände automatisiert nach relevanten Informationen abzusuchen, ergeben sich dabei nicht nur Möglichkeiten für das effiziente Finden vorgegebener Daten sondern insbesondere auch die Möglichkeit durch die Beobachtung von Themen und Konzepten über einen zeitlichen Verlauf hinweg, Risiken für die Unternehmensreputation frühzeitig zu erkennen und Aktionsspielräume zu erhöhen (Hoffmann [Hof03]).

Literaturverzeichnis

[Bee98] Douglas Beeferman. Lexical discovery with an enriched semantic network. In *Proceedings of the ACL/COLING Workshop on Applications of WordNet in Natural Language Processing Systems*, pages 358–364, 1998.

[CEE+04] K.-U. Carstensen, Ch. Ebert, C. Endriss, S. Jekat, R. Klabunde, and H. Langer. *Computerlinguistik und Sprachtechnologie*. Spektrum Verlag, 2004.

[DDF+90] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[FNR03] Jürgen Franke, Golamreza Nakhaeizadeh, and Ingrid Renz. *Text Mining*. Physica-Verlag Heidelberg, 2003.

[Gri97] Ralph Grishman. Information Extraction: A Multdisciplinary Approach to an Emerging Information Technology. In *Information Extraction: Techniques and challenges*, pages 10–27, 1997.

[Hac99] Richard D. Hackathorn. *Web Farming for the Data Warehouse: Exploiting Business Intelligence and Knowledge Management*. CA: Morgan Kaufmann Publishers, Inc., 1999.

[Hea99] Marti A. Hearst. Untangling Text Data Mining. In *Proceedings of ACL99: the 37th Annual Meeting of the Association for Computational Linguistics*, pages 3–10, 1999.

[Hea03] Marti A. Hearst. What is Text Mining? http://www.sims.berkeley.edu/~hearst/text-mining.html, 2003.

[HK99] Jiawei Han and Micheline Kamber. *Data Mining*. Academic Press, San Diego, 1999.

[Hof03] Matthias Hoffmann. Wenn es in der Presse steht, ist es zu spät. http://www.wissensmanagement.net/online/archiv/2004/09_2004/text-mining.shtml, 2003.

[Kod99] Yves Kodratov. Knowledge Discovery in Texts: A Definition, and Applications. *Lecture Notes in Computer Science*, 1609:16–30, 1999.

[LD96] Thomas K. Landauer and Susan T. Dumais. A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction and Representation of Knowledge. http://lsa.colorado.edu/papers/plato/plato.annote.html, 1996.

[Lyc00] Blaine Lycyk. Controlling the Digital Deluge: IBM Content Manager. *DB2 Magazine*, 10(2), 2000.

[Mar98] Wolfgang Martin. *Data Warehousing, Data Mining – OLAP*. International Thomson Publishing, 1998.

[RL94] Ellen Riloff and Wendy Lehnert. Information Extraction as a Basis for High Precision Text Classification. *ACM Transactions on Information Systems*, 12(3):296–333, 1994.

[Sal83] G. Salton. *Introduction to Modern Information Retrieval*. G. & McGill, M. J, 1983.

[Sch04a] Peter Schüler. Detektei Allwissend. *c't*, (8):86, 2004.

[Sch04b] Elisabeth Schwarz. Automatische Inhaltserschließung von Textdokumenten. http://www.infomanager.at/KM/working_papaer2004-2.pdf, 2004.

[SS94] Don R. Swanson and N.R. Smalheiser. Assessing a gap in the biomedical literature: Magnesium defiency and neurologic disease. *Neuroscience Research Communications*, 15:1–9, 1994.

[SS97] Don R. Swanson and N.R. Smalheiser. An interactive system for finding complementary literatures: a stimulus to scientific discovery. *Artificial Intelligence*, 91(2):183–203, 1997.

[SS99] Don R. Swanson and N.R. Smalheiser. Complementary structures in disjoint science literatures. In *Proceedings of the 14th Annual International ACM/SIGIR Conference*, pages 280–289, 1999.

[Sul01] Dan Sullivan. *Document Warehousing and Text Mining*. John Wiley & Sons, Inc., 2001.

[Tan98] Ah-Hwee Tan. Text Mining: The state of the art and the challenges. http://www.ewastrategist.com/papers/text_mining_kdad99.pdf, 1998.

[Tan99] Ah-Hwee Tan. Text Mining: The state of the art and the challenges. http://citeseer.ist.psu.edu/tan99text.html, 1999.

[Vis01] Ari Visa. Technology of Text Mining. In *Proceedings of MLDM 2001, the Second International Workshop on Machine Learning and Data Mining in Pattern Recognition*, pages 9–11, 2001.

[Wil00] S. Williams. A survey of Natural Language Processing Techniques for Text Data Mining. *CSIRO Mathematical and Information Sciences*, 127, 2000.

Markus Bestehorn

4 Part-of-Speech Tagging

4.1 Einleitung

Beispiel 4.1.1 *„Vier Reiche haben 8 Arme. Acht Arme haben 16 Beine. Also haben Arme mehr Beine als Reiche Arme."*

Beim Durchlesen des obigen Satzes müssen die meisten Menschen sich zumindest zweimal überlegen, an welcher Stelle im Satz es sich beim Wort „Arme" um ein Körperteil handelt und an welcher Stelle damit eine Person bzw. Personengruppe gemeint ist.

Die Problematik beim Verstehen des Satzes liegt darin, dass das Wort „Arme" des Satzes nur dann verstanden werden kann, wenn der Kontext, in dem es verwendet wird, beim Verstehen des einzelnen Wortes miteinbezogen werden kann.

Von denjenigen, die sich während der Zeit am Gymnasium mit einer humanistisch geprägten Bildung „herumplagen" mussten, dürften viele die täglichen Übungen aus dem Lateinunterricht kennen, in denen beim Übersetzen eines Satzes zunächst für jedes Wort bestimmt werden musste, zu welchem Satzteil das Wort gehört. Für die Übersetzung des Textes war diese Übung unerlässlich, und ein Fehler in der Übersetzung ließ sich häufig auf einen Fehler in der Satzanalyse, also bei der Zuordnung der Wörter zu Satzteilen, zurückführen.

Vor einem ähnlichen Problem, nämlich einem Übersetzungsproblem, steht die maschinelle Spracherkennung, denn es muss ein menschlicher Text in eine für Maschinen verständliche Form übersetzen. Einer der ersten Schritte, die für ein solches maschinelles Textverständnis notwendig sind, ist wie im Lateinunterricht, die korrekte strukturelle Analyse des Satzes.

Im folgenden Kapitel wird es nun darum gehen, die verschiedenen Verfahren, die zur Markierung von Satzteilen entwickelt wurden, vorzustellen und deren Arbeitsweise zu verstehen.

4.1.1 Was ist Part-of-Speech Tagging?

Bei der Entwicklung von Computer-Programmen wird ebenfalls ein Quelltext, also der Programm-Code, der z. B. in C++ oder Java verfasst wurde, in eine für den Computer verwendbare Form umgesetzt, die sogenannte Maschinensprache. Nachdem ein Programm in Maschinensprache vorliegt kann, der Computer die Anweisungen, die der Programmierer dem Computer gegeben hat, umsetzen und entsprechende Ergebnisse berechnen.

Warum geht das nicht mit menschlicher Sprache?

Computer-Sprachen haben eine Eigenschaft, die menschliche Sprachen im Allgemeinen nicht haben: Sie sind *kontextfrei*. Das heisst (umgangssprachlich formuliert), dass sie keine Kontextabhängigkeiten oder Doppeldeutigkeiten enthalten.

Da wegen der Irregularität menschlicher Sprache eine Verarbeitung, wie sie bei Computer-Sprachen verwendet wird, auf Grund des derzeitigen Forschungsstandes nicht möglich ist, versucht man derzeit, Teilprobleme auf dem Weg zur Lösung des Problems der Sprachverarbeitung zu lösen. Eines dieser Teilprobleme ist die Gewinnung von Strukturinformationen aus dem Text, ohne dass Informationen über den Inhalt des Textes vorhanden sein müssen. Einer der ersten Schritte, analog zur Bestimmung von Satzteilen bei der Übersetzung aus dem Lateinischen, ist das sogenannte *Part-of-Speech Tagging*:

Part-of-Speech Tagging

Definition 4.1.2 *Part-of-Speech Tagging, im folgenden auch kurz* Tagging *genannt, ist der Vorgang, bei dem jedem Wort in einem Satz ein sogenanntes* Part-of-Speech Tag*, also eine (grammatikalische) Markierung, zugeordnet wird.*

Ziel des Part-of-Speech Taggings ist es beispielsweise, aus dem Satz *„Buchen Sie den Flug“* folgende Markierung der Satzteile zu erhalten (Die Bedeutung der Markierungen wird im Abschnitt 4.1.2 noch genauer erklärt):

Beispiel 4.1.3 *Buchen/VVIMP Sie/PPER den/ARTDEF Flug/NN .$*

Die Tags werden im Allgemeinen immer durch einen Slash (/) vom Wort getrennt und direkt an das dazugehörige Wort angefügt. Die hier dargestellten Tags stammen aus dem STTS – dem Stuttgart-Tübingen Tagset [STST95]; eine genauere Einführung bezüglich Tagsets findet sich in Abschnitt 4.1.2. Neben der oben genannten richtigen Variante könnte man den Satz auch wie folgt taggen:

Beispiel 4.1.4 *Buchen/NN Sie/PPER den/ARTDEF Flug/NN.$*

In der ersten Variante würde das Wort „Buchen“ als ein Verb im Imperativ interpretiert und entsprechend mit einem Tag versehen. Die zweite Möglichkeit markiert das Wort „Buchen“ als Plural des Nomens „Buche“, was offensichtlich falsch ist, aber verdeutlicht, dass es auch bei solch einfachen, kurzen Sätzen durchaus nicht trivial ist, maschinell die richtige Variante herauszufinden.

4.1.2 Tagsets

Um die Strukturinformationen, die aus dem Part-of-Speech Tagging gewonnen werden, auch nutzen zu können bzw. die Bedeutung der zugewiesenen Tags festzulegen, werden sogenannte Tagsets benutzt:

Definition 4.1.5 *Ein Tagset ist eine Menge von Markierern, die für verschiedene Satzteile während des Part-of-Speech Taggings benutzt werden.* Tagset

Ein Beispiel für ein solches Tagset für deutsche Sprache ist das bereits erwähnte Stuttgart-Tübingen-Tagset [STST95]. Ein kleiner Ausschnitt, aus den insgesamt 54 verschiedenen Tags, ist in Tabelle 4.1 zu sehen.

Tag	Beschreibung	Beispiele
ADJA	Adverb	schon, bald, doch
APPR	Präposition; Zirkumposition links	in [der Stadt], [ihm] zufolge
ARTDEF	Artikel	der, die, das, ein, eine
CARD	Kardinalzahl	zwei [Männer], [im Jahre] 1994
NN	normales Nomen	Frau, Haus, München, Ostern, [das] Gehen
PPER	irreflexives Personalpronomen	ich, er, ihm, mich, dir, sie
VVFIN	finites Verb, voll	[du] gehst, [wir] holen
VVIMP	Imperativ, voll	komm [!], Buchen [Sie]
VVINF	Infinitiv, voll	gehen, erzeugen, malen

Tabelle 4.1: Ausschnitt aus dem STTS

In den Beispielen dieses Kapitels wird das STTS als Referenz für das Tagging deutscher Texte benutzt. Da auch einige Beispiele in englischer Sprache vorkommen, soll an dieser Stelle das Tagset vorgestellt werden, das für das Tagging der englischen Beispiele benutzt wurde (Siehe Tabelle 4.2).

POS Tag	Beschreibung	Beispiel
CC	coordinating Conjunction	and
CD	cardinal number	1, third
JJ	adjective	green, strong, bad
NN	Noun, singular or mass	house, snow, salt
NNS	Noun, plural	houses
POS	Possessive ending	friend's
VB	Verb, base form	take, make, draw
VBD	Verb, past tense	took, made

Tabelle 4.2: Das Penn Treebank Tagset

Insgesamt beinhaltet das Penn Treebank Tagset [MSM94] 45 Tags, womit dieses Tagset eines der kleineren englischen Tagsets ist. Tagsets wie das C7-Tagset [Ray] benutzen bis zu 146 Tags, aus Gründen der Übersichtlichkeit und um die Beispiele kurz und einfach zu halten, wurde aber das kürzere Penn Treebank Tagset verwendet. Penn Treebank Tagset

4.2 Regelbasierte Tagger

Nachdem im vorangegangenen Abschnitt eine kurze Einführung in die Thematik des Part-of-Speech Taggings gemacht wurde, wird die erste Klasse von Taggern, die *regelbasierten Tagger*, vorgestellt. Die Eingabe für einen regelbasierten Tagger besteht im wesentlichen aus zwei Teilen:

- Einem Text, der keine Tags enthält, also eine Aneinanderreihung von Wörtern bzw. Buchstaben.
- Einem Tagset, ähnlich denen, die in Abschnitt 4.1.2 vorgestellt wurden.

Wie bereits in dem vorangegangenen Beispiel 4.1.3 „Buchen Sie den Flug“ gezeigt wurde, ist es selbst bei so einfachen Sätzen nicht trivial, ein korrektes Tagging zu finden. Ein trivialer Ansatz wäre, sich zu überlegen, welches Tag das Wort „Buchen“ in den meisten Fällen hat. Diese Methode wurde von Greene und Rubin [GR71] bereits umgesetzt und führt dazu, dass der Tagger nur eine Genauigkeit von 77% aufweist.

Dafür gibt es mehrere Gründe: Das Hauptproblem beim Taggen ist, dass es zahlreiche Wörter gibt, die mehrere mögliche Verwendungen haben und folglich auch mehrere mögliche Tags. Untersuchungen, zum Beispiel von DeRose [DeR88], zeigten, dass nur ein relativ geringer Prozentsatz der Wörter im Englischen, bzw. den zur Untersuchung verwendeten Texten, mehr als einen möglichen Tag hat. Problematisch aber ist, dass gerade diese Wörter relativ häufig benutzt werden. In den meisten modernen Sprachen werden bereits bestehende Wörter durch Verwendung in einem neuen Zusammenhang einer weiteren Wortklasse hinzugefügt, so dass zum Beispiel Wörter, die vorher nur als Nomen auftreten konnten, plötzlich in einem Text auch als Verb verwendet werden. Ein Beispiel für eine solche „Verwandlung“ im Englischen ist der Satz „He will *web* our work tomorrow.“; hier wird ein Wort, welches vor der Verbreitung des Internets wohl kaum als Verb aufgetreten ist, als Verb benutzt.

Eine der Möglichkeiten ein eindeutiges Tagging zu finden und damit doppeldeutige Taggings zu vermeiden, bilden die regelbasierten Tagger, die jetzt genauer vorgestellt werden.

4.2.1 Grundlagen des regelbasierten Part-of-Speech Taggings

Die ersten regelbasierten Tagger wurden in den 1960er Jahren entwickelt (Klein und Simmons [KS63]; Greene und Rubin [GR71]) und hatten alle eine sogenannte 2-Phasen Architektur. In der ersten Phase werden aus einem „Wörterbuch“ für jedes Wort in dem zu taggenden Satz alle möglichen Tags gesucht. Diese Liste der möglichen Tags wird dann dem Wort zugewiesen. Bei allen Wörtern, denen eine Liste mit mehr als einem Element zugewiesen wurde, wird dann durch Anwendung eines komplexen, grammatikalischen Regelsystems versucht, die Liste auf ein Element zu reduzieren, damit das Tagging eindeutig ist.

2-Phasen Architektur

Spätere Tagger, wie der in Abschnitt 4.2.2 vorgestellte ENGTWOL, der 1995 von Voutilainen [Vou95] veröffentlicht wurde, basieren immer noch auf dieser 2-Phasen Architektur, allerdings sind bei diesen regelbasierten Taggern sowohl das Wörterbuch als auch das Regelsystem wesentlich ausgefeilter. In den nächsten zwei Abschnitten werden daher zunächst die beiden Phasen der regelbasierten Tagger genauer vorgestellt.

ENGTWOL

Phase 1: Das Wörterbuch und die initiale Annotation

Die zentrale Komponente der ersten Phase des Taggens ist das Wörterbuch. An Hand des folgenden, bereits bekannten, Beispiels 4.1.3 soll diese Phase nun verdeutlicht werden: “Buchen Sie den Flug“. Ein Ausschnitt aus einem fiktiven und relativ simplen Wörterbuch könnte, wie in Tabelle 4.2.1 dargestellt, aussehen.

Es ist deutlich sichtbar, dass ein Wort in dem Wörterbuch durchaus mehrere Einträge haben kann. Um nun die erste Phase abzuschließen, würde ein Suchalgorithmus

Wort	POS Tag
Buchen	VVIMP
Buchen	VVINF
Buchen	NN
den	ARTDEF
den	PRELS
den	PDS
Flug	NN
Sie	PPER

Tabelle 4.3: Beispiel für ein einfaches Wörterbuch

für jedes der Worte aus dem Beispiel-Satz eine Liste anlegen und diese mit den für dieses Wort möglichen Tags füllen. Ein mögliches Ergebnis für das regelbasierte Part-of-Speech Tagging des Beispielsatzes steht in Tabelle 4.4.

Buchen	{VVIMP, VVINF, NN}
Sie	{PPER}
den	{ARTDEF, PRELS, PDS}
Flug	{NN}

Tabelle 4.4: Ergebnis der ersten Phasen für „Buchen Sie den Flug“

Für die beiden Worte “Flug“ und “Sie“ ist damit das Tagging abgeschlossen, da die Mengen bereits einelementig sind und daher eine Auflösung von Doppeldeutigkeiten durch Anwendung des Regelsystems nicht notwendig ist. Dies vereinfacht den Prozess der Regelanwendung im folgenden Abschnitt, denn es kann davon ausgegangen werden, dass diese beiden Tags richtig sind.

Phase 2: Die Anwendung von Regeln zur Auflösung von Doppeldeutigkeiten

Die zweite Phase zur Eliminierung von Doppeldeutigkeiten innerhalb des Taggings wird nur dann benutzt, wenn nicht jedes Wort bereits in der ersten Phase ein eindeutiges Tagging erhalten hat, also die Liste der Tags aus dem Wörterbuch mehrelementig ist. In dem Beispiel aus dem vorherigen Abschnitt 4.2.1 gibt es noch zwei Wörter, denen eine Liste mit mehr als einem Element zugeordnet wurde. In Systemen, wie sie heute verwendet werden, kommen an dieser Stelle die Regelsysteme mit bis zu 1100 verschiedenen, teilweise recht komplexen Regeln zum Einsatz. Für das Beispiel seien zwei einfache Regeln definiert, die “zufälligerweise“ recht gut passen: [1]

Die linke der beiden Regeln bewirkt, dass bei einem Wort „den“ geprüft wird, ob dem Wort danach ein Tag NN zugeordnet wurde und sich davor ein Wort befindet, dass mit PPER markiert wurde. Wenn beide Bedingungen wahr sind, dann wird jeder Tag aus der Tag-Liste entfernt, außer dem ARTDEF Tag, das heisst diese Regel findet heraus, ob es sich um einen Artikel handelt oder eben nicht. Sollte eine der Bedingungen

[1] Es geht hier um die Anwendung der Regeln, nicht um deren grammatikalische Richtigkeit. Die beiden Regeln sind nur für dieses Beispiel sinnvoll und für die allgemeine Anwendung wahrscheinlich nicht geeignet.

INPUT: “dem“ , “den“ if ((+1 NN) && (-1 PPER)) then removeAll(NON ARTDEF-Tags) else remove(ARTDEF-Tag)	INPUT: ’„Verb“ if ((-1 Satzgrenze) && (+1 PPER)) then removeAll(NON VVIMP-Tags) else remove(VVIMP-Tag)

Tabelle 4.5: Regelanwendung am Beispiel 4.1.3 „Buchen Sie den Flug.“

nicht wahr sein, dann kann der ARTDEF Tag entfernt werden, weil es sich dann sicher um keinen Artikel handelt. Diese Art der Regelanwendung nennt man *negative Regelanwendung*.

Negative Regelanwendung

Ähnlich wie bei der linken Regel verhält es sich auch bei der rechten: Der Unterschied ist, dass hier Verben im allgemeinen als Eingabe verwendet werden dürfen. Die Regel besagt, dass Worte, in deren Tag-Menge ein Verb-Tag vorkommt, die am Anfang eines Satzes stehen und die von einem Personalpronomen (PPER) gefolgt werden, Verben im Imperativ sind und daher mit dem Tag VVIMP markiert werden müssen.

4.2.2 Der ENGTWOL Tagger

Wie bereits in den vorhergehenden Abschnitten erwähnt, sind echte regelbasierte Systeme wesentlich komplexer, als das bisher vorgestellt wurde. Um einen kurzen Einblick in ein solches System zu geben, wird nun an Hand des ENGTWOL Taggers [Vou95] ein weiteres Beispiel mit Part-of-Speech Tags versehen:

Beispiel 4.2.1 *It is expected that Mr. Montoya will race tomorrow.*

Eine wesentliche Erweiterung des ENGTWOL Taggers gegenüber den älteren Systemen sind die Struktur und der Inhalt des Wörterbuches. Neben Stammformen für die meisten Verben werden auch Wortstämme von Substantiven sowie morphologische und syntaktische Informationen in sogenannten „Additional Part-of-Speech Features“ gespeichert. Ein Ausschnitt aus dem Wörterbuch des ENGTWOL Taggers könnte daher so aussehen wie in Tabelle 4.2.2.

Additional POS features

Es ist deutlich sichtbar, dass versucht wird, neben dem POS-Tag zu jedem Wort eine Reihe anderer Informationen, wie zum Beispiel die Stammform, zu bestimmen und für das Tagging zur Verfügung zu stellen. Außerdem werden Markierer verwendet um anzuzeigen, dass Wörter in bestimmten Umgebungen besonders oft einen bestimmten Tag haben, zum Beispiel zeigt das POS-Feature *Vcog* bei “expected“, dass dieses Wort besonders häufig in einem Satz vor “that“ steht. POS-Features wie *Vcog* können dann beim Tagging benutzt werden, dass heißt, wenn man auf ein Wort mit *Vcog* stößt und danach ein “that“ auftaucht, dann kann man alle Tags aus der Liste entfernen, die das POS-Feature *Vcog* nicht enthalten. Da auch der ENGTWOL Tagger auf der 2-Phasen Architektur basiert, ist es nicht verwunderlich, dass das Ergebnis der ersten Phase sich analog zu dem des vorhergehenden Beispiels 4.4 verhält, wie man in Tabelle 4.2.2 sehen kann.

Durch die Erweiterung des Wörterbuches um die Additional POS features sind nun wesentlich feinere Regeln möglich, als das bei den früheren Systemen der Fall war. Nach der 1. Phase wird nun analog zu dem vorhergehenden Beispiel durch Anwendung von Regeln ein eindeutiges Tagging erzeugt.

Word	POS	Additional POS features
it	PRON	“it“ NonMod ACC SG3
it	PRON	“it“ NonMod ACC SG3 SUBJ @SUBJ
is	V	“be“ SV SVC/N SVC/A PRES SG3 VFIN
expected	V	“expect“ Vcog SVO P/of PAST VFIN @+FMAINV
expected	PCP2	“expect“ Vcog SVO P/of
that	ADV	“that“ AD-A @AD-A
that	PRON	“that“ NonMod **CLB Rel SG/PL
that	PRON	“that“ PRON DEM SG
that	DET	“that“ CENTRAL DEM SG @DN>
that	CS	“that“ **CLB @CS
to	INFMARK	“to“ @INFMARK
to	PREP	“to“
race	N	“race“ NOM SG
race	V	“race“ SV SVO INF
...	...	...

Tabelle 4.6: Ausschnitt aus dem Wörterbuch des ENGTWOL Taggers

Die Qualität der Ergebnisse des ENGTWOL Taggers sind hauptsächlich aus diesem Grund besser als bei den älteren Systemen. Zusätzlich zu der Erweiterung des Wörterbuches werden zur Auflösung von Doppeldeutigkeiten auch probalistische Annahmen und andere syntaktische Informationen verwendet, die aber hier nicht weiter vertieft werden.

4.2.3 Evaluierung: Regelbasierte Tagger

Im Gegensatz zu den im Folgenden vorgestellten stochastischen Taggern benötigen regelbasierte Tagger keinen manuell annotierten Trainingskorpus[2]. Ein Nachteil ist jedoch, dass es einen enormen Aufwand bedeutet, ein so komplexes Regelsystem aufzustellen, wie es z. B. der ENGTWOL Tagger benutzt. Um qualitativ hochwertige Ergebnisse zu erzielen, benötigt man zudem Sprachwissenschaftler und andere Spezialisten, die dann die Regeln aufstellen und testen. Ein weiterer Nachteil ist, dass die Benutzung von existierenden Regeln einer Sprache (z. B. Englisch) für das Tagging einer anderen Sprache (z. B. Deutsch) so gut wie unmöglich ist, so dass für jede Sprache ein komplett neues Regelsystem erzeugt werden muss.

Da man versuchen wollte, diesen enorm hohen materiellen und intellektuellen Aufwand zur Erstellung der Regeln zu vermeiden, versuchten andere Forscher einen gänzlich anderen Ansatz zu verfolgen und entwickelten die *Stochastischen Tagger*.

4.3 Stochastische Tagger

Die wichtige Idee hinter dem stochastischen Part-of-Speech Tagging ist, dass es bestimmte Kombinationen von Wörtern und Tags gibt, die besonders häufig sind und wiederum andere Kombinationen eher unwahrscheinlich sind. Es leuchtet zum Beispiel ein, dass es sehr wahrscheinlich ist, dass ein Artikel vor einem Nomen steht

[2]Was das genau heisst, wird in Abschnitt 4.3 erklärt.

word	POS-Tags
It	{PRON, “it“ NonMod ACC SG3}, {V, “be“ SV SVC/N SVC/A PRES SG3 VFIN}
is	{V, “be“ SV SVC/N SVC/A PRES SG3 VFIN}
expected	{V, “expect“ Vcog SVO P/of PAST VFIN @+FMAINV}, {PCP2, “expect“ Vcog SVO P/of}
that	{DET, “that“ CENTRAL DEM SG @DN}, {CS, “that“ **CLB @CS}, {ADV, “that“ AD-A @AD-A}, {PRON, “that“ PRON DEM SG}, {PRON, “that“ NonMod **CLB Rel SG/PL}
Mr.	{ABBR, “mr“ Title NOM SG}
Montoya	{N, “Montoya“ proper NOM SG}
will	{V, „will“ AUXMOD VFIN @+FAUXV}, {N, „will“ NOM SG}
race	{N, “race“ NOM SG}, {V, “race“ SV SVO INF}
tomorrow	{N, „tomorrow“ NOM SG}, {ADV, „tomorrow“ ADVL @ADVL}

Tabelle 4.7: Ergebnis der Phase 1 bei ENGTWOL

(„das Haus“), und im Gegensatz dazu die Wahrscheinlichkeit, dass ein Artikel vor einem Verb steht, eher gering ist. Analog dazu dürfte es jedem einleuchten, dass die Wahrscheinlichkeit, dass das Wort „Buchen“ (vgl. Beispiel 4.1.3) als Verb benutzt wird, wesentlich größer ist als die Wahrscheinlichkeit, dass es als Nomen zur Beschreibung einer Baumart oder Baumgruppe benutzt wird. Die stochastischen Part-of-Speech Tagger basieren genau auf dieser Eigenschaft und versuchen, für einen Satz die wahrscheinlichste Tagging-Variante zu finden.

4.3.1 Mathematische Grundlagen

Die bereits in Kapitel 2 vorgestellten Verfahren zur Berechnung von bedingten Wahrscheinlichkeiten werden im Folgenden auf das Part-of-Speech Tagging übertragen bzw. es wird gezeigt, wie die Anwendung dieser mathematischen Grundlagen auf das Problem des Part-of-Speech Taggings funktioniert. Für die Anwendung im Bereich des stochastischen Part-of-Speech Tagging werden im wesentlichen drei bedingte Wahrscheinlichkeiten immer wieder benutzt:

$P(t_n|t_1, ..., t_{n-1})$: Die Wahrscheinlichkeit, dass ein Wort mit dem Tag t_n markiert wird unter der Voraussetzung, dass die vorherigen $n-1$ Worte mit den Tags t_1 bis t_{n-1} markiert wurden.

$P(\textbf{Wort}|t)$: Die Wahrscheinlichkeit, dass ein Wort mit einem bestimmten Tag t markiert wird.

$P(t|\textbf{START})$: Die Wahrscheinlichkeit, dass ein bestimmter Tag t am Anfang eines Satzes steht.

Das Markov-Modell

Die meisten stochastischen Part-of-Speech Tagger benutzen die *Markov Annahme*: Markov Annahme

Definition 4.3.1 *Sei $X = \{X_1, X_2, ..., X_t\}$ eine Kette von Zufallsvariablen, die Werte aus einem Wertebereich $S = \{s_1, s_2, ..., s_n\}$ annehmen können. Wenn für die Vorhersage des Wertes der Variable X_{t+1} die Markov Eigenschaften*

- Lokalität*:* $P(X_{t+1} = s_k|X_1, ..., X_t) = P(X_{t+1} = s_k|X_t)$
- Zeitliche Invarianz*:* $P(X_{t+1} = s_k|X_1, ..., X_t) = P(X_2 = s_k|X_1)$

gelten, dann wird X als Markov Kette *bezeichnet.* Markov Kette

Stochastische Part-of-Speech Tagger (genauer: Bigramm-Tagger), die von dieser Annahme ausgehen, benutzen also nur den Tag des vorhergehenden Wortes, um den Tag des aktuellen Wortes zu bestimmen. Zunächst mag diese Annahme relativ radikal erscheinen, aber es hat sich gezeigt, dass diese Annahme relativ gute Ergebnisse ($\approx 90\%$ korrekt markierte Wörter) erzielt. Eine Aufweichung der Annahme dahingehend, dass der Tag des aktuellen Wortes von den beiden vorhergehenden Tags abhängt (Trigramm-Tagger) liefert zwar bessere Ergebnisse, aber der Aufwand, für die Berechnung des Taggings steigt dramatisch an, wie im Folgenden noch deutlich wird.

4.3.2 Training stochastischer Tagger

In der Einleitung zu diesem Abschnitt wurde angedeutet, dass es sich beim dem Wort „Buchen“ deutlich wahrscheinlicher um ein Verb als um ein Nomen handelt. Die Frage, die sich an dieser Stelle aufdrängt ist, woher die Wahrscheinlichkeiten für das Tagging überhaupt kommen. Woher „weiss“ ein Tagger, welches Wort in welchem Zusammenhang wahrscheinlich ein Nomen ist? Zur Erzeugung einer solchen Wissensbasis werden große, von Hand annotierte *Textkorpora* benutzt, also Sammlungen von Texten, die von Hand mit den grammatikalisch richtigen Part-of-Speech Tags versehen wurden. Es gibt eine ganze Reihe von solchen Textkorpora, von denen hier aber lediglich zwei vorgestellt werden, damit man einen Eindruck vom Aufwand bekommt, der für diesen Schritt des stochastischen Part-of-Speech Taggings notwendig ist:

NEGRA: Korpus mit mehr als 20.000 Sätzen (ca. 350.000 Wörter) in deutscher Sprache der Frankfurter Rundschau. [NEG]

Brown Korpus: Mehr als 1 Mio. Wörter aus ca. 500 Texten aus verschiedenen Bereichen (z. B. Zeitungen, wissenschaftliche Arbeiten, Anleitungen...). [MSM94]

Das Training für einen stochastischen Part-of-Speech Tagger ist im Prinzip nichts anderes als ein Durchlaufen des Textes und Zählen von Häufigkeiten. Es wird zum Beispiel gezählt, wie oft Wort mit einem Tag für Adjektiv vor einem Nomen steht, wie oft ein bestimmter Tag, z. B. für Nomen, ein Wort am Anfang eines Satzes annotiert und wie oft ein bestimmtes Wort (z. B. “Buchen“) mit einem bestimmten Tag versehen wird.

Aus der Zählung können dann Wahrscheinlichkeiten berechnet werden, die dann in die Matrizen des Hidden Markov Modells eingetragen werden.

4.3.3 Part-of-Speech Tagging mit dem Hidden Markov Modell

Nachdem die mathematischen Grundlagen und das „Erlernen“ von Wahrscheinlichkeiten vorgestellt wurden, geht es in diesem Abschnitt darum, mit Hilfe des Hidden Markov Modells für einen gegebenen Text ein Part-of-Speech Tagging zu erhalten.

Beispiel 4.3.2 *Der zu taggende Satz ist “Buchen Sie den Flug“ aus Beispiel 4.1.3. Damit ist die Menge der Beobachtungen $O = \{o_1 = Buchen, o_2 = Sie, o_3 = den, o_4 = Flug\}$. Die Matrizen A, B und Π sehen wie folgt aus:*

	NN	VVIMP	ARTDEF	PPER	PDS	Buchen	Sie	den	Flug	Π
NN	0,1	0	0,3	0,4	0,2	0,2	0	0	1	0,2
VVIMP	0	0	0,2	0,4	0,4	0,8	0	0	0	0,3
ARTDEF	0,4	0,1	0	0,2	0,3	0	0	0,7	0	0,2
PPER	0,1	0,1	0,1	0,3	0,4	0	1	0	0	0,1
PDS	0,1	0,2	0,2	0,3	0,2	0	0	0,3	T	0,2

Tabelle 4.8: Die Matrizen A, B und Π für das Beispiel „Buchen Sie den Flug“

Mit Hilfe der Matrizen soll nun für die gegebenen Beobachtungen ein Tagging erzeugt werden, welches die größte mögliche Wahrscheinlichkeit hat.

Berechnung der Wahrscheinlichkeiten Allgemein berechnet sich die Wahrscheinlichkeit eines Taggings wie folgt:

$$\underbrace{\pi_{i_1} \times b_{i_1 1}}_{1.\,Wort} \times \underbrace{b_{i_2 2} \times a_{i_1 i_2}}_{2.\,Wort} \times \underbrace{b_{i_3 3} \times a_{i_2 i_3}}_{3.\,Wort} \times \ \ldots \ \times \underbrace{b_{i_n n} \times a_{i_{n-1} i_n}}_{n-tes\ Wort}$$

Beim ersten Wort wird die Wahrscheinlichkeit $P(\text{Wort}|\text{Tag})$ aus der Matrix B multipliziert mit der Wahrscheinlichkeit $P(\text{Tag}|\text{Satzanfang})$ aus der Matrix Π. Für alle folgenden Wörter wird dann die Wahrscheinlichkeit $P(\text{Wort}|\text{Tag}) \times P(\text{Tag}|\text{vorheriger Tag})$ berechnet.

Für die Berechnung des Taggings, welches die größte Wahrscheinlichkeit hat, gibt es nun verschiedene Varianten, die nachfolgend beschrieben werden:

Brute-Force: Es wird versucht, alle Kombinationen von Tags für alle Wörter durchzuprobieren und das wahrscheinlichste auszuwählen: Unter Verwendung der Zahlen aus Beispiel 4.3.2 sähe das dann so aus:

$$P(\text{NN},\text{NN},\text{NN},\text{NN}|O,A,B,\Pi) = \underbrace{0,2 \times 0,2}_{Buchen} \times \underbrace{0 \times 0,1}_{Sie} \times \underbrace{0 \times 0,1}_{den} \times \underbrace{1 \times 0,1}_{Flug} = 0$$

$$P(\text{NN},\text{NN},\text{NN},\text{VVIMP}|O,A,B,\Pi) = \underbrace{0,2 \times 0,2}_{Buchen} \times \underbrace{0 \times 0,1}_{Sie} \times \underbrace{0 \times 0,1}_{den} \times \underbrace{0 \times 0}_{Flug} = 0$$

...

$$P(\text{NN},\text{PPER},\text{ARTDE}F,\text{NN}|...) = \underbrace{0,2 \times 0,2}_{Buchen} \times \underbrace{1 \times 0,4}_{Sie} \times \underbrace{0,7 \times 0,1}_{den} \times \underbrace{1 \times 0,4}_{Flug} = 0,000448$$

...

$$P(\text{VVIMP}, \text{PPER}, \text{ARTDEF}, \text{NN}|...) = \underbrace{0,3 \times 0,8}_{Buchen} \times \underbrace{1 \times 0,4}_{Sie} \times \underbrace{0,7 \times 0,1}_{den} \times \underbrace{1 \times 0,1}_{Flug} = 0,000672$$

$$\ldots$$

$P(\text{PDS}, \text{PDS}, \text{PDS}, \text{PPER}|O, A, B, \Pi) = 0$
$P(\text{PDS}, \text{PDS}, \text{PDS}, \text{PDS}|O, A, B, \Pi) = 0$

Das Problem hier ist, dass bei diesem Verfahren schon für den Satz mit 4 Worten und dem „Tagset“ mit 5 Tags $5^4 = 625$ Möglichkeiten zu prüfen sind. Bei einem normalen Satz mit ca. 20 Worten und einem Tagset, wie dem STTS mit ca. 50 Tags würde das dazu führen, dass $50^{20} \approx 10^{34}$ Möglichkeiten zu überprüfen sind.

Dynamische Programmierung: Die Berechnung des Ergebnisses erfolgt hierbei schrittweise und es werden nur Berechnungen fortgesetzt, die einen Beitrag zum Ergebnis leisten könnten. Wichtig ist hier, dass die Berechnung der Wahrscheinlichkeit für ein Tagging nur Multiplikationen enthält, was dazu führt, dass eine Berechnung automatisch abgebrochen werden kann, wenn eines der Elemente darin gleich 0 ist. Diese Eigenschaft wird durch den *Viterbi Algorithmus*, der im Folgenden vollständig vorgestellt wird, ausgenutzt.

Der Viterbi-Algorithmus

Der Viterbi-Algorithmus besteht im wesentlichen aus drei Phasen und baut einen Graphen auf, auf dessen Grundlage dann das wahrscheinlichste Tagging bestimmt werden kann:

Initialisierung: In dieser Phase wird ein Startknoten S erzeugt.

Aufbauen des Graphen: Mittels Induktion wird hier ein Graph aufgebaut, aus dem das wahrscheinlichste Tagging abgelesen werden kann. Die Knoten des des Graphen enthalten die Wahrscheinlichkeit eines Taggings, welches durch den Pfad vom Startknoten zu diesem Knoten repräsentiert wird, und Verweise auf den jeweiligen Vorgängerknoten.

Auslesen des wahrscheinlichsten Taggings: Unter Benutzung des Verweises auf den Vorgänger-Knoten wird ausgehend vom wahrscheinlichsten Endknoten die Belegung bestimmt, die die höchste Wahrscheinlichkeit hat.

Jeder Knoten des Graphen entspricht einem Wort und einem dazugehörigen Tag, z. B. {“Buchen“,NN}. Das Aufbauen des Graphen erfolgt im wesentlichen mit zwei Funktionen: Die Funktion $\delta(i, t)$ berechnet die Wahrscheinlichkeit, dass das i-te Wort mit dem Tag t versehen wird. Die Berechnung erfolgt genau so, wie sie oben bei der Berechnung der Brute-Force-Wahrscheinlichkeiten erfolgte. Beispielsweise wäre

$$\delta(i, t) = (\text{VVIMP}, \text{PPER}, \text{ARTDEF}, \text{NN}|O, A, B, \Pi) = 0,000672$$

wie oben zu sehen ist. Die zweite Funktion $\psi(i, t)$ berechnet für das i-te Wort mit dem Tag t den Vorgänger-Knoten.

Phase 1: Initialisierung

Bei der Initialisierung wird nur ein Startknoten S erzeugt. Außerdem wird diesem Knoten $\delta(0, START) = 1$ zugewiesen.

Phase 2: Aufbau des Graphen

Hier werden für jedes Wort i und jeden Tag im Tagset t die beiden Funktionen $\delta(i,t)$ und $\psi(i,t)$ berechnet. Für das Beispiel 4.3.2 wird damit der Graph (siehe Abbildung 4.1) wie folgt aufgebaut:

Nach dem durch den Initialisierungsschritt der Knoten S erzeugt wurde und $\delta(0, START) = 0$ gesetzt wurde, wird für jeden Tag t ein Knoten {“Buchen“,t} erzeugt. Am Beispiel des Knotens {“Buchen“,NN} sollen die nachfolgenden Schritte verdeutlicht werden:

a) Es wird $\psi(1, NN) = S$ gesetzt, d.h. der Vorgängerknoten von {“Buchen“,NN} ist S.

b) Um den Wert der Wahrscheinlichkeitsfunktion δ zu berechnen werden die folgenden Werte aus den Matrizen benötigt:

   a) Der Wert $P(Buchen|NN) = b_{11} = 0,2$, der die Wahrscheinlichkeit angibt, mit der das Wort “Buchen“ mit NN markiert wird.

   b) Der Wert $P(NN|START) = \Pi_1 = 0,2$, der die Wahrscheinlichkeit angibt, mit der ein Nomen (NN) am Anfang eines Satzes steht.

   Durch die Multiplikation der beiden Werte $\Pi_1 \times b_{11} = 0,04$ kann nun der endgültige Funktionswert von $\delta(1, NN)$ berechnet werden. Dieser ergibt sich aus dem Produkt des eben berechneten Wertes und dem Wert von $\delta(0, \underbrace{\psi(1, \text{NN})}_{\text{START}}) = 1,0$, also dem δ-Wert des Vorgängerknotens.

Die vorgestellten Schritte werden für jeden Tag t durchgeführt, wobei die Knoten, deren δ-Wert gleich 0 ist, nicht weiter bearbeitet werden müssen (in Abbildung 4.1 sind das die grauen Knoten). Am Ende gibt es zwei Knoten, deren δ-Wert ungleich 0 ist, nämlich {“Buchen“,NN} und {“Buchen“,VVIMP}. Diese beiden Knoten sind nun die einzigen beiden Knoten, die weiter bearbeitet werden müssen. Am Beispiel des Knotens {“Sie“,PPER} wird nun gezeigt, wie der Graph weiter aufgebaut wird, bis alle Knoten expandiert sind:

a) Es gibt zwei Möglichkeiten, zu diesem Knoten zu gelangen, die eine ist der Pfad über den Knoten {“Buchen“,NN}, die andere Variante wäre über den Knoten {“Buchen“,VVIMP}. Es müssen also die δ-Werte für beide Varianten errechnet werden:

   a) {“Buchen“,NN}: Die Wahrscheinlichkeit $P(Sie|PPER) = b_{42} = 1$ wird multipliziert mit $P(ARTDEF|NN) = a_{13} = 0,4$, was einen Wert von 0,4 ergibt. Dieser Wert wird nun mit $\delta(1, NN) = 0,04$ multipliziert, was 0,016 ergibt.

   b) {“Buchen“,VVIMP}: Hier wird analog zu oben verfahren, indem zunächst das Produkt $P(Sie|PPER) \times P(PPER|VVIMP) = b_{42} \times a_{23} = 0,4$ gebildet wird. Das Ganze wird nun mit dem δ-Wert des Vorgänger-Knotens 0,24 multipliziert, womit $\delta(2, VVIMP) = 0,096$ gilt.

b) Die anderen Vorgänger-Knoten müssen nicht miteinbezogen werden, da hier sich eine Multiplikation mit 0 ergeben würde.

c) Aus den beiden Varianten wird nun diejenige ausgewählt, die die größere Wahrscheinlichkeit hat, also letztere. Für den Knoten {„Sie“,PPER} gilt damit $\delta(2, PPER) = 0,096$ und $\psi(2, PPER) =$ {“Buchen“,VVIMP}.

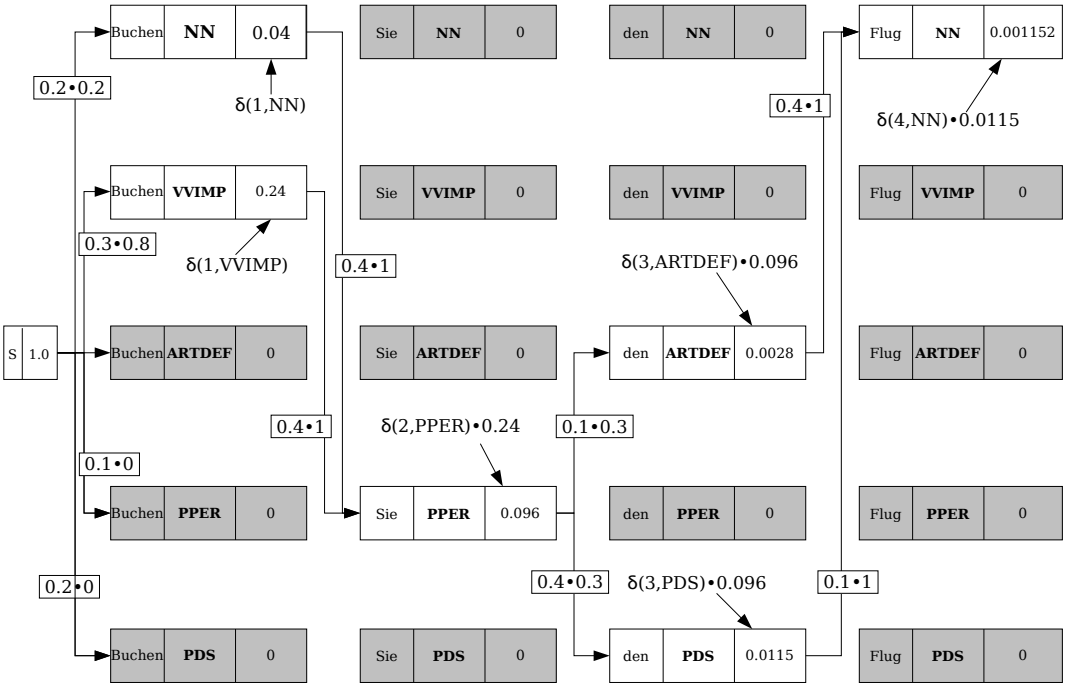


Abbildung 4.1: Graph am Ende der zweiten Phase des Viterbi-Algorithmus

Wie beschrieben werden die restlichen Knoten für die restlichen Wörter aufgebaut. Wenn der Graph (vgl. Abbildung 4.1) fertig ist, kann in der dritten Phase des Algorithmus ein Pfad ausgelesen werden, der das Tagging mit der größten Wahrscheinlichkeit repräsentiert.

Phase 3: Auslesen des Part-of-Speech Taggings

Aus den Knoten des letzten Schrittes beim Graphaufbau, also im Beispiel mit $i = 4$, wird nun derjenige ausgewählt, der die größte Wahrscheinlichkeit, also den größten δ-Wert hat. Für das Beispiel wäre das der Knoten {“Flug“,NN} mit $\delta(4, NN) = 0,001152$. Von diesem Knoten aus kann nun solange mit Hilfe der ψ-Funktion immer der Vorgänger-Knoten ermittelt werden, bis der Startknoten S erreicht ist. Für jeden Knoten werden also die folgenden Schritte durchgeführt:

a) Berechne den Vorgänger des aktuellen Knotens mit der ψ-Funktion.

b) Speichere den berechneten Knoten in einer FIFO-Warteschlange (First in – First out) ab.

Wie zu erwarten, ergibt sich dadurch das gewünschte Tagging des Satzes “Buchen Sie den Flug“:

Buchen/VVIMP Sie/PPER den/ARTDEF Flug/NN

Formale Spezifikation des Algorithmus

Definition 4.3.3 *Eine Implementierung des Viterbi-Algorithmus im Pseudo-Code sieht wie folgt aus*[3]*, wobei* w_i *das* i*-te Wort des Eingabesatzes ist:*

Algorithmus 1 Pseudo-Code des Viterbi-Algorithmus

Require: Satz der Länge n
{Phase 1:}
createNode(S);
δ(0,START) = 1.0;
{Phase 2:}
for all Tag t **do**
  δ(1,t) = $\pi_t \times P(w_1|t)$;
  ψ(1,t) = S;
end for
for i=1 to n-1 **do**
  for all Tag t_{akt} **do**
    if ($\delta(i, t_{akt}) > 0$) **then**
      for all Tag t_{next} **do**
        tmp = $\delta(1, t_{akt}) \times P(w_{i+1}|t_{next}) \times P(t_{next}|t_{akt})$;
        if ($\delta(i+1, t_{next}) < t_{next}$) **then**
          $\delta(i+1, t_{next}) = tmp$;
          $psi(i+1, t_next) = t_{akt}$;[4]
        end if
      end for
    end if
  end for
end for
{Phase 3:}
max = 0;
for all Tag t **do**
  if (max < δ(n,t)) **then**
    max = δ(n,t);
    t_{max} = t;
  end if
end for
X[n] = t_{max};
i=n;
while (i > 1) **do**
  X[i-1] = $\psi(i, t_{max})$;
  $t_{max} = \psi(i, t_{max})$;
  i = i - 1;
end while
return X; {Vektor $X = X_1, ..., X_n$ wobei X[i] den Tag des i-ten Wortes enthält}

In den Zeilen 3 bis 6 wird für das erste Wort des Satzes der Aufbau des Graphen durchgeführt. Dies muss seperat erfolgen, weil die Wahrscheinlichkeiten mit Hilfe des

[3] Eine weitere Implementation findet sich in [JM00]

Vektors Π berechnet werden und nicht wie in den Zeilen 7 bis 19 unter Benutzung der Matrix A ($P(t_{next}|t_{akt})$). Durch die IF-Anweisung in Zeile 9 wird eine Verarbeitung von Knoten verhindert, deren Wahrscheinlichkeit 0 ist.

In den Zeilen 21 bis 26 wird die größte Wahrscheinlichkeit ermittelt und von dem Knoten aus, der die größte Wahrscheinlichkeit hat, wird dann durch Benutzung der ψ-Funktion in den Zeilen 27 bis 33 der Pfad, der das beste Tagging repräsentiert in ein Feld der Länge n eingetragen. Zeile 34 gibt dann dieses Feld als Repräsentation des Taggings zurück.

4.3.4 Evaluierung: Stochastische Tagger

Der Vorteil der stochastischen Tagger gegenüber den regelbasierten Tagger ist, dass das Aufstellen komplexer Regelsysteme zur Elimination von Doppeldeutigkeiten im Tagging wegfällt. Dieser Vorteil wird aber recht schnell relativiert, wenn man bedenkt, dass für die Berechnung der Wahrscheinlichkeiten ein recht großer Trainingskorpus vorhanden sein muss, der von Hand annotiert werden muss. Die Qualität des Taggings hängt daher auch wesentlich mit der des Trainingskorpus zusammen. Ein weiterer, allerdings kleiner Nachteil ergibt sich aus der Tatsache, dass die für die Repräsentation der Wahrscheinlichkeiten relativ große Matrizen notwendig sind: Bei einem Tagset wie dem STTS ist das, wenn es sich um einen echten HMM-Tagger handelt, alleine für die Matrix A eine 54×54 Matrix. Vor allem aber die Matrix B kann extrem groß werden, denn ihre Größe hängt von der Anzahl der verschiedenen Wörter ab, die im Trainingskorpus verwendet wurden. Beide Arten von Taggern haben Vor- und vor allem gravierende Nachteile. Deswegen hat Eric Brill einen weiteren Tagger entworfen, der versucht von beiden Seiten die Vorteile zu benutzen und die Nachteile zu umgehen. Das Ergebnis dieser Arbeit ist der sogenannte Brill Tagger und dieser wird im nächsten Abschnitt vorgestellt.

4.4 Der Brill Tagger

Grundlage des von Eric Brill entwickelten Brill Taggers ist das sogenannte *Transformation Based Learning (TBL)* [Bri92], welches ebenfalls von Eric Brill eingeführt wurde. Transformation Based Tagging hat sowohl regelbasierte Anteile als auch stochastische Anteile.

Transformation Based Tagging

In Anlehnung an die regelbasierten Tagger hat auch der Brill Tagger Regeln, die zur Korrektur von falsch gesetzten Tags und für das Taggen von unbekannten Wörtern benutzt werden. Analog zu den stochastischen Taggern müssen Wahrscheinlichkeiten gelernt werden oder anderweitig vor der Anwendung vorhanden sein. Ein Brill-Tagger hat mehrere verschiedene Komponenten:

Lexikon: Dies ist die Komponente, die analog zu den Wörterbüchern der anderen beiden Tagger-Arten aufgebaut ist.

Transformation Rules (TR): Ähnlich wie bei regelbasierten Taggern werden diese Regeln zur Korrektur von Tagging-Fehlern benutzt. Durch Benutzung des Transformation Based Learning können die Regeln jedoch automatisch „gelernt“ werden und müssen nicht manuell definiert werden.

Lexical Rules (LR): Regelwerk, welches vor allem mit Hilfe von morphologischen Regeln versucht, *unbekannten Wörtern* den richtigen Tag zuzuweisen.

Bigramme: Zerlegung des zu taggenden Satzes in Wortpaare, die dann im wesentlichen von den Lexical Rules benutzt werden.

Die einzelnen Komponenten werden jetzt im Detail vorgestellt.

Das Lexikon

Wie bereits angemerkt wurde, hat das Lexikon eine ähnliche Struktur, wie das Wörterbuch bei den regelbasierten Taggern: Jedem Wort wird eine Folge möglicher Tags zugeordnet. Der einzige Unterschied ist, dass die Tags zu einem Wort nach Wahrscheinlichkeit absteigend sortiert in dieser Folge vorkommen:

Wort	POS Tags (Sortiert nach Wahrscheinlichkeit.)
Buchen	{VVIMP, NN}
Sie	{PPER}
den	{ARTDEF, PDS, PRELS}
Flug	{NN}

Für das Taggen wird dann später zunächst jedem Wort der wahrscheinlichste Tag zu diesem Wort zugeordnet.

Transformation Rules

Da für das Taggen zunächst nur der wahrscheinlichste Tag des Wortes benutzt wird, entstehen offensichtlich Fehler:

a) „Buchen/VVIMP Sie/PPER den/ARTDEF Flug/NN."

b) „Die/ARTDEF Buchen/VVIMP sind/VVAIMP groß/ADJ."

Im zweiten Satz ist das Wort „Buchen" offensichtlich kein Imperativ und daher ist das Tagging falsch. In Anlehnung an die regelbasierten Tagger werden nun Regeln benutzt, um solche Fehler zu vermeiden. Der wesentliche Unterschied ist, dass die Regeln beim Brill Tagger automatisch gelernt werden können, wie nachfolgend vorgestellt.

Wie oben bereits beispielhaft vorgeführt wurde, wird zunächst ein Trainingskorpus so markiert, dass immer der wahrscheinlichste Tag für das entsprechende Wort benutzt wird. Dadurch entstehen Fehler, die durch den Vergleich mit dem manuell annotierten Trainingskorpus gefunden werden können. Diese Fehler werden dann in eine Tabelle eingetragen, die die Fehler kategorisiert: Die Tabelle 4.9 enthält Tripel $\{t_{\text{Haben}}, t_{\text{Soll}}$,Anzahl$\}$ in denen die Anzahl der Fehler gespeichert werden, bei denen ein Tag t_{Haben} statt eines Tags t_{Soll} verwendet wurde:

t_{Haben}	t_{Soll}	Anzahl
VVFIN	NN	134
VVIMP	NN	1
ADJ	VVFIN	54

Tabelle 4.9: Beispiel für Fehler-Tripel beim Transformation Based Learning

Die erste Zeile besagt, dass es an 134 Stellen Fehler gab, bei denen ein Wort, welches richtig markiert ein Nomen (NN) ist, stattdessen als Verb (VVFIN = finites Vollverb) markiert wurde. Der Brill Tagger versucht, aus diesen Fehlern zu lernen und automatisch Regeln zu erzeugen. Problem dabei ist, dass es theoretisch möglich wäre unendlich viele Regeln automatisch zu generieren: „Ändere Tag VVFIN in NN wenn das erste Word davor den Tag x hat, das 2. Wort davor den Tag y,... das n-te Wort davor den Tag... usw."

Um die Anzahl der Regeln überschaubar und dadurch auch die Zeit für die Anwendung dieser Regeln in einem gewissen Rahmen zu halten, werden sog. *Templates* eingeführt. Diese Templates sorgen dafür, dass automatisch erzeugte Regeln immer eine bestimmte Form haben müssen[5]:

Templates

a) „Ändere Tag a in Tag b, wenn das vorgehende (nachfolgende) Wort x ist."

b) „Ändere Tag a in Tag b, wenn eines der drei vorhergehenden (nachfolgenden) Wörter x ist."

c) „Ändere Tag a in Tag b, wenn das vorhergehende (nachfolgende) Wort mit dem Tag t markiert wurde und das vorhergehende (nachfolgende) Wort x ist."

In den obigen Regeln sind a,b,t und x jeweils Variablen für beliebige Wörter bzw. Tags. Aus jedem Template und jedem Fehler-Tripel wird nun genau eine Regel erzeugt. Diese Regel wird angewendet und die *Fehlerreduktion* berechnet:

Korrigierte Fehler: Die Anzahl der Fehler, die durch Anwendung der Regel korrigiert wurden.

Beispiel: Es wäre möglich, dass von den 134 fälschlich markierten Verben nach Anwendung einer Regel nur noch 50 übrig sind. Die Anzahl der korrigierten Fehler wäre demnach 84.

Erzeugte Fehler: Die Anzahl der Fehler, die durch Anwendung der Regel erst gemacht wurden.

Beispiel: Nach der Anwendung der Regel sind 21 neue Fehler hinzugekommen, weil die Regel unter bestimmten Voraussetzungen einen Fehler erzeugt.

Die Fehlerreduktion ergibt sich demnach aus Differenz der Anzahl der korrigierten Fehler und der Anzahl der erzeugten Fehler. In dem Beispiel oben sind das 63.

Diejenige Regel, die die beste Fehlerreduktion hat, wird dann dem Regelsystem hinzugefügt.

Nachteil dieser Methode zum automatischen Erlernen von Regeln ist, dass immer noch ein manuell annotierter Korpus vorhanden sein muss. Dieses Problem wurde teilweise von Eric Brill [Bri95] gelöst, wird aber im Folgenden nicht betrachtet.

Bigramme und Kontextregeln

Diese beiden Komponenten werden zusammen vorgestellt, weil sie bei dem Versuch Wörter zu taggen, die nicht im Trainingskorpus vorkamen und daher unbekannt sind, zusammenarbeiten. Grundsätzlich gibt es verschiedene Ansätze, um diese unbekannten Wörter zu taggen:

[5] Dies ist nur eine kleine Auswahl, es gibt insgesamt 21 solcher Templates in der entsprechenden Publikation von Eric Brill [Bri92]

Die „dümmste“ Möglichkeit mit diesen Wörtern umzugehen wäre, ihnen einfach einen konstanten oder zufälligen Tag zuzuweisen, zum Beispiel NN, das heißt alle unbekannten Wörter werden als Nomen markiert. Die „Lösung“ ist jedoch sehr problematisch, da es sehr wahrscheinlich ist, dass hier ein Fehler gemacht wird. Das Problem dabei ist nicht der einzelne Fehler, sondern die Tatsache, dass das Auftreten eines falschen Tags bzw. eines Fehlers wahrscheinlich zu einer Reihe von Folgefehlern führt. Dieser Ansatz ist also weniger geeignet.

Eine weitere Möglichkeit ist die Bearbeitung mit Bigrammen: Der zu taggende Text wird in Bigramme zerlegt, also für das Beispiel 4.2.1, wie folgt:

{{It, is}, {is,expected}, {expected, that}, {that, Mr.}, {Mr.,Montoya}, {Montoya, will}, {will,race}, {race, tomorrow}[6]

Es ist wahrscheinlich, dass das Wort „Montoya“ nicht im Trainingskorpus vorhanden war und daher das Zuweisen eines initialen Tags nicht möglich war. Mit Hilfe der Bigramme wäre es aber möglich, an Hand der umliegenden Wörter den wahrscheinlichsten Tag zu finden.

Ähnlich zu den Transformation Rules ist der Aufbau der Contextual Rules: Hier wird an Hand von Suffixen, Affixen und Präfixen, der Schreibweise oder anderen Wortmerkmalen versucht, den richtigen Tag zu finden. Zum Beispiel sind im Englischen unbekannte Wörter, die groß geschrieben sind, relativ häufig Eigennamen, womit der richtige Tag im obigen Beispiel leicht zu finden ist. Ähnlich könnte es bei einem Wort funktionieren, dass zum Beispiel auf die Endung „-ous“ endet, womit ein Tagging des Wortes als Adjektiv wahrscheinlich richtig ist.

4.4.1 Ablauf des Brill Taggings

Der Ablauf des Taggens mit dem Brill-Tagger ist sehr ähnlich zu dem eines regelbasierten Taggers:

a) Zuweisen eines initialen Tags zu allen bekannten Wörtern

b) Zuweisung von Tags bei Wörtern, die nicht im Trainingskorpus vorkamen mit Hilfe von Kontextregeln und Bigrammen.

c) Anwendung der Lexical Rules zur Korrektur von Fehlern, die während der ersten beiden Schritte gemacht wurden.

4.4.2 Evaluierung: Brill Tagger

Im Gegensatz zu den anderen regelbasierten Ansätzen in [KS63] und [GR71] ist die Qualität des Ergebnisses des Brill Taggers wesentlich besser. Die Qualität der Ergebnisse der stochastischen Tagger, welche bisher von den regelbasierten Systemen nicht erreicht werden konnten, werden sogar noch geringfügig verbessert. Ein weiterer Vorteil gegenüber den stochastischen Taggern ist die Tatsache, das wesentlich weniger Information gespeichert werden muss: Wie bereits bei der Evaluation der stochastischen Tagger in Abschnitt 4.3.4 angesprochen wurde, müssen die Kontextinformationen in riesigen Matrizen abgespeichert werden, während der Brill Tagger mit weniger als 100 Regeln auskommt.

[6] Auf die Betrachtung der Satzzeichen als eigene Tokens wurde hier verzichten, weil es für das Beispiel unerheblich ist.

Ferner ist es zusätzlich möglich, die Regeln ohne einen manuell annotierten Korpus zu lernen, d.h. es genügt ein nicht annotierter Korpus und ein Wörterbuch, in dem lediglich die möglichen Tags zu einem Wort in beliebiger Reihenfolge (also nicht wie bei dem vorgestellten Modell nach Wahrscheinlichkeit geordnet) enthalten sind, um entsprechende Regeln zu lernen. Dieses Verfahren wird in [Bri95] vorgestellt.

Durch die Möglichkeit Regeln vollkommen automatisch zu lernen ist der Brill Tagger damit auch sehr portabel und für beliebige Sprachen mit geringem Aufwand einsetzbar.

In [RS95] wurde desweiteren gezeigt, dass die Regeln des Brill Taggers in endliche Automaten umgewandelt werden können. Eine solche Umwandlung führt dazu, dass der Brill Tagger für das Taggen eines Satzes mit n Wörtern genau n Schritte benötigt. Damit ist der Brill Tagger zehnmal schneller als der schnellste stochastische Part-of-Speech Tagger.

4.5 Ausblick

Neben den bisher vorgestellten Taggern gibt es noch einige Varianten verschiedener Tagger, die die grundlegenden Ideen aus den vorhergehenden Abschnitten benutzen, dabei aber meist Nachteile durch die Verwendung von speziellen Mechanismen abmildern oder gänzlich umgehen.

4.5.1 Der Hepple Tagger

Wie in Abschnitt 4.4 gezeigt, werden die Korrekturregeln des Brill-Taggers mit Hilfe von Templates gelernt und später analog zu den Regeln der regelbasierten Tagger angewendet, um Fehler, die durch das Zuweisen des wahrscheinlichsten Tags im ersten Schritt des Taggings entstehen, zu korrigieren. Durch das automatisch erzeugte Regelsystem kann es passieren, dass Regeln sich gegenseitig beeinflussen. Dieses Phänomen nennt man *Rule Interaction*.

Rule Interaction

Beispiel 4.5.1 *Es seien zwei Regeln gegeben:*

a) „Ändere Tag B zu Tag D wenn der vorhergehende Tag A ist.“

b) „Ändere Tag C zu Tag E wenn der vorhergehende Tag D ist.“

Bei einer Tag-Folge über 3 Wörter „ABC“ führt die Ausführung der Regeln 1 dazu, dass der Tag B zu Tag D wird, also die Folge „ADC“ übrig bleibt. Danach ist die Bedingung für die zweite Regel erfüllt und diese wird dann ausgeführt, was dazu führt, dass die Folge zu „ADE“ wird.

Diese Art der rule interaction ist offensichtlich meist erwünscht, denn sonst wäre an dieser Stelle (unter der Voraussetzung das die Regeln hier einen Fehler korrigieren und keinen Fehler erzeugen) ein Fehler gemacht worden. Eine andere Art der Rule Interaction ist dagegen weniger erwünscht:

Beispiel 4.5.2 *Es seien wieder zwei Regeln gegeben:*

a) „Ändere Tag B zu Tag D, wenn der vorhergehende Tag A ist.“

b) „Ändere Tag D zu Tag E, wenn der nächste Tag C ist.“

Analog zu dem Beispiel oben, sei nun wieder eine Tag-Folge „ABC“ gegeben. Die Regel 1 würde zuerst den Tag B in einen Tag D ändern, und danach würde der gleiche Tag wieder zu E geändert.

Es ist offensichtlich, dass solche eine Anwendungskette von Regeln eher unerwünscht ist und dazu führt, dass der Tagger langsam wird oder unnötige Änderungen vornimmt. Von Mark Hepple [Hep00] wurden daher einige Verbesserungen des Brill Taggers vorgeschlagen, die solche Interaktionen zwischen Regeln vermeiden. Für diese Verbesserungen werden zwei wichtige Annahmen eingeführt:

Definition 4.5.3 *Commitment / Festlegung:*
Ein Tag der einmal durch Regelanwendung verändert wurde, muss nicht mehr verändert werden.

Definition 4.5.4 *Independence / Unabhängigkeit:*
Die Häufigkeit, mit der eine Änderung einer Regel den Kontext einer anderen Regel so verändert, dass diese ausgelöst wird, ist so gering, dass die Möglichkeit vernachlässigt werden kann.

Die Festlegungsannahme lässt offen, ob nach der Zuweisung der wahrscheinlichsten Tags im ersten Schritt des Brill Taggers die Regeln für jedes Wort der Reihe nach einzeln angewendet werden sollen (um die beste Regel auszuwählen) oder ob jede Regel für sich auf den Text angewendet werden soll. Beide Varianten für sich könnten „rule interaction“ auslösen, betrachtet man jedoch zusätzlich die Independence-Annahme, dann ist eine Interaktion zwischen verschiedenen Regeln über mehrere Tags nicht mehr möglich. Diese beiden Annahmen sind extrem wichtig für den Algorithmus, mit dem die Erweiterung des Brill Taggers von Mark Hepple die Transformationsregeln „lernt“. Der Algorithmus für das Lernen von Regeln wird dabei in mehrere Phasen aufgeteilt, so dass in jeder Phase nur Regeln gelernt werden, die einen bestimmten Tag modifizieren. Wenn in einer Phase alle Regeln zu einem Tag t gelernt werden, dann ist es wegen der Independence-Annahme nicht notwendig, dass Änderungen die andere Regeln (die bereits gelernt wurden oder die noch gelernt werden müssen) machen könnten, beim diesem Lernvorgang beachtet werden. Gleichzeitig kann wegen der Commitment-Annahme vernachlässigt werden, dass irgendeine Regel einen Tag s in den Tag t ändert und dieser Tag dann ebenfalls bearbeitet werden muss.

Desweiteren ermöglichen diese beiden Annahmen, dass jede Phase nur einen kleinen Ausschnitt bzw. eine Menge kleiner Ausschnitte bearbeiten muss, die für das Taggen mit dem Tag t relevant sind: Es müssen also genau die Stellen im Trainingskorpus bearbeitet werden, die nach der initialen Phase den Tag t haben und mindestens einen weiteren alternativen Tag[7].

Der Ablauf für das Erlernen der Transformations-Regeln ist daher wie folgt:

a) Lade den Trainingskorpus (Wörter und zugehörige korrekte Tags) als Array in den Speicher und weise jedem Wort seinen wahrscheinlichsten Tag t_{begin} zu.

b) Für jeden Tag t im Tagset führe die folgenden Schritte durch

   a) Durchsuche den Korpus nach allen Textpunkten, denen im ersten Schritt der Tag t zugewiesen wurde, wo also $t = t_{begin}$ gilt. Speichere die Textpunkte in einem Array P_t (als Indizes auf das Array, in dem der Trainingskorpus in Schritt 1 gespeichert wurde).

[7] Diese Stellen heißen im folgenden Textpunkte

b) Berechne die Fehlerreduktion f_r (jede Regel bekommt bei der erstmaligen Erzeugung einen Eintrag in einer listenähnlichen Datenstruktur) für jede mögliche Regel r, bei der der Tag t zu einem anderen Tag verändert wird:
   - Inkrementiere die Fehlerreduktion f_r um 1 für jeden Textpunkt, bei dem im ersten Schritt fälschlicherweise der Tag t zugewiesen wurde und bei dem die Regel r den Tag t so verändert hätte, dass der Tag korrekt gewesen wäre.
   - Dekrementiere die Fehlerreduktion f_r um 1 für jeden Textpunkt, welcher einen korrekten Tag t nach dem ersten Schritt hatte, aber durch Anwendung der Regel r nicht mehr korrekt markiert wäre.

c) Suche die beste Regel r_{max}, also diejenige Regel r bei der $f_r > f_{r(i)} \forall i$ gilt, und füge die Regel zur endgültigen Regelliste hinzu.

d) Wenn $f_{r(max)}$ unterhalb eines bestimmten Abbruchkriteriums liegt, dann fahre mit dem nächsten Tag t aus dem Tagset fort.

e) Ansonsten müssen alle Textpunkte, an denen die Regel r_{max} ausgelöst werden würde, gefunden werden und die Fehlerreduktion aller anderen Regeln, die ebenfalls an einem dieser Textpunkte ausgelöst werden, verändert werden.

Zum letzten Schritt gibt es genau zwei Varianten, die im Prinzip gleichwertig bezüglich Performanz und Ergebnisgüte sind

Append: Hierbei wird bei einer neuen Regel (im Algorithmus r_{max}) davon ausgegangen, dass alle diese Textstellen nur von dieser einen Regel bearbeitet werden und von keiner anderen. Deswegen werden alle Textpunkte, an denen diese Regel ausgelöst wird, aus dem Array P_t entfernt. Danach muss noch für die jetzt fehlenden Textpunkte die Fehlerreduktion der anderen Regeln angepasst werden, das heisst der Wert f_r muss für genau diejenigen Regeln r, die an einem dieser Textpunkte eine Korrektur durchführen, um jeweils 1 dekrementiert werden (diese Korrekturen können nicht mehr der Fehlerkorrektur dieser Regeln zugerechnet werden). Umgedreht müssen alle Textpunkte, an denen durch Anwendung der Regel r ein neuer Fehler entstanden wäre, wieder der Fehlerkorrektur gutgeschrieben werden.

Prepend: Hierbei wird davon ausgegangen, dass eine Regel Änderungen der bisherigen Regeln „überschreibt“, also ob mit dem bisherigen Regelwerk bereits ein korrektes Tagging erreicht worden wäre oder nicht. Dementsprechend muss die Fehlerreduktion f_r einer Regel r verändert werden.

Der wichtigste Parameter für den Lern-Algorithmus des Brill Taggers ist das Abbruchkriterium (wenn die Fehlerreduktion unter diesen Wert sinkt, so wird die Regel nicht benutzt), denn die Qualität des Ergebnisses des Taggings, wie auch die Effizienz bezüglich Rechenzeit und Speicherbedarf hängt ganz entscheidend davon ab wie groß dieser Wert ist: Ist das Abbruchkriterium zu niedrig angesetzt, so steigt der Speicherbedarf und die Rechenzeit auf Grund der gestiegenen Anzahl an benutzten Regeln stark an. Umgekehrt sinkt die Qualität des Ergebnisses, wenn der Wert zu klein ist, einfach weil einige Regeln „fehlen“. Messungen von Mark Hepple [Hep00] zeigten jedoch, dass durch die Verwendung der beiden Varianten sich sowohl die Rechenzeit für das Tagging als auch die Rechenzeit für das Learning nicht mehr wesentlich ändern, wenn das Abbruchkriterium sehr niedrig angesetzt wird.

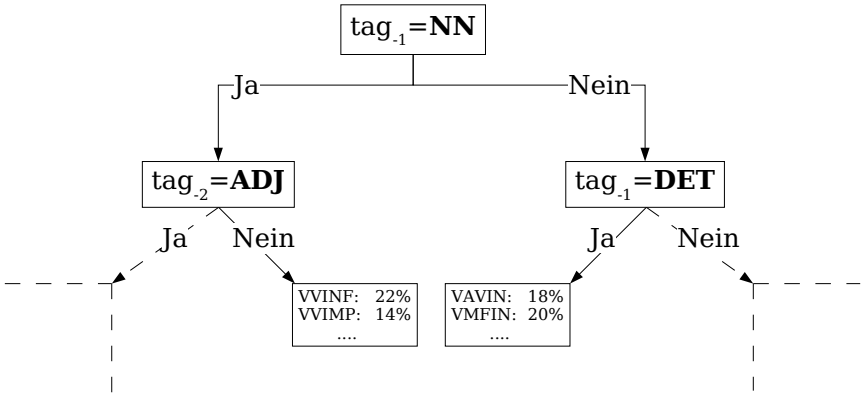


Abbildung 4.2: Beispiel für einen binären Entscheidungsbaum des Tree Taggers

4.5.2 Part-of-Speech Tagging mit Entscheidungsbäumen

Bei Training stochastischer Tagger (siehe Abschnitt 4.3.2) kann es vorkommen, das ein zum Beispiel ein Wort sehr selten ist, also vielleicht im Trainingskorpus nur 1 bis 2 mal vorkommt. Bei einem Wort, dass so selten ist, kann nur sehr grob abgeschätzt werden, welche Tags zum Beispiel für dieses Wort überhaupt benutzt werden. In Tabelle 4.3.2 sind in Matrix A zahlreiche Einträge 0. Solche Einträge können zweierlei Bedeutung haben: entweder es handelt sich um einen Übergang, der syntaktisch nicht korrekt ist (dann wäre die Wahrscheinlichkeit 0 richtig), oder es handelt sich um eine Abfolge von Tags, die relativ selten ist, nicht im Trainingskorpus vorhanden war und daher eine falsche Wahrscheinlichkeit eingetragen wurde. In Implementierungen von stochastischen Taggern wird daher versucht, solche Nullwerte in der Matrix A zu vermeiden, in dem Anstelle der Nullwerte sehr kleine positive Werte eingetragen werden. Der Nachteil dieser Methode ist offensichtlich, denn syntaktisch nicht korrekte Abfolgen von Tags bekommen eine Wahrscheinlichkeit größer als 0 zugewiesen, und dadurch werden Fehler möglich, die durch die Multiplikationskette bei der Berechnung der Wahrscheinlichkeit eigentlich vermieden werden sollen. Zusätzlich leidet die Performanz des Taggers, weil Pfade im Graphen (vgl. Abbildung 4.1) berechnet werden, die syntaktisch inkorrekte Taggings repräsentieren. Im Gegensatz zu den reinen stochastischen Taggern, wie sie oben eingeführt wurden, verwendet der Tree Tagger [Sch94] einen binären Entscheidungsbaum, um Übergangswahrscheinlichkeiten zu bestimmen. Für den Aufbau des Entscheidungsbaumes wird der ID3-Algorithmus [Qui83] verwendet, welcher im wesentlichen auf der Information Gain Heuristik basiert. Ein Beispiel für einen solchen binären Entscheidungsbaum ist in Abbildung 4.2. Der erste Knoten repräsentiert die Prüfung, ob es sich bei dem vorhergehenden Wort um ein Adjektiv handelt. Wenn das der Fall ist, wird der linke Teilbaum benutzt, ansonsten der rechte. Dieses Verfahren wird solange fortgesetzt, bis ein Blatt erreicht ist. In den Blättern stehen die Wahrscheinlichkeiten für die möglichen Tags und mit diesen kann nun das Tagging ähnlich zum dem in Abschnitt 4.3.3 vorgestellten Verfahren fortgesetzt werden.

Der Tree Tagger erreicht auf dem Penn-Treebank Korpus eine Genauigkeit von 96,36%, während die besten stochastischen Tagger knapp 96% erreichen.

Literaturverzeichnis

[Bri92] Eric Brill. A simple rule-based part-of-speech tagger. In *Proceedings of ANLP-92, 3rd Conference on Applied Natural Language Processing*, pages 152–155, Trento, IT, 1992. http://citeseer.ist.psu.edu/brill92simple.html.

[Bri95] Eric Brill. Unsupervised Learning of Disambiguation Rules for Part of Speech Tagging. In David Yarovsky and Kenneth Church, editors, *Proceedings of the Third Workshop on Very Large Corpora*, pages 1–13, Somerset, New Jersey, 1995. Association for Computational Linguistics. http://citeseer.ist.psu.edu/article/brill95unsupervised.html.

[DeR88] S. DeRose. Grammatical category disambiguation by statistical optimization. *Computational Linguistics*, 14:31–39, 1988. http://www.derose.net.

[GR71] B. Greene and G. Rubin. Automatic Grammatical Tagging of English. *Technical Report, Brown University, Providence, RI*, 1971.

[Hep00] Mark Hepple. Independence and Commitment: Assumptions for Rapid Training and Execution of Rule-based POS Taggers. In *ACL*, 2000. http://www.aclweb.org/anthology/P00-1036.

[JM00] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall PTR, 2000.

[KS63] Sheldon Klein and Robert F. Simmons. A Computational Approach to Grammatical Coding of English Words. *Journal of the ACM*, 10(3):334–347, 1963. http://doi.acm.org/10.1145/321172.321180.

[MSM94] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1994. http://citeseer.ist.psu.edu/marcus93building.html.

[NEG] NEGRA Korpus Version 2. http://www.coli.uni-saarland.de/projects/sfb378/negra-corpus/negra-corpus.html.

[Qui83] J.R. Quinlan. Learning efficient classification procedures and their application to chess end games. *Machine Learning: An artificial Intelligence approach*, pages 463–482, 1983.

[Ray] Paul Rayson. The CLAWS Web Tagger. http://citeseer.ist.psu.edu/291455.html.

[RS95] Emmanuel Roche and Yves Schabes. Deterministic Part-of-Speech Tagging with Finite-State Transducers. *Computational Linguistics*, 21(2):227–253, 1995. http://citeseer.ist.psu.edu/roche95deterministic.html.

[Sch94] Helmut Schmid. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *International Conference on New Methods in Language Processing*, Manchester, UK, 1994. http://citeseer.ist.psu.edu/schmid94probabilistic.html.

[STST95] Anne Schiller, Simone Teufel, Christine Stöckert, and Christine Thielen. Stuttgart Tübingen Tagset, 1995. Vorläufige Guidelines für das Tagging deutscher Textcorpora mit STTS http://www.sfs.nphil.uni-tuebingen.de/Elwis/stts/stts.html.

[Vou95] A. Voutilainen. A syntax-based part of speech analyser, 1995. http://citeseer.ist.psu.edu/579502.html.

Thomas Gitzinger

5 Berechnung von Koreferenzketten

In einem Zeitungsbericht, zum Beispiel über die SAP AG, wird diese sehr wahrscheinlich unter verschiedenen Namen erwähnt, etwa „SAP“, „SAP AG“, „Deutschlands grösster Softwarehersteller“ oder auch einfach „das Unternehmen“. Für ein Textanalysesystem, zum Beispiel einen automatischen Zusammenfasser, ist es unerlässlich zu wissen, dass sich all diese Begriffe auf die selbe Sache beziehen. Dies wird durch die Bildung von Koreferenzketten erreicht.

5.1 Einleitung

Zunächst erklären wir einige Begriffe, mit denen wir arbeiten werden. In Abschnitt 5.3 berichten wir über einige historische Entwicklungen im Zusammenhang mit Koreferenzberechnung. Der unter Zeitmangel leidende oder ausschliesslich an den heutigen Fakten interessierte Leser kann diese Ausführungen überspringen, ohne dass ihm bei den späteren Teilen der Ausarbeitung ein Nachteil entsteht.

Nachfolgend legen wir die Koreferenzberechnung in GATE [Cun02, CMBT02] dar (Abschnitt 5.4), bevor wir zum Hauptthema dieses Textes kommen, der Koreferenzberechnung mit Hilfe von Fuzzy-Methoden (Abschnitt 5.5). Anschliessend betrachten wir die Bewertung berechneter Koreferenzen (Abschnitt 5.6) samt der wichtigsten dabei auftretenden Probleme. Den Abschluss dieses Kapitels bildet ein Blick auf besondere Fälle der Koreferenzauflösung (biomedizinische Literatur) (Abschnitt 5.7) beziehungsweise auf die praktische Bedeutung der Koreferenzkettenbildung für darauf aufbauende Anwendungen.

5.2 Begriffe

Benannte Entitäten In einem Text, den wir untersuchen wollen, interessieren uns in erster Linie die darin vorkommenden Personen, Personengruppen, Orte, Organisationen und Gegenstände. Man fasst diese Personen und Objekte durch den abstrakten Begriff *benannte Entitäten* (englisch *named entities*) zusammen.

Benannte Entitäten

Bei der maschinellen Verarbeitung natürlichsprachiger Texte können benannte Entitäten durch eine Kombination von Wortlisten und Grammatiken erkannt werden. Sinnvolle Wortlisten enthalten etwa Vor-/Nachnamen von Personen, Städte- und Ländernamen, Namen von Firmen et cetera. Das Erkennen benannter Entitäten mit Hilfe von Wortlisten heisst *Gazetteering*. Ein Beispiel für eine Grammatik, die erkennen soll, dass eine Person vorliegt, ist *Person = Titel + Vorname + mittlerer Name + Nachname*. In GATE werden JAPE-Grammatiken verwendet, die durch den *Named Entity Transducer* Anwendung finden.

Gazetteering

Nominalphrasen Aus grammatikalischer Sicht sind benannte Entitäten enthalten in *Nominalphrasen* (englisch *noun phrases, NPs*). Nominalphrasen sind aufgebaut aus

Nominalphrasen

- Keinem oder einem *Artikel* („der“, „eine“, . . .)
- Keinem, einem oder mehreren *Modifiern*. Hierzu zählen unter anderem Adjektive („ein *grosses* Unternehmen“) und Partizipien („die *treibende* Kraft“, „ein *verstossener* Mann“).
- Genau einem *Head*. Dieses Wort bestimmt Geschlecht und Anzahl der gesamten Nominalphrase.

Beispiel 5.2.1 *Beispiele für Nominalphrasen:*

Artikel	*Modifier*	*Head*
das	*grosse*	*Unternehmen*
	viele	*Firmen*
		SAP
		sie
der		*20. Juni*
		$20

Wie wir in der Tabelle sehen, kann der Head einer Nominalphrase unter anderem ein Substantiv („Unternehmen“, „Firmen“), der Name einer Organisation oder Person („SAP“), ein Pronomen („sie“) oder auch eine Datums- oder Währungsangabe („20. Juni“, „$20“) sein. Im Falle eines Pronomens fallen Artikel und Modifier weg („die sie“ oder „schöne sie“ geht nicht oder ist zumindest zweifelhaftes Deutsch).

Zu bemerken ist, dass die genauen Möglichkeiten, die für den Aufbau einer NP in Frage kommen, von der Sprache abhängen. So ist eine benannte Entität im Englischen ohne weiteres als Modifier möglich: In „the *tin foil* roll“ sind „tin“ und „foil“ benannte Entitäten, die „roll“ modifizieren, in "the *Microsoft* policy“ ist der Name „Microsoft“ Modifier für „policy“. Im Deutschen geht das traditionell nicht, man schreibt nicht „die Aluminium Folie Rolle“, sondern „die Aluminiumfolienrolle“. Interessanterweise setzt sich diese Art der NP-Bildung jedoch in der Praxis auch im Deutschen durch, vor allem bei Buch- und Filmtiteln („Die *Bourne* Identität“). Lebendige Sprachen sind in ständigem Wandel, und Systeme zur Sprachverarbeitung müssen dies berücksichtigen.

Referenz und Koreferenz Jede Nominalphrase bezieht sich auf eine Entität. Man sagt, sie *referiert* die entsprechende Entität. Die Entität wiederum heisst *Referent* der NP. Zwei NPs, die sich auf die selbe Entität beziehen, heissen *koreferierend* (englisch *to corefer*). Zwischen ihnen herrscht *Koreferenz*.

Referent

Koreferenz

Beispiel 5.2.2 *„Luke verlor die Kontrolle. Er konnte das Schiff nicht mehr steuern.“*

Hier koreferieren die Nominalphrasen „Luke“ und „er“, denn sie beziehen sich auf die selbe Entität, eine Person namens Luke.

Koreferenz definiert eine Äquivalenzrelation: Eine Nominalphrase koreferiert offensichtlich mit sich selbst (denn sie bezieht sich auf die selbe Entität wie sie selbst), sie ist symmetrisch (wenn A mit B koreferiert, so koreferiert auch B mit A) sowie transitiv (wenn A mit B koreferiert und B mit C, so koreferieren auch A und C).

Koreferenzkette Eine *Koreferenzkette* $c \in C$ ist eine Menge von Nominalphrasen: $c \subseteq$ NP. Hierbei wird nicht die Forderung erhoben, dass *alle* enthaltenen Nominalphrasen miteinander koreferieren müssen, auch wenn dies natürlich angestrebt wird. Der Umgang mit Fehlern in Koreferenzketten wird im Abschnitt 5.6 betrachtet. Koreferenzkette

Weitere Begriffe Bevor wir den Abschnitt abschliessen, sind noch die Begriffe *Anapher* (englisch *anaphor*), *Katapher* (englisch *cataphor*) und *Antezedens* (englisch *antecedent*) zu erwähnen. Eine Anapher ist ein Ausdruck, der auf etwas bereits vorher genanntes verweist: Anapher

Beispiel 5.2.3 *„Luke verlor die Kontrolle. Er konnte das Schiff nicht mehr steuern.“*

Hier ist „er“ eine Anapher, denn es verweist auf das bereits vorher genannte „Luke“. Umgekehrt ist eine Katapher ein Vorwärtsverweis: Katapher

Beispiel 5.2.4 *„Nachdem er gelandet war, machte sich Luke auf die Suche.“*

Ein Antezedens schliesslich bezeichnet denjenigen Ausdruck, auf den eine Anapher verweist: Zu dem „er“ im ersten Beispiel ist „Luke“ das Antezedens. Antezedens

Ein wichtiger Teil der Koreferenzauflösung ist es offensichtlich, zu einer Anapher das richtige Antezedens zu finden. Dieser Prozess heisst im Englischen *anaphora resolution*.

5.3 Geschichte

In diesem Abschnitt stellen wir exemplarisch drei unterschiedliche Vorgehensweisen vor, pronominale Koreferenzen aufzulösen. Es geht dabei um den Baumsuchalgorithmus von Hobbs, den salienzbasierten Algorithmus von Lappin und Leass sowie um heutige statistische Ansätze. Die Unterschiede der Verfahren zeigen die historische Entwicklung der Methoden der Pronomenauflösung.

Am Ende des Abschnitts gehen wir auf die Message Understanding Conferences (MUCs) des amerikanischen National Institute for Standards and Technology (NIST) ein, welche viel zur Forschung auf dem Gebiet der automatischen Textanalyse im allgemeinen und zur Weiterentwicklung der Koreferenzberechnung im besonderen beigetragen haben.

5.3.1 Algorithmus von Hobbs

Seit den Anfängen der Computerlinguistik in den frühen 1970er Jahren wurde vorwiegend versucht, einen zu analysierenden Text möglichst in seiner Gesamtheit zu erfassen. Für die Pronomenauflösung bedeutete dies, dass man bemüht war, möglichst viele semantische Implikationen eines Pronomens und des zugehörigen Verbs bei der Bestimmung eines Antezedens in Betracht zu ziehen. Dies zu realisieren war sehr kompliziert, und der Erfolg stellte sich nicht ein.

In einem 1978 veröffentlichten Papier [Hob78] beschreibt Jerry R. Hobbs einen Algorithmus, der im Gegensatz zu den bisherigen Methoden *syntaxbasiert* ist. Vor der Anwendung dieses Algorithmus muss zunächst ein Parser den vorliegenden (englischen) Text mit Hilfe einer kontextfreien Grammatik in Syntaxbäume umwandeln. Dabei entspricht jeder Satz einem Parse-Baum. Wir betrachten in Tabelle 5.1 einen Ausschnitt aus der Grammatik.

$$
\begin{array}{lll}
S & \rightarrow \text{NP VP} & \\
\text{NP} & \rightarrow \text{(Det) } \overline{N} \text{ (PP|Rel)}^* & \mid \quad \text{pronoun} \\
\text{Det} & \rightarrow \text{article} \mid \text{NP}'s & \\
\overline{N} & \rightarrow \text{noun (PP)}^* & \\
\text{PP} & \rightarrow \text{preposition NP} & \\
\text{Rel} & \rightarrow \text{wh-word } S & \\
\text{VP} & \rightarrow \text{verb NP (PP)}^* &
\end{array}
$$

Tabelle 5.1: Produktionen der von Hobbs verwendeten CH-2-Grammatik

Man sieht an der ersten, zweiten und vorletzten Regel, dass die Hauptnominalphrase (NP) in einem Satz (S) einen Relativsatz (Rel) hervorbringen kann, der wiederum einen S-Knoten produziert. Diese S-Knoten sind wichtig für den Algorithmus. Der Algorithmus arbeitet wie folgt (wir verzichten hier auf die genaue Wiedergabe der 9 Schritte des Originaldokuments und hoffen, mit unserer Beschreibung ein etwas intuitiveres Verständnis für die Vorgehensweise zu vermitteln), wobei p das Pronomen bezeichnet, dessen Antezedens man ermitteln möchte:

a) Wir gehen vom Pronomen p aus den Baum aufwärts, bis wir auf einen S-Knoten treffen (Satz oder Teilsatz)

b) Wir traversieren in diesem Teilsatz alle Zweige LINKS von dem Pfad, der zu p führt. Dabei traversieren wir von links nach rechts mittels Breitensuche.

c) Wenn wir ein passendes Antezedens finden, so wählen wir dieses. Der Algorithmus terminiert. Andernfalls:

d) Ist der S-Knoten die Wurzel dieses Satzes? Wenn ja, durchsuchen wir nacheinander die vorherigen Sätze (von links nach rechts, Breitensuche). Wenn der S-Knoten nicht die Wurzel ist: Baum weiter nach oben gehen, bis wir auf einen NP- oder S-Knoten treffen. Diesen nennen wir X.

e) Wenn X ein NP-Knoten ist und der Pfad, den wir gegangen sind, nicht durch X führt, so wählen wir X.

f) Sonst: Wir traversieren alle Pfade links von dem Pfad, den wir gegangen sind (von links nach rechts, Breitensuche). Wir wählen das erste passende (Anzahl, Geschlecht) Antezedens.

g) Wurde immer noch nichts gefunden, so suchen wir auch rechts des aktuellen Pfads (aber nicht unterhalb eines NP- oder S-Knotens). Wir wählen den ersten passenden NP-Knoten.

h) Ergibt auch die Suche rechts des aktuellen Pfads kein Resultat, so wiederholen wir unser Vorgehen ab Schritt 4.

Hobbs selbst bemerkte, dass dieser Algorithmus trotz seiner „Naivität“ erstaunlich gute Ergebnisse liefert. Er testete ihn an 100 aufeinanderfolgenden Pronomen aus 3 „sehr unterschiedlichen Texten“ und stellte fest, dass der Algorithmus in Fällen mit mehr als einem möglichen Antezedens in 81,8% das richtige Ergebnis liefert. Das war weit mehr als bis dahin jeder semantische Ansatz erreichen konnte. Er erinnerte jedoch daran, dass dies keine akzeptable Lösung sei, da jeder Beispiele konstruieren könne, in denen der Algorithmus fehlschlägt. Hobbs war der Ansicht, dass auf lange Sicht kein Weg an einem semantischen Ansatz vorbei führe.

5.3.2 Algorithmus von Lappin und Leass

Der nächste Algorithmus, der vergleichbare und zum Teil bessere Ergebnisse lieferte, liess bis 1994 auf sich warten und kam von Shalom Lappin und Herbert Leass [LL94]. Er ist einfacher einzusetzen insofern als dass er keinen vollständigen syntaktischen Parser für die Vorverarbeitung benötigt, sondern nur einen Mechanismus, um Nominalphrasen zu identifizieren und ihre grammatikalischen Rollen festzustellen.

Der Algorithmus von Lappin und Leass basiert auf einem Gewichtungsschema für die benannten Entitäten, die im zu analysierenden Text vorkommen. Zentraler Begriff hierbei ist die *Salienz*. Ist eine Entität, zum Beispiel eine Person, zu einem Zeitpunkt beim Durchlesen des Textes sehr präsent, aktuell, hervorstehend, so spricht man von einer hohen Salienz dieser Entität. Saliente Entitäten werden bei der Pronomenauflösung bevorzugt.

Salienz

Ursachen für hohe Salienz sind zum Beispiel, dass die Entität im aktuellen Satz genannt wird, vielleicht sogar Subjekt dieses Satzes ist, und möglichst Head der Nominalphrase, in der sie genannt wird. Um einen Salienzwert zu berechnen, summiert man einzelne Gewichte auf, die sogenannten *Salienzfaktoren*. Eine Auflistung einiger Salienzfaktoren (für Englisch) bietet Tabelle 5.2. Die Punkte für die einzelnen Faktoren wurden experimentell bestimmt. In Klammern steht gegebenenfalls ein Beispielsatz, wobei der Satzteil, um den es geht, *hervorgehoben* ist.

Factor	*Value*
Sentence recency	100
Subject emphasis („*A car* is parked in the driveway.“)	80
Existential emphasis („There is *a car* parked in the driveway.“)	70
Accusative emphasis („John parked *his car* in the driveway.“)	50
Indirect object (Dativ) („John showed *his friend* the car.“)	40
Non-adverbial emphasis („Inside *his car*, John was looking for something.“)	50
Head noun emphasis („They met on the *car* park.“)	80

Tabelle 5.2: Salienzfaktoren im System von Lappin und Leass

Besondere Aufmerksamkeit verdienen in Tabelle 5.2 die Punkte *Non-adverbial emphasis* und *Head noun emphasis*. Bei der non-adverbial emphasis handelt es sich um

eine Strafe für Entitäten, die Teil eines Adverbials sind, so wie „car“ in „Inside his car“. Die Absicht dieser Regel wird durch zwei Beispielsätze deutlicher:

- Inside his car, John tried out his CD player. It was new.
- The car had a CD player. It was new.

Während man im ersten Beispiel stark dazu tendiert, das „It“ dem CD-Player zuzuordnen, könnte sich „It“ im zweiten Beispiel durchaus auf das Auto beziehen. Entitäten, die für sich alleine stehen, haben also intuitiv eine höhere Salienz als solche, die Teil einer Ortsangabe, Zeitangabe oder ähnlichem sind.

Ähnlich ist die head noun emphasis eine Abwertung für alle Nominalphrasen, die Teil einer grösseren Nominalphrase sind und daher nicht der bestimmende Ausdruck ihrer NP sind. Betrachtet man den Beispielsatz „They met on the car park.“, so würde zum Beispiel bei einem anschliessenden „It was very big.“ kein Mensch das Pronomen „it“ der Nominalphrase „car“ zuordnen. Daher würde gemäss Lappin und Leass die NP „car“ hier bestraft, im Sinne des Salienzwerts. Dies ist ein Phänomen der englischen Sprache und lässt sich nicht ohne weiteres ins Deutsche übertragen.

Eine wesentliche Eigenschaft des Algorithmus ist es, dass alle Salienzwerte positiv sein müssen. Daher wird in diesen Fällen nicht der Wert der betreffenden Entität herabgesetzt, sondern der Wert aller Entitäten, auf die das jeweilige Kriterium nicht zutrifft, um 50 (Adverbial) beziehungsweise 80 (head noun) erhöht.

Es ist leicht einsehbar, dass sich die Salienz einer Entität von Satz zu Satz ändert. Da Entitäten, die seit mehreren Sätzen nicht mehr erwähnt wurden, nicht mehr den hohen Salienzwert haben sollten, den sie vielleicht einmal hatten, wird der Wert jeder Entität bei Beginn eines neuen Satzes halbiert.

Pronomenauflösung Trifft man nun auf ein Pronomen, werden folgende Schritte durchgeführt:

a) Sammle potentielle Referenten (bis zu 4 Sätze zurück).

b) Entferne potentielle Referenten, die in Anzahl oder Geschlecht nicht zum Pronomen passen.

c) Entferne potentielle Referenten, die aus syntaktischen Gründen nicht mit dem Pronomen koreferieren können.

Dann kommen noch zwei weitere Salienzfaktoren zum Tragen, die in Tabelle 5.3 gezeigt sind.

Factor	*Value*
Grammatical Role Parallelism („John talked to *Bob*. John showed *him* what he had discovered.“)	35
Cataphora („Before *he* got out, *John* switched off the radio.“)	-175

Tabelle 5.3: Salienzfaktoren, die pronomenspezifisch wirken

Bezeichne p das Pronomen, dessen Referent gesucht wird. Der erste Faktor in der Tabelle bevorzugt Nominalphrasen, welche die gleiche grammatikalische Rolle haben

wie p, also zum Beispiel Subjekt oder Akkusativobjekt. Kataphorische Vorkommen werden als selten beziehungsweise unwahrscheinlich betrachtet, deshalb werden durch den zweiten Faktor solche NPs in ihrer Salienz vermindert, durch die p kataphorisch würde (das heisst, wenn p sich auf sie beziehen würde). Im Beispielsatz in der Tabelle würde also die Salienz für die Nominalphrase „John“ um 175 vermindert.

Diese beiden Faktoren kann man nur per Pronomen entscheiden, deshalb werden sie nicht schon im Vorfeld der Pronomenauflösung miteinbezogen. Schliesslich wird diejenige Nominalphrase als Referent gewählt, deren Äquivalenzklasse beziehungsweise Referent im Augenblick über die höchste Salienz verfügt.

Beispiel Betrachten wir den Algorithmus in Aktion. Da er für die englische Sprache mit ihren Besonderheiten entworfen wurde, wählen wir ein englischsprachiges Beispiel:

- Luke showed Han a new space ship. He liked it a lot.

Durch Bearbeiten des ersten Satzes bekommen wir drei potentielle Referenten für spätere Pronomen, nämlich „Luke“, „Han“ und „space ship“. Die folgende Tabelle zeigt die Zuweisung der Salienzwerte:

	Rec	Subj	Exist	Obj	Ind-Obj	Non-Adv	Head Noun	Total
Luke	100	80				50	80	310
Han	100				40	50	80	270
space ship	100			50		50	80	280

Da der Satz keine Pronomen enthält, fahren wir mit dem nächsten fort, wobei wir die eben errechneten Salienzwerte halbieren (nächste Tabelle). Die Spalte *Nominalphrasen* zeigt an, mittels welcher Nominalphrasen der Referent bisher genannt wurde.

Referent	Nominalphrasen	Salienzwert
Luke	*{Luke}*	155
Han	*{Han}*	135
space ship	*{space ship}*	140

Wir treffen auf das Pronomen „he“, welches sowohl auf „Luke“ als auch auf „Han“ passen würde. In solch einer Situation kommen eventuell die beiden pronomenspezifischen Salienzfaktoren aus Tabelle 5.3 zur Geltung. Keiner der beiden potentiellen Referenten würde das Pronomen kataphorisch machen, jedoch besteht grammatikalische Rollenparallelität zwischen „Luke“ und „he“, da beide Subjekt ihres jeweiligen Satzes sind. Daher geht der Salienzwert für „Luke“ um 35 nach oben, auf 190. Dies ist höher als die 135 von „Han“, also wird „Luke“ als Referent ausgewählt.

Anschliessend wird der Salienzwert für „Luke“ aktualisiert und „he“ in die Menge der Nominalphrasen aufgenommen, welche sich auf „Luke“ beziehen. „He“ trägt 100 Punke aufgrund seines Vorkommens im aktuellen Satz bei, 80 als Subjekt, 50 für das nichtadverbiale Vorkommen und 80 als head noun, somit werden 310 Punkte zu den 190 von „Luke“ hinzuaddiert. Das Resultat zeigt die nächste Tabelle. „He“ wird hier mit Index versehen (he_1), um es eindeutig zu kennzeichnen.

Referent	Nominalphrasen	Salienzwert
Luke	*{Luke,* he_1*}*	500
Han	*{Han}*	135
space ship	*{space ship}*	140

Als nächstes kommt das Pronomen „it“ an die Reihe. Hier werden durch Schritt 2 des Algorithmus die potentiellen Referenten „Luke“ und „Han“ entfernt, so dass „space ship“ als einzige Möglichkeit bleibt und daher als Referent ausgewählt wird. Übrigens besteht auch ein Rollenparallelismus, da „space ship“ und „it“ beides Akkusativobjekte sind.

Es erfolgt eine Aktualisierung des Wertes für „space ship“ (aktueller Satz + Objekt + Non-Adv + head noun = 100 + 50 + 50 + 80 = 280):

Referent	Nominalphrasen	Salienzwert
Luke	*{Luke,* he_1*}*	500
Han	*{Han}*	135
space ship	*{space ship,* it_1*}*	420

Schliesslich ist auch der zweite Satz zu Ende bearbeitet, da alle Nominalphrasen abgearbeitet wurden. Alle Salienzwerte werden halbiert und der Algorithmus analysiert den nächsten Satz.

5.3.3 Statistische Algorithmen

Heutzutage liefern statistische Methoden die präzisesten und vollständigsten Ergebnisse [Niy]. Hierfür ist jedoch eine umfangreiche Vorverarbeitung nötig:

Korpus
- In einer repräsentativen Trainingsmenge von Texten, genannt *Korpus*, werden von Hand alle Koreferenzen markiert (man sagt: der Text wird *annotiert*).

Features
- Über diesen Trainingskorpus lässt man eine Reihe von Algorithmen laufen, die Daten über gewisse Merkmale (englisch *features*) sammeln. Die Kunst ist hier, diejenigen Merkmale zu erfassen, die später gute Entscheidungen zwischen möglichen Antezedenzen ermöglichen.

- Beispiele für Merkmale sind:
  - (Durchschnittlicher) Abstand zwischen Anapher und Antezedens
  - Syntaktische Rolle der Anapher
  - Häufigste Anzahl, häufigstes Geschlecht des Antezedens je nach Pronomen
  - Anzahl der Vorkommen einer Entität im Korpus

Am letzten Punkt der Aufzählung, „Anzahl der Vorkommen einer Entität im Korpus“ erkennt man am deutlichsten, dass die Vorverarbeitung eventuell sehr domänenabhängig ist. Je mehr spezielles Wissen über den vorliegenden Korpus man in Merkmalsparameter einbaut, desto bessere Ergebnisse erzielt man typischerweise bei ähnlichen Texten, desto weniger leicht übertragbar auf andere Domänen sind aber auch die aufgrund der Trainingsmenge ermittelten Parameter.

Einige Beispielwahrscheinlichkeiten für einen Trainingskorpus sehen wir in Tabelle 5.4. In dieser Tabelle wird (potentiell) für jede benannte Entität aufgezeichnet, mit welcher Wahrscheinlichkeit die Entität mit welchem Pronomen referenziert wird. Diese Werte helfen nicht nur bei der Pronomenauflösung, sie teilen einem System in vielen Fällen auch das grammatikalische Geschlecht der vorkommenden Entitäten mit.

Den Mechanismus der Pronomenauflösung stellen wir exemplarisch an [Niy] vor, wo ein Bayes-Modell mit bedingten Wahrscheinlichkeiten verwendet wird. Es sei darauf

Word	count	p(he)	p(she)	p(it)
COMPANY	7052	0.0764	0.0060	0.9174
WOMAN	250	0.172	0.708	0.12
PRESIDENT	931	0.8206	0.0139	0.1654
GROUP	1096	0.0602	0.0054	0.9343
MR. REAGAN	534	0.882	0.0037	0.1142
MAN	441	0.8480	0.0385	0.1133
PRESIDENT R.	455	0.8439	0.0043	0.1516

Tabelle 5.4: Beispielwahrscheinlichkeiten, aus einem Trainingskorpus extrahiert. Aus [Niy]

hingewiesen, dass auch andere wahrscheinlichkeitstheoretische Modelle praktischen Einsatz finden.

Soll bei der Analyse eines Textes ein Pronomen p aufgelöst werden, so werden in Frage kommende Antezedensen $a_1, a_2, \ldots, a_k$ untersucht. In eine Funktion F, die eine bedingte Wahrscheinlichkeit berechnet, werden für jedes a_i spezifische Werte als Bedingungen eingesetzt. Die Funktion $F(p)$, die ein Pronomen p auf sein Antezedens abbildet, lautet:

$$F(p) = arg \max_a P(A(p) = a|p, h, \vec{W}, t, l, s_p, \vec{d}, \vec{M})$$

Dabei ist

- $A(p)$ eine Zufallsvariable, die den Referenten des Pronomens p bezeichnet
- a ein vorgeschlagenes Antezedens
- h das Verb, von dem das Pronomen abhängt
- $\vec{W}$ die Liste der Antezedens-Kandidaten
- t der Typ des vorgeschlagenen Ausdrucks a (hier: immer Nominalphrase)
- l der Typ des Verbs
- s_p eine Beschreibung der syntaktischen Struktur, in der sich p befindet
- $\vec{d}$ der Vektor mit den Abständen zwischen p und jedem der a_i
- $\vec{M}$ der Vektor mit den Häufigkeiten, wie oft jedes der a_i genannt wurde

Es wird also dasjenige a als Antezedens gewählt, welches die Wahrscheinlichkeit P maximiert. Für die Aufgabe der Pronomenauflösung erzielten Ge, Hale und Charniak in ihren Messungen 84,2% Treffgenauigkeit (Precision) [Niy].

5.3.4 MUCs - Message Understanding Conferences

In diesem Abschnitt wird eine Reihe von Forschungskonferenzen vorgestellt, der wir wichtige Meilensteine der automatischen Textanalyse verdanken. Die Rede ist von den Message Understanding Conferences oder kurz MUCs. Ihrer gab es sieben Stück, bezeichnet durch MUC-1 (1987) bis MUC-7 (1998) [GS95]. MUC-1 wurde noch vom amerikanischen Naval Ocean Systems Center (NOSC) initiiert, später führt allerdings das

National Institute of Standards and Technology (NIST) die Konferenzen durch. Die Motivation war anfangs, militärische Nachrichten automatisch zu analysieren. Ab MUC-5 verschob sich der Fokus der behandelten Texte in Richtung Wirtschaft und Technik.

Die teilnehmenden Forscher mussten ihre Systeme Aufgaben, sogenannte *Tasks*, erfüllen lassen. Bis zu MUC-5 bestanden diese Tasks im Prinzip aus dem Sammeln von Informationen über Ereignisse und dem Füllen von Schablonen (Templates) über diese Ereignisse. Der Teilnehmer erhielt die Beschreibung einer Klasse von Ereignissen, die sein System im Text identifizieren muss. Das System musste dann das Template mit Informationen über das betreffende Ereignis füllen. Der Wettbewerb bestand aus dieser einzigen, (bei MUC-5: zwei) grossen Task. Wie ein teilnehmendes System die gesuchten Informationen ermittelte, wurde nicht bewertet.

MUC-6 änderte dies grundlegend: Nun gab es auch mehrere kleinere Aufgaben, in denen Systeme antreten konnten, ohne die übrigen Tasks zu bearbeiten. Das Problem der automatischen Textanalyse wurde also in kleinere Teilprobleme aufgeteilt. Unter ihnen waren die Erkennung benannter Entitäten und Koreferenzauflösung. Die Verantwortlichen erkannten hiermit die Berechnung von Koreferenzen als wichtigen Vorverarbeitungsschritt bei der automatischen Textanalyse an. Zusätzlich erfolgte im Rahmen der MUC-6 zum ersten Mal eine breit angelegte Auswertung und Bewertung verschiedener Systeme zur Koreferenzberechnung. Auf Details des verwendeten Bewertungsschemas gehen wir in einem späteren Abschnitt (5.6) ein.

Grundlage waren handannotierte Texte, mit denen der Output der teilnehmenden Systeme verglichen wurde. Die Teilnehmer erhielten als Trainingsdaten SGML-annotierte Texte, ein Beispiel aus [GS95] zeigt Abbildung 5.1.

```
Maybe <COREF ID="136" REF="134">he</CSREF>'ll even leave something
from <COREF ID="138" REF="139"><COREF ID="137" REF="136">his</COREF>
office</COREF> for <CSREF ID="140" REF="91">Mr. Dooner</COREF>.
Perhaps <COREF ID="144">a framed page from the New York Times,
dated Dec. 8, 1987, showing a year-end chart of the stock market
crash earlier that year</COREF>. <COREF ID="141" REF="I37">Mr.
James</COREF> says <COREF ID="142" REF="141">he</COREF> framed <COREF
ID="143" REF="144" STATUS="OPT">it</COREF> and kept <COREF ID="145"
REF="144">it</COREF> by <COREF ID="146" REF="142">his</COREF> desk as
a "personal reminder. It can all be gone like that."
```

Abbildung 5.1: Koreferenz-Annotierung mittels SGML

Wie man schnell sieht, werden erkannte Koreferenzen mit Hilfe des `REF`-Attributs vermerkt. In den Trainingstexten war dieses Attribut gefüllt, im Wettbewerb selbst musste es von den teilnehmenden Systemen gefüllt werden. Die Nominalphrasen waren dabei schon „erkannt", die Koreferenzsysteme mussten sich also um deren Erkennung nicht mehr kümmern.

Bei MUC-7 bildete ein Korpus von etwa 158 000 Artikeln aus der New York Times die Datengrundlage, daraus wurden für die Koreferenz-Aufgabe 30 Trainingsartikel und 30 Wettbewerbsartikel genommen. Die teilnehmenden Systeme lieferten bis zu 63% Recall und bis zu 72% Precision bei MUC-6. Bei MUC-7 reichte das F-Measure, dies ist das harmonische Mittel zwischen Recall und Precision, bis 62%. Diese Zahlen sind niedriger als die Zahlen, die üblicherweise bei der Vorstellung von Systemen angegeben werden. Das hängt damit zusammen, dass es bei einer MUC die eine oder andere „böse Überraschung" geben kann: Waren im Trainingsset noch fast ausschliesslich

Artikel über Unternehmensfusionen zu finden, kann es plötzlich sein, dass im Wettbewerbsdurchlauf Flugzeugabstürze den Mittelpunkt des Interesses bilden. Solche Wechsel der Domäne wirken sich mitunter drastisch auf die Ergebnisse von Systemen aus und werden selten vom Forscherteam selbst vorgenommen.

5.4 Koreferenzberechnung in GATE

Die *General Architecture for Text Engineering* [Cun02, CMBT02] bietet mehrere Module zur Koreferenzauflösung (*coreference resolution modules*), darunter den *OrthoMatcher* zur Koreferenzherstellung zwischen benannten Entitäten (Personen, Organisationen), den (schlecht bis gar nicht dokumentierten) *Nominal Coreferencer* und den *Pronominal Coreferencer* zur Pronomenauflösung.

Der OrthoMatcher vergleicht Namen auf Ähnlichkeit (Stringvergleich) und nimmt zusätzlich eine Tabelle mit Alias-Namen zur Hilfe, um zum Beispiel die Identität von „IBM“ und „Big Blue“ erkennen zu können. Wir betrachten im folgenden das Pronominal Coreference Module.

5.4.1 Pronomen-Koreferenzmodul

GATE ist modulartig aufgebaut. Das Pronomen-Koreferenzmodul (Pronominal Coreference Module) ist solch ein Modul und damit eine eigenständige Berechnungskomponente. Es erfordert die vorherige Ausführung aller folgender Module:

- *Tokenizer* (Identifizierung von Wortgrenzen)
- *Sentence Splitter* (Identifizierung von Satzgrenzen)
- *Named Entity Transducer* (Erkennen von Personen, Orten, Organisationen et cetera)
- *OrthoMatcher* (Erkennen von ähnlichen Namen, welche die gleiche Entität bezeichnen)

Diese Ablaufstruktur ist in Abbildung 5.2 veranschaulicht. Dabei ist zu ergänzen, dass dem eigentlichen *Pronominal Resolution Module* zwei Untermodule (*submodules*) vorgeschaltet sind, die seine Arbeit unterstützen:

- Das *Quoted Speech Submodule*, welches wörtliche Rede im Text identifiziert.
- Das *Pleonastic It Submodule*, welches pleonastische Vorkommen des Pronomens „it“ findet (pleonastisch = ohne Referent, „*es* ist warm“).

Die darauf folgende Pronomenauflösung arbeitet in drei Schritten:

Vorverarbeitung Zur Vorverarbietung gehören diese Punkte:

- *Initialisierung von Datenstrukturen:* Für jeden Satz wird je eine Liste für die darin vorkommenden Personen, Orte und Organisationen erzeugt.
- *Geschlecht von Personen:* Der Named Entity Transducer und der OrthoMatcher versuchen unter anderem, das Geschlecht von Personen zu ermitteln und generieren entsprechende Annotationen. Diese Annotationen werden benutzt.

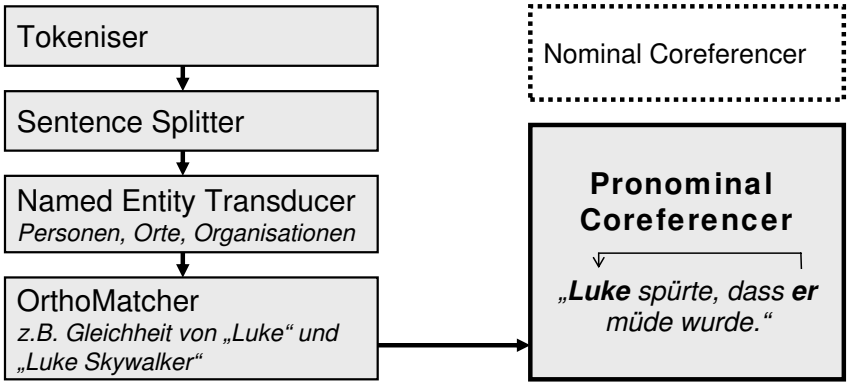


Abbildung 5.2: Verarbeitungspipeline in GATE

- *Pleonastisches „it“:* Pleonastische Vorkommen von „it“ werden in einer separaten Liste abgelegt.
- *Wörtliche Rede:* Aufbau spezieller Datenstrukturen beim Vorkommen von wörtlicher Rede.

Pronomenauflösung Für die Pronomenauflösung werden zunächst alle Pronomen – sowohl Possessivpronomen („my“, „your“ et cetera) als auch Personal- und Reflexivpronomen („she“, „herself“) – in einer Liste abgelegt, sortiert nach Erscheinen im Text. Der Text wird dann Satz für Satz durchgegangen. Für jedes Pronomen im aktuellen Satz werden vier Schritte durchgeführt:

a) Wenn es sich um „it“ handelt, wird geprüft, ob es ein pleonastisches Vorkommen ist. Ist dies der Fall, so wird kein Versuch der Auflösung unternommen.

b) Es wird ein angemessener *Kontext* für das Pronomen bestimmt, das heisst ein Bereich, der nach möglichen Referenten durchsucht wird. Der Kontext enthält immer den Satz, in dem das Pronomen vorkommt, plus 0 oder mehr vorherige Sätze. Mehr als 3 vorherige Sätze werden in der Regel nicht betrachtet.

c) Aus dem Kontext wird eine Menge von Antezedens-Kandidaten ermittelt. Die Kandidaten müssen in Anzahl und Geschlecht zu dem Pronomen passen, oder Anzahl und Geschlecht müssen unbekannt sein.

d) Aus dieser Kandidatenmenge wird ein Kandidat als Antezedens ausgewählt. Die Auswahl erfolgt gemäss für das Pronomen spezifischen Kriterien.

Die pronomenspezifischen Kriterien von Schritt 4 besagen zum Beispiel bei Pronomen in der 3. Person („he“, „she“), dass der Kandidat, der am nächsten beim Pronomen und *vor* dem Pronomen steht, ausgewählt wird. Dieses Vorgehen spiegelt die Berücksichtigung des *recency factor* wieder, also die Gewichtung nach Aktualität im Text. Ausserdem sieht man, dass Anaphern gegenüber Kataphern bevorzugt werden. Pronomen in der 1. Person („I“, „me“) treten meist in Verbindung mit wörtlicher Rede auf, und die meisten dieser Pronomen sind kataphorisch. Hier kommen entsprechende, besondere Regeln zur Anwendung.

Koreferenzkettenbildung Nachdem alle Pronomen behandelt wurden, bleibt noch die Bildung von Koreferenzketten. Dieser Schritt des Algorithmus durchläuft alle Anapher-Antezedens-Paare, welche die Pronomenauflösung produziert hat. Für jedes solche Paar werden das Antezedens sowie, falls welche exisiteren, alle seine orthographischen Treffer (*OrthoMatcher*) genommen und die Anapher, also das Pronomen hinzugefügt. Das Ergebnis ist eine Koreferenzkette.

5.5 Fuzzy-Koreferenzberechnung

Zur Motivation betrachten wir folgenden Textausschnitt:

- Luke betrat das Zimmer. Der junge Jedi bemerkte nichts.

Koreferieren „Luke“ und „der junge Jedi“? Selbst für einen Menschen ist dies ohne weiteres Wissen (Ist Luke ein Jedi? Ist Luke jung?) und ohne Kontextwissen (Ist Luke der einzige junge Jedi, der im Augenblick präsent ist?) nicht mit Sicherheit entscheidbar. Dies gilt natürlich auch für ein automatisches Analysesystem. Viele Systeme stehen nun vor der Entscheidung, die Koreferenz entweder anzunehmen oder auszuschliessen. Wird sie angenommen, so ist sie eine Koreferenz wie jede andere auch, das heisst es wird nicht modelliert, dass die Entscheidung mit einer beträchtlichen Menge an Unsicherheit behaftet ist. Nachfolgende Verarbeitungsschritte sind eventuell auf sehr sichere Koreferenzen angewiesen.

Wird die Koreferenz andererseits strikt ausgeschlossen, verlieren wir eventuell ein gültiges Element einer Koreferenzkette und unsere Vollständigkeit verringert sich.

Der Fuzzy-Ansatz Eine elegante und intuitive Möglichkeit, dieses Dilemma zu umgehen, liefert René Witte durch das Konzept der Fuzzy-Koreferenzketten (siehe [Wit02, WB03]). Dieses Konzept wurde in dem System *Fuzzy-ERS* (*Experimental Resolution System*) umgesetzt. Fuzzy-ERS orientiert sich an ERS [Ber97], einem System zur Bestimmung von Nominalphrasen-Koreferenzen, welches unter der Leitung von Sabine Bergler von der Concordia University in Montréal entwickelt wurde.

Der Grundgedanke des Fuzzy-Ansatzes ist, dass eine Nominalphrase *mit einer gewissen Sicherheit* Mitglied einer Koreferenzkette ist. Diese Sicherheit liegt zwischen 0 (Zugehörigkeit ausgeschlossen) und 1 (NP ist sicher Mitglied der Kette), einschliesslich.

Definition 5.5.1 (Fuzzy-Menge) *Eine Fuzzy-Menge μ von Ω ist eine Funktion von der Referenzmenge Ω in das Einheitsintervall:*

$$\mu : \Omega \longrightarrow [0, 1]$$

Definition 5.5.2 (Fuzzy-Koreferenzkette) *Eine Fuzzy-Koreferenzkette C ist eine Fuzzy-Menge μ_C, wobei die Referenzmenge die Menge aller Nominalphrasen im betreffenden Text ist:*

$$\mu_C : \{np_1, np_2, \ldots, np_n\} \longrightarrow [0, 1]$$

Die Interpretation von Definition 5.5.2 ist die, dass eine Fuzzy-Kette C für jede NP np_i im Text einen *Zugehörigkeitsgrad* $\mu_C(np_i)$ hält. Es ist nun also möglich, die Unsicherheit von Koreferenzen explizit zu machen. So wird die Vernichtung wichtiger Information vermieden und nachfolgende Schritte können zwischen sicheren und eher

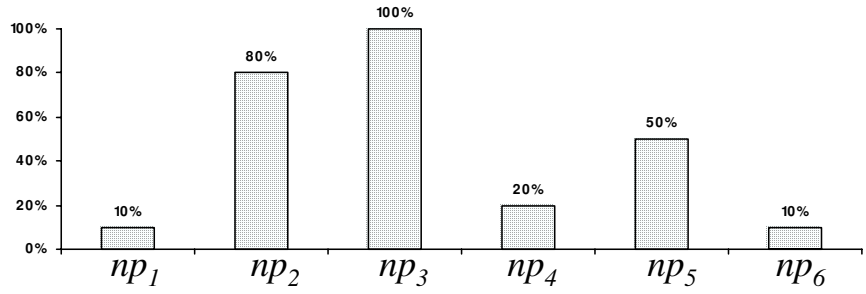


Abbildung 5.3: Fuzzy-Koreferenzkette C

riskanten Koreferenzen unterscheiden. Eine grafische Veranschaulichung einer Fuzzy-Koreferenzkette sehen wir in Abbildung 5.3.

Im folgenden werden die einzelnen Schritte zur Erstellung von Fuzzy-Koreferenzketten erläutert. Der erste dieser Schritte ist die Anwendung verschiedener Fuzzy-Heuristiken, welche Einzelergebnisse für jeweils zwei Nominalphrasen liefern. Diese Einzelergebnisse werden anschliessend zu gesamten Ketten kombiniert und diese gegebenenfalls defuzzifiziert für nachfolgende Anwendungen.

5.5.1 Fuzzy-Heuristiken

Da kein bekanntes Verfahren (mitunter nicht einmal das menschliche Sprachvermögen) Koreferenz zwischen zwei Nominalphrasen völlig sicher feststellen oder ausschliessen kann, spricht man von *Heuristiken*, die Nominalphrasen auf Koreferenz untersuchen. Scharfe Heuristiken nehmen dabei ein Paar von Nominalphrasen und liefern als Ergebnis einen Wert *true*, falls die Nominalphrasen nach Ansicht der Heuristik koreferieren, und *false* sonst. *NP* bezeichne im folgenden die Menge aller Nominalphrasen im Text:

$$H : \mathrm{NP} \times \mathrm{NP} \rightarrow \{\mathrm{true}, \mathrm{false}\}$$

Das Problem hierbei wurde bereits erwähnt: Unsichere Koreferenzen muss man entweder zulassen (risikoreich) oder ausschliessen (Verlust von Vollständigkeit). *Fuzzy*-Heuristiken erlauben es, die Unsicherheit der Koreferenz explizit zu benennen. Das Ergebnis einer Fuzzy-Heuristik ist der Sicherheitsgrad für die Koreferenz der beiden Argumente:

$$H : \mathrm{NP} \times \mathrm{NP} \rightarrow [0, 1]$$

Je nach Heuristik wird der Sicherheitsgrad mittels linguistischer Analysen „stufenlos“ berechnet oder einer von wenigen „diskreten“ Graden vergeben (zum Beispiel „sicher“, „wahrscheinlich“, „möglich“, „unwahrscheinlich“, „nicht“ für 1.0, 0.75, 0.5, 0.25, 0.0).

Die meisten Heuristiken beinhalten Ausnahmen von der üblichen Regel. So koreferieren buchstabengleiche Nominalphrasen, vor allem Eigennamen, zwar oft, bei Pronomen wäre eine entsprechende Regel jedoch äusserst kontraproduktiv. Damit einzelne

Heuristiken nicht mit Ausnahmen überfrachtet werden, wurden die Ausnahmeregelungen in Fuzzy-ERS als eigene Heuristiken realisiert. Zu vielen Heuristiken gibt es also auch die „Anti-Heuristik“. Betrachten wir einige Beispiele für Fuzzy-Heuristiken:

Fuzzy-CommonHead-Heuristik Die CommonHead-Heuristik zielt, wie der Name nahelegt, darauf ab, bei zwei Nominalphrasen ein gemeinsames head noun (Erinnerung: das Substantiv, das Geschlecht und Anzahl der NP festlegt) zu identifizieren. Als koreferierend gelten zum Beispiel: „the King's *castle*“, „the *castle*“, „the *castle* of Camelot“.

Voraussetzung zur Anwendung dieser Heuristik ist eine Analyse der Nominalphrasen eines Satzes. Diese liefert das head noun. Da allerdings eine Übereinstimmung im head noun nicht immer sicher eine Koreferenz impliziert, liefert die CommonHead-Heuristik einen entsprechenden Sicherheitswert als Gesamtergebnis, also als Sicherheit der Koreferenz zwischen den beiden Argumenten.

Fuzzy-Anti-CommonHead Koreferenz zwischen zwei Nominalphrasen wird ausgeschlossen, wenn deren Head einen Betrag kennzeichnet. Bestimmt wird dies durch eine Liste von Substantiven wie „million“ oder „shares“.

Fuzzy-Synonym/Hypernym-Heuristik Synonyme sind unterschiedliche Wörter mit gleicher Bedeutung („immer“ ↔ „stets“), ein Hypernym ist ein Oberbegriff eines anderen („Transportmittel“ ist ein Hypernym von „Raumschiff“). Die Fuzzy-Synonym/Hypernym-Heuristik wurde entwickelt, um Koreferenzen zu identifizieren, die auf Synonym- oder Hypernym-Beziehungen basieren. Rein syntaktisch ist das nicht machbar: Man benötigt ein Lexikon, das die semantischen Beziehungen zwischen unterschiedlich aussehenden Wörtern herstellt. Ein weit verbreitetes Lexikon für diesen Zweck ist *WordNet* [Fel98]. Abbildung 5.4 zeigt die Synonyme und Hypernyme, die WordNet für den Begriff „workforce“ liefert.

```
Synonyms/Hypernyms (Ordered by Frequency) of noun workforce

1 sense of workforce

Sense 1
work force, workforce, manpower, hands, men
       => force, personnel
           => organization, organisation
               => social group
                   => group, grouping
```

Abbildung 5.4: Ausgabebeispiel von WordNet: Synonyme / Hypernyme für die Nominalphrase „workforce“

In WordNet sind nur einzelne Nominalphrasen abrufbar, also etwa nur „player“ und nicht „world class tennis player“. Daher beschränkt man sich bei der Heuristik auf den Vergleich des Head der beiden Nominalphrasen. Koreferenz wird als *sicher* angenommen, wenn der Head der ersten Nominalphrase ein Synonym des Heads der zweiten Nominalphrase ist.

Schlägt die Synonymprüfung fehl, wird untersucht, ob eine Hypernym-Beziehung vorliegt. Bei Betrachten der WordNet-Ausgabe in Abbildung 5.4 leuchtet ein, dass Koreferenz nicht in jedem dieser Fälle angenommen werden sollte. Zum Beispiel ist in

> "The *workforce* protested against the closing of the plant, which a *group* of executives was aiming at."

die Nominalphrase „group" gemäss WordNet ein Hypernym für „workforce". Der Beispielsatz macht aber deutlich, dass Koreferenz nicht ohne weiteres angenommen werden sollte. Grund ist der grosse Unterschied in der Abstraktheit der beiden Begriffe: Während „workforce" einen relativ klaren, kleinen Bereich eingrenzt, kann „group" alles mögliche sein. Offensichtlich muss eine Heuristik, die WordNet benutzt, also der verschiedenen Abstraktionsebenen in der Ausgabe gewahr sein. Fuzzy-ERS benutzt eine einstellbare maximale Distanz d. Die Distanz zwischen zwei Nominalphrasen np_j und np_k ist im Moment als die Länge des Pfades zwischen np_j und np_k definiert. Liegen die NPs auf der selben Ebene in der WordNet-Ausgabe, so ist die Distanz 0 und die Begriffe werden als synonym angesehen: Ein Sicherheitsgrad von 1.0 wird angesetzt. Sind np_j und np_k d Ebenen voneinander entfernt, so wird der Sicherheitsgrad 0.0 angesetzt. Dazwischen nimmt der vergebene Sicherheitsgrad linear ab beziehungsweise zu.

Fuzzy-Pronomen-Heuristik Da Pronomen knapp 20% aller Koreferenzen ausmachen [Wit02], ist eine gute Pronomen-Heuristik besonders wichtig. Die Fuzzy-Pronomen-Heuristik dient der Identifizierung von Koreferenz zwischen Pronomen und nicht-pronominalen Nominalphrasen. Dabei werden mehrere Schritte abgearbeitet:

- Mit Hilfe eines Lexikons wird festgestellt, ob bei der aktuellen Nominalphrase ein Pronomen vorliegt. Ist dies nicht der Fall, so bricht die Heuristik ab und liefert den Sicherheitsgrad *nicht* zurück, so dass sie nicht zum Tragen kommt.

Pleonastisches Pronomen

- Ist das Pronomen *pleonastisch*, das heisst ohne Referent („*Es* regnet."), so bricht die Heuristik ebenfalls ab.
- Ist das Pronomen ein *Possessivpronomen* („his", „their"), so wird das entsprechende nicht-Possessivpronomen gesucht („he", „they") und die Heuristik mit diesem rekursiv aufgerufen.
- Schliesslich wird das Pronomen auf Koreferenz mit der anderen übergebenen Nominalphrase überprüft (welche kein Pronomen sein darf).

Einige dieser Schritte, insbesondere der letzte, sind sehr komplex. Hier wird zunächst geprüft, ob Pronomen und nichtpronominale NP in Anzahl und Geschlecht übereinstimmen. Von letzterer wird dabei der Head verwendet, da dieser allein Anzahl und Geschlecht bestimmt. Zu beachten ist, dass es im Englischen sogenannte Gruppen-Nominalphrasen gibt, die im Singular stehend durch ein Pronomen im Plural referenziert werden können: „The workforce protested because *they* didn't want to live with these conditions." „Workforce" steht im Singular, koreferiert aber mit „they", das im Plural steht. Um derartige Koreferenzen aufzudecken, benötigt man ein Wörterbuch, das die entsprechenden Gruppen-Nominalphrasen enthält.

Falls Übereinstimmung bezüglich Geschlecht und Anzahl besteht, wird die Distanz zwischen Pronomen und anderer Nominalphrase als Grundlage für die Berechnung des Sicherheitsgrades gewählt. Auch dann, wenn die nichtpronominale NP nicht im Wörterbuch gefunden wurde, geht man immerhin noch von einer *möglichen* Koreferenz aus. Es ist leicht vorstellbar und empirisch erwiesen, dass ein Pronomen und eine nichtpronominale NP umso weniger wahrscheinlich koreferieren, je weiter sie

voneinander entfernt stehen. Die Distanz zwischen beiden Ausdrücken wird dabei in Sätzen gemessen. Stehen beide im gleichen Satz (Distanz $\delta = 0$), so geht man von einer Sicherheit von 1,0 aus. Diese Sicherheit nimmt linear ab bis zu einer einstellbaren maximalen Distanz δ_{max} (in [Wit02] ist von $\delta_{max} = 10$ die Rede), bei der sie 0,0 beträgt.

Weitere Fuzzy-Heuristiken

Fuzzy-Substring-Heuristik Die Fuzzy-Substring-Heuristik nimmt zwei Nominalphrasen np_i und np_j entgegen und prüft, ob np_i in np_j enthalten ist. Als Grad der Koreferenz wird zurückgegeben:

- 1.0, falls np_i und np_j identisch sind („Luke Skywalker“, „Luke Skywalker“)
- 0.0, falls np_i nicht in np_j enthalten ist („Luke“, „Han Solo“)
- den Quotienten
$$\frac{|np_i|}{|np_j|}$$
sonst. Bei $np_i =$ „Luke“ und $np_j =$ „Luke Skywalker“ ergibt dies circa 0.29, bei Vertauschen der Argumente 0.0. Die Fuzzy-Substring-Heuristik ist also nicht symmetrisch.

Fuzzy-Appositions-Heuristik Die Fuzzy-Appositions-Heuristik erkennt Appositionen („Luke, *der junge Jedi*, ...“). Sie arbeitet sehr zuverlässig, jedoch machen Appositionen nur circa 3% aller Nominalphrasen aus.

Fuzzy-Akronym-Heuristik Die Fuzzy-Akronym-Heuristik markiert Nominalphrasen als koreferierend, bei denen die eine NP eine Abkürzung der anderen darstellt (zum Beispiel „General Motors Corporation“ und „GMC“).

5.5.2 Kettenbildung

Die Anwendung der verschiedenen Heuristiken liefert Sicherheitsgrade für die Koreferenz zwischen Nominalphrasenpaaren. Man könnte sie Fuzzy-Koreferenzpaare nennen. Unser Ziel ist jedoch die Konstruktion von ganzen Fuzzy-Koreferenzketten.

Bevor wir den Weg zu diesem Ziel betrachten, möchten wir hier noch zwei grundlegende Operationen auf Fuzzy-Mengen einführen, die wir benötigen werden.

Definition 5.5.3 (Schnitt) *Der Schnitt zweier Fuzzy-Mengen A und B, geschrieben $A \cap B$, ist definiert als*

$$A \cap B = \mu_{A \cap B} := \min\{\mu_A, \mu_B\}$$

Definition 5.5.4 (Vereinigung) *Die Vereinigung zweier Fuzzy-Mengen A und B, geschrieben $A \cup B$, ist definiert als*

$$A \cup B = \mu_{A \cup B} := \max\{\mu_A, \mu_B\}$$

Im Kontext der Fuzzy-Koreferenzketten bedeutet das: Ist eine Nominalphrase np_i in der Fuzzy-Koreferenzkette c_1 mit dem Zugehörigkeitsgrad $\mu_{c_1}(np_i)$ vertreten und in der Kette c_2 mit dem Zugehörigkeitsgrad $\mu_{c_2}(np_i)$, so ist sie

- im Schnitt $c_1 \cap c_2$ mit dem Grad $\min\{\mu_{c_1}(np_i), \mu_{c_2}(np_i)\}$
- in der Vereinigung $c_1 \cup c_2$ mit dem Grad $\max\{\mu_{c_1}(np_i), \mu_{c_2}(np_i)\}$

vertreten.

Mit den Operationen Fuzzy-Schnitt und Fuzzy-Vereinigung ausgerüstet machen wir uns nun an die Konstruktion der gewünschten Koreferenzketten. Zunächst wird für jede der Nominalphrasen im Text, deren Anzahl mit n bezeichnet sei, eine eigene Kette aufgebaut, und zwar durch die Vereinigung aller Heuristik-Ergebnisse, welche die jeweilige Nominalphrase betreffen. Die angewendeten Heuristiken seien $H_1, H_2, \cdots, H_m$. Eine Kette für die Nominalphrase np_i wird dann aufgebaut durch

$c_i \leftarrow \emptyset$

für alle $np_k \neq np_i$

$$c_i \leftarrow c_i \ \cup \ H_1(np_i, np_k) \ \cup \ H_2(np_i, np_k) \ \cup \ \ldots \ \cup \ H_m(np_i, np_k)$$

Für den Aufbau der Kette für die Nominalphrase np_i werden also alle Heuristiken mit Parameter np_i und allen anderen Nominalphrasen betrachtet und die Ergebnisse vereinigt.

Beispiel Sagen wir, wir bauen eine Kette für np_1 auf. Nach Ansicht von Heuristik H_1 koreferieren np_1 und np_2 nur unwahrscheinlich, sagen wir zu 10%, während Heuristik H_4 findet, dass die beiden NPs durchaus koreferieren könnten, mit 80% Sicherheit. Werden die Ergebnisse der beiden Heuristiken verschmolzen, so hat np_2 im Resultat den Zugehörigkeitsgrad 80%, veranschaulicht in Abbildung 5.5. Die Heuristik-Ergebnisse sind dabei durch die Fuzzy-Mengen $\mu^{H_l}_{np_i,np_k}$ repräsentiert.

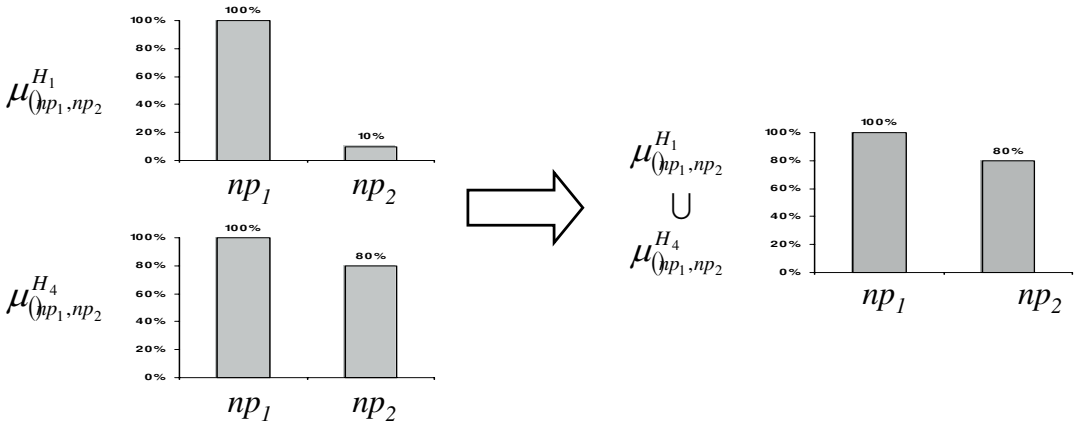


Abbildung 5.5: Vereinigung zweier Heuristikergebnisse

Da jede Nominalphrase sicher mit sich selbst koreferiert und wir in 5.5 eine Kette „aus der Sicht“ von np_1 anstreben, hat np_1 in allen betrachteten Mengen den Zugehörigkeitsgrad $1,0$.

Die beschriebene Prozedur wird also für jede Nominalphrase im Text durchgeführt, wodurch wir n Fuzzy-Koreferenzketten $c_1, \ldots, c_n$ erhalten.

5.5.3 Kettenverschmelzung

Dies ist jedoch noch nicht das Ergebnis, auf das wir abzielen, da die *Transitivität* der Koreferenzrelation unter Umständen noch nicht richtig umgesetzt ist. Betrachten wir Abbildung 5.6: Hier koreferieren aus der Sicht von np_1 die Nominalphrasen np_1 und np_2 mit 80% Sicherheit, aus der Sicht von np_2 koreferieren np_2 und np_4 mit 80% Sicherheit. Gemäss der Transitivität der Koreferenz sollten demnach auch np_1 und np_4 mit hoher Sicherheit koreferieren, wie in dem Diagramm rechts angedeutet.

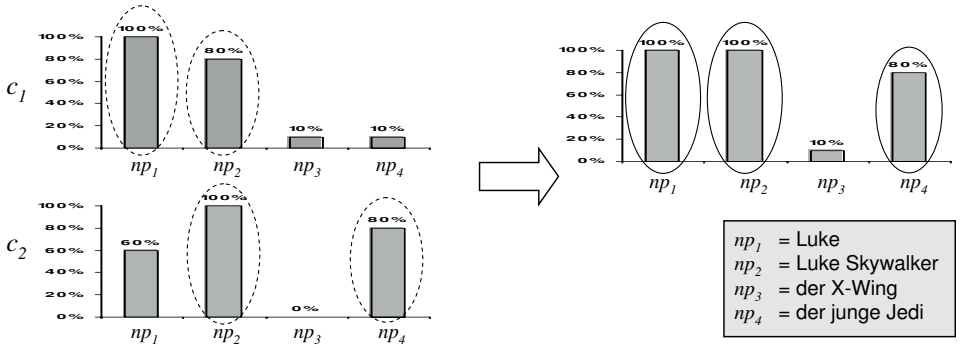


Abbildung 5.6: Verschmelzen zweier Ketten führt zu Realisierung der Transitivität (np_1 und np_4)

Hierfür werden „angemessen ähnliche“ Ketten c_j und c_k miteinander *verschmolzen* (englisch *to merge*), was bedeutet, dass die beiden Ketten durch ihre Fuzzy-Vereinigung $c_j \cup c_k$ ersetzt werden.

Was bedeutet „angemessen ähnliche“ Ketten? Betrachten wir Abbildung 5.7. Die beiden dort abgebildeten Ketten sollten auf keinen Fall verschmolzen werden, da sie „sehr wenig gemeinsam haben“ im Sinne von: Keine Nominalphrase hat zu *beiden Ketten* einen hohen Zugehörigkeitsgrad. Das Ergebnis einer Verschmelzung wäre eine Kette, in der alle sechs betrachteten Nominalphrasen mit hoher Sicherheit koreferieren, was nicht der Wirklichkeit entspräche und diese Kette unbrauchbar machen würde.

Deshalb fordern wir für zwei Ketten c_1 und c_2, die verschmolzen werden sollen, einen (einstellbaren) *Konsistenzgrad* γ, zum Beispiel $0,8$. Das bedeutet, dass mindestens eine Nominalphrase zu $c_1 \cap c_2$ einen Zugehörigkeitsgrad von γ oder höher haben muss. Ist dies der Fall, so werden beide Ketten verschmolzen, also die Vereinigung $c_1 \cup c_2$ gebildet, welche die beiden ursprünglichen Ketten im Endergebnis ersetzt: Konsistenzgrad

$$\max_{np_k}((\mu_{c_1} \cap \mu_{c_2})(np_k)) \geq \gamma \quad \Rightarrow \quad \mu_{c_{(1,2)}} := \mu_{c_1} \cup \mu_{c_2}$$

Die Auswirkung unterschiedlicher geforderter Konsistenzgrade veranschaulicht Abbildung 5.8. Bei $\gamma = 0.7$ wird verschmolzen, bei $\gamma = 0.8$ bleiben die beiden einzelnen Ketten für sich.

Ketten werden so lange verschmolzen, bis keine konsistente Verschmelzung mehr möglich ist. Das Ergebnis sind die gewünschten Fuzzy-Koreferenzketten. Der genaue Algorithmus ist in [Wit02] in Abschnitt 18.3.2.3 angegeben.

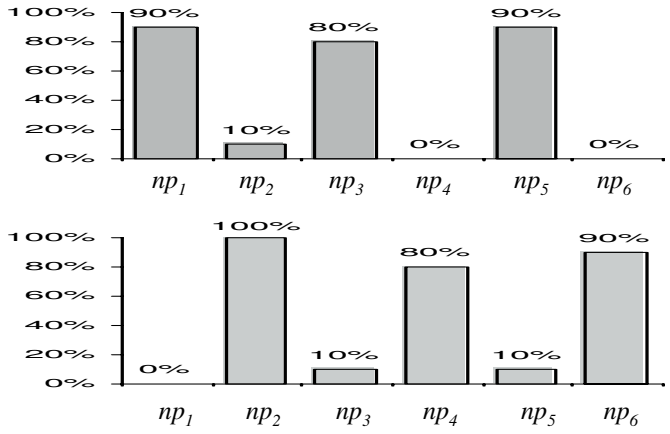


Abbildung 5.7: Ketten, die nicht verschmolzen werden sollten

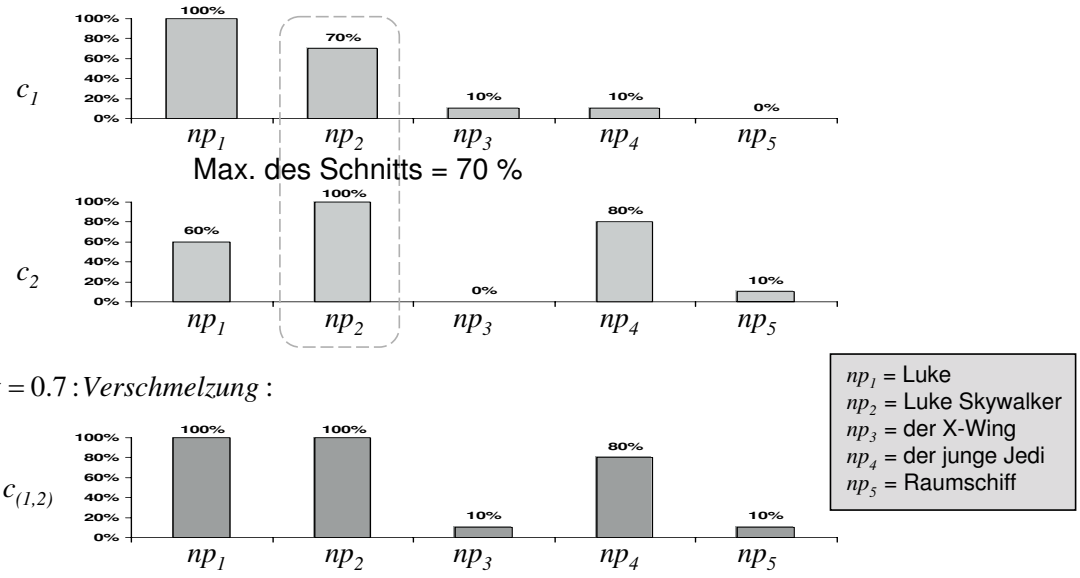


Abbildung 5.8: Auswirkung unterschiedlicher Konsistenzgrade: Bei $\gamma = 0.7$ findet eine Verschmelzung statt, bei $\gamma = 0.8$ nicht mehr.

5.5.4 Ketten-Defuzzifizierung

Falls ein folgender Verarbeitungsschritt keine Fuzzy-Ketten kennt und daher scharfe Ketten verlangt, müssen die Fuzzy-Ketten *defuzzifiziert* werden. Bei diesem Schritt wird ein Mindestzugehörigkeitsgrad γ angewendet: Die scharfe Referenzkette enthält genau diejenigen Nominalphrasen, die in der Fuzzy-Kette mit einem Zugehörigkeitsgrad von mindestens γ vertreten sind. Eine Veranschaulichung bietet Abbildung 5.9.

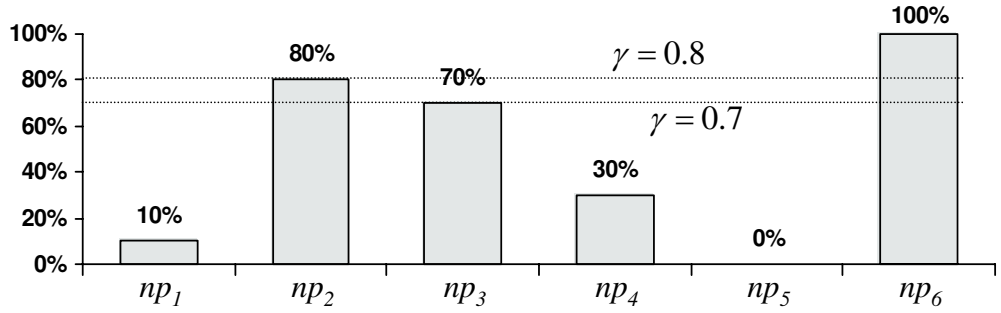


Abbildung 5.9: Defuzzyfizierung einer Fuzzy-Kette: Bei $\gamma = 0.7$ erhält man die scharfe Ergebnismenge $\{np_2, np_3, np_6\}$, bei $\gamma = 0.8$ nur $\{np_2, np_6\}$.

5.6 Bewertung berechneter Koreferenzketten

Motivation Wendet man mehrere Algorithmen zur Koreferenzbestimmung auf den selben Text an, so ist man an qualitativen Unterschieden zwischen den verschiedenen Ergebnissen interessiert. Wir betrachten folgende Beispieleingabe, auf die zwei Algorithmen angewendet werden (Algorithmus 1 und Algorithmus 2):

> „Luke Skywalker stieg in den X-Wing. Luke hatte das Raumschiff repariert, aber er hatte lange gebraucht."

Als Ausgabe werden folgende Koreferenzketten geliefert:

- Algorithmus 1: <Luke Skywalker – X-Wing – Luke> <er> <Raumschiff>
- Algorithmus 2: <Luke Skywalker – Luke> <X-Wing – Raumschiff – er>

Beide Ergebnisse sind weder fehlerfrei noch vollständig. Welches Ergebnis ist nun besser? Was genau bedeutet „besser" in Zusammenhang mit der Koreferenzkettenberechnung?

5.6.1 Precision und Recall

Einen Algorithmus zur Koreferenzauflösung kann man als *Suche* betrachten: Die Koreferenzen existieren im Text, der Algorithmus muss sie finden. Allgemein für Suchergebnisse lassen sich folgende zwei Gütekriterien definieren:

Definition 5.6.1 (Precision) *Eine Suche liefere eine Teilmenge $O \subseteq \Omega$ von Objekten einer Grundmenge Ω. Die gemäss der Suchanfrage korrekten (erwünschten, relevanten) Objekte seien mit $C \subseteq \Omega$ bezeichnet. Dann heisst der Quotient*

$$P = \frac{|O \cap C|}{|O|}$$

die Precision (Genauigkeit) des Suchergebnisses.

Definition 5.6.2 (Recall) *O, Ω und C seien wie in Definition 5.6.1 vereinbart. Dann heisst der Quotient*

$$R = \frac{|O \cap C|}{|C|}$$

der Recall (Vollständigkeit) des Suchergebnisses.

Etwas salopp in Worte gefasst: Im Zähler steht das korrekt Gefundene, im Nenner das gesamte Gefundene (Precision) beziehungsweise das gesamte Korrekte (Recall). Die Precision einer Suche S sagt also aus, ob S überwiegend nützliche Objekte geliefert (hohe Precision) hat oder ob sehr viel Irrelevantes dabei ist (niedrige Precision). Der Recall zeigt an, wie viel von dem, das S hätte finden müssen, sie auch gefunden hat.

Precision und Recall bei Koreferenzketten Betrachten wir nun das Konzept von Precision und Recall im Zusammenhang mit Koreferenzketten. Hier stellt ein Modellierungsproblem. Werfen wir noch einmal einen Blick auf die folgenden wirklichen beziehungsweise von einem Algorithmus ermittelten Koreferenzketten:

Wirklich:	<Luke Skywalker – Luke – er>	<X-Wing – Raumschiff>
Algorithmus:	<Luke Skywalker – Luke>	<X-Wing – Raumschiff – er>

Man beobachtet, dass keine der ermittelten Koreferenzketten mit einer korrekten Koreferenzkette übereinstimmt. Eine naive Anwendung der Precision- und Recall-Formeln würde also jeweils 0 ergeben (weil der Zähler 0 ist), obwohl der Algorithmus offensichtlich Koreferenzen erkannt hat. Wir halten also fest, dass es zu irreführenden Werten führt, wenn man ganze Koreferenzketten betrachtet und die Berechnungen darauf aufbaut.

5.6.2 Die Vilain-Metrik

Diese Problematik war selbstverständlich auch den Organisatoren von MUC-6 klar, dennoch brauchte man ein Bewertungsschema, um die Ergebnisse von teilnehmenden Systemen zu evaluieren und zu vergleichen. Marc Vilain und andere entwarfen dieses Schema, vorgestellt in [VBA+].

Hier werden Koreferenzketten als Äquivalenzklassen von Nominalphrasen betrachtet. Dies ist intuitiv einleuchtend, wenn man sich in Erinnerung ruft, dass Koreferenz eine Äquivalenzrelation auf der Menge der Nominalphrasen des zugrundeliegenden Texts definiert (siehe Abschnitt 5.2: Begriffe). Um eine Äquivalenzklasse, also eine Koreferenzkette, der Mächtigkeit n herzustellen, genügt es dank der Transitivität der Koreferenzrelation, $n-1$ verschiedene Einzelverbindungen, sogenannte *Links*, zwischen jeweils zwei Nominalphrasen aufzubauen. Zugrundeliegende Idee hierbei ist die des minimalen Spannbaums bei Graphen. Illustriert wird das Modell durch Abbildung 5.10, wo eine Äquivalenzklasse von 5 Elementen durch 4 Koreferenzlinks erstellt wird.

Links

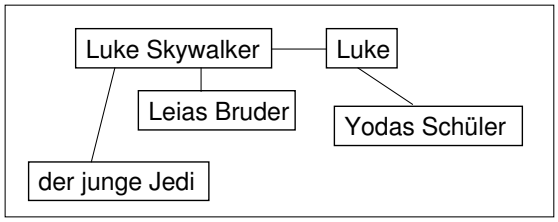


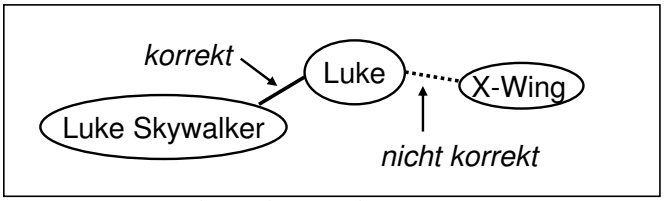
Abbildung 5.10: Eine Äquivalenzklasse mit 5 Elementen kann durch 4 Links hergestellt werden

Precision und Recall können nun mit Hilfe dieser Links modelliert werden. Für die Precision-Berechnung betrachten wir zunächst eine einzelne *gefundene* Koreferenzkette c. Der Precision-Wert P_c für diese Kette berechnet sich durch

Precision-Berechnung

$$P_c = \frac{|\text{korrekte Links}|}{|\text{alle Links}|} = \frac{|\text{korrekte Links}|}{|c| - 1}.$$

Eine Veranschaulichung zeigt Abbildung 5.11 mit einer gefundenen Koreferenzkette mit drei Elementen, also 2 Links. Ein Link davon ist korrekt, der andere falsch, was zu einem Precision-Wert von 50% führt.



$$P = \frac{|\text{korrekte Links}|}{|\text{alle Links}|} = \frac{1}{2} = 50\%$$

Abbildung 5.11: Berechnung der Precision P für eine einzelne gefundene Koreferenzkette

Will man nun die Precision für ein gesamtes Ergebnis berechnen, so summiert man im Zähler und Nenner über alle gefundenen Koreferenzketten. Diese Menge sei mit C_r (r wie *response* entsprechend der Terminologie des Vilain-Teams) bezeichnet.

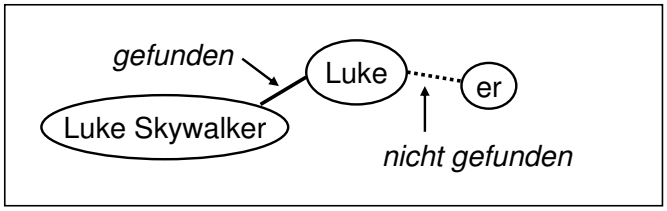
$$P = \frac{\sum_{c \in C_r} |\text{korrekte Links in c}|}{\sum_{c \in C_r} (|c| - 1)}$$

Wollen wir den Recall berechnen, so dreht sich die Sichtweise um: Dieses Mal betrachten wir eine *tatsächliche* Koreferenzkette c. Der Recall R_c bezüglich dieser Kette ergibt sich dann aus der Anzahl der gefundenen Links, die zu dieser Kette beitragen, und der Anzahl der nötigen Links, um die Kette herzustellen:

Recall-Berechnung

$$R_c = \frac{|\text{gefundene Links}|}{|\text{nötige Links}|} = \frac{|\text{korrekte Links}|}{|c| - 1}.$$

Die Recall-Berechnung demonstriert Abbildung 5.12, ebenfalls mit einer Kette von drei Elementen. Einer von zwei notwendigen Links wurde gefunden, der Recall beträgt also 50%.



$$R = \frac{|\text{gefunden und richtig}|}{|\text{benötigt}|} = \frac{1}{2} = 50\%$$

Abbildung 5.12: Berechnung des Recalls R bezüglich einer einzelnen tatsächlichen Koreferenzkette

Auch bei der Berechnung des Recall für das Gesamtergebnis summiert man im Zähler und Nenner über alle Koreferenzketten, wohlgemerkt dieses Mal über alle *tatsächlichen*. Diese seien mit C_k (*k* wie *key*) bezeichnet:

$$R = \frac{\sum_{c \in C_k} |\text{gefundene Links von c}|}{\sum_{c \in C_k} (|c| - 1)}$$

Precision und Recall für die Beispiele Betrachten wir mit diesen Formeln unsere anfänglichen Beispiele. Zur Erinnerung:

- Tatsächliche Koreferenzketten: <Luke Skywalker – Luke – er> <X-Wing – Raumschiff>
- Ergebnis 1: <Luke Skywalker – X-Wing – Luke> <er> <Raumschiff>
- Ergebnis 2: <Luke Skywalker – Luke> <X-Wing – Raumschiff – er>

Wir berechnen die Precision-Werte, betrachten also die *berechneten* Koreferenzketten: In Ergebnis 1 hat die erste Kette eine richtige Verbindung, die zweite und dritte enthalten jeweils nur ein Element und daher keine Links. Im Zähler steht also 1. Im Nenner steht $(3-1)+(1-1)+(1-1) = 2$, die Precision von Ergebnis ist daher $\frac{1}{2}$.

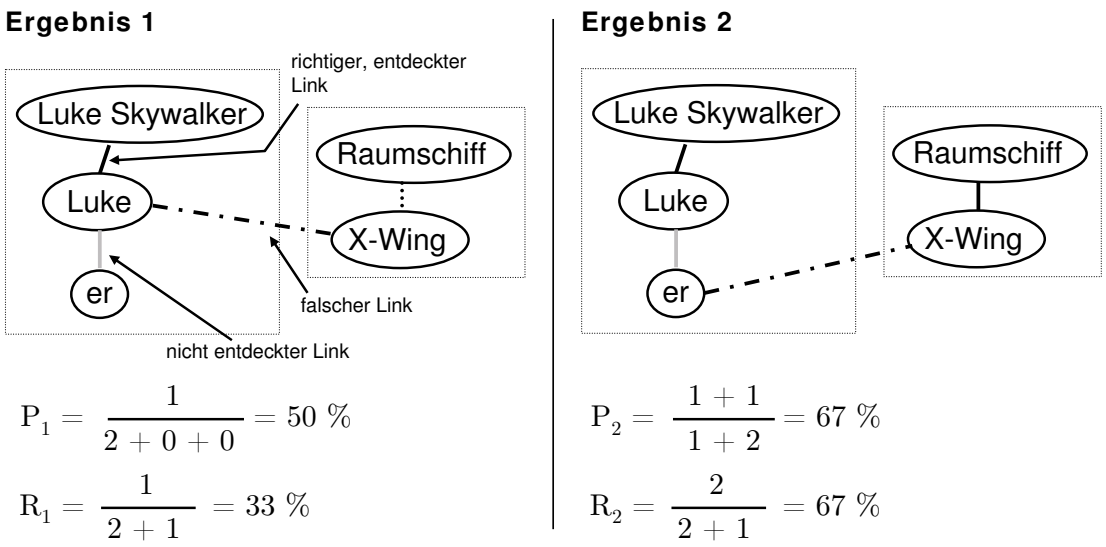
Bei Ergebnis 2 finden wir in jeder der zwei gefundenen Ketten einen korrekten Link vor (Luke Skywalker – Luke, X-Wing – Raumschiff), im Zähler steht also $1+1 = 2$. Im Nenner steht $(2-1)+(3-1) = 3$, somit hat Ergebnis 2 eine Precision von $\frac{2}{3}$.

Für die Recall-Berechnungen sehen wir uns die *tatsächlichen* Koreferenzketten an. Betreffs der ersten Kette <Luke Skywalker – Luke – er> entdeckte Ergebnis 1 einen Link von zwei (Luke Skywalker – Luke), betreffs der zweiten Kette <X-Wing – Raumschiff> keinen. Im Zähler steht also 1. Im Nenner stehen die Kardinalitäten der tatsächlichen Koreferenzketten aufsummiert, jeweils minus 1, also $(3-1)+(2-1) = 3$. Der Recall des Ersten Ergebnisses ist also $\frac{1}{3}$.

In Ergebnis 2 ist ein Link der ersten Kette enthalten (Luke Skywalker – Luke) und ein Link der zweiten Kette (X-Wing – Raumschiff). Der Nenner ist bekanntermassen 3, das Ergebnis also $\frac{2}{3}$.

Die Situation und die Berechnungen sind in Abbildung 5.13 noch einmal veranschaulicht.

In [VBA+] werden die Recall- und Precision-Berechnung formaler eingeführt.



$$\mathrm{P}_1 = \frac{1}{2+0+0} = 50\ \%$$

$$\mathrm{R}_1 = \frac{1}{2+1} = 33\ \%$$

$$\mathrm{P}_2 = \frac{1+1}{1+2} = 67\ \%$$

$$\mathrm{R}_2 = \frac{2}{2+1} = 67\ \%$$

Abbildung 5.13: Precision- und Recall-Berechnung für die anfänglichen Beispiele

5.6.3 Defizite der Vilain-Metrik

Obwohl die Vilain-Metrik in vielen Fällen zu intuitiven Ergebnissen führt, ist sie nicht ohne Schwächen. Auf zwei dieser Schwächen, die Bewertung trivialer Ergebnisse und die Gleichbehandlung unterschiedlich schwerwiegender Fehler, möchten wir hier hinweisen. Eine eingehende Betrachtung der Vilain-Metrik und anderer Metriken mit ihren individuellen Stärken und Schwächen liefert [PB].

Bewertung trivialer Ergebnisse Es ist sehr leicht, von der Vilain-Metrik einen Recall von 100% bescheinigt zu bekommen: Man fügt einfach alle Nominalphrasen in eine einzige Kette ein. Pro tatsächlicher Koreferenzkette c sind dann alle notwendigen Links, also $|c|-1$ Links, erstellt, und die Recall-Formel wird zu

$$R = \frac{\sum_{c \in C_k} |\text{gefundene Links von c}|}{\sum_{c \in C_k} (|c|-1)} = \frac{\sum_{c \in C_k} (|c|-1)}{\sum_{c \in C_k} (|c|-1)} = 1.$$

Die Precision ist bei dieser „Methode“ nur dann 0, wenn alle Links in dieser einen Kette falsch sind, wenn also in Wirklichkeit kein Paar von Nominalphrasen koreferiert. Dies ist sehr selten.

Eine bessere Metrik müsste diese Trivialalgorithmen durch sehr niedrige Recall- *und* Precision-Werte bestrafen.

Unterschiedlich schwerwiegende Fehler Betrachten wir die in Abbildung 5.14 dargestellte Situation: Die Wirklichkeit besteht aus zwei fünfelementigen und einer zweielementigen Koreferenzkette, verschiedene Ergebnisse verschmelzen die beiden fünfelementigen Ketten beziehungsweise eine fünfelementige und eine zweielementige.

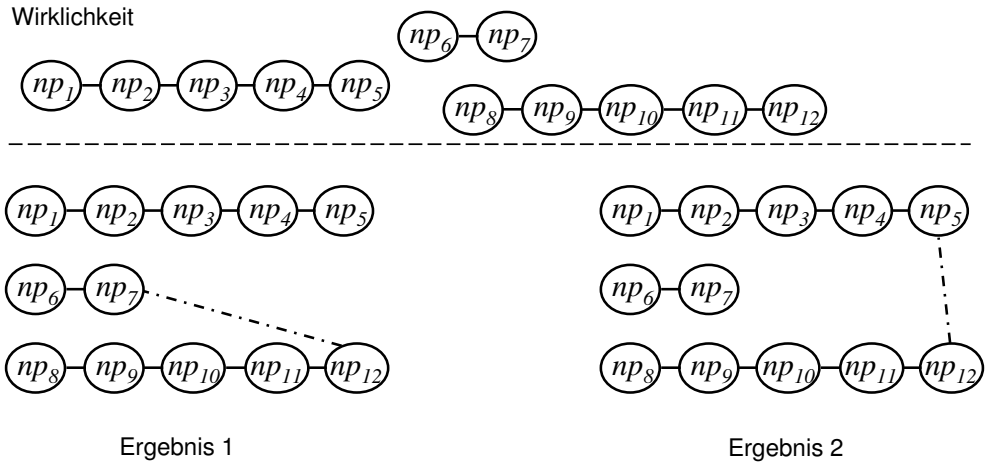


Abbildung 5.14: Verschiedene Precision-Fehler

Beide Ergebnisse enthalten einen falschen Link, also einen Precision-Fehler. Gemäss der Precision-Formel der Vilain-Metrik würden beide Ergebnisse gleich abschneiden:

$$P_1 = \frac{\sum_{c \in C_r} |\text{korrekte Links in c}|}{\sum_{c \in C_r} (|c| - 1)} = \frac{8+1}{9+1} = \frac{9}{10} = \frac{5+4}{6+4} = P_2$$

Die Autoren von [BMBa], an deren Beispiel unser Beispiel angelehnt ist, vertreten die durchaus nachvollziehbare Meinung, dass diese beiden Fehler nicht gleich schwerwiegend sind, wie durch die Rechnung nahegelegt wird. Denn durch den fehlerhaften Link in Ergebnis 2 werden mehr Nominalphrasen fälschlicherweise als koreferierend erklärt als durch den fehlerhaften Link in Ergebnis 1. Deshalb richtet er mehr Schaden an als der andere. Die Autoren beheben diese Schwäche in ihrer Metrik, der B-CUBED-Metrik, indem sie Precision- und Recall-Werte für jede einzelne Nominalphrase np_i berechnen und dabei die Grösse der wirklichen beziehungsweise der gefundenen Koreferenzkette, die np_i enthält, berücksichtigen.

Fazit Man sieht, dass der Entwurf einer guten Metrik für berechnete Koreferenzen, die in allen Fällen intuitive Ergebnisse liefert, keine triviale Aufgabe ist. Auch sollte man bei einer Angabe von Precision und Recall immer die verwendete Metrik angeben, da diese unter Umständen die Werte noch einmal stark beeinflusst.

5.7 Anwendungen und Ausblick

Dieser Abschnitt soll zeigen, wie wichtig eine zuverlässige Koreferenzkettenbildung als Voraussetzung für weitere Aufgaben des TextMining ist. Ausserdem wird ein Ausblick auf die Besonderheiten der Koreferenzbestimmung in sehr domänenspezifischen Texten gegeben.

5.7.1 Automatische Textzusammenfassung und Informationsextraktion

Für die *automatische Zusammenfassung von Texten* (automatic text summarization) und die automatische Informationsextraktion ist es unerlässlich, Koreferenzen aufzulösen. Das automatische Textzusammenfassungssystem ERSS (beschrieben in [BWL+04] und [BWK+03]) ermittelt näherungsweise die Wichtigkeit einer Entität im Text durch die Anzahl, wie oft sie referiert wird, also durch die Länge ihrer Koreferenzkette. Wie bereits ganz am Anfang bemerkt, wird besonders auf Organisationen und Institutionen, aber auch auf Personen in Ämtern, in einem Bericht sehr wahrscheinlich durch unterschiedliche Ausdrücke Bezug genommen. Werden Koreferenzen nicht aufgelöst, so bezeichnen „Gerhard Schröder“, „Schröder“ und „der Bundeskanzler“ aus Sicht des Summarizers verschiedene Entitäten und werden dementsprechend getrennt gezählt, was eine eventuell wichtige Entität hinter anderen Entitäten abrutschen lässt.

Auch im Zusammenhang der *automatischen Informationsextraktion* ist Koreferenzauflösung eine Voraussetzung für gute Ergebnisse. Betrachten wir als Beispiel folgenden Textauszug (aus [GS95]):

> „We are striving to have a strong renewed creative partnership with Coca-Cola,“ Mr. Dooner says. However, odds of that happening are slim since word from Coke headquarters in Atlanta is that . . .

Ein automatisches Informationsextraktionssystem sollte im Idealfall mitteilen können, wo Coca-Cola seine Zentrale hat (nämlich in Atlanta). Ohne die Koreferenz zwischen „Coca-Cola“ und „Coke“ zu erkennen ist dies nicht möglich. Je mehr Koreferenzen zwischen Entitäten, die von Interesse sind, also erkannt werden, desto vollständiger kann die ermittelte Information werden.

5.7.2 Anaphorenauflösung in biomedizinischer Fachliteratur

Zum Schluss möchten wir einen Eindruck von Koreferenzauflösung auf einem sehr speziellen Gebiet, nämlich der Domäne der Biomedizin, vermitteln [Jos02]. Da die Forschung immer schneller neue Kenntnisse produziert, ist es für Menschen schwer, die daraus resultierende Informationsmenge im Überblick zu behalten. Daher wird die maschinelle Extraktion von Informationen aus Fachartikeln und Kurzfassungen von Artikeln (*abstracts*) voraussichtlich eine immer grössere Rolle spielen.

Da biomedizinische Texte voll von Fachbegriffen und Akronymen sind, bleiben viele Koreferenzen ohne spezielles Wissen unentdeckt. Zum Beispiel können die Begriffe „NtrC“ und „the homodimeric enhancer-binding protein“ koreferieren, was aber für ein System unmöglich herauszufinden ist, wenn es nicht Zugang zu einem wissenschaftlichen Lexikon der entsprechenden Domäne hat oder zu einer vergleichbaren Quelle.

Die Autoren von [Jos02] befassen sich in dem Artikel, der ein System zur Anaphernauflösung beschreibt, unter anderem mit *sortal anaphors*. Damit ist eine Anapher gemeint, die den Typ einer Entität nennt und dadurch die Referenz zu dieser Entität herstellt. Wenn zum Beispiel in einem Bericht über ein Treffen zwischen Gerhard Schröder und Michael Schumacher der eine als „der Politiker“, der andere als „der Rennfahrer“ und beide gemeinsam wiederum als „die beiden Männer“ referenziert werden, so sind dies alles *sortal anaphors*. Die Untersuchung eines repräsentativen Textkorpus ergab, dass in der biomedizinischen Literatur ungefähr 60% aller Anaphern *sortal* sind, was die Notwendigkeit unterstreicht, solche Anaphern aufzulösen. Beispiele in dieser Domäne sind etwa „das Protein“ oder „beide Enzyme“.

Um mit *sortal anaphors* umgehen zu können, benötigt man im biomedizinischen Bereich ein Typisierungssystem. Dies ist durch das *Unified Medical Language System (UMLS)* gegeben. Bevor im beschriebenen System Koreferenzen aufgelöst werden, läuft eine Vorverarbeitung, die alle Nominalphrasen im Text identifiziert und für jede unter anderem den passenden UMLS-Typ bereitstellt. Der Koreferenzalgorithmus macht sich diese Typinformationen zunutze.

Zusätzlich wird noch eine Datenbank benutzt, welche fachspezifische Relationen zwischen biomedizinischen Entitäten enthält (zum Beispiel „A inhibits B“, „X regulates Y“). Solche Relationen sollen auch automatisch extrahiert und in die Datenbank eingefügt werden.

Durch diese hoch spezialisierten Massnahmen erzielen die Autoren nach eigenen Angaben eine Precision von 77% und einen Recall von 71%. Der hohe Anteil der typbezogenen (*sortal*) Anaphern macht deutlich, dass allgemeine Methoden zur Koreferenzberechnung schnell scheitern können, wenn sie auf sehr domänenspezifische Texte angewandt werden. Zusatzwissen in Form von Lexika oder Fachdatenbanken sind nötig, um befriedigende Ergebnisse zu erzielen.

Literaturverzeichnis

[Ber97] Sabine Bergler. Towards reliable partial anaphora resolution. In *Proceedings of the ACL'97/EACL'97 Workshop on Operational Factors in Practical, Robust Anaphora Resolution for Unrestricted Texts*, Madrid, Spain, July 1997. http://www.cs.concordia.ca/~bergler/publications/eaclwkshp97.ps.

[BMBa] Breck Baldwin, Tom Morton, Amit Bagga, and al. Description of the UPenn Camp System as Used for Coreference. MUC-7 Proceedings, http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_toc.html.

[BWK+03] Sabine Bergler, René Witte, Michelle Khalife, Zhuoyan Li, and Frank Rudzicz. Using knowledge-poor coreference resolution for text summarization. 2003. http://www-nlpir.nist.gov/projects/duc/pubs/2003final.papers/concordia.final.pdf.

[BWL+04] Sabine Bergler, René Witte, Zhuoyan Li, Michelle Khalifé, Yunyu Chen, Monia Doandes, and Alina Andreevskaia. Multi-ERSS and ERSS 2004. In *Document Understanding Workshop*, Boston Park Plaza Hotel and Towers, Boston, USA, May 6-7 2004. Document Understanding Conference (DUC), http://duc.nist.gov/pubs.html#2004.

[CMBT02] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*, 2002. http://gate.ac.uk/gate/doc/papers.html.

[Cun02] H. Cunningham. GATE, a General Architecture for Text Engineering. *Computers and the Humanities*, 36:223–254, 2002. http://gate.ac.uk.

[Fel98] Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[GS95] Ralph Grishman and Beth Sundheim. Message Understanding Conference - 6: A Brief History. 1995. http://acl.ldc.upenn.edu/C/C96/C96-1079.pdf.

[Hob78] Jerry R. Hobbs. Resolving pronoun references. *Lingua*, 44:311–338, 1978.

[Jos02] José Castaño and J. Zhang and J. Pustejovsky. Anaphora Resolution in Biomedical Literature. In *International Symposium on Reference Resolution, Alicante, Spain*, 2002. http://medstract.org/papers/coreference.pdf.

[LL94] Shalom Lappin and Herbert Leass. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–562, 1994.

[Niy] Niyu Ge and John Hale and Eugene Charniak. A statistical approach to anaphora resolution. http://acl.ldc.upenn.edu/W/W98/W98-1119.pdf.

[PB] Andrei Popescu-Belis. Evaluating reference resolution – a guide to numeric measures. Vorabversion unter http://andreipb.free.fr/textes/apb-corefappl-sub.ps.gz.

[VBA+] Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. A Model-Theoretic Coreference Scoring Scheme. http://acl.ldc.upenn.edu/M/M95/M95-1005.pdf.

[WB03] René Witte and Sabine Bergler. Fuzzy Coreference Resolution for Summarization. In *Proceedings of 2003 International Symposium on Reference Resolution and Its Applications to Question Answering and Summarization (ARQAS)*, pages 43–50, Venice, Italy, June 23–24 2003. Università Ca' Foscari. http://rene-witte.net.

[Wit02] René Witte. *Architektur von Fuzzy-Informationssystemen*. BoD, 2002. ISBN 3-8311-4149-5, http://www.ubka.uni-karlsruhe.de/vvv/2002/informatik/3/.

Johannes Leitner

6 Extraktion von Ontologien aus natürlichsprachlichen Texten

Ontologien sind wesentlicher Bestandteil vieler Systeme die mit verteilter Information arbeiten. Das manuelle Erstellen von Ontologien durch den Menschen ist jedoch aufwendig und kostspielig. Daher ist es erstrebenswert, Ontologien automatisch aus natürlichsprachlichen Dokumenten zu extrahieren. Dieses Kapitel gibt eine Einführung in allgemeine Methoden zum Finden von Ontologien in Textsammlungen und stellt einige Beispielsysteme vor.

6.1 Einführung

6.1.1 Was sind Ontologien?

Um miteinander zu kommunizieren und Wissen auszutauschen ist zwischen den Kommunikationspartnern ein gewisses Einverständnis über die Grundstrukturen der Welt erforderlich. Formal repräsentiertes Wissen benötigt – wenn es austauschbar sein soll – als Grundlage eine abstrakte, vereinfachte Sicht auf einen geeigneten Ausschnitt der Welt. Eine *Ontologie* ist eine explizite Formalisierung einer solchen Sicht [Gru93], eine formale Beschreibung der grundlegenden existierenden Konzepte und ihrer Beziehungen. Man unterscheidet zwischen generischen Ontologien, die die Beziehungen allgemeiner Begriffe darstellen und in vielen Fachgebieten Anwendung finden können, und Domänenontologien, die auf ein Fachgebiet wie Medizin oder Biochemie eingeschränkt sind.

Domänenontologien und generische Ontologien

Die Art der Formalisierung einer Ontologie ist im allgemeinen nicht näher spezifiziert und kann sich je nach Anwendungsfall unterscheiden. Abbildung 6.1 zeigt in einer Beispielontologie Merkmale, die in vielen Ontologiemodellen auftreten. In diesem Beispiel sind drei typische Elemente von Ontologien ersichtlich:

Konzepte Konzepte sind wichtiger Bestandteil aller Ontologien und stellen abstrakte Objekte der modellierten Welt dar. In der Grafik sind sie repräsentiert durch

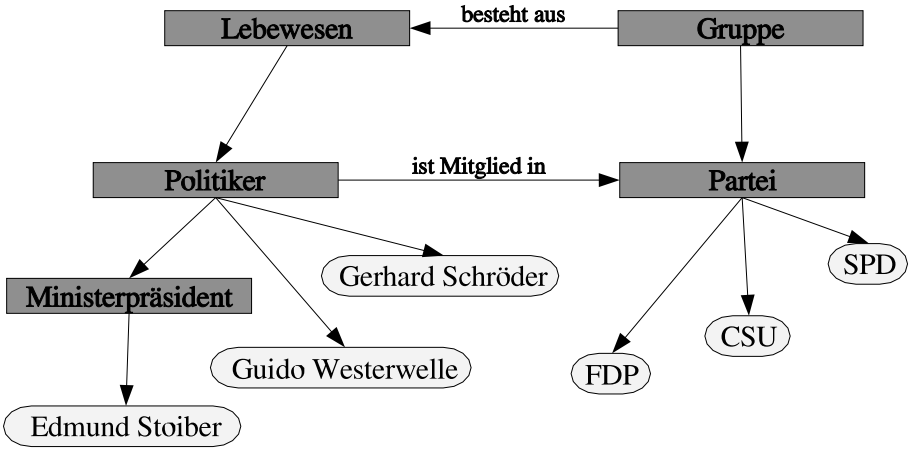


Abbildung 6.1: Eine Beispielontologie aus der Politik-Domäne

Vierecke (zum Beispiel *Partei*, *Lebewesen*).

Beziehungen Benannte oder unbenannte Beziehungen (im Bild die Kanten des Graphen) zwischen diesen Konzepten stellen die Zusammenhänge in dieser Wissensmodellierung dar. Unbenannte Beziehungen stehen in diesem Beispiel für so genannte taxonomische, d.h. Unterkonzeptsbeziehungen.

Instanzen In der Abbildung in abgerundeten Vierecken dargestellt, stellen Instanzen das konkrete Auftreten eines Konzeptes dar. Der Abstraktionsgrad, d.h. die Unterscheidung zwischen Unterkonzepten und Instanzen ist anwendungsabhängig wählbar: So könnte in einem anderen Kontext durchaus *Mensch* eine Instanz, anstelle eines Unterkonzeptes, von *Lebewesen* sein.

Taxonomien

Ontologien, die nur Unterkonzeptsbeziehungen und Instanzen enthalten, die so genannten *Taxonomien*, stellen eine wichtige Unterklasse der Ontologien dar. Eine Taxonomie sollte keine Zyklen enthalten, in vielen Fällen wird auch eine Baumstruktur (mit den Instanzen als Blättern) gefordert. Eine weitere spezielle Ontologie ist der *Thesaurus*, der nur Kanten enthält die eine Synonymbeziehung ausdrücken.

Thesaurus

Viele Ontologieformalisierungen orientieren sich am Ontologiemodell des Semantic Web, der *Ontology Web Language (OWL)* [W3C04]. Dort sind lediglich taxonomische (unbenannte) Beziehungen zulässig, allerdings können andere, konzeptuelle Beziehungen über den allgemeineren Mechanismus der Eigenschaften (*properties*) modelliert werden. Mit *properties* können Konzepten beliebig typisierte Attribute zugeordnet werden, als Typ sind auch andere Konzepte zulässig. Diese Eigenschaften werden dann in den Instanzen konkret ausgefüllt.

6.1.2 Aufbau dieses Kapitels

Dieses Kapitel organisiert sich wie folgt: In den nächsten beiden Abschnitten werden die beiden elementaren Bestandteile eines Ontologieerkennungssystems, das Identifizieren von Konzepten und das Finden von Beziehungen zwischen diesen, sowohl

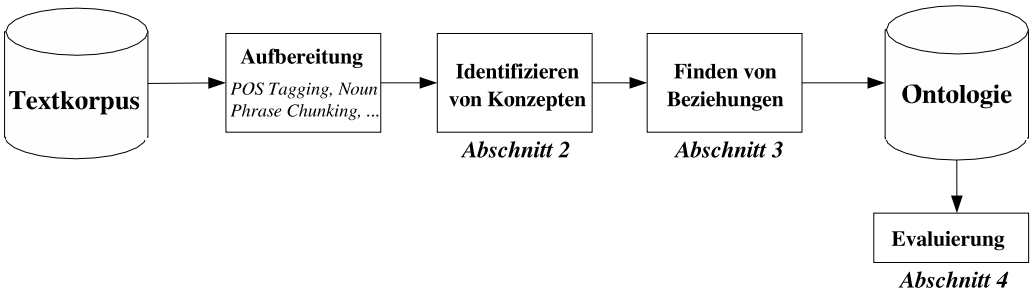


Abbildung 6.2: Aufbau eines Ontologieerkennungssystems

abstrakt als auch mit Verweisen auf konkrete Implementierungen, untersucht. Das vierte Kapitel wendet sich der Evaluierung solcher Systeme zu, die aufgrund der häufigen Unvergleichbarkeit von Ontologien ein schwieriges Problem darstellt. Der Rest dieser Einführung zeigt einige Anwendungsgebiete von Ontologien auf und zeigt, wie diese – mit großem Aufwand – manuell erstellt werden.

Wegen der meist englischsprachigen Quellen und Systeme sind viele Beispiele ebenfalls in englischer Sprache. Viele Methoden sind jedoch in andere Sprachen übertragbar, da sie auf bereits grammatikalisch annotiertem Text arbeiten. *Beispieltext* wird innerhalb des Fließtexts kursiv hervorgehoben.

6.1.3 Einsatzgebiete

Es ist offensichtlich, dass sich die eigentliche Bedeutung des Satzes „*Edmund Stoiber redet mit Guido Westerwelle.*“ erst erschließt, wenn man das in der Ontologie in Abbildung 6.1 dargestellte Hintergrundwissen in Betracht zieht. Eine solche gemeinsame semantische Grundlage ist die Voraussetzung für eine konsistente Kommunikation, für die Weitergabe und das Wiederverwenden von Wissen. Letztendlich wird es angestrebt, auf Grundlage der in Ontologien formalisierten Zusammenhänge automatisiert die Korrektheit von Aussagen zu beweisen oder zu widerlegen.

Semantic Web

Haupteinsatzgebiet von Ontologien sind die so genannten Web Ontologies des Semantic Web. Das „automatische Verstehen“ von Webseiten, Vision des Semantic Web, erfordert eine klare semantische Grundlage. In dieser Schicht kommen im Modell des Semantic Web Ontologien in einer strikten Formalisierung, der Ontology Web Language, zum Einsatz. Deren Syntax hat einen weitgehenden Einfluss auf viele Spezifizierungen von Ontologien und stellt heute einen Quasi-Standard der Formalisierung von Ontologien dar. Eine Web Ontology in OWL besteht aus Klassen, die den Konzepten entsprechen, und Eigenschaften von Klassen. Für beides können taxonomische Beziehungen, also Unterklassen und Untereigenschaften, formuliert werden. Instanzen von Klassen heißen in OWL Individuen [W3C04].

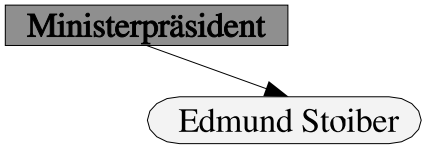
Menschlich nicht mehr überschaubare Domänen

In vielen naturwissenschaftlichen Bereichen sind in den letzten Jahrzehnten unüberschaubare Datenmengen zum Gegenstand der Forschung geworden. In Zusammenfas-

sungen medizinischer Veröffentlichungen der letzten zehn Jahre wurden hunderttausende Proteinnamen gezählt [MSCV04]. Die Interaktionen und Beziehungen in solchen Systemen lassen sich menschlich nur schwer oder gar nicht überblicken. Ontologien erlauben automatischen Systemen, Wissen über diese Domänen auszutauschen und bieten die Möglichkeit strukturierter Suchfunktionen.

Text Mining

Eine große Menge an Information lässt sich nicht ohne Hintergrundwissen aus natürlichsprachlichen Texten erschließen. Selbst die reine grammatikalische Analyse ist ohne diese oft unmöglich. Die Bestimmung von Koreferenzketten beispielsweise, Mengen von Nominalphrasen die sich auf dasselbe Objekt beziehen, liefert ohne zusätzliche Information oft nur wenig Ergebnisse. In Abbildung 6.3 ist ohne Hintergrundwissen nicht ersichtlich, ob sich *Ministerpräsident* auf *Gerhard Schröder* oder auf *Edmund Stoiber* bezieht. Die Miteinbeziehung einer Ontologie, die diese Information enthält, löst dieses Problem.



Später spricht Bundeskanzler Gerhard Schröder in Köln. Er hatte ***Stoiber*** *schon gestern "Realitätsverlust" vorgeworfen.* ***Der bayrische Ministerpräsident*** *hatte die Arbeitsmarktpolitik der rot-grünen Regierung für den Erfolg der NPD verantwortlich gemacht.*

Abbildung 6.3: Einsatz von Ontologien beim Bestimmen von Koreferenzketten

Die Text Mining Architektur GATE[1] unterstützt Ontologien. Terme können mit ihren entsprechenden Instanzen (oder Konzepten) in einer Ontologie annotiert werden. Die so hergestellte Verbindung zwischen Text und Ontologie versieht den Text mit einer Semantik, die zum Beispiel der Mustererkenner JAPE nutzen kann. Dieser erkennt dann nicht nur die spezifizierten Muster, sondern auch Spezialisierungen davon. Enthält ein Muster also Teile, die zum Beispiel mit *Gruppe* annotiert sein sollen, werden – mit der Ontologie aus Abbildung 6.1 – auch mit *Partei* annotierte Terme akzeptiert.

6.1.4 Erstellen von Ontologien

Ontologieerstellung mit Protégé

Traditionell werden Ontologien manuell erstellt. Als unterstützendes Werkzeug zur Ontologieerstellung ist vor allem das weitverbreitete Werkzeug *Protégé*[2] erwähnenswert. In Protégé kann eine Taxonomie komfortabel modelliert werden. Konzeptuelle Beziehungen werden wie in OWL über Properties ausgedrückt. Mit einer speziellen Anfragesprache kann dann die Menge der Instanzen durchsucht werden. Protégé lässt sich in das GATE-System einbinden [CMB+05] – die Erstellung von Ontologien zur Verwendung in GATE erfolgt daher häufig mit Protégé.

Trotz guter Werkzeugunterstützung ist das manuelle Erstellen von Ontologien langwierig, aufwendig und kostspielig. In den meisten Fällen werden hierzu sowohl Domänenfacheute als auch Informatiker benötigt. Die Erstellung einer nützlichen Ontologie

[1] http://gate.ac.uk
[2] http://protege.stanford.edu

mit mehreren hunderttausend Einträgen ist daher oft wirtschaftlich inpraktikabel und unzumutbar – man spricht hier vom *knowledge acquisition bottleneck*.

Knowledge Acquisition Bottleneck

Die Möglichkeit, Ontologien voll- oder zumindest halbautomatisch aus natürlichsprachlichen Texten zu erstellen, birgt also großen Nutzen. In den nächsten zwei Abschnitten dieses Kapitels werden Methoden und Systeme vorgestellt, die dies zumindest ansatzweise möglich machen.

Die Suche nach Ontologien in Texten unterliegt allerdings einigen inhärenten Beschränkungen. Könnte man Ontologien vollständig aus Texten extrahieren, wären sie überflüssig, da man offensichtlich keine Zusatzinformation benötigte und direkt auf den Texten arbeiten könnte. Fast alle Texte setzen Hintergrundwissen (sprich: das Vorhandensein einer Ontologie) voraus, welches *nicht* im Text enthalten ist. Dieses Hintergrundwissen kann nicht aus diesen Texten gewonnen werden – es ist bestenfalls ein iterativer Prozess möglich, der sich Hintergrundwissen aus Texten aneignet, um das Text Mining im weiteren damit anzureichern [BHS02]. Meist wird nur ein halbautomatisches System angestrebt, das dem menschlichen Ontologieersteller möglichst viele Aufgaben abnimmt oder die Auswahl einschränkt.

6.2 Identifizierung relevanter Konzepte

Im ersten Schritt der Ontologieextraktion werden im Textkorpus relevante Konzepte identifiziert. Die Konzepte stellen eine Auswahl der während der Textaufbereitung identifizierten Nominalphrasen (*noun phrases*) dar. Häufig ist bereits bei dieser Auswahl menschliche Unterstützung vonnöten, da eine ungeeignet große (oder kleine) Menge von Konzepten die Qualität der im nächsten Schritt gefundenen Beziehungen stark einschränken kann. Es existieren jedoch Methoden, die Relevanz eines Terms bezüglich einer bestimmten Domäne automatisch einzuschätzen.

Eine Auswahl relevanter Konzepte ist nicht nur für die Erstellung von Ontologien, sondern auch an sich sinnvoll. So können beispielsweise Dokumente automatisch kategorisiert werden, oder das Auftauchen und die Verwendung neuer Fachbegriffe automatisch überwacht werden. Man spricht hier von *terminology detection*.

Terminology Detection

In Fachdokumenten treten domänenspezifische Begriffe meist überdurchschnittlich häufig auf. Umgekehrt ist ein überdurchschnittlich häufig auftretender Begriff ein Indiz dafür, dass es sich um einen wichtigen oder domänenrelevanten Begriff handelt [MSCV04].

Das Verhältnis der Auftretenswahrscheinlichkeiten im allgemeinen und in Fachdokumenten liefert also einen Hinweis auf die Relevanz eines Begriffes. Dieses Verhältnis wird häufig als *Term Frequency Inverse Document Frequency (TF·IDF)* bezeichnet. Sind diese Verhältnisse bestimmt, können alle Begriffe in die Menge der Konzepte aufgenommen werden, deren Relevanz einen bestimmten Schwellwert überschreitet. Alternativ kann an dieser Stelle die nach Relevanz geordnete Liste der Terme dem Menschen zur Auswahl präsentiert werden, wie dies zum Beispiel im KAON Workbench (TextToOnto)[3] System der Fall ist.

TF·IDF

Zu allgemeine Begriffe erweisen sich als nachteilig beim Erstellen von Ontologien, da sie sehr häufig auftreten und zu zu vielen anderen Konzepten in Beziehung stehen. Dennoch können sie in Texten zufällig – zum Beispiel abhängig vom Schreibstil – überdurchschnittlich häufig auftauchen und daher als domänenrelevant eingestuft

[3] http://kaon.semanticweb.org

werden. Diese Begriffe können in einer existierenden Menge von Konzepten durch Nachschlagen in Lexika wie WordNet eliminiert werden.

Die wirkungsvollste Methode benutzt Domänenwissen zum Identifizieren relevanter Konzepte. In vielen naturwissenschaftlichen Domänen lassen sich Wörter anhand ihrer Endungen (wie „...ase“, „..itis“) als relevante Begriffe identifizieren.

6.3 Finden von Beziehungen zwischen Konzepten

Wie sonst auch bei der Verarbeitung natürlicher Sprache wird beim Suchen nach Ontologiebeziehungen zwischen statistischen und symbolischen Ansätzen unterschieden. Während bei symbolischen Ansätzen die sprachliche Struktur des Textes im Vordergrund steht, betrachten statistische Ansätze einen Text meist ohne Semantik und analysieren nur die Häufigkeiten des Auftretens von Termen. Diesen beiden grundlegenden Vorgehensweisen werden wir uns in den beiden folgenden Abschnitten zuwenden.

Die Trennline zwischen symbolischen und statistischen Ansätzen ist in der Praxis meist nicht eindeutig gegeben. Insbesondere werden statistische Ansätze oft mit einem geringen Maß an Sprachstruktur (zum Beispiel grammatikalische Annotierung) ausgestattet. Eine dieser „hybriden Methoden“, das Verb-Objekt-Clustering, wird in Abschnitt 6.3.3 dargestellt. Im letzten Abschnitt werden die Methoden betrachtet, die anstelle der aus dem Text gewonnenen Informationen externes Wissen einbeziehen, meist um eine bereits gefundene Ontologie zu verfeinern.

6.3.1 Symbolische Ansätze

Symbolische Ansätze identifizieren taxonomische Beziehungen zwischen Konzepten anhand der in Sätzen implizit enthaltenen Subkonzeptbeziehungen oder Instantiierungen. So enthält zum Beispiel der beispielhafte Satz

> *Ministerpräsident Edmund Stoiber machte die Arbeitsmarktpolitik der rot-grünen Bundesregierung für den Erfolg der NPD verantwortlich.*

unter anderem die implizite Information, dass Edmund Stoiber eine Instanz des Konzepts Ministerpräsident ist. Ein Großteil dieser Informationen lässt sich anhand einfacher Mustererkennung in den vorher grammatikalisch annotierten Sätzen finden. Da es sich bei den gesuchten Mustern meist um reguläre Ausdrücke handelt, ist die Implementierung dieser Methoden sehr einfach und generisch möglich.

Interessante Muster sind *definierende* Sprachmuster, wie:

Appositionen (wie "*Der* NP_{type} NP_{token}")
Beispiel: *Der Parteivorsitzende Gerhard Schröder sagte ...*

Copula (wie "NP_{token} *ist ein* NP_{type}")
Beispiel: *Gerhard Schröder ist Parteivorsitzender.*

Benennungen (wie "NP_{type} *genannt/namens* NP_{token}")
Beispiel: *Ein Abgeordneter namens Gerhard Schröder hatte ...*

und *exemplifizierende Sprachmuster*, wie:

- NP_1 (, NP_i)* *und andere* NP_{type}
  Beispiel: *Gerhard Schröder, Edmund Stoiber und andere Politiker ...*

- NP_{type} *wie* NP_{token}
  Beispiel: *Abgeordnete, wie Gerhard Schröder ...*
- *Unter den* NP_{type} Verb NP_{token}
  Beispiel: *Unter den Abgeordneten sprach Gerhard Schröder als ...*
- NP_{type}, *außer/ausgenommen/bis auf* NP_{token}
  Beispiel: *Die Abgeordneten, ausgenommen Gerhard Schröder ...*

Letztere werden aufgrund ihrer Einführung in [Hea92] oft auch als *Hearst patterns* bezeichnet. Wird eines dieser Muster gefunden, kann die entsprechende taxonomische Beziehung in die Ontologie aufgenommen werden. Hearst patterns

[...] ASH respectfully submits that it is just as wrong to use ***Fred Flintstone, Bugs Bunny, Barnie the Dinosaur, Willy the Penguin, Tinker Bell, and other cartoon characters*** *yet to be created to push cigarettes as it is to use Joe Camel, argues the [...]*
Quelle: `http://ash.org`

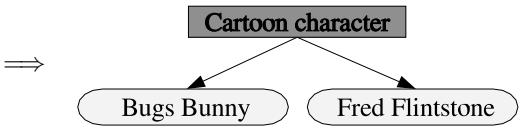


Abbildung 6.4: Beispiel für die Funktionsweise symbolischer Methoden

Eine weiteres, besonders einfaches Muster ist die *Teilphrase*. Dieser Methode liegt die linguistische Annahme zugrunde, dass jede Nominalphrase, die eine andere enthält, eine Instanz (oder ein Unterkonzept) des enthaltenen Konzeptes ist. So ist zum Beispiel *Grünkohl* ein Unterkonzept von *Kohl*. Hier kann also ausschließlich auf den gefundenen Konzepten gearbeitet werden, es ist kein Kontext im Satz notwendig. Die Verwendung der Teilphrasenmethode ist – vor allem in stark domänenspezifischen Ontologien mit langen Nominalphrasen als Konzepten (zum Beispiel *critical immonoregulatory cytokine*) – eine ergiebige Methode zum Aufdecken ganzer Konzepthierarchien und Instanzbeziehungen. Die linguistische Annahme ist allerdings nicht immer korrekt, so ist zum Beispiel *Schneemann* in den meisten Taxonomien kein erwünschtes Unterkonzept von *Mann*.

Der Vorteil symbolischer Vorgehensweisen, im Gegensatz zu den im folgenden Abschnitt besprochenen statistischen Methoden, ist ihre Genauigkeit. Da die Beziehungen direkt aus der sprachlichen Information des Textes stammen, werden derartige Muster (bis auf die Teilphrasenmethode) nur in seltenen Fällen[4] eine falsche Beziehung identifizieren. Der Nachteil dieser Methoden ist der – im Vergleich zur meist überraschend geringen Ausbeute – unangemessen hohe Aufwand der Mustererstellung [MSCV04, CPSTS04]. Die Definition einer hinreichend großen, sprachabhängigen Menge von Mustern erfordert wiederum die menschliche Arbeit, die ein Ontologieerkennungssystem eigentlich vermeiden sollte. Rein symbolische Ansätze werden wegen dieses ungünstigen Kosten-Nutzen-Faktors selten als einzige Methode verwendet.

In [OC04] wird eine vordefinierte Menge solcher Muster auf einen Textkorpus angewendet um Unterkonzepte (hier als *subtypes* bezeichnet) zu finden. Die so gefundenen Beziehungen werden durch die Prädikate subtype(a, b) und properSubtype(a, b)

[4] Da sie nur einen begrenzten Kontext miteinbeziehen, sind symbolische Methoden natürlich nicht *immer* korrekt. Bei indirekter Rede oder einer Negation im entfernteren Kontext kann eine unzuverlässige oder falsche Beziehung erkannt werden.

(nicht reflexiv) repräsentiert. Auf Aussagen dieser Form werden in der logischen Programmiersprache Prolog definierte Regeln angewandt um die Ontologie schrittweise zu verbessern.

Systeme, die eine Kombination verschiedener Methoden benutzen, beziehen die Suche nach Sprachmustern meist mit ein, allerdings spielen die dadurch gewonnenen taxonomischen Beziehungen mengenmäßig im Vergleich zu den statistischen Methoden eine untergeordnete Rolle [CPSTS04].

6.3.2 Statistische Ansätze

Statistische Methoden analysieren den Text nicht auf sprachlicher, sondern auf Worthäufigkeitsebene. Meist wird folgende Vorgehensweise zumindest prinzipiell eingehalten:

a) Analysieren der Häufigkeit gemeinsamen Auftretens (englisch *co-occurrence*) von Wörtern, meist Wortpaaren.

b) Aus den bestimmten Häufigkeiten kann die *Ähnlichkeit* zweier Worte bestimmt werden und die Menge der Konzepte so in einem metrischen Raum angeordnet werden.

c) In diesem Raum können die Techniken des *Clustering* angewandt werden, um Klassen ähnlicher Konzepte zu identifizieren.

Es werden nicht in jedem Fall alle drei Schritte ausgeführt – nach jedem Schritt liegt ein präsentierbares Zwischenergebnis vor, das aufbereitet in eine Ontologie eingehen kann.

Co-occurrence-Analyse

In der Co-occurence-Analyse wird die relative Wahrscheinlichkeit des gemeinsamen Auftretens zweier Wörter innerhalb eines Kontextes bestimmt. Bei dem Kontext kann es sich um einen Satz, einen vordefinierten Wortabstand, oder einen Absatz handeln. Zuerst wird für alle Konzeptpaare die Häufigkeit der Co-occurence bestimmt, wie in folgendem Beispiel für die Terme *Sprache*, *Beziehung* und *Konzept* auf Absatzebene in diesem Kapitel:

cooccurence ($p(x \cup y)$)	*Sprache*	*Beziehung*	*Konzept*
Sprache	16 ($p = 0.064$)	6 ($p = 0.024$)	3 ($p = 0.012$)
Beziehung	6 ($p = 0.024$)	26 ($p = 0.104$)	17 ($p = 0.068$)
Konzept	3 ($p = 0.012$)	17 ($p = 0.068$)	39 ($p = 0.156$)

Die Einträge in der Tabelle geben die absolute und relative Häufigkeit des gemeinsamen Auftretens der in Spalte und Zeile eingetragenen Wörter an. In Klammern ist die Wahrscheinlichkeit, dass die beiden Wörter gemeinsam in einem Absatz auftreten,[5] angegeben. Insbesondere besagen die diagonalen Einträge, wie häufig ein Wort überhaupt auftrat. Aus dieser Tabelle lassen sich die *bedingten Wahrscheinlichkeiten* $p(a|b)$ und $p(b|a)$ errechnen.

$$p(a|b) := \frac{p(a \cup b)}{p(b)}$$

[5] Insgesamt enthält dieses Kapitel etwa 250 Absätze.

Nun steht $p(a|b)$ für die Wahrscheinlichkeit, dass in einem Kontext, in dem b auftritt, auch a auftritt. Offensichtlich sind diese Auftretenswahrscheinlichkeiten nicht mehr symmetrisch, der Zusammenhang zwischen $p(a|b)$ und $p(b|a)$ ist gegeben durch *Bayes' Regel* (siehe Kapitel 2).

Im Beispiel entsteht so die neue Tabelle der bedingten Wahrscheinlichkeiten:

$p(x\|y)$	*Sprache*	*Beziehung*	*Konzept*
Sprache	1.000	0.230	0.077
Beziehung	0.375	1.000	**0.435**
Konzept	0.187	**0.436**	1.000

Die Ergebnisse der Co-occurrence-Analyse können nun zum Clustering verwendet werden, allerdings können aus ihnen auch direkt Ontologiebeziehungen gewonnen werden. So kann man alle Regeln, die ein bestimmtes Maß an Wahrscheinlichkeit überschreiten, direkt als Beziehungen in die Ontologie aufnehmen. Im Beispiel könnten dies alle Beziehungen sein, die eine Wahrscheinlichkeit von 0.4 überschreiten (im der Tabelle fett markiert), dies führt zu einer bidirektionalen Beziehung zwischen *Konzept* und *Beziehung*.

In [MS00] werden die Ergebnisse der Co-occurence-Analyse benutzt, um allgemeine, konzeptuelle Beziehungen bezüglich einer existierenden Taxonomie zu finden und diese auf der richtigen Abstraktionsebene (siehe Abbildung 6.5) in die Taxonomie einzuordnen. Dabei werden für *Assoziationsregeln* der Form Konzept$_1$ $\Rightarrow$ Konzept$_2$ die Maße *confidence* und *support* errechnet. Hier wird eine gegebene Taxonomie miteinbezogen, indem jedes Vorkommen eines Unterkonzeptes (zum Beispiel *city* in Abbildung 6.5) auch als Vorkommen der Überkonzepte (zum Beispiel *area* in Abbildung 6.5) zählt. Dann werden Regeln, die einen bestimmten *support* überschreiten als konzeptuelle Beziehungen in die Ontologie aufgenommen, *es sei denn* es gibt eine andere Regel $\{Konzept'_1\} \Longrightarrow \{Konzept'_2\}$ mit mindestens so hohem *confidence* und *support*, wobei $\{Konzept'_1\}$ ein Überkonzept von $\{Konzept_1\}$ und $\{Konzept'_2\}$ ein Überkonzept von $\{Konzept_2\}$ in der zugrundeliegenden Taxonomie ist. Auf diese Weise werden die aus den Assoziationsregeln gefolgerten konzeptuellen Beziehungen stets auf der höchstmöglichen Abstraktionsebene zusammengefasst.

Assoziationsregeln

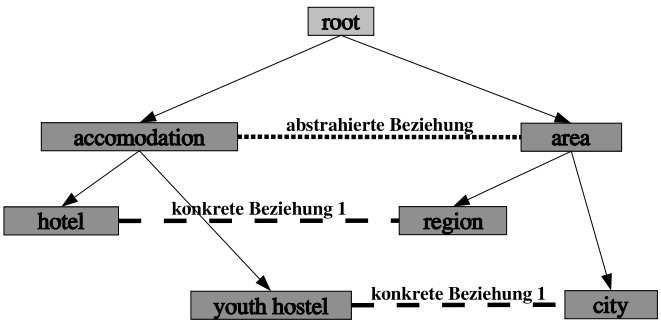


Abbildung 6.5: Einordnen gefundener Assoziationsregeln auf verschiedenen Abstraktionsebenen

Ähnlichkeitsmetriken

Nach der Co-occurence-Analyse kann die Menge der Konzepte als m-dimensionaler Raum aufgefaßt werden (wobei m die Anzahl der vorhandenen Konzepte ist), indem jedem Konzept x die Verteilung der Co-ocurrence-Wahrscheinlichkeit bezüglich aller anderen Worte w, ϕ_x, zugeordnet wird:

$$\phi_x(w) = p(x|w)$$

So ist im vorangehenden Beispiel $\phi_{Sprache} = (1.000, 0.230, 0.077)$. Auf diesem Vektorraum lassen sich verschiedene Metriken definieren, drei der bekannteren sind die *euklidische Norm* (6.1), die L_1-Norm (6.2) und die Cosinusnorm (6.3).

$$\text{Euklid}(x,y) = \sqrt{\sum_w (\phi_x(w) - \phi_y(w))^2} \tag{6.1}$$

$$L_1(x,y) = \sum_w |\phi_x(w) - \phi_y(w)| \tag{6.2}$$

$$\cos(x,y) = \frac{\sum_v \phi_x(v)\phi_y(v)}{\sqrt{\sum_v \phi_x(v)^2}\sqrt{\sum_v \phi_y(v)^2}} \tag{6.3}$$

Die Wahl der Norm ist deswegen von Interesse, weil man es bei der statistischen Analyse natürlichsprachlicher Dokumente – und insbesondere dem gemeinsamen Auftreten zweier Terme – nie mit einer ausreichenden Datenmenge zu tun hat („*sparse data*“). So treten in noch so großen Textkorpora Wortpaare nicht auf, deren Wahrscheinlichkeit eigentlich nicht Null sein sollte. Die Verteilungsfunktionen ϕ_x sind also je nach Korpusgröße meist nur schlechte Schätzungen. Die gewählte Metrik sollte mit dieser Art von Daten „angemessen umgehen“ – so ist beispielsweise die Euklidische Norm aufgrund ihrer hohen Empfindlichkeit gegenüber sog. Ausreißern hier ungeeignet. [MPS03] erwähnen als häufig verwendete Metriken die Kosinus- und die Leibler-Kullback-Norm. Ein detaillierter Vergleich von sieben für die Co-occurence-Analyse geeigneten Ähnlichkeitsmetriken findet sich in [Lee99].

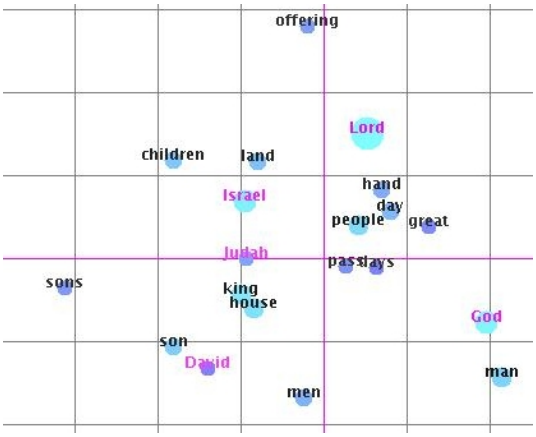


Abbildung 6.6: Ausschnitt einer Concept Map

Die Darstellung einer Menge von Konzepten als metrischer Raum liefert zwar noch keine Ontologie, ist jedoch auch allein ein Mittel, Wissen aus Texten zu gewinnen. In einer Ebene visualisiert, erhält man hieraus eine so genannte *Concept Map*. Abbildung 6.6 zeigt einen Ausschnitt einer Concept Map die von Leximancer (siehe Abschnitt 6.5.2) anhand der Co-occurence-Analyse der Bibel erstellt wurde. Concept Maps können dazu dienen, dem Menschen schnell und vollautomatisch einen Überblick über Themengebiete zu verschaffen.

Concept Maps

Clustering

Zum Finden von Ontologiebeziehungen zwischen derartig angeordneten Konzepten können nun die Methoden des *Clustering* verwendet werden. Hierbei werden ähnliche Objekte zu Klassen gruppiert. Clustering ist auch hierarchisch möglich. So lassen sich zum Teil ganze Objekthierarchien finden.

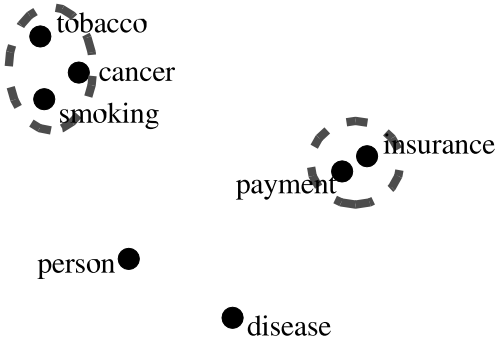


Abbildung 6.7: Identifizieren von Klassen von Objekten in einem metrischen Raum

Man unterscheidet (einfach) partitionierende und hierarchische Clusteringmethoden. Zu den einfachsten und bekanntesten Methoden des partitionierenden Clusterings zählt die k-Nearest-Neighbor-Methode (kNN). Die hierarchischen Clusteringmethoden lassen sich wiederum in zwei grundlegende Arten unterteilen:

Top-down clustering Das top-down oder auch unterteilende Clusteringverfahren unterteilt eine Menge von Elementen rekursiv immer wieder in zwei (nicht zwangsweise gleichmächtige) Teile. Am Ende erhält man so eine geschachtelte Klassenhierarchie, deren Blätter gerade die einzelnen Elemente einer Menge und deren Wurzel die gesamte Menge ist.

Bottom-up clustering Bei den bottom-up oder auch anhäufenden Verfahren wird zunächst jedes Element als ein Cluster aufgefasst. In jedem Schritt werden nun die einander am nächsten liegenden Cluster zusammengefasst. Sind alle Cluster zu einem vereinigt worden, oder überschreitet der Abstand aller noch getrennten Cluster ein bestimmtes Maximum, endet das Verfahren. Auch hier ist das Ergebnis eine Hierarchie von Klassen. Bottom-up Clustering wird in der Praxis häufiger verwendet als Top-down-Clustering.

Das Ergebnis eines hierarchischen Clusterings wird von der Definition des *Clusterabstands* signifikant beeinflusst. Der Clusterabstand basiert stets auf dem elementaren Abstand – dem oben eingeführten Ähnlichkeitsmaß – und kann zum Beispiel

das Maximum (*single linkage clustering*), das Minimum (*complete linkage clustering*) oder der Durchschnitt (*average linkage clustering*) des Abstands der in beiden Klassen enthaltenen Elemente sein.

Abbildung 6.8: Unklare Interpretation der durch Clustering gefundenen Beziehungen

In einem Ontologieerkennungssystem besteht die größte Schwierigkeit beim Verwenden von Clustering in der Interpretation der so gefundenen Konzeptklassen, wie in Abbildung 6.8 zu sehen ist. Es ist lediglich bekannt, dass zwei Konzepte besonders häufig zusammen vorkommen – nicht, was für eine Beziehung zwischen den beiden besteht, oder ob es überhaupt eine direkte gibt. So können die beiden Terme auch nur Unterkonzepte oder Instanzen eines gemeinsamen, unbekannten Überkonzepts sein (in Abbildung 6.8 links), welches in deren Kontext jedoch nur selten auftritt.

6.3.3 Hybride Methoden

Rein symbolische oder rein statistische Verfahren werden in der Praxis nur selten verwendet. Im Folgenden werden zwei Ansätze beschrieben, die statistische Methoden mit grammatikalischen Informationen anreichern, um so besser verwertbare Ergebnisse zu erzielen.

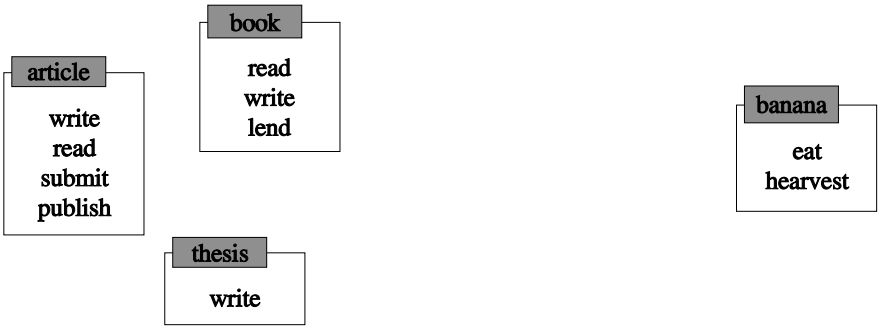


Abbildung 6.9: Mit Termen assoziierte Verbklassen

Nicht jedes Konzept in natürlichsprachlichen Texten wird gleich häufig mit denselben Verben verwandt, sicherlich ist die Kombination „Buch lesen“ häufiger als „Buch essen“. Den folgenden beiden Ansätzen liegt die Annahme zugrunde, dass das Auftreten eines Konzepts als Objekt eines Verbs die Bedeutung des Konzepts mehr oder weniger stark einschränkt. So wäre in unserem Beispiel „Buch“ eine Instanz oder Unterkategorie des abstrakten Konzepts „Lesbares Objekt“ aber nicht – oder mit einer

geringeren Wahrscheinlichkeit – „Essbares Objekt“. Man kann diese Annahme nutzen, indem man die für das Clustering erforderliche Ähnlichkeitsfunktion nicht auf gemeinsamen Auftreten in einem Kontext basiert, sondern auf häufigem gemeinsamem Auftreten als Objekt desselben Verbs.

In einem ersten Schritt wird über alle Konzept hinweg die bedingte Wahrscheinlichkeit des Auftretens als Objekt eines Verbs $p(t|v)$ berechnet. Ein Konzept wird dann mit der Menge der am häufigsten auftretenden Verben assoziiert. So erhält man eine Menge von (nicht-disjunkten) Verbklassen, wie in Abbildung 6.9 dargestellt.

Anhand dieser Verb-Objekt-Häufigkeiten werden in [CST03] taxonomische Beziehungen in der Ontologie angelegt. Konzepte, die häufig mit dem gleichen Verb auftreten, werden als Unterkonzepte eines neuen abstrakten Konzepts, dessen Name sich aus dem Verbstamm und der Nachsilbe „-able“ ergibt, eingestuft. Wichtig ist hierbei die zusätzliche Annahme, dass sich die so gewonnenen abstrakten Konzepte zu einer Hierarchie anordnen lassen. Wird also in einem Text das Verb „lesen“ häufig mit den Konzepten *Buch*, *Zeitschrift* und *Hinweis*, das Verb *kaufen* jedoch nur mit *Buch* und *Zeitschrift* verwendet, so würde das abstrakte Konzept *kaufbar* als Unterkonzept von *lesbar* erscheinen, wie in Abbildung 6.10 gezeigt. In der praktischen Anwendung in TextToOnto erzeugt diese Annahme oft ein undurchschaubares Geflecht taxonomischer Beziehungen.

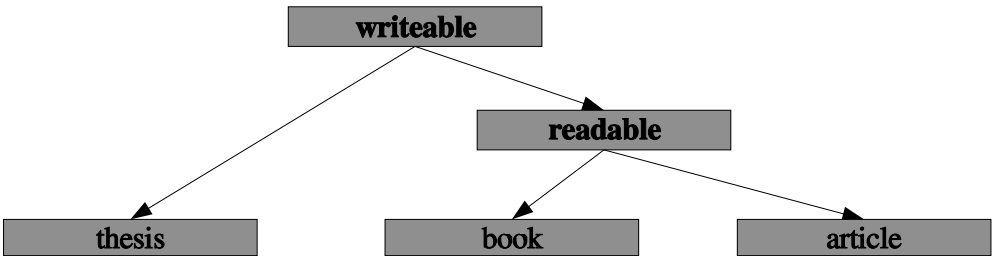


Abbildung 6.10: Eine anhand von Verbklassen erstellte Konzepthierarchie

In [RS04] wird auf Klassen wie in Abbildung 6.9 zuerst ein Clusteringalgorithmus angewendet. Die verwendete Ähnlichkeitsfunktion basiert auf der Anzahl der gleichen und der verschiedenen Verben in beiden Klassen. Durch dieses Clustering entsteht eine Anzahl von *Konzept*klassen, deren Elemente in einer (unbekannten) Beziehung zueinander stehen.

In einem zweiten Clusteringschritt werden nun die präpositionalen Verbindungen zwischen den gefundenen Konzeptklassen untersucht. Hier wird als Ähnlichkeit von Konzepten nicht das Auftreten als Objekt desselben Verbs, sondern das Vorkommen im gleichen Präpositionalphrasenkontext untersucht, zum Beispiel *...transmission of disease...*, *...transmission of infection...*. So erhält man Cluster der Form

transmission-of -cluster: { *disease*, *infection*, ... }.

Man beachte, dass als Kopf der Präpositionalphrase nicht nur elementare Konzepte, sondern die im ersten Schritt gefundenen Klassen auftreten können, zum Beispiel

{book, article, thesis} by {author, institution, ...}

Die derart gefundenen Termklassen werden mit den im ersten Schritt erstellten verglichen und bei genügend Ähnlichkeit zusammengeführt.

Ein Problem bei beiden Ansätzen ist die nicht vorhandene Information über den Abstraktionsgrad eines Objekts. Selbst in Fällen, in denen die linguistische Annahme korrekt ist, also das Verb tatsächlich seine Objekte zu einer gemeinsamen Oberklasse gruppiert, ist aufgrund des geringen Maßes an verwendeter Sprachinformation eine korrekte Einordnung in die Ontologie oft nicht möglich. Es wird noch eine Auswahl der vorherigen Methoden benötigt, vor allem um Instanzbeziehungen aufzudecken, da gleiche Verben oft auf unterschiedlichem Abstraktionsgrad verwendet werden, zum Beispiel

6.3.4 Weitere Ansätze

Nicht alle Methoden lassen sich eindeutig symbolischen oder statistischen Ansätzen zuordnen. Im Folgenden werden Ansätze erläutert, die nicht ausschließlich auf Information arbeiten die aus der zugrunde liegenden Textsammlung gewonnen wurden, sondern zusätzliche Informationen, wie das Hintergrundwissen einer Domäne oder existierende Ontologien, miteinbeziehen.

Externe Quellen

Das Einbeziehen externer Quellen, wie Lexika, Thesauri und vor allem bereits existierende Ontologien kann Fehlentscheidungen beim automatischen Aufbau einer Ontologie vermeiden. Häufig werden generische Ontologien wie WordNet als „Allgemeinwissen“ beim Aufbau einer Domänenontologie hinzugezogen [CPSTS04]. Thesauri können viele der oben erläuterten Methoden deutlich verbessern, da sie a priori bestimmte Termklassen (Synonyme) vorgeben. So kann zum Beispiel das Verb-Objekt-Clustering Informationen über synonyme Verben verwenden um die Ähnlichkeitsfunktion auf Termen zu verbessern.

Verwendet man andere Domänenontologien während des Ontologieaufbaus, vermeidet dies vor allem das Entstehen widersprüchlicher Information. Man spricht hier von *ontology aligning*.

ontology aligning

Es gibt prinzipiell zwei Möglichkeiten, verschiedene Ontologien in Übereinstimmung zu halten [KDFO03]. Zum Einen kann das Wissen der einen Ontologie direkt in eine andere integriert werden. Man spricht dann von *ontology merging* oder *ontology integration*. Eine alternative Möglichkeit, das Wissen verschiedener Ontologien zu verbinden, ist das *ontology linking*. Hier werden Knoten, die dieselben Konzepte repräsentieren, in beiden Ontologien identifiziert und miteinander verknüpft. Auf diese Weise sind zum Beispiel die einzelnen Teile der GeneOntology[6] miteinander verwoben.

ontology merging

ontology linking

GeneOntology

Das Nachschlagen in anderen Ontologien führt umgekehrt auch zu einer iterativen Herangehensweise. Eine Ontologie kann stufenweise mit neuen Informationen angereichert werden, man nennt dies *ontology enrichment*.

Domänenspezifisches Wissen

Wie auch beim Finden von Konzepten kann domänenspezifisches Wissen viel Hintergrundinformation enthalten. Zum Beispiel ist jeder biologische Term, der auf *-ase* endet, ein Enzym – kann also ohne weiteres als Instanz des Konzeptes *Enzym* in die Ontologie eingehen. In Softwaredokumentation [Sab04] sind häufig Funktions- oder

[6] http://www.geneontology.com

Klassenbezeichner (zum Beispiel *getValue* oder *WidgetFactory*) enthalten, die Rückschlüsse auf ihre Einordnung in eine Taxonomie erlauben. Dieser Schritt kann auch zur gleichen Zeit wie die Identifizierung von Konzepten erfolgen.

6.3.5 Zusammenfassung

In diesem Abschnitt wurden die grundlegenden Mechanismen zum Finden von Beziehungen zwischen gegebenen Konzepten in natürlichsprachlichen Texten vorgestellt. Obwohl die Trennlinien zwischen statistischen und symbolischen Ansätzen in der Praxis nicht immer gegeben ist, ist diese Einteilung dennoch sinnvoll. Es lässt sich beobachten, dass symbolische Methoden im allgemeinen zwar wenig ergiebig sind [MSCV04, CPSTS04], die erzielten Ergebnisse aber mit hoher Wahrscheinlichkeit korrekt sind. Bei der Benutzung von statistischen Methoden kehrt sich dieses Verhältnis erfahrungsgemäß um. Desweiteren werden in allen untersuchten Systemen symbolische Methoden nur zum Finden taxonomischer Beziehungen eingesetzt – statistische hingegen oft auch zum Finden von (namenlosen) konzeptuellen Beziehungen. Ausgereifte Systeme benutzen meist eine Anzahl verschiedener Methoden (in [MSCV04] als *heterogenous evidence* bezeichnet), die in einem letzten Schritt gewichtet miteinander kombiniert werden.

6.4 Bewertung von Ontologien

Die Bewertung der Qualität einer Ontologie – und damit der Qualität eines Ontologieerkennungssystems – stellt ein schwieriges Problem dar. Ontologien definieren die Grundlage der Wissensmodellierung und -vermittlung; gäbe es die eine, „richtige“ Ontologie, würde dies eine implizite Übereinstimmung über die grundlegenden Konzepte einer Domäne bedeuten – eine Ontologie wäre dann überflüssig. Ontologien können sich, auch bei gleicher Domäne und gleichem Anwendungsgebiet, stark unterscheiden: Im Abstraktionsgrad der Konzepte, Art und Abstraktionsebene der konzeptuellen Beziehungen, Anzahl der gefundenen Konzepte und Beziehungen, Granularität der Taxonomie, und vielem mehr. Aus keinem dieser Merkmale lässt sich direkt ein Qualitätsmerkmal ableiten.

Unvergleichbarkeit von Ontologien

Es wird oft darauf zurückgegriffen, eine gegebene Ontologie als „perfekt“ anzunehmen, und die automatisch erstellte Ontologie mit dieser zu vergleichen. Hierzu können bereits existierende Ontologien [RS04], sowie speziell zur Evaluierung von Menschen erstellte Ontologien verwendet werden. Es ist jedoch auch dabei nicht davon auszugehen, dass manuell erstellte Ontologien qualitativ besonders hochwertig sind und sich zur Evaluierung eignen [MS00]. Die sinnvollste Evaluierungsmöglichkeit ist daher oft, die gefundenen Ontologien „bei der Benutzung“ zu evaluieren, also dem System, für das die Ontologie bestimmt ist, mehrere Ontologie als Eingabe zu geben, und *dessen* Ergebnisse zu evaluieren. Dies nennt man *extrinsische Evaluierung*, im Gegensatz zur *intrinsischen*, bei der die Ontologie direkt bewertet wird.

Extrinsische und intrinsische Evaluierung

Als Grundlage des Vergleichs von Ontologien dienen meist die klassischen Qualitätsmaße des Information Retrieval, *precision* und *recall*.

$$\text{recall} = \frac{|\{\text{relevante Konzepte/Relationen}\} \cap \{\text{gefundene Konzepte/Relationen}\}|}{|\{\text{relevante Konzepte/Relationen}\}|}$$

$$\text{precision} = \frac{|\{\text{relevante Konzepte/Relationen}\} \cap \{\text{gefundene Konzepte/Relationen}\}|}{|\{\text{gefundene Konzepte/Relationen}\}|}$$

Diese können sowohl auf die gefundenen Konzepte, wie auch auf Relationen angewandt werden. Offensichtlich ist in Ontologien, die viele unterschiedliche Konzepte enthalten, eine große Anzahl von Relationen überhaupt nicht vergleichbar. Dies wirkt sich negativ auf den $recall$ des betrachteten Systems aus. Häufig werden daher dem System die Konzepte des Vergleichssystems vorgegeben, so das diese in beiden Ontologien gleich sind. Aufgrund der geringen Anwendbarkeit dieser Maße erreichen Ontologien meist Werte unter 10%, sowohl für *precision* als auch für *recall*. Ist ein stärkeres Maß an Gemeinsamkeit, zum Beispiel eine gemeinsame Taxonomie, beiden Systemen als Grundlage vorgegeben, können zum Vergleich von Ontologien auch komplexere Graphvergleichsmethoden Anwendung finden, wie die in [MS00] eingeführte und im folgenden kurz erläuterte *Conceptual* beziehungsweise *Relation Learning Accuracy*.

Conceptual Learning Accuracy

Mit der *Conceptual Learning Accuracy (CLA)* lässt sich die Genauigkeit eines in eine vorhandene Taxonomie eingefügten Knotens bewerten.

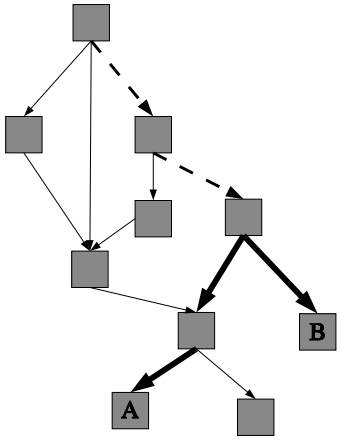


Abbildung 6.11: Berechnung der Conceptual Learning Accuracy

Sie berechnet sich aus der Länge des kürzesten Pfades vom Originalknoten zum gefundenem Knoten $d_{A,B}$ (in Abbildung 6.11 fett), und dem Abstand des nächsten gemeinsamen Vorfahrens zur Wurzel $root_{A,B}$ (gestrichelt) wie folgt:

$$\text{CLA}(A,B) = \frac{root_{A,B}}{root_{A,B} + d_{A,B}}$$

Für zwei gleiche Knoten ist die CLA also genau 1, bei steigendem Abstand sinkt sie linear, wobei die Knotenentfernung mit steigender Tiefe im Baum immer weniger ins Gewicht fällt.

Relation Learning Accurracy

In [MS00] wird eine komplette Taxonomie vorgegeben und nur das Einfügen der gefundenen konzeptuellen Beziehungen evaluiert. Dazu wird eine Vergleichsmetrik ent-

wickelt, die *Relation Match Accuracy (RMA)*, die nicht nur exakte Gleichheit von Relationen erkennt, sondern auch „fast richtige“ Relationen bewerten kann. Sie beruht auf der Conceptual Learning Acccuracy:

$$\mathrm{MA}(R,S) = \sqrt{\mathrm{CLA}(source(R), source(S)) \cdot \mathrm{CLA}(target(R), target(S))}$$

Die MA ist also das geometrische Mittel aus den CAs von Quelle und Senke der gefundenen Beziehung. Wählt man für jede Beziehung die am besten passende und mittelt über die so bestimmten Match Accuracies, erhält man die *Relation Learning Accuracy (RLA)*, eine Bewertung der Ähnlichkeit aller auf einer Taxonomie gefundenen konzeptuellen Beziehungen:

$$\mathrm{RLA}(R,S) = \frac{\sum_{r \in R}(\min_s(\mathrm{CLA}(r,s)))}{|R|}$$

6.5 Beispielsysteme

In diesem Abschnitt werden zwei Beispielsysteme vorgestellt, die die abstrakt eingeführten Methoden zur Ontologieerkennung umsetzen. Das Ontologieerstellungssystem *TextToOnto*, im folgenden Unterabschnitt vorgestellt, implementiert viele der in Abschnitt 6.2 und 6.3 erläuterten statistischen und symbolischen Methoden. In 6.5.2 wird das Werkzeug *Leximancer* vorgestellt, das so genannte *Concept Maps* aus natürlichsprachlichen Textsammlungen erzeugen kann.

Ein weitergehender Vergleich verschiedener Systeme zur halb- oder vollautomatischen Ontologieerstellung findet sich in [SB03].

6.5.1 TextToOnto

TextToOnto [GSV04] ist eine Sammlung von Werkzeugen zur halb- und vollautomatischen Erstellung von Ontologien anhand von natürlichsprachlichen Korpora. TextToOnto baut auf dem *KArlsruhe ONtology and Semantic Web (KAON)* System auf, einem in Java geschriebenen Open Source Framework zum Arbeiten mit Ontologien. Nachdem ein Korpus angegeben wurde, kann mit folgenden Werkzeugen eine Ontologie erstellt werden:

Term Extraction Mit Hilfe des Werkzeugs *Term Extraction* können Konzepte zu einer Ontologie hinzugefügt werden. Terme, die eine bestimmte, vom Benutzer festlegbare Häufigkeitsschwelle überschreiten, werden dem Benutzer in einer Liste präsentiert, die nach Maßen wie der TFIDF (beschrieben in Abschnitt 6.2) sortiert werden kann. Die Entscheidung, welche Terme als Konzepte in die Ontologie aufgenommen werden sollen, liegt beim Benutzer.

Instance Extraction Sind Konzepte in die Ontologie eingefügt worden, kann TextToOnto im Korpus Instanzen dieser Konzepte identifizieren. TextToOnto bedient sich dazu einer umfangreichen Sammlung von Mustern, wie in Abschnitt 6.3.1 beschrieben. Die Erkennung von Instanzen erfolgt vollautomatisch.

Relation Extraction Auf der bisher ermittelten Taxonomie können nun allgemeine, konzeptuelle Beziehungen bestimmt werden. Hierzu verwendet TextToOnto zwei verschiedene Ansätze: Einen, der anhand der Co-occurrence-Analyse wie in Abschnitt 6.3.2 beschrieben Assoziationsregeln ermittelt und einen symbolischen,

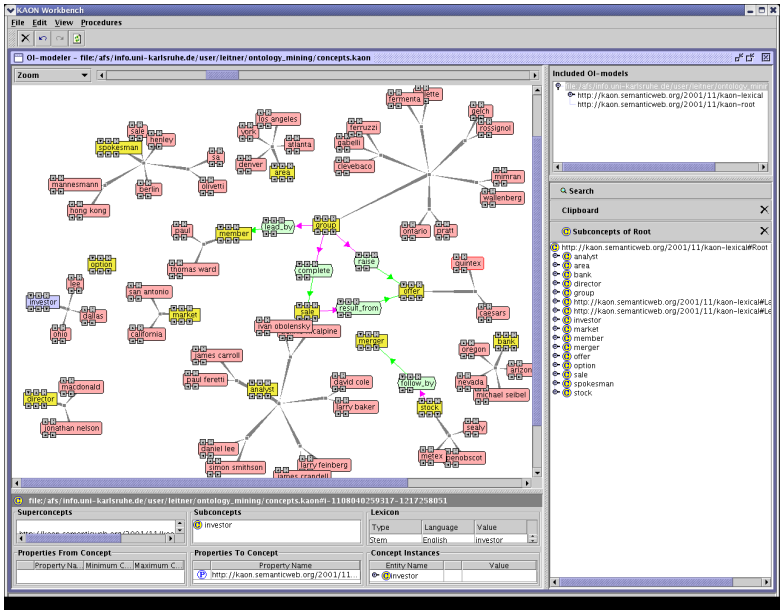


Abbildung 6.12: Das Ontologieerkennungssystem TextToOnto

der eine Menge von Mustern, ähnlich den *Hearst patterns* auf den Textkorpus anwendet. Letzterer Ansatz kann auch benannte konzeptuelle Beziehungen identifizieren, wie in Abbildung 6.12 die Beziehungen *lead_by* und *result_from*. Beim Finden von Transaktionsregeln findet die in Abschnitt 6.3.2 beschriebene Methode zur Einordnung auf höchstmöglicher Abstraktionsebene Anwendung. Eine reichhaltige Taxonomie, insbesondere die im letzten Schritt gefundenen Instanzbeziehungen, können das Ergebnis daher stark verbessern. Beide Methoden können nur semiautomatisch verwendet werden.

Relation Learning Im Gegensatz zur *Relation Extraction* kann das *Relation Learning* auch vollautomatisch ausgeführt werden. Hier wird zum Auffinden von Beziehungen zwischen Konzepten die in Abschnitt 6.3.3 erläuterte Methode des Verb-Objekt-Clusterings verwendet. Für Konzepte die häufig als Objekt des gleichen Verbs, zum Beispiel *write*, auftauchen, wird ein gemeinsames Oberkonzept, zum Beispiel *writeable*, in der Ontologie angelegt.

Zusätzlich stellt TextToOnto noch Funktionen zur Verfügung um Ontologien miteinander zu vergleichen, aufbauend auf *Precision* und *Recall* sowohl für Relationen als auch für Konzepte.

6.5.2 Leximancer

Leximancer [Smi03] ist ein Werkzeug, das den Inhalt einer Sammlung von Dokumenten analysiert und als *Concept Map* darstellt. Leximancer gewinnt also keine Ontologien aus Texten, allerdings sind Concept Maps eine Visualisierung eines wichtigen "Zwischenergebnisses", dass bei vielen Ontologieerkennungssystemen eine Rolle spielt: Der Anordnung der Terme in einem metrischen Raum anhand der Ergebnisse der Co-occurrence-Analyse (siehe Abschnitte 6.3.2 und 6.3.2).

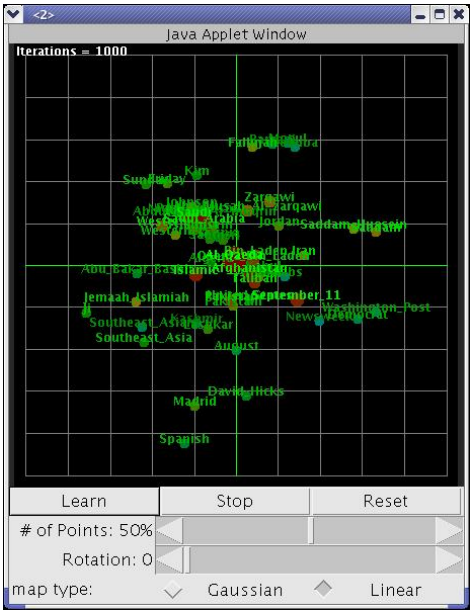


Abbildung 6.13: Ausschnitt einer von Leximancer erstellten Concept Map

In Abbildung 6.13 ist ein Teil einer aus der Bibel[7] erstellten Concept Map dargestellt. Die Helligkeit, in der Konzepte gezeichnet sind, drückt ihre Relevanz aus, d.h. wie oft die Terme überhaupt aufgetreten sind. Der Benutzer kann einen Schwellwert für die Relevanz geben, unter dem Konzepte überhaupt nicht mehr dargestellt werden. Die Nähe zweier Terme wird ermittelt aus einer auf der Co-occurrence-Analyse basierenden Ähnlichkeitsfunktion.

Obwohl Leximancer selbst keine Ontologien erstellt, ist das Ergebnis ein Hinweis für die gute Ausbeute bei der Verwendung rein statistischer Methoden. Zusätzlich können Concept Maps sehr einfach in Thesauri, also spezielle Ontologien, überführt werden.

6.6 Zusammenfassung

In diesem Kapitel wurden die wesentlichen Bestandteile eines Ontologieerkennungssystems eingeführt. Es wurden die grundlegenden Methoden der *terminology detection* erläutert, die dazu dienen, Konzepte einer Ontologie zu identifizieren, nämlich

- Identifizierung von Begriffen, die mit höherer Häufigkeit auftreten als in normalen Dokumenten;
- Eliminierung von zu allgemeinen Begriffen mit Hilfe von Lexika;
- Domänenspezifische Mustererkenner.

In Abschnitt 6.3 wurden verschiedene Methoden vorgestellt, Ontologiebeziehungen in natürlichsprachlichen Dokumenten zu finden. Hierbei wurde zwischen symbolischen und statistischen Ansätzen unterschieden. Symbolische Ansätze durchsuchen

[7] Dieses und weitere Beispiele für die Benutzung von Leximancer finden sich auf `http://www.leximancer.com`.

den Text nach vordefinierten Sprachmustern, die implizite Information über taxonomische Beziehungen der Konzepte enthalten, unter anderem den so genannten *Hearst patterns*. Solche symbolischen Methoden finden taxonomische Beziehungen, die mit hoher Wahrscheinlichkeit korrekt sind, allerdings mit einer sehr niedrigen Ausbeute. Statistische Methoden analysieren die Häufigkeit des Auftretens von Termen, meist das gemeinsame Auftreten (*cooccurrence*) eines Wortpaares. Anhand dessen kann die Menge der Konzepte zu einem Raum angeordnet werden, in dem Clusteringtechniken verwendet werden können, um Klassen zusammenhängender Terme zu identifizieren. Zwischen den Elementen dieser Klassen wird dann eine Beziehung angenommen – Art und Name dieser Beziehung lassen sich aber nicht feststellen. Statistische Methoden liefern viel mehr Ergebnisse als symbolische, allerdings mit einer geringeren Korrektheit.

Im Abschnitt 6.4 wurde erläutert, dass die Evaluierung eines Ontologieerkennungssystems aufgrund der Unvergleichbarkeit von Ontologien nur schwer möglich ist. Als Grundlage der Wissensmodellierung lassen sich Ontologien oft nur informell oder anhand eines anderen Systems, welches die Ontologie als Eingabe nimmt, vergleichen. Wird ein gewisses Maß an Gemeinsamkeit (zum Beispiel gleiche Konzeptmenge oder eine gleiche zugrundeliegende Taxonomie) angenommen, lassen sich auch komplexere Vergleichsmöglichkeiten wie die Relation Learning Accuracy verwenden.

Literaturverzeichnis

[BHS02] B. Berendt, A. Hotho, and G. Stumme. Towards semantic web mining. In *Proceedings of the International Semantic Web Conference (ISWC02)*, 2002. citeseer.ist.psu.edu/berendt02towards.html.

[CMB+05] Hamish Cunningham, Diana Maynard, Kalina Bontcheva, Valentin Tablan, Cristian Ursu, Marin Dimitrov, Mike Dowman, Niraj Aswani, and Ian Roberts. Developing Language Processing Components with GATE - Version 3a. Technical report, The University of Sheffield, 2005. http://gate.ac.uk/sale/tao/tao.pdf.

[CPSTS04] Philipp Cimiano, Alexander Pivk, Lars Schmidt-Thieme, and Steffen Staab. Learning Taxonomic Relations from Heterogeneous Evidence. In *Proceedings of the ECAI Workshop on Ontology Learning and Population* [Eca04]. http://www.aifb.uni-karlsruhe.de/WBS/pci/ECAI04OLWS.pdf.

[CST03] Philipp Cimiano, Steffen Staab, and Julien Tane. Automatic acquisition of taxonomies from text: FCA meets NLP. In *Proceedings of the International Workshop on Adaptive Text Extraction and Mining*, 2003. http://www.dcs.shef.ac.uk/~fabio/ATEM03/cimiano-ecml03-atem.pdf.

[Eca04] *Proceedings of the Workshop on Ontology Learning and Population at the European Conference on Artifical Intelligence (ECAI)*, 2004. http://olp.dfki.de/ecai04/.

[Gru93] Thomas R. Gruber. A translation approach to portable ontologies. *Knowledge Acquisition*, 5(2):199–220, 1993.

[GSV04] Thomas Gabel, York Sure, and Johanna Voelker. KAON – An Overview. Karlsruhe Ontology Management Infrastructure. Technical report, Institut für Angewandte Informatik und Formale Beschreibungsverfahren

(AIFB), Universität Karlsruhe, 2004. http://kaon.semanticweb.org/main_kaonOverview.pdf.

[Hea92] M. A. Hearst. Automatic acquistion of hyponyms from large text corpora. In *Proceedings of the 14th International Confernce on Computational Linguistics*, 1992.

[KDFO03] M. Klein, Y. Ding, D. Fensel, and B. Omelayenko. *Towards the Semantic Web: Ontology Driven Knowledge Management*, chapter 5: Ontology Management: Storing, aligning and mantaining ontologies. Wiley, 2003.

[Lee99] Lillian Lee. Measures of Distributional Similarity. In *37th Annual Meeting of the Association for Computational Linguistics*, pages 25–32, 1999. http://www.cs.cornell.edu/home/llee/papers/cf.pdf.

[MPS03] Alexander Maedche, Viktor Pekar, and Steffen Staab. *Web Intelligence*, chapter 10: Ontology Learning Part One – On Discovering Taxonomic Relations from the Web. Springer, 2003.

[MS00] Alexander Maedche and Steffen Staab. Discovering Conceptual Relations from Text. In *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI-2000)*, 2000. http://www.aifb.uni-karlsruhe.de/WBS/Publ/2000/ecai_amaetal_2000.pdf.

[MSCV04] Inderjeet Mani, Ken Samuel, Kris Concepcion, and David Vogel. Automatically Inducing Ontologies from Corpora. In *Proceedings of the CompuTerm Workshop on Computational Terminology (COLING04)*, 2004.

[OC04] Norihiro Ogata and Nigel Collier. Ontology Express: Statistical and Non-monotonic Learning of Domain Ontologies from Text. In *Proceedings of the ECAI Workshop on Ontology Learning and Population* [Eca04]. http://olp.dfki.de/ecai04/final-ogata.pdf.

[RS04] Marie-Laure Reinberger and Peter Spyns. Discovering Knowledge in Texts for the learning of DOGMA-inspired ontologies. In *Proceedings of the ECAI Workshop on Ontology Learning and Population* [Eca04]. http://olp.dfki.de/ecai04/final-reinberger.pdf.

[Sab04] Marta Sabou. Extracting Ontologies from Software Documentation: a Semi-Automatic Method and its Evaluation. In *Proceedings of the ECAI Workshop on Ontology Learning and Population* [Eca04]. http://olp.dfki.de/ecai04/final-sabou.pdf.

[SB03] Mehrnoush Shamsfard and Abdollahzadeh Barforoush. The state of the art in ontology learning: A framework for comparison. *Knowledge Engineering Review*, 2003. http://www.sepehrs.com/~shams/papers/ontolearn03.pdf.

[Smi03] Andrew E. Smith. Automatic Extraction of Semantic Networks from Text using Leximancer. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, 2003. http://www.leximancer.com/documents/hlt2003.pdf.

[W3C04] W3C. The OWL Web Ontology Language Overview. Technical report, W3C, 2004. http://www.w3.org/TR/owl-features/.

Ralf Krestel

7 Automatische Textzusammenfassung

In diesem Kapitel werden die Grundlagen der automatischen Textzusammenfassung erörtert. Anschließend werden wir die Document Understandig Conference (DUC) vorstellen. Ein konkretes automatisches Textzusammenfassungssystem, ERSS, das an den DUC-Wettbewerben 2003 und 2004 teilnahm, werden wir näher beschreiben, und die Funktionsweise erläutern. Abschließend folgt eine Einführung in das Gebiet der automatischen Evaluierung von Zusammenfassungen und eine konkrete Metrik (ROUGE) wird exemplarisch vorgestellt.

7.1 Einführung

Informationszeitalter. Informationsgesellschaft. Was sagen diese oder ähnliche Ausdrücke über die Welt, in der wir leben aus? Online-Bibliotheken, Wissensmanagement, Tageszeitungen, Spam-Filter, Informationsterminal, PDAs, Mailing-Liste, Internetrecherche, Information Overkill, etc. Es scheint immer wichtiger zu werden, Informationen zu haben, oder zu bekommen. „Wissen ist Macht!“, lautet ein Sprichwort, das den Puls der Zeit zu treffen scheint. Doch was für Wissen hierbei gemeint ist kann nicht eindeutig erschlossen werden. Dass Wissen grundsätzlich erstrebenswert ist, und per se einen Wert besitzt, ist wohl genauso abwegig, wie die konträre Annahme. Die Aufgabe lautet also zu diskriminieren. Wichtiges von Unwichtigem zu trennen, Informationsmüll zu detektieren und aus dem vermeintlich wichtigen Rest eine Auswahl zu treffen. Bei der unwahrscheinlichen Flut an Informationen ein geradezu unlösbares Problem. Nicht zuletzt die Verbreitung des World Wide Web hat dazu beigetragen, dass wir einer Welt mit allgegenwärtiger Information immer näher kommen. Informationen werden in naher Zukunft überall, für jeden und zu jeder Zeit zugänglich sein. Ein immer wichtiger werdender Bereich stellt daher die Informationsverarbeitung im wörtlichen Sinne dar. So haben nicht nur Privatpersonen, Unternehmen und Staatsregierungen ein reges Interesse, relevante Informationen für die unterschiedlichsten Zwecke zu sammeln und auszuwerten, sondern auch immer mehr kommerzielle Anbieter möchten ihren Kunden den Zugang zu für sie wichtigen Informationen gewährleisten. Ein

Hauptaugenmerk liegt hierbei auf der Reduzierung großer Informationsmengen.

7.1.1 Was ist eine Textzusammenfassung?

Zusammenfassungen begegnen wir in unserem täglichen Leben in vielerlei Kontexten. Eine ihrer Aufgaben kann in der Verkürzung von Texten bestehen, ohne jedoch wesentlich deren Informationsgehalt zu verringern. Beispielsweise sei hier die Inhaltsangabe eines Buches genannt. Diese Art der Zusammenfassung nennt man im Englischen einen „Abstract“. Er wird meistens als Surrogat für den langen Originaltext verwendet, und wird von Menschen angefertigt. Es ist viel Hintergrundwissen notwendig, um die einem Text wesentlichen Punkte zu identifizieren und entsprechend aufzuarbeiten. Man erwartet ein grammatikalischen- und semantischen Regeln entsprechendes, zusammenhängendes, gut lesbares Ergebnis. Dies stellt alles in allem eine nicht-triviale Aufgabe dar, welche sehr aufwendig und somit sehr teuer ist. Einer anderen Art, einen Text zu verkürzen, entspricht der sogenannte „Extract“, oder Textauszug. Er beinhaltet im Gegensatz zum Abstract keinen neuen, auf welche Art und Weise auch immer erzeugten Text, sondern extrahiert aus dem Quelltext wichtig erscheinende Informationen. Dies kann in Einheiten von Wörtern, Sätzen, Paragraphen oder ganzen Kapiteln geschehen. Als Beispiel sei hier der Kino-Trailer erwähnt, der zwar keinen Text zusammenfasst, aber dennoch eine Zusammenfassung darstellt. Wie aus dem Beispiel ersichtlich, dient diese Art der Zusammenfassung nicht als Ersatz für das Original, sondern lediglich als grobe Richtlinie, ob man sich das Original ansehen möchte oder nicht. Entsprechend ihrer Anforderungen nennt man diese Art der Zusammenfassung „indikativ“, im Kontrast zur „informativen“ Variante, die das Betrachten des Originals dem Nutzer aus zeitlichen Gründen gerade ersparen möchte. Des Weiteren lassen sich Zusammenfassungen auch noch auf Grund ihrer unterschiedlichen Zielgruppen unterscheiden. So gibt es zum einen generische Zusammenfassungen, die einer möglichst breiten Leserschaft eine allgemeine Zusammenfassung liefern, und zum anderen benutzerorientierte Zusammenfassungen, die ihre Ergebnisse den Wünschen und Vorlieben des jeweiligen Nutzers anpassen. Dies kann aufgrund einer Benutzeranfrage geschehen, über das Speichern von Benutzerprofilen, durch Einschränkung auf bestimmte Themengebiete oder durch benutzerspezifische Schwerpunkte. Jemand, der an den Börsenkursen interessiert ist, braucht nur börsenrelevante Informationen. Und eine Zusammenfassung von Tagesnachrichten sollte die entsprechenden Daten herausfiltern.

Abstract

Extract

Indikativ

Informativ

Soweit zur theoretischen Einteilung von Textzusammenfassungen. Im allgemeinen, umgangssprachlichen Gebrauch versteht man unter einer Zusammenfassung einen informativen Abstract, der meistens generisch ist. Das langfristige Ziel von automatischen Textzusammenfassungssystemen geht sicher auch in diese Richtung, mit der Tendenz zur Benutzerorientierung. Aufgrund der Komplexität dieser Aufgabe konzentriert man sich zur Zeit noch auf den Teilbereich der indikativen Extracts, nicht zuletzt weil diese sich für die Automatisierung mehr eignen. Das schlichte Extrahieren führt dabei oft zu unzusammenhängenden, unleserlichen Fragmenten, die weder kohärent noch kohäsiv sind. Hier besteht die Möglichkeit, durch weitere Forschungsarbeit bessere Ergebnisse zu erzielen. Fehlendes Hintergrundwissen und mangelndes Textverstehen sind weitere Schwierigkeiten solcher automatischer Systeme.

7.1.2 Einteilung von automatischen Textzusammenfassungssystemen

Um ihrer Aufgabe gerecht zu werden, verwenden automatische Textzusammenfassungssysteme unterschiedliche Ebenen der Verarbeitung. Wir folgen hier der Einteilung von [MM99]. Ein eher oberflächlicher Ansatz arbeitet beispielsweise mit Worthäufigkeitsstatistiken. Das heißt, die Wörter, die häufig in einem Text vorkommen, werden als den Text gut klassifizierende Wörter angesehen, wobei das häufige Vorkommen von Stoppwörtern entsprechend behandelt werden muss, um eine aussagekräftige Zusammenstellung zu erhalten. Aussagekräftiger werden diese Zählungen, wenn nicht die absolute Worthäufigkeit berechnet wird, sonderen die Worthäufigkeit im Vergleich zu der relativen Häufigkeit in der Sprache allgemein Betrachtung findet. Damit kommt man zu statistisch relevanten, den Quelltext charakterisierenden Begriffen. Eine derartige Berechnung lässt sich mit Hilfe der tf.idf-Metrik, durchführen, welche Wörter entsprechend gewichtet. Eine Variante ([Sal89]) zur Berechnung ist:

Oberflächenebene

$$w_{ij} = tf_{ij} * log_2 \frac{N}{n}$$

Hierbei ist w_{ij} das Gewicht für das Wort t_i in Dokument d_j, tf_{ij} ist die Häufigkeit des Wortes t_i im Dokument d_j, N ist die Anzahl der Dokumente im Vergleichskorpus, und n ist die Anzahl der Dokumente im Korpus, in denen das Wort t_i vorkommt. Diese Gewichtung von Wörtern hat zur Folge, dass Fachbegriffe, die nur in einer begrenzten Anzahl von Dokumenten vorkommen, von allgemeinen Wörtern, die in vielen Texten vorkommen, unterschieden werden können. Des Weiteren lassen sich auch aufgrund der Position eines Begriffes innerhalb eines Textes Rückschlüsse auf dessen Wichtigkeit ziehen. So wird angenommen, dass einem Wort als Teil einer Überschrift mehr Bedeutung zukommt, als einem Wort mitten in einem Absatz. Auch das Vorkommen zu Beginn eines Abschnittes oder innerhalb eines besonderen Absatzes kann berücksichtigt werden. Entsprechend können Begriffe innerhalb eines Textes, die schon in der Überschrift enthalten waren, eine höhere Priorität eingeräumt bekommen. Hat man zusätzliche Informationen des Benutzers, beispielsweise eine konkrete Anfrage, so sind die Stellen eventuell von Interesse, an denen Begriffe aus der Anfrage im Text auftauchen. Zuletzt lassen sich noch Methoden, die Schlüsselwörter berücksichtigen, zu dem Oberflächenebenenansatz hinzuzählen. Das Vorkommen bestimmter Ausdrücke, wie „wichtig“, „zusammenfassend“, „entscheidend“ oder „schlussfolgernd“, aber auch ganzer Phrasen, wie „auf den Punkt gebracht“, deuten auf wichtige Stellen in einem Text hin, und können entsprechend ausgewertet werden. Bewegt man sich nur in speziellen Domänen, so lassen sich auch Bonus- und Stigmawörter festlegen, die bestimmte Passagen als relevant, beziehungsweise uninteressant, kennzeichnen.

Ein weiterer Ansatz geht eine Ebene tiefer und befasst sich mit einzelnen Entitäten (Entitätenebenenansatz). Hierbei wird versucht mit Hilfe von Beziehungen von Ausdrücken eine gewisse Struktur in den Quelltext zu bekommen. Durch das Verbinden einzelner Entitäten entsteht so eine Graphtopologie, mit der anschließend weiter gearbeitet werden kann. Diese Beziehungen können auf unterschiedlichste Weise bestehen, beispielsweise in Hinblick auf die Ähnlichkeit von Ausdrücken. So könnte man die Ausdrücke „...des schwarzen Pferdes...“ und „Dieses Pferd...“ aufgrund ähnlicher Bestandteile als eine Entität auffassen. Der Abstand zwischen zwei in Betracht kommenden Begriffen kann auch als ein Indiz dazugenommen werden. So ist die Wahrscheinlichkeit, das es sich um die gleiche Entität handelt, größer, je näher die Ausdrücke innerhalb eines Textes zusammenstehen. Die Betrachtung des Kontextes, in dem ein Wort vorkommt, kann auch in die Berechnung mit einfließen. Stellt sich

Entitätenebene

heraus, dass innerhalb des gleichen Kontextes nur zwei Begriffe ausgetauscht werden, so besteht die Vermutung, dass es sich um nur eine Entität handeln könnte. Zur Verdeutlichung: „in der Wohnung des Mannes…“ und „in der Wohnung des Opfers..“, sind zwei Fragmente, aus denen man schlussfolgern könnte, dass der Mann das Opfer sei. Ein auf Wörterbüchern beruhendes Verfahren untersucht Synonyme, Hyperonyme oder Ist-Teil-von-Beziehungen, und versucht damit Entitäten festzustellen. Auch mit Hilfe von Koreferenzen, das heißt sich aufeinander beziehenden Ausdrücken, lassen sich sinnvolle Verbindungen zwischen Wörter aufbauen. Logische Beziehungen, wie beispielsweise innerhalb einer Kontradiktion oder einer Zustimmung können genauso zum Aufbau von Verbindungen herangezogen werden. Untersucht man die zu Grunde liegende Syntax eines Quelltextes näher, so lassen sich mit Hilfe von Syntaxbäumen grammatikalische Strukturen erkennen, die dann zur Konstruktion von Beziehungen einzelner Begriffe benutzt werden können. All diese Ansätze auf Entitäten-Ebene entwickeln eine innere Struktur durch den Aufbau von Beziehungen von Ausdrücken.

Diskursebene

Ein weiterer Ansatz betrachtet den Originaltext als Einheit, und versucht eine globale Struktur aufzubauen. Dieser Diskursebenenansatz arbeitet mit dem Textformat und gegebenenfalls metasprachlichen Konstrukten. Des Weiteren versucht man durch Identifizierung von Handlungs- beziehungsweise Erzählsträngen Grundaussagen des Textes zu erfassen. Das Verfolgen von Argumentationslinien und anderen rhetorischen Strukturen kann auch zum Erfolg führen. Diese drei unterschiedlichen Ansätze werden in der Praxis meist durch hybride Ansätze realisiert. Man versucht mit Hilfe von mehreren Verfahren die Schwächen einzelner Methoden zu kompensieren. Grundsätzlich lassen sich alle Verfahren noch hinsichtlich ihrer Präferenz von statistischen oder linguistischen Methoden unterscheiden, wobei linguistische Methoden, aufgrund ihres komplexeren Hintergrundwissens, das sie benötigen, um mit der Sprachsemantik oder ihrer Syntax arbeiten zu können, statistischen Methoden theoretisch überlegen sind. In der Praxis zeigen jedoch gerade statistische Lösungen überraschend gute Resultate. Ein Indiz dafür ist die Verwendung statistischer Methoden auf dem Gebiet der automatischen Textzusammenfassung seit nunmehr fast 50 Jahren, wobei [Luh58] als Pionierarbeit auf diesem Sektor diente.

7.1.3 Geschichtliche Entwicklung

Begonnen hat die Entwicklung von automatischen Textzusammenfassungssystemen in den späten *1950ern*, geprägt durch einen Oberflächenebenenansatz, zum Beispiel mit Hilfe von Worthäufigkeiten. In den *1960ern* kamen dann erste Ansätze auf Entitätenebene hinzu, welche mit einer syntaktischen Analyse an die Aufgabenstellung herangingen. Etwas später wurde auch die Position der Wörter innerhalb eines Textes zur Gewinnung von Informationen herangezogen. Die *1970er* brachten einen erneuten Aufschwung des Bereiches, und auf der Oberfächenebene wurde die Auswertung von Schlüsselwörtern zur Erstellung von Zusammenfassungen genutzt. Auch das erste kommerzielle System [PZ75] fällt in jene Zeit. Ende der 1970er wurde sowohl die Entitätenebene wieder weiterentwickelt, als auch die ersten Diskursebenensysteme entworfen. Die *1980er* brachten viele verschiedene neue Ansätze in die Textzusammenfassung. So wurde versucht, mit Hilfe von Künstlicher Intelligenz, Logik- und Produktionsregeln oder semantischen Netzen brauchbare Ergebnisse zu erzielen. Die Entwicklung von hybriden Ansätzen fällt auch in jene Zeit. In den *1990ern* wurden dann alle drei Ansätze weiterentwickelt, jedoch konzentrierte man sich hauptsächlich auf „Extracts“, und fand wieder mehr Gefallen an der früheren Herangehensweise auf

der Oberflächenebene. Für einen detaillierteren Einblick in die Geschichte empfehlen wir [MM99], aus dem auch dieser kurze Abriss stammt.

7.2 DUC 2004 & ERSS 2004

In diesem Abschnitt wollen wir uns zu Beginn mit der Document Understanding Conference, einer Konferenz, die das Automatische Textzusammenfassen zum Thema hat, beschäftigen. Anschließend betrachten wir ein bestimmtes Automatisches Textzusammenfassungssystem genauer, um einen Einblick in die Funktionsweise solcher Systeme zu gewinnen.

7.2.1 Die Document Understanding Conference

Die *Document Understanding Conference (DUC)* [Cona], ist eine jährlich stattfindende Konferenz, welche ein Forum für die weltweite Forschung auf dem Gebiet des automatischen Textzusammenfassens bietet. Der Fortschritt soll dokumentiert und vorangetrieben werden, wobei das Hauptaugenmerk auf dem Vergleich von bestehenden Systemen liegt. Da es eo ipso nicht möglich ist, zwei verschiedenartige Zusammenfassungssysteme zu vergleichen, stellt die DUC eine gute Plattform für gerade diese Aufgabe dar. Indem den Teilnehmern einheitliche Daten zur Verfügung gestellt werden, auf denen klare Aufgaben zu bearbeiten sind, bietet sich die Möglichkeit, die entwickelten Systeme zu testen und in Relation zu konkurrierenden Teilnehmern das eigene Leistungsvermögen festzustellen.

Entstanden ist die Konferenz im Jahre 2000. Im Jahr darauf fand sie zum erstenmal statt. Organisiert wird die Konferenz von dem *National Institute of Standards and Technology (NIST)* und gesponsert von der DARPA und ARDA, Abteilungen des US-Verteidigungsministeriums. Auch hier zeigt sich, dass das Gebiet der automatischen Textzusammenfassung keine rein akademische Spielwiese darstellt, sondern eine große praktische Relevanz besitzt, und von unterschiedlichsten Seiten Beachtung findet. Die Bewertung der Kandidatensysteme fand 2004 [Doc04] sowohl durch Menschen, mit Hilfe der *Summary Evaluation Environment*, als auch automatisch mit *ROUGE* statt. Auf diese beiden Bewertungsmethoden wird im Zusammenhang mit der Evaluierung von Systemen noch genauer einzugehen sein.

Daten und Aufgaben

2004 gab es für die Kandidaten fünf Aufgaben, wie in [PO04] beschrieben, zu bewältigen. Die Texte, die zusammengefasst werden sollten, waren in themenverwandte Gruppen eingeteilt, sogenannten Clustern.

Aufgabe 1. Die erste Aufgabe erforderte die Erzeugung einer Zusammenfassung von 75 Bytes Länge, aus einem englischsprachigen Zeitungsartikel. Für diese Aufgabe standen 50 Textsammlungen mit jeweils 10 thematisch verwanden Artikeln, aus der New York Times (NYT) und der Associated Press (AP) zur Verfügung. Es mussten also 500 Kurzzusammenfassungen pro teilnehmendem System erzeugt werden.

Aufgabe 2. Für die zweite Aufgabe wurde auf den selben Daten gearbeitet, jedoch war die Zielsetzung aus den jeweiligen 10 Dokumenten einer Sammlung, also aus

```
said Jacobs the speaking on behalf hospital May in
Rochester in Minnesota that “the treatment make as
expected . said the Jordanian monarch in the United
States, where receive treatment in a telephone call
by with him television official Jordanian yesterday
evening Friday ” with regard to the chemotherapy
ended last stage during the the first 10 days recent
and there is no impact of the disease same item . “
in his statement to the Palestinian people Jordanian
emplacement television official, he said Prince
El Hassan brother King Hussein of the smallest “
while aid to my words this be Hussein had left the
hospital ) . . ( and may recovery and discovery of
disease .
```

Abbildung 7.1: Ein automatisch übersetzter Zeitungsartikel für Aufgabe 4

Texten mit ähnlichem Inhalt, eine Zusammenfassung zu erzeugen. Diese Zusammenfassung mehrerer Dokumente zu einem Text nennt man "multi-document summary". Die Länge war hierbei auf 665 Bytes beschränkt.

Aufgabe 3 + 4. Für die Aufgaben drei und vier wurden 24 Textsammlungen mit jeweils 10 arabischen Zeitungsartikeln der Agence France Press verwendet. Warum hier gerade arabische Zeitungsartikel ausgewählt wurden, bleibt der Phantasie des Lesers zur Einschätzung überlassen. Es wurde hier allerdings nicht auf den Originalartikeln gearbeitet[1], sondern auf maschinell ins Englische Übersetzten, ein Beispiel ist in Abbildung 7.1 zu sehen. Die Übersetzung fand mit maschinellen Übersetzungssystemen von IBM und ISI statt. Bei einem zweiten Durchlauf wurde mit von Hand übersetzten Zeitungsartikeln gearbeitet; bei einem optionalen dritten Durchlauf konnten neben den automatisch übersetzten Texten noch thematisch ähnliche, englische Texte mit in die Textsammlung aufgenommen werden. Umfang der Zusammenfassungen war analog zu Aufgabe eins und zwei, also 75 Bytes für die einfachen Zeitungsartikel und 665 Bytes für die Zusammenfassung von 10 Artikeln zu einer Zusammenfassung.

Aufgabe 5. Bei der abschließenden fünften Aufgabe musste die Zusammenfassung eine Antwort auf eine Frage der Form „Wer ist X?“ geben, wobei X für eine Person, Organisation oder Gruppe von Personen stehen konnte. Dazu standen den Teilnehmern pro Frage jeweils 10 einschlägige Artikel in englischer Sprache (AP, New York Times und Xinhua News Agency) zur Verfügung. Es kamen hier keine TDT-Cluster zum Einsatz, sondern insgesamt 50 TREC Textsammlungen [Conb], das heißt auch wieder nach Themen vorsortierte Zeitungsartikel. Die Zusammenfassung, die die Antwort auf die jeweilige Frage liefern sollte, war auch hier auf 665 Bytes Länge beschränkt.

7.2.2 ERSS 2004 – Ein automatisches Textzusammenfassungssystem

Unter den 25 teilnehmenden Systemen bei DUC 2004 befand sich auch ein System, das an der Concordia University[2] entwickelt wurde. Dieses System, mit Namen *ERSS*

[1] bis auf ein System, dass auf den Originaldaten arbeitete und anschließend das Ergebnis maschinell übersetzte

[2] Montreal, Quebec, Canada, http://www.cs.concordia.ca/CLAC/

2004, dokumentiert unter anderem in [BWL+04], wird im folgenden genauer beschrieben. ERSS steht für Experimental Resolution System Summarizer und die grundlegende Idee ist, einen indikatorischen Textauszug zu generieren, wobei das System entsprechend der unterschiedlichen Aufgabenstellung parametrisiert wurde. In leichter Abwandlung kam für die Aufgaben 2, 4 und 5, also den Multi-Dokument-Zusammenfassungen, Multi-ERSS zum Einsatz. Des Weiteren werden wir noch an geeigneter Stelle auf das Vorgängersystem ERSS [BWK+03] eingehen, das an der Document Understanding Conference 2003 teilnahm. ERSS, dessen Weiterentwicklung ERSS 2004 darstellt, basiert genau wie dieses auf der GATE-Architektur [CMBT02], und benutzt einige ANNIE-Komponenten, die von GATE zur Verfügung gestellt werden. Der Aufbau des Systems lässt sich grob in vier Teilkomponenten untergliedern:

NPE: Ein Noun Phrase Extractor

Fuzzy-ERS: Ein Koreferenzkettenberechner, der mit Fuzzy-Theorie arbeitet

Klassifizierer: Ein bayes'scher Textklassifizierer

ERSS: Das Zusammenfassungssystem

Den Kern stellt der Fuzzy-Koreferenzkettenberechner, englisch *fuzzy coreferencer*, nach [Wit02, WB03] dar. Details hierzu lassen sich im Kapitel 5 ab Seite 83 nachlesen. Mit seiner Hilfe lässt sich die Wichtigkeit von Entitäten für einen Text abschätzen, indem Koreferenzketten mit Hilfe von ein paar Heuristiken, u.a. durch Nutzung von WordNet [Fel98], aufbaut werden. Anhand deren Länge kann die Wichtigkeit der enthaltenen Referenz abgeschätzt werden. Da solche Heuristiken nicht absolut verläßlich sind, werden die Unsicherheiten, die damit grundsätzlich im System enthalten sind, durch einen Fuzzy-Ansatz direkt zugänglich gemacht. Der Vorteil eines solchen Vorgehens besteht in der unmittelbaren Einstellmöglichkeit des Grenzwertes. Dadurch lässt sich eine großzügigere oder striktere Auslegung für die Zugehörigkeit einer Entität zu einer Kette einstellen, so dass auch unterschiedliche Anforderungen mit Hilfe unterschiedlicher Schwellwerte erfüllt werden können.

Grenzwert

7.2.3 Die Komponenten von ERSS 2004

In der ersten Phase der Verarbeitung muss die entsprechende Vorverarbeitung eines Eingabetextes stattfinden. Der *Noun Phrase Extractor* (NPE) wird anschließend mit einem durch einen Wortartmarkierer (POS-Tagger) annotierten Text gefüttert, und liefert die einzelnen Noun Phrases. Diese werden dann dem Fuzzy-Koreferenzkettenberechner zugeführt. Der bei DUC 2004 nicht mehr eingesetzte *bayes'sche Klassifizierer* (Classifier) wurde 2003 zur Textklassifikation benutzt. Aufgrund des Bewertungsalgorithmus bei DUC 2004 wäre ein erneuter Einsatz dem Ergebnis abträglich gewesen. Die vierte und letzte Phase ist bestimmt durch den eigentlichen *Summarizer*. Er generiert die Ausgabe, indem er entsprechend den Aufgaben die als wichtig erkannten Teile aus dem Text extrahiert. In den folgenden Abschnittenen besprechen wir die einzelnen Komponenten im Detail.

Der Noun Phrase Extractor (NPE)

Benannte Entitäten, *Named Entities* (NE), wie Personen, Organisationen, Datumsangaben oder Länder, werden mit Hilfe von Wortlisten und einer Reihe von Grammatiken

identifiziert. Der eigentliche Noun Phrase Extractor benutzt eine kontextfreie Grammatik und eine auf dem Earley-Algorithmus basierende Syntaxanalyse. Das Ergebnis sind minimale Nominalphrasen (Noun Phrases), das heißt, Entitätenbezeichnungen ohne Relativsätze oder Appositionen (siehe Abbildung 7.2). Er arbeitet hauptsächlich

Nominalphrasen

```
Marilyn Monroe, then Norma Jean Baker, was
discovered working at an aircraft factory during
the second world war.
```

Abbildung 7.2: Das Ergebnis des Noun Phrase Extractors

auf den erkannten benannten Entitäten, nur falls eine nicht erkannte Entität vorliegt greift er auf Part-of-Speech-Tags zurück. Für die Aufgabe fünf bei DUC 2004 wurde als weiterer Schritt versucht, die Nominalphrasen auf maximale Länge zu erweitern, so dass durch das Hinzunehmen von Appositionen und Präpositionen ein Informationsgewinn erzielt werden konnte.

Der Fuzzy Coreferencer

Koreferenzketten

Der Fuzzy Coreferencer ordnet die extrahierten Nominalphrasen in Koreferenzketten an, also Gruppen von Nominalphrasen, die sich auf die selbe Entität beziehen. Es wird ein Fuzzy-Algorithmus benutzt, um auf die bei der Referenzauflösung entstehende Unsicherheit direkt Einfluss nehmen zu können. Der Vorteil einer Fuzzy-Auswertung liegt nicht nur in der Möglichkeit, einen Grenzwert vorher nicht festlegen zu müssen, für den eine Nominalphrase zu einer bestimmten Kette als zugehörig angenommen wird, sondern auch in der Möglichkeit durch Variieren des Grenzwertes eine strengere oder nachsichtigere Zuordnung von Nominalphrasen zu Ketten zu erreichen (graphisch dargestellt in Abbildung 7.3). Dies führt je nachdem zu besseren Recall- oder Precision-Werten. Ein strengerer Grenzwert führt dabei zu einem Aufbrechen von Ketten in mehre Teilketten mit höherem Precision-Wert, was aber bei automatischen Textzusammenfassungen zu schlechteren Gesamtergebnissen führt, da die Länge der Referenzkette ausschlaggebend für deren Repräsentation in der finalen Zusammenfassung ist.

Für die Zusammenfassung mehrerer Dokumente ist das *Multi-ERSS* zuständig. Es baut nur Koreferenzketten auf, die aus Nominalphrasen bestehen, die unterschiedlichen Dokumenten angehören. In einer Kette können also nicht zwei Nominalphrasen

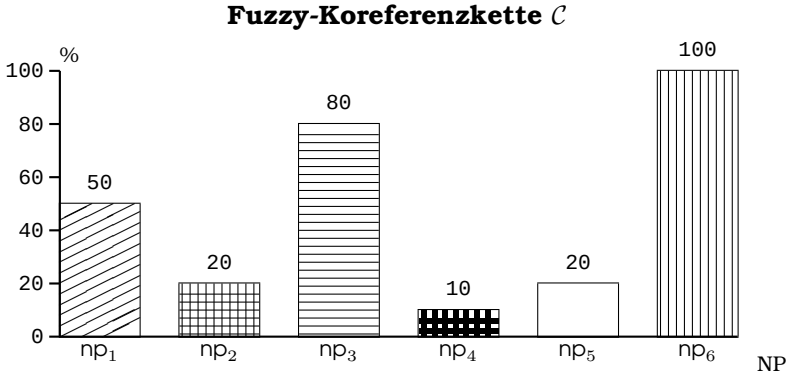


Abbildung 7.3: Ein Beispiel für die Zuordnung von Nominalphrasen zu einer Fuzzy-Koreferenzkette

aus einem Text enthalten sein. Da der gleiche Koreferenzalgorithmus wie bei ERSS 2004 benutzt wird, kann nur eine Teilmenge der Heuristiken für den Aufbau dieser Inter-Dokument-Koreferenzketten benutzt werden. Es liegt auf der Hand, dass zum Beispiel die Pronomenauflösung nicht sinnvoll ist. Das Ergebnis dieser zweiten Phase ist jedoch bei ERSS 2004 und Multi-ERSS prinzipiell das Gleiche: Verschiedene Koreferenzketten mit Nominalphrasen, die sich auf die selbe Entität beziehen, und deren Länge mit unterschiedlich justiertem Grenzwert beeinflusst werden kann.

Der Classifier

Der Classifier wurde in ERSS 2004 und Multi-ERSS zwar nicht mehr benutzt, in der Vorgängerversion war er jedoch Bestandteil des Zusammenfassungssystems. Aufgrund der Anforderungen, die die Document Understanding Conference an ihre Teilnehmer stellte, hätte die Textklassifizierung innerhalb der Zusammenfassung zu weniger guten Bewertungen geführt. Dies ist darauf zurückzuführen, dass extra-textuelles Material bei dem Bewertungsalgorithmus zu niedrigeren Bewertungen führt. Da diese Klassifizierung jedoch gerade bei einer indikativen Zusammenfassung dem Leser zusätzliche Informationen bietet, werden wir diese Komponente in ihrer Funktion kurz beschreiben.

Der Classifier ist ein naiver bayes'scher Klassifizierer, der mit Hilfe von thematisch eng begrenzten, kleinen Ontologien trainiert wurde. Implementiert wurde er mit dem Bow toolkit [Kha04]. Jede der Ontologien konzentriert sich auf eine Textkategorie, wie Sportereignisse oder Naturkatastrophen. Damit lässt sich anschließend eine Klassifizierung des Quelltextes vornehmen. Verwendet man beispielsweise drei Ontologien im Trainingsprozess, so lässt sich ein Zeitungsartikel in eine der drei Kategorien einordnen. Das Ergebnis in diesem Fall wäre ein Punkt im dreidimensionalen Raum. Abbildung 7.4 zeigt eine Zeitungsartikel, und 7.5 die zugehörige Zusammenfassung aus DUC 2003 einschließlich der Klassifizierung des Textes.

Klassifizierung

```
HOUSTON - The Hubble Space Telescope got smarter
and better able to point at distant astronomical
targets on Thursday as spacewalking astronauts
replaced two major pieces of the observatory's
gear. On the second spacewalk of the shuttle
Discovery's Hubble repair mission, the astronauts,
C. Michael Foale and Claude Nicollier, swapped
out the observatory's central computer and one of
its fine guidance sensors, a precision pointing
device. The spacewalkers ventured into Discovery's
cargo bay, where Hubble towers almost four stories
above, at 2:06 p.m. EST, about 45 minutes earlier
than scheduled, to get a jump on their busy day of
replacing some of the telescope's most important
components. ...
```

Abbildung 7.4: Ein Zeitungsartikel über das Hubble Space Telescope

Der Summarizer

Der Summarizer bietet durch verschiede Konfigurationseinstellungen die Möglichkeit, unterschiedliche Strategien bei der Umsetzung der verschiedenen Aufgaben einzu-

```
Space News: [the shuttle Discovery's Hubble repair
mission, the observatory's central computer]
```

Abbildung 7.5: Eine von ERSS 2003 erzeugte Zusammenfassung des Textes 7.4 mit einer Klassifizierung des Textes durch die Classifier-Komponente

setzen. Grundsätzlich bestehen die erzeugten Zusammenfassungen aus extrahierten Textpassagen des Quelltextes. Diese Teile bestehen ihrerseits aus Textelementen wie Nominalphrasen oder Sätzen, die zuvor aufgrund strategiespezifischer Merkmale Punkte erhielten. Die Punktbesten werden selektiert und in der Ausgangsreihenfolge ausgegeben, bis die geforderte Länge erreicht ist. Betrachten wir nun die genauen Konfigurationen für die unterschiedlichen Aufgaben, die bei DUC 2004 bearbeitet werden mussten.

```
Castro; London; dictator Augusto Pinochet;
Pinochet's arrest; Ibero-America
```

Abbildung 7.6: Eine von ERSS 2004 erzeugte Zusammenfassung für die Aufgabe 1

Summarizer: Aufgabe 1 & 3. Die Aufgaben 1 und 3 wurden mit der gleichen Strategie gelöst, da es sich jeweils um einzelne Zeitungsartikel handelte, die zusammengefasst werden sollten. Der Unterschied ist, dass bei der ersten auf englischen Originalartikeln und bei der dritten auf maschinell übersetzten Zeitungsartikeln gearbeitet wurde. Der Summarizer bewertet alle Nominalphrasen *entsprechend der Koreferenzketten*, in denen sie enthalten sind. Je länger die Kette ist, der eine Nominalphrase angehört, desto höher die Punktzahl für die Nominalphrase. Als zweites Kriterium werden noch Punkte an Nominalphrasen verteilt, die in den *ersten zwei Sätzen* eines Quelltextes vorkommen. Beide Faktoren werden gleich stark gewichtet. Als Ergebnis erhält man nun die punktbesten Nominalphrasen, die von Artikeln und Satzzeichen befreit wurden. Des Weiteren werden eventuell vorhandene redundante Nominalphrasen eliminiert. Als letzter Schritt erfolgt die Ausgabe, bis die geforderte Länge erreicht ist. Aufgrund der bei Aufgabe eins und drei geforderten Länge von 75 Zeichen entspricht die erzeugte Zusammenfassung einfach einer Aneinanderreihung von Nominalphrasen, stellt also eher eine Index oder eine Liste von Schlüsselwörtern dar und hat nicht die Form einer wohlgeformten Zeitungsüberschrift. Dennoch sind die Ergebnisse von Aufgabe eins in der Lage, ihrer indikativen Funktion nachzukommen, wie beispielhaft in Abbildung 7.6 zu sehen ist. In beschränktem Umfang gilt dies auch für Aufgabe 3, die Ergebnisse sind hier aber aufgrund schlechter Übersetzungen nicht so gut. Abbildung 7.7 gibt hier einen Einblick.

```
King Hussein; hospital; treatment; statement; end;
Jordanian monarch; cancer
```

Abbildung 7.7: Die Zusammenfassung des Textes aus Abbildung 7.1

Summarizer: Aufgabe 2 & 4. Für die Aufgaben zwei und vier kam nun *Multi-ERSS* zum Einsatz. Beim Aufbau von Koreferenzketten über Dokumentgrenzen hinweg ist es nicht sinnvoll, Heuristiken, die Pronomen auflösen zu verwenden. Ebenso verhält es sich mit dem Ausfindigmachen von Synonymen oder Hyperonymen. Daher wird

für die Berechnung von Koreferenzketten mehrerer Dokumente nur eine Teilmenge der Heuristiken von ERSS 2004 benutzt. Die für diese beiden Aufgaben vorgegeben Länge beträgt 665 Zeichen. somit werden hier nicht nur die einzelnen punktbesten Nominalphrasen wiedergegeben, sondern die *kompletten Sätze*, in denen die punktbesten Nominalphrasen vorkommen. Auch die Bewertung der Nominalphrasen unterscheidet sich von der Einzeltextzusammenfassung. Eine Nominalphrase erhält Punkte entsprechend der *Länge der Inter-Dokument Referenzkette*, der sie angehört, und entsprechend ihrer *eigenen Länge*. Es lassen sich hier noch weitere Einschränkungen vornehmen, wie beispielsweise Zitate unberücksichtigt zulassen. Genauso ist es sinnvoll Wiederholungen zu vermeiden. Die Ausgabe der gefundenen Sätze erfolgt in sortierter Reihenfolge, so dass die Reihenfolge, wie die Originaltexte eingelesen wurden beibehalten wird. Ein Ergebnis ist in Abbildung 7.8 zu sehen. Eine Nachbearbeitung fand hier aufgrund der begrenzter Entwicklungszeit nicht statt.

```
President Yoweri Museveni insists they will remain
there until Ugandan security is guaranteed, despite
Congolese President Laurent Kabila's protests that
Uganda is backing Congolese rebels attempting to
topple him. After a day of fighting, Congolese
rebels said Sunday they had entered Kindu, the
strategic town and airbase in eastern Congo used
by the government to halt their advances. The rebels
accuse Kabila of betraying the eight-month rebellion
that brought him to power in May 1997 through
mismanagement and creating divisions among Congo's
400 tribes. A day after shooting down a jetliner
carrying 40 people, rebels clashed with government
troops near a strategic airstrip in eastern Congo on
Sunday.
```

Abbildung 7.8: Eine von Multi-ERSS erzeugte Zusammenfassung mehrere Texte für Aufgabe 4

Summarizer: Aufgabe 5. Die 5. Aufgabe erforderte eine Zusammenfassung von 10 Texten zu erstellen, die auf eine Frage der Form „Wer ist ...?“ eine Antwort liefert. Diese Zusammenfassungen werden im Prinzip wie die Multi-Dokument Zusammenfassungen erzeugt, mit dem Unterschied, dass die Anfrage als weiteres Dokument mit in die Textsammlung aufgenommen wird. Als relevante Sätze werden nur solche angesehen, die eine Nominalphrase enthalten, die Teil einer Koreferenzkette ist, welche auch eine Nominalphrase aus der Frage enthält. Die Punkte, die bei dieser Aufgabe den Nominalphrasen zugeteilt werden, werden wieder durch die *Länge der Referenzkette* und die *Länge der Nominalphrase* selbst bestimmt. Hier liegt die Gewichtung jeweils bei 1.0. Als weiterer Faktor fließt mit einer Gewichtung von 3.0 die Position der Nominalphrase innerhalb des Textes mit ein. Das heißt, eine Nominalphrase, die Teil einer *Apposition* ist, bekommt zusätzliche Punkte. Dies trägt der Tatsache Rechnung, dass in Appositionen oft wesentliche Aussagen gemacht werden, die im Hinblick auf eine Entität wichtige Informationen beinhalten (Abbildung 7.9). Das von Multi-ERSS erzeugte Endergebnis zu einer Frage von Aufgabe 5 sehen wir in Abbildung 7.10.

Apposition

```
Marilyn Monroe, then Norma Jean Baker, was
discovered working at an aircraft factory during
the second world war.
```

Abbildung 7.9: Eine Apposition, die zusätzliche Informationen über eine Entität beinhaltet

7.3 Vergleich und Evaluierung verschiedener Textzusammenfassungssysteme

Im nun folgenden Teil gehen wir von der automatischen Erzeugung von Textzusammenfassungen über zur prinzipiellen Bewertung unterschiedlicher Textzusammenfassungen und der sie erzeugenden Systeme.

7.3.1 Wie vergleicht man Zusammenfassungen?

Ein wichtiges Gebiet, das unmittelbar an die Entwicklung automatischer Zusammenfassungssysteme anschließt, ist die Evaluierung solcher Systeme. Grundsätzlich gibt es zwei Methoden (siehe [Man01b]), mit deren Hilfe man Zusammenfassungen vergleichen kann. Die intrinsische Methode bewertet den erzeugten Text entweder nur im Hinblick auf den erzeugten Text oder im Vergleich mit anderen Texten. Dies können Referenzzusammenfassung sein oder der Quelltext, aus dem die Zusammenfassung erzeugt wurde. Die Aussagekraft der ersteren beschränkt sich auf grammatikalische, bzw. formale Aspekte. Die beide letzteren intrinsischen Ansätze sind aufschlussreicher, haben aber auch Schwächen: Die Evaluierung mit dem Quelltext ist problematisch, da per se noch nicht bekannt ist, welche Teile oder Wörter besonders informativ und den Text gut beschreibend sind. Es muss also noch eine Annotation von Menschen durchgeführt werden, die die wesentlichen Teile der Quelle kennzeichnet. Der Vergleich mit Referenzzusammenfassungen setzt voraus, dass diese von Menschen erzeugt wurden. Hierbei ist es sinnvoll, nicht nur mit einem Referenztext zu vergleichen, sondern mit möglichst vielen. Da es mehrere gute, unterschiedliche Zusammenfassungen eines Textes gibt, sollte die Bewertung eines Testkandidaten nicht nur von einer mehr oder

Intrinsisch

```
Hawking, 56, is the Lucasian Professor of
Mathematics at Cambridge, a post once held by Sir
Isaac Newton. Hawking, 56, suffers from Lou Gehrig's
Disease, which affects his motor skills, and speaks
by touching a computer screen that translates his
words through an electronic synthesizers. Stephen
Hawking, the Cambridge University physicist, is
renowned for his brains. Hawking, a professor of
physics an mathematics at Cambridge University in
England, has gained immense celebrity, written a
best-selling book, fathered three children, and done
a huge amount for the public image of disability.
Hawking, Mr. Big Bang Theory, has devoted his life
to solving the mystery of how the universe started
and where it's headed.
```

Abbildung 7.10: Eine von Multi-ERSS erzeugte Zusammenfassung von 10 Texten für Aufgabe 5 auf die Frage: „Who is Stephen Hawking?“

weniger subjektiven Referenzzusammenfassung abhängen. Da es jedoch immer nur eine begrenzte Anzahl von Referenztexten gibt, besteht auch immer die Möglichkeit, dass eine sehr gute, informative und kohärente, erzeugte Zusammenfassung nicht mit den Referenzen übereinstimmt, und somit schlecht bewertet wird.

Die zweite Methode, um verschiedene Zusammenfassungen zu bewerten und zu vergleichen ist die extrinsische. Hier ist die Idee, die erzeugte Zusammenfassung nicht mit den Quellen oder mit Referenzen zu vergleichen, sondern in Hinblick auf eine externe Fragestellung. Der erzeugte Text wird daraufhin bewertet, wie gut er zu der Lösung dieser Frage beiträgt. Denkbar wäre beispielsweise eine konkrete Frage über den Quelltext, oder aber auch die Aufgabe, den Quelltext anhand der erzeugten Zusammenfassung einem Themenkomplex zuzuordnen. Ein weiterer Ansatz stellt Verständnisfragen zu dem Quelltext, die zum Einen von Personen beantwortet werden, die den Quelltext kennen, und zum Anderen von Personen, die nur die Zusammenfassung gelesen haben. Als weitere Testgruppe nimmt man noch Personen, die weder den einen noch den anderen Text kennen, und vergleicht anschließend die Antworten aller drei Personengruppen.

Extrinsisch

Qualitätsmerkmale von Zusammenfassungen. Was eine gute Zusammenfassung noch ausmacht, unabhängig von deren Inhalt, sind Faktoren wie *Kohäsion*, *Kohärenz* oder *Kompression* (siehe [Man01a]). Die Kohäsion beschreibt den Grad des Zusammenhaltens unterschiedlicher Sätze mit sprachlichen Mitteln wie Konjunktionen, Koreferenzen oder Substitutionen. Die Kohärenz dagegen entsteht durch inhaltliche Verknüpfung der Sätze. Eine wahllose Aneinanderreihung von Sätzen bildet noch keinen Text, dazu gehören sowohl Kohäsion, als auch Kohärenz. Die Kompression bestimmt den Grad der Reduktion eines Quelltextes, und damit wie detailiert eine Zusammenfassung ist. Eine weiterführende Einführung in linguistische Merkmale findet man in [Cry87]. Ebenso stellt sich die Frage, wie vollständig eine Zusammenfassung sein sollte. Informationsgehalt und Genauigkeit sind weitere Kriterien, die man bei einer Zusammenfassung berücksichtigen kann. Es ist klar, das unterschiedliche Typen von Zusammenfassungen unterschiedliche Anforderungen an eine Bewertung stellen. Ein Abstract ist von Natur aus kohärenter als ein Extract, und eine indikative Zusammenfassung erhebt gar nicht den Anspruch auf Vollständigkeit. Man muss an dieser Stelle abwägen, was für Arten von Zusammenfassungen man vergleicht und auf welche, der sich zum Teil gegenseitig ausschließenden Kriterien man besonderen Wert legen möchte.

Kohäsion
Kohärenz
Kompression

7.3.2 Manuelles Vergleichen

Die Herausforderung des Evaluierens unterschiedlicher Zusammenfassungen ist am besten zu begegnen, indem man Personen bittet, über die Güte von Kandidatenzusammenfassungen zu entscheiden. Man braucht aber zumindest noch Referenzzusammenfassungen, damit die Personen eine Vorstellung bekommen, wie eine mehr oder weniger optimale Zusammenfassung in dem betrachteten Fall auszusehen hat, um eine graduelle Einteilung der Kandidaten vornehmen zu können. Zuvor sind von geübten Zusammenfassungsschreibern möglichst viele Referenztexte anzufertigen, um die subjektive Sicht eines einzigen Autors zu relativieren. Damit man einheitliche und möglichst reproduzierbare Ergebnisse erhält, muss der Vorgang des Vergleichens der Zusammenfassungen mit den Referenzen möglichst standardisiert ablaufen. Als Beispiel stellen wir das neben der automatischen Evaluierung bei DUC 2004 verwendete

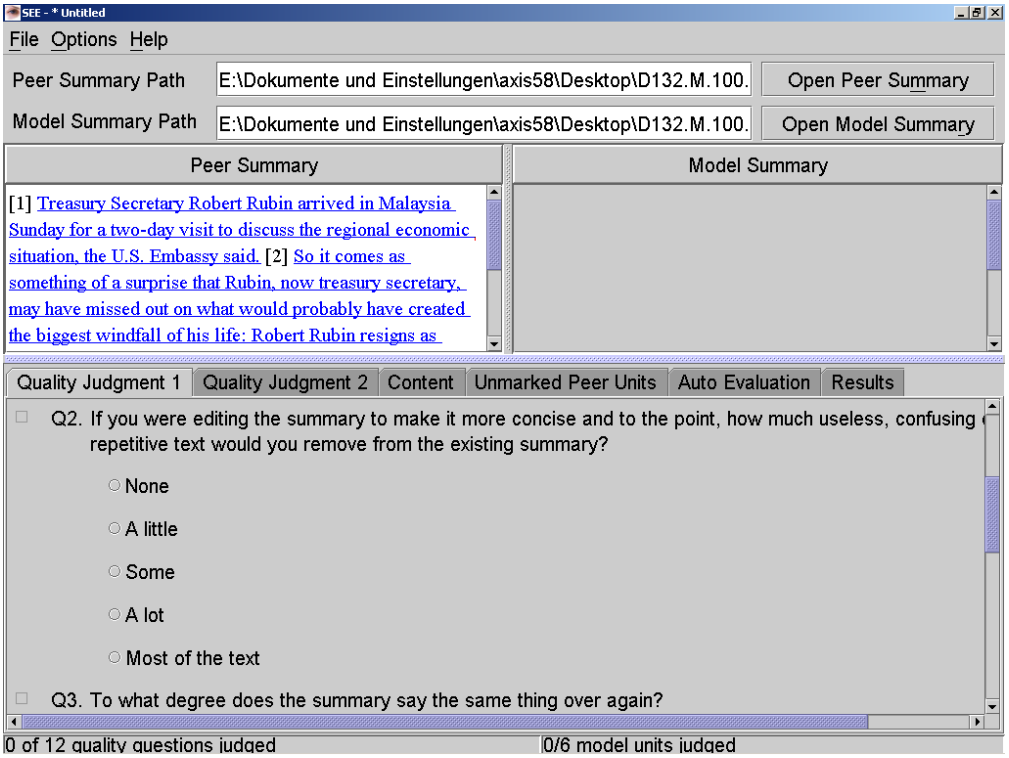


Abbildung 7.11: Die SEE Qualitätsbewertung

Summary Evaluation Environment (SEE)

SEE vor. SEE steht für Summary Evaluation Environment, und bietet eine Benutzeroberfläche, die es ermöglicht, zwei Zusammenfassungen zu vergleichen. Mit Hilfe von Multiple-Choice-Fragen wird eine Person durch den Bewertungsprozess geführt. Dabei wird auf Kohärenz, Wiederholungen, Vollständigkeit usw. eingegangen. Abbildung 7.11 zeigt die Benutzeroberfläche des Bewertungsprogramms. Auch die einzelnen Sätze werden mit den jeweiligen Sätzen der Referenz verglichen, und der Übereinstimmungsgrad auf einer Skala von 0–100 Prozent notiert (siehe Abbildung 7.12).

Das alles ist mit *hohem Personalaufwand* verbunden. Bei der DUC 2002 [Doc02], waren ungefähr 7000 Vergleiche durchzuführen. Doch neben den Kosten, die eine manuelle Evaluierung verursacht, gibt es noch andere Faktoren, die ein automatisches Vergleichen attraktiv machen. *Reproduzierbare Ergebnisse* sind wünschenswert, aber noch wichtiger ist eine ständige *Verfügbarkeit*. Gerade beim Entwickeln und Testen eines Zusammenfassungssystems ist man auf eine schnelle Überprüfung der Leistung angewiesen. Justiert man einen Parameter neu, möchte man eine unmittelbare Rückmeldung, ob sich das System verbessert oder verschlechtert hat. Des Weiteren möchte man unabhängig sein von subjektiven Vorlieben einer oder mehrerer Personen. Die Schwierigkeiten liegen hier in der Umsetzung dieser Anforderungen.

7.3.3 Automatisches Vergleichen

Bittet man mehrere Menschen, einen Text zusammenzufassen, so kommen mit hoher Wahrscheinlichkeit sehr unterschiedliche Ergebnisse zustande, je nach Vorwissen,

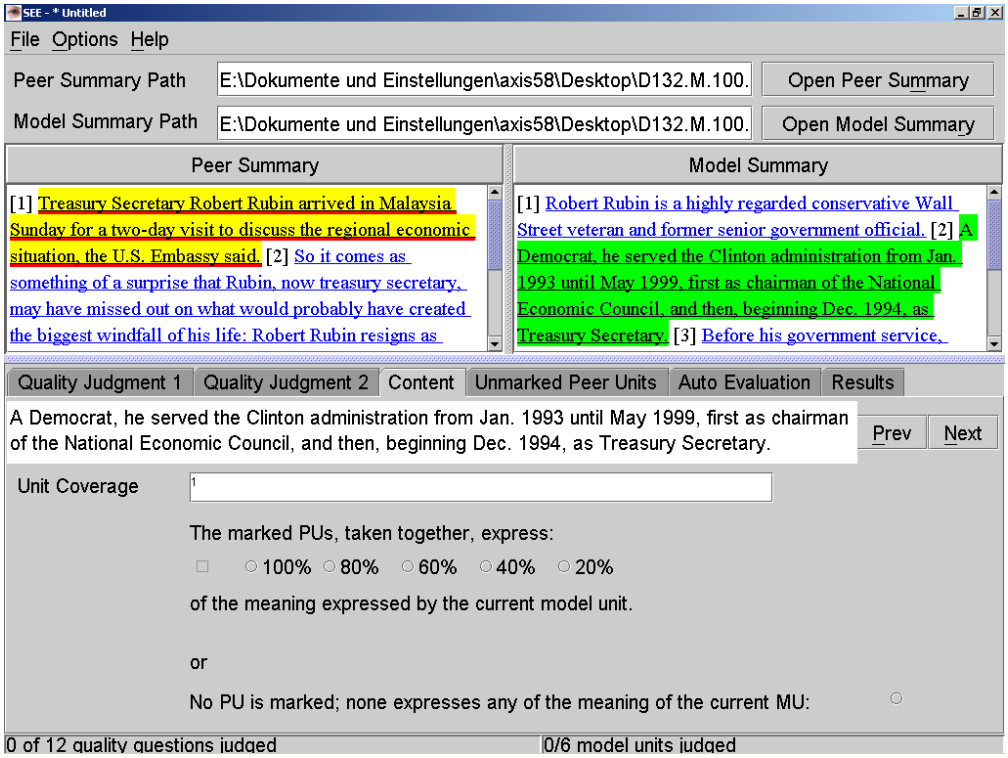


Abbildung 7.12: Die SEE Qualitätsbewertung

Kenntnissen über Hintergründe, Bildungsgrad oder Vorstellungen über eine Zusammenfassung. Zwei Zusammenfassungen können inhaltlich übereinstimmen, aber der Form nach unterschiedlich sein. Ebenso gibt es gute Schreibstile und weniger gute, genau wie unterschiedliche Gliederungen. Man muss sich also als erstes Gedanken machen, was man automatisch vergleichen will, bevor man an die Umsetzung geht. Ein allgemeines Maß für die Güte einer Zusammenfassung ist noch nicht gefunden worden, denn es gibt unterschiedliche Ansichten, was eine gute Zusammenfassung ausmacht. Eine weitere Schwierigkeit ergibt sich daraus, dass es noch kein perfektes Zusammenfassungssystem gibt. Hätte man ein solches, könnte man dieses zur Qualitätsbestimmung von erzeugten Zusammenfassungen heranziehen. Von der anderen Seite aus betrachtet ergibt sich ein analoges Bild. Hätten wir ein Evaluierungssystem, das perfekte Ergebnisse lieferte, also genau die Stärken und Schwächen einer Zusammenfassung erkennen könnte, dann hätten wir das entscheidende Mittel in der Hand, selbst gute Zusammenfassungen zu generieren. Die Zusammenfassungsevaluierung und die Zusammenfassungserstellung sind zwei voneinander abhängige Gebiete, deren Betrachtung losgelöst voneinander wenig sinnvoll ist.

Korrelation von Bewertungen. Aus der Tatsache, dass es noch kein perfektes Evaluierungssystem gibt, erwächst der Wunsch, unterschiedliche Evaluierungsmethoden miteinander vergleichen zu können. Dafür bietet sich an, die Ergebnisse der Evaluierungssysteme mit Evaluierungen von Menschen zu vergleichen. Je stärker die Ergebnisse mit denen von Menschen korrelieren, desto geeigneter ist eine Methode, und desto

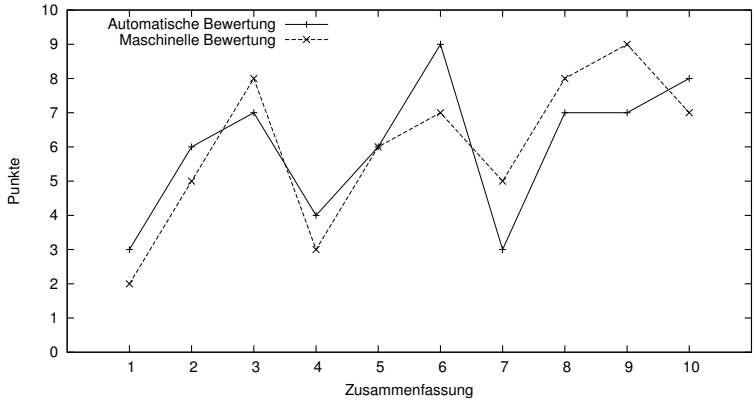


Abbildung 7.13: Ein Beispiel für die Korrelation zweier Bewertungen

eher ist es sinnvoll, verschiedene Zusammenfassungen von einem automatischen System evaluieren zu lassen. Dabei soll die Korrelation von einer automatisch erzeugten Bewertung mit einer von Menschen aufgestellten Bewertung positiv und konsistent sein. Das heißt, die automatische Bewertung sollte immer genau dann einen hohen Wert haben, wenn die menschliche Bewertung einen hohen Wert aufzeigt. Es gibt nun verschiedene Metriken, die den Korrelationsgrad von zwei Ergebniskurven berechnen.

Spearman-Metrik

Ein wichtiges Maß stellt die Spearman-Metrik dar:

$$p = 1 - 6 \sum_{i=1}^{n} \frac{d_i^2}{[n(n^2-1)]}$$

Mit ihrer Hilfe lässt sich die Übereinstimmung von menschlichen mit automatisch generierten Bewertungen bestimmen und auf einen Wert zwischen 0,0 und 1,0 abbilden. Für das Beispiel in Abbildung 7.13 ergibt sich ein Wert von 0,89, was recht nahe an die 1,0 heranreicht, und somit eine starke Korrelation der beiden Bewertungen ausweißt. Aber schauen wir uns nun eine konkrete Evaluierungsmethode an.

7.3.4 ROUGE

Das bei DUC 2004 eingesetzte ROUGE [Lin03] steht für *Recall-Oriented Understudy for Gisting Evaluation* und bietet eine Möglichkeit, Zusammenfassungen mit Hilfe von Referenzzusammenfassungen zu bewerten. ROUGE baut dabei auf einer *Recall basierten Metrik* auf und bietet die Möglichkeit, die Schnittmenge von Kandidatentexten und Referenztexten zu bestimmen:

$$ROUGE_n = \frac{\sum_{C \in ModelUnits} \sum_{n-gram \in C} Count_{match}(n-gram)}{\sum_{C \in ModelUnits} \sum_{n-gram \in C} Count(n-gram)}$$

$ROUGE_1$
$ROUGE_2$

Die praktische Anwendung hat gezeigt, das die besten Ergebnisse zustande kommen, wenn man als zu vergleichende Einheiten Unigramme, oder Bigramme zu Grunde legt. Prinzipiell lässt sich auch mit n-Grammen mit n>2 arbeiten. Die gewünschte Korrelation mit menschlichen Bewertungen leidet jedoch darunter.

Eine hoher Spearman-Wert im Vergleich zu menschlichen Bewertungen spricht für ein System wie ROUGE. So erreichte $ROUGE_1$ bei DUC 2001 [Doc01] einen Spearman-

Wert zwischen 0,879 und 0,993 für die Bewertung von Multi-Dokumenten-Zusammenfassungen. Weitere Zahlen sind zu entnehmen aus [Lin03]. An Hand des in Abbildung 7.14 gezeigten Beispiels werden jedoch auch gleich die Schwächen eines auf Statistik basierenden Systems sichtbar. Da die Metrik keinerlei *linguistische Kompetenz* besitzt

Referenz: „Anschließend verließ der Dieb
langsam und leise das Haus“

Kandidat1:„dannach Dieb Haus ruhig verlassen“
Kandidat2:„das Fenster anschließend leise und“

$ROUGE_{1-gram}$(Kandidat1): $2/9$
$ROUGE_{1-gram}$(Kandidat2): $4/9$

Abbildung 7.14: Beispiel zur Berechnung eines ROUGE Scores

wird auf den Inhalt der Texte nicht eingegangen, lediglich die Übereinstimmung von Wörtern wird gezählt. Für die Bewertung von $ROUGE_1$ bedeutet das zum Beispiel, das es keinen Unterschied macht, ob die Wörter in einem sinnvollen Satz angeordnet sind, oder einfach alphabetisch sortiert wurden. Für die Brauchbarkeit einer solchen Zusammenfassung, und natürlich auch für deren Qualität, macht es jedoch einen großen Unterschied. Um einigermaßen repräsentative Ergebnisse zu erzielen ist es auch hier nicht ausreichend, nur einen Referenztext zu verwenden. Bei DUC 2004 wurden vier benutzt, wobei die Gefahr trotzdem besteht, das eine Kandidatenzusammenfassung sehr gut in allen Belangen ist, die Wortwahl der Referenzzusammenfassungen jedoch keine Schnittmenge mit dem Kandidatentext aufweißt. Das Ergebnis ist in Folge dessen ein niedriger ROUGE Wert, im Gegensatz zu einem hohen Wert von einem System, das lediglich die ersten Sätze des Quelltextes wiederholt, und somit überhaupt keine Zusammenfassung des kompletten Quelltextes darstellt. Um dies zu verdeutlichen, hier die Ergebnisse der ROUGE-Evaluierung bei DUC 2004 für Aufgabe 2 (siehe Abbildung 7.15). Der Baseline-Algorithmus macht genau die oben beschriebenen Schritte, er nimmt die ersten 665 Bytes des ersten Textes, und gibt diese unverändert aus. Als weitere Orientierungspunkte wurden neben den automatisch generierten auch noch von Menschen erstellte Zusammenfassungen durch ROUGE bewertet. Auch hier fielen die Ergebnisse nicht sehr befriedigend aus, da die von Menschen erstellten Texte nur unwesentlich besser bewertet wurden, obgleich sie eine höhere Qualität aufwiesen.

Verwendung von ROUGE

Neben den erwähnten Nachteilen ist vor allem die undifferenzierte Bewertung ein Problem. Innerhalb der zu testenden Systeme sind die Bewertungen nur marginal unterscheidbar, und ROUGE zufolge werden alle Systeme ungefähr gleichgut eingestuft. Wichtig wäre beispielsweise die Gewichtung von einzelnen Wörtern zu verändern, so dass irrelevante Beiträge nicht genauso hoch bewertet werden wie wichtige Schlüsselwörter. Der rein statistische Ansatz schließt jedoch so etwas aus, und so wird auch die Wohlgeformtheit von Sätzen keine Rolle spielen können. Einen weiteren Nachteil stellt der mangelnde Einfluss der Länge der Zusammenfassung dar. Intuitiv würde man sagen, dass eine kurze Zusammenfassung, die den gleichen Informationsgehalt hat wie eine längere, eine höhere Bewertung verdienen müsste. Dieses Kriterium spielte bei

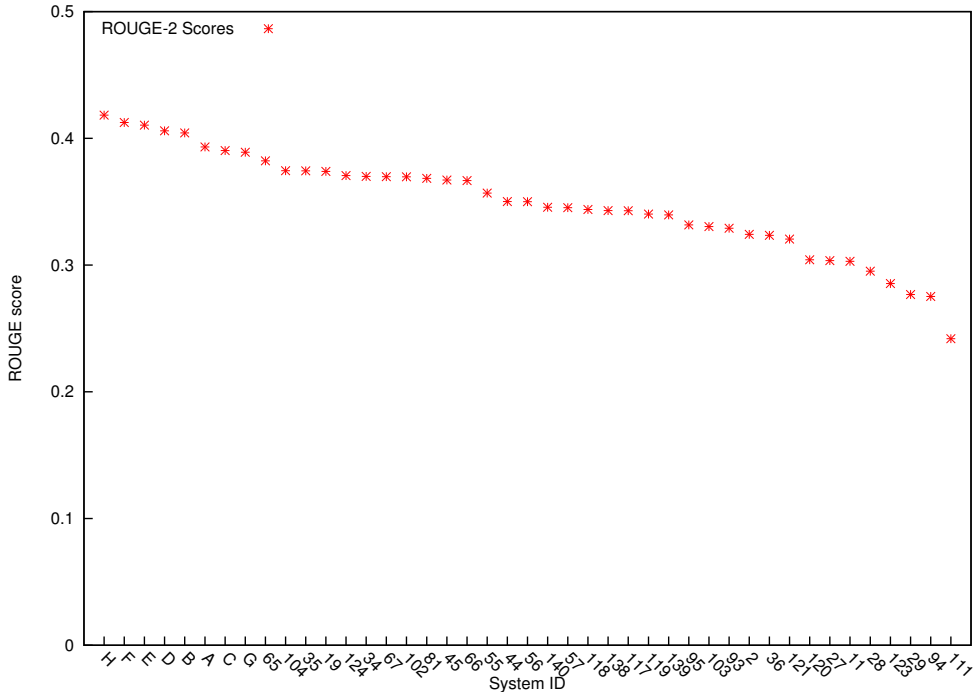


Abbildung 7.15: Die Rouge-2 Scores von Aufgabe 2 bei DUC 2004

DUC aufgrund der festgeschriebenen Länge jedoch keine Rolle, wäre für andere Aufgaben sicherlich sinnvoll. Als positive Anmerkung sei noch erwähnt, dass eine Metrik während der Entwicklung von automatischen Zusammenfassungssystemen eine tragende Rolle spielen kann. Gerade in der Phase der Parameterjustierung möchte man schnell wissen, ob sich ein System verbessert hat oder nicht. Dafür kann ROUGE unkompliziert eingesetzt werden und ohne großen Aufwand lassen sich die Auswirkungen von Veränderungen am zu entwickelnden System feststellen. Abschließend sei noch bemerkt, dass grundsätzlich nur ein Vergleich von ähnlichen Systemen Sinn macht, das heißt, die Anforderungen an die Zusammenfassungen müssen von vornherein gleich sein (indikativ oder informatitiv, Extract oder Abstract, usw.). Es ist einleuchtend, dass selbst für Menschen sonst ein Vergleich nicht sinnvoll durchzuführen ist.

7.4 Ausblick

Benutzerorienterte Aufgaben

Für das Jahr 2005 sehen die Organisatoren von DUC ein paar wesentliche Änderungen vor. Der Schwerpunkt soll nun stärker auf benutzerorientierten Aufgaben liegen. Konkret wird das so aussehen, dass dem System, das Zusammenfassungen erzeugen soll, neben den Texten noch ein Benutzerprofil sowie das Thema der Texte zur Verfügung gestellt wird (siehe 7.16). Ziel ist es dann, eine Zusammenfassung zu generieren, die auf das Thema zugeschnitten ist und die Interessen des Benutzers wiederspiegelt. Die Länge der Zusammenfassung ist dabei auf 250 Wörter begrenzt. Bei diesen Multi-Dokumenten-Zusammenfassungen wird auch eine wohlformuliert Ausgabe erwartet,

so dass es nicht ausreichend ist, einfach bestimmte Namen oder Daten zu extrahieren. Man möchte hier etwas in Richtung Abstract gehen, um auch die Lesbarkeit der erzeugten Texte zu erhöhen.

> International Organized Crime
> Identify and describe types of organized crime that crosses borders or involves more than one country. Name the countries involved. Also identify the perpetrators involved with each type of crime, including both individuals and organizations if possible.
> granularity: specific

Abbildung 7.16: Beispiel einer Aufgabe aus DUC 2005

Neue Evaluierungmethoden. Ein weiterer Schwerpunkt liegt auf der automatischen Evaluierung. Da die Probleme bei ROUGE nicht zu übersehen waren, ist man hier bemüht ein besseres Verfahren zu entwickeln. Zwei neue Verfahren kommen 2005 zum Einsatz, zum Einen ein auf minimanlen, semantischen Einheiten basierendes Verfahren mit Namen BE [BE], zum Anderen ein manuelles Bewertungsverfahren, Pyramid Method [Met] genannt, das versuchsweise angewendet werden wird. Dabei wird nicht eine Modellzusammenfassung isoliert herangezogen, sonder eine Bewertung von Modellzusammenfassungen vorgenommen, um eine feinere Auflösung der Wichtigkeit von einzelnen Informationen zu erreichen. Ein möglicher Ansatz wäre auch, über die Differenzen innerhalb von Referenzzusammenfassungen zu versuchen, die wesentlichen Informationsträger zu isolieren.

Vergleichendes Evaluieren

Man darf mit Blick auf benachbarte Forschungsgebiete, wie beispielsweise die Question-Answering-Gemeinde, gespannt sein, wie sich Entwicklungen auf diesen Gebieten in der automatischen Textzusammenfassung niederschlagen werden.

Literaturverzeichnis

[BE] Basic elements. http://www.isi.edu/~cyl/BE/.

[BWK+03] Sabine Bergler, René Witte, Michelle Khalife, Zhuoyan Li, and Frank Rudzicz. Using Knowledge-poor Coreference Resolution for Text Summarization. In *Proceedings of the HLT-NAACL Workshop on Text Summarization DUC 2003*. Document Understanding Conference, 2003. http://www-nlpir.nist.gov/projects/duc/pubs/2003final.papers/concordia.final.pdf.

[BWL+04] Sabine Bergler, René Witte, Zhuoyan Li, Michelle Khalife, Yunyu Chen, Monia Doandes, and Alina Andreevskaia. Mult-ERSS and ERSS 2004. In *Proceedings of the HLT-NAACL Workshop on Text Summarization DUC 2004*. Document Understanding Conference, 2004. http://www-nlpir.nist.gov/projects/duc/pubs/2004papers/concordia.witte.pdf.

[CMBT02] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*, 2002.

[Cona] Document Understanding Conference. http://duc.nist.gov/.

[Conb] Text Retrieval Conference. http://trec.nist.gov/.

[Cry87] David Crystal. *The Cambridge Encyclopedia of Language*. Cambridge:UP, 1987.

[Doc01] Document Understanding Conference. *DUC 2001*, New Orleans, Louisiana USA, September 13-14 2001. http://duc.nist.gov/pubs.html#2001.

[Doc02] Document Understanding Conference. *DUC 2002*, Philadelphia, Pennsylvania, USA, July 11-12 2002. http://duc.nist.gov/pubs.html#2002.

[Doc04] Document Understanding Conference. *DUC 2004*, Boston, MA, USA, May 6-7 2004. http://duc.nist.gov/pubs.html#2004.

[Fel98] Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[Kha04] Michelle Khalife. Examining Orthogonal Concepts-Based Micro-Classifiers and their Correlations with Noun-Phrase Coreference Chains. Master's thesis, Concordia University, Sept 2004.

[Lin03] Chin-Yew Lin. Cross-Domain Study of N-grams Co-Occurrence Metrics – A Case in Summarization, 2003. http://www.isi.edu/~cyl/papers/CrossDomainsNgramStudy.pdf.

[Luh58] H.P. Luhn. *The Automatic Creation of Literature Abstracts*. IBM J. Res. Develop. 2, 1958.

[Man01a] Inderjeet Mani. *Automatic Summarization*. John Benjamins, 2001.

[Man01b] Inderjeet Mani. Summarization Evaluation: An Overview. In *Proceedings of the Second NTCIR Workshop on Research in Chinese & Japanese Text Retrieval and Text Summarization*, 2001.

[Met] Pyramid Method. http://www1.cs.columbia.edu/~ani/DUC2005/.

[MM99] Inderjeet Mani and Mark T. Maybury, editors. *Advances in Automatic Text Summarization*. MIT Press, 1999.

[PO04] James Yen Paul Over. An Introduction to DUC-2004, 2004. http://www-nlpir.nist.gov/projects/duc/pubs/2004slides/duc2004.intro.pdf.

[PZ75] J.J. Pollock and A. Zamora. Automatic abstraction research at chemical abstracts service. In *Proceedings of the 169th National Meeting of the American Chemical Society*, Philadelphia, PA, April 8 1975.

[Sal89] Gerard Salton. *Automatic text processing : the transformation, analysis, and retrieval of information by computer*. Addison-Wesley, 1989. ISBN 0-201-12227-8.

[WB03] René Witte and Sabine Bergler. Fuzzy Coreference Resolution for Summarization. In *Proceedings of 2003 International Symposium on Reference Resolution and Its Applications to Question Answering and Summarization (ARQAS)*, pages 43–50, Venice, Italy, June 23–24 2003. Università Ca' Foscari. http://rene-witte.net.

[Wit02] René Witte. *Architektur von Fuzzy-Informationssystemen*. BoD, 2002. ISBN 3-8311-4149-5.

Benjamin Heitmann

8 NewsBlaster: Zusammenfassungen von Nachrichten aus mehreren Quellen

Durch das Ausnutzen der Eigenschaften von Meldungen aus Pressetickern und Nachrichtenseiten, lässt sich ein robustes und praxistaugliches System zum Zusammenfassen von thematisch verwandten Dokumenten aus Nachrichten implementieren. Das NewsBlaster System der Natural Language Processing Group der University of Columbia implementiert alle Schritte eines solchen Systems. Im folgenden werden vergleichbare Systeme vorgestellt, es wird gezeigt welche Art von Zusammenfassungen NewsBlaster erzeugt und der Aufbau des Systems wird dargestellt. Dabei werden auch die beiden getrennt voneinander entwickelten Zusammenfassungskomponenten MultiGen und DEMS, welche für unterschiedliche Gruppen von Dokumenten zuständig sind, behandelt.

8.1 Einleitung

Ein tradioneller Bereich der Forschung über die Zusammenfassung von Dokumenten, ist das Zusammenfassen von Nachrichten, also kurzen Dokumenten mit bis zu 500 Worten, verfasst von Presseagenturen, Nachrichtendiensten und Zeitungen.

Die Forschung in diesem Bereich hat sich über lange Zeit auf das Zusammenfassen einzelner Dokumente beschränkt (single document summary). Die Sätze im Dokument werden dafür anhand verschiedener Kriterien untersucht und bewertet. Diese Kriterien können rein statistischer Natur sein, sich auf die Semantik des Textes beziehen oder aber Eigenheiten der Textgattung, zu der das Dokument gehört, ausnutzen.

Forschungsfortschritte

Verschiedene Fortschritte in der Erforschung der Verfahren, welche nicht nur das Betrachten eines einzelnen Dokuments ermöglichen, sondern auch den Kontext des Dokuments untersuchen, insbesondere die chronologisch oder thematisch verwandten Dokumente, haben es ermöglicht, Ansätze zur Zusammenfassung mehrerer Dokumente zu realisieren.

Diese Fortschritte ermöglichen es unter bestimmten Umständen sehr robuste und praxistaugliche Ergebnisse zu erreichen, sowohl was die Qulität des Inhalts angeht, als auch die Lesbarkeit der Zusammenfassung.

Informationsüberflutung im Internet

Das vorteilhafte Einbeziehen möglichst vieler Quellen zum Erzeugen einer Zusammenfassung als echte Hilfe zur Bewältigung der sich schon jetzt abzeichnenden Informationsüberflutung im Internet, ist auch eine Antwort auf die Inflation des Wertes einer einzelnen Quelle und eines einzelnen Artikels. Die ähnlichen Textelemente der konsultierten Quellen spiegeln die höhere Relevanz der in ihnen enthaltenen Informationen wieder, und damit einen guten Anhaltspunkt dafür, das die Information auch in einer Zusammenfassung enthalten sein sollte. Bei den seltener vorkommenden Informationsfragmenten, kann dann anhand komplexerer nicht-statistischer Kriterien versucht werden, die Relevanz dieser seltenen aber nicht notwendigerweise überflüssigen Informationen zu bewerten, und dann kann für oder gegen das „Garnieren“ der Zusammenfassung mit diesen weiteren Fakten entschieden werden.

Ein Projekt welches nicht nur die theoretische Erforschung der Verfahren zum Kategorisieren von Texten, und der Zusammenfassung thematisch verwandter Texte als Ziel hat, sondern vor allem die Implementierung eines seit 2001 für den täglichen Einsatz geeigneten Systems, ist das NewsBlaster System der Natural Language Processing Group der University of Columbia.

Anhand der Zusammenfassungen, welche das NewsBlaster System erstellt, kann eine interessierte Person, etwa ein Analyst, Journalist oder Student, sich über den allgemeinen Verlauf der Geschehnisse aus einem Bereich des täglichen Lebens auf dem laufenden halten, ohne von der Menge der zur Verfügung stehenden Quellen überwaltigt zu werden und ohne sich von unsignifikanten Details ablenken zu lassen.

Auf der NewsBlaster Webseite [dUoC05] befindet sich der täglich aktualisierte Überblick der Nachrichten des Tages aus den Bereichen „U.S. Politik“, „Globale Politik“, „Finanzen“, „Wissenschaft“, „Entertainment“ und „Sport“. Neben den eigentlichen Zusammenfassungen, können alle Quellen einer Zusammenfassung, Listen von ähnlichen Ereignissen, die zeitliche Einordnung des Ereignisses in den kontext thematisch verwandter Ereignisse, sowie die verschiedenen nationalen Perspektiven auf das Ereigniss eingesehen werden, allerdings zur Zeit nur für englisch-sprachige Quellen.

Die Bereiche in denen die jüngsten Forschungsergebnisse am meisten zum Fortschritt der multi-document summary beigetragen haben, sind in den Bereichen Clustering und dem komplexeren Topic Detection and Tracking (TDT) zu finden, also dem Gruppieren von Artikeln und dem Entdecken enger thematischer Beziehungen zwischen Artikeln. Ausserdem gibt es große Fortschritte beim Zuordnen der Textkategorie eines Artikels, sowie beim Erzeugen der Zusammenfassung selbst.

Document Understanding Conference

Der Forschungsbereich der Dokumentzusammenfassung ist besonders durch das Interesse von Geheimdiensten an diesem Verfahren, welches sonst kostspielig von Menschen realisiert werden muss, motiviert. Dies zeigt sich in der jüngeren Vergangenheit an der Veranstaltungsreihe der Document Understanding Conference (DUC), welche jedes Jahr seit 2000 stattfindet, und welche vom National Institute of Standards and Technology (NIST) und der Advanced Research and Development Activity (ARDA) gesponsort wird. Das ARDA ist die Nachfolge-Organisation der DARPA, einer Technologie-Initiative des US Militärs, welche damals dem Internet zur Geburt verholfen hat.

Im weiteren Verlauf des Textes werden die verschiedenen Teile des NewsBlaster Systems behandelt, welche alle Aspekte der realen Anforderungen und Arbeitsschritte eines solchen Systems abdecken, vom Vorverarbeiten der Quell-Webseiten, über das thematische Gruppieren der Artikel, bis hin zum eigentlichen Erzeugen der Zusam-

menfassungen und dem Präsentieren auf der Webseite.

Ausserdem gehen wir auch auf andere Implementierungen der Idee des Textzusammenfassens und der Kategorisierung von Nachrichten ein, und geben eine kurze Tour durch die Zusammenfassung eines typischen im täglichen Ablauf gefundenen Ereignisses. Im Anschluss an die Betrachtung der einzelnen Komponenten des NewsBlaster Systems, werden die zukünftigen Erweiterungen und Forschungsthemen kurz behandelt.

8.2 Vergleichbare Implementierungen zur Zusammenfassung von Dokumenten

Bevor das NewsBlaster System analysiert wird, welches auf dem Prinzip der Zusammenfassung mehrere Dokumente (multi document summary) beruht, werden wir nun den alternativen Ansatz der Zusammenfassung eines einzelnen Dokuments (single dokument summary) betrachten, sowie sehr kurz auf Google News und ein anderes System, welches NewsBlaster sehr ähnlich ist, eingehen.

8.2.1 Single Document Summary

Bei der single document summary können die für eine Zusammenfassung benötigten Sätze anhand ihrer numerischen Bewertung und anhand der gewünschten Zusammenfassungslänge ermittelt werden. Zur Bewertung der Sätze kann Wissen über die Semantik des Dokuments verwendet werden, dazu ist aber die a priori Kenntniss der Art des Dokuments, oder eine gute Heuristik zum Einordnen der Dokumentart anhand von oberflächlichen Eigenschaften des Textes notwendig. Dieses domänenspezifische Wissen lässt sich etwa sinnvoll bei der Analyse medizinischer Texte anwenden, Krankheitsbeschreibungen haben zum Beispiel meistens einen sehr einheitlichen Textaufbau. Viel häufiger eingesetzt werden aber statistische Verfahren zur Bewertung der Relevanz eines Satzes, da diese Verfahren sehr robust entworfen werden können. Die Bewertungen aller Terme eines Satzes werden aufgerechnet, dann wird eine Rangfolge der Sätze erstellt.

Die Länge der Zusammenfassung wird entweder als Prozentzahl des Ursprungsdokuments oder als Anzahl von Sätzen vorgegeben. Die obersten Sätze des Satz-Rankings werden ohne Veränderung in der Zusammenfassung wiedergegeben, bis die gewünschte Länge erreicht ist. In der Praxis werden meistens Kombinationen aus statistischen und linguistischen Verfahren zum Erstellen des Rankings benutzt.

Zwei gängige Systeme unter Microsoft Windows zum Zusammenfassen eines einzelnen Textes sollen nun stellvertretend betrachtet werden, nämlich die AutoSummarize Funktion in Microsoft Word und die Applikation Copernic Summarizer.

Der Copernic Summarizer, hergestellt von Copernic Technologies kann PDFs, Microsoft Office Dokumente, WebSeiten und alle Texte, welche sich in die Windows Zwischenablage kopieren lassen, zusammenfassen. WebSeiten, PDFs und Office Dokumente lassen sich direkt aus den jeweiligen Applikationen heraus zusammenfassen, für alle anderen Formate muss der Text zuerst in die Zwischenablage kopiert werden, und kann dann im Copernic Summarizer zusammengefasst werden.

Vorgehen des Copernic Summarizers

Zum Zusammenfassen wird eine Mischung aus statistischen und linguistischen Methoden verwendet, siehe [Tec03]. Der erste Schritt ist das Umwandeln des ursprünglichen Dokumentformats in ein standardisiertes Eingangsformat. Die von Copernic

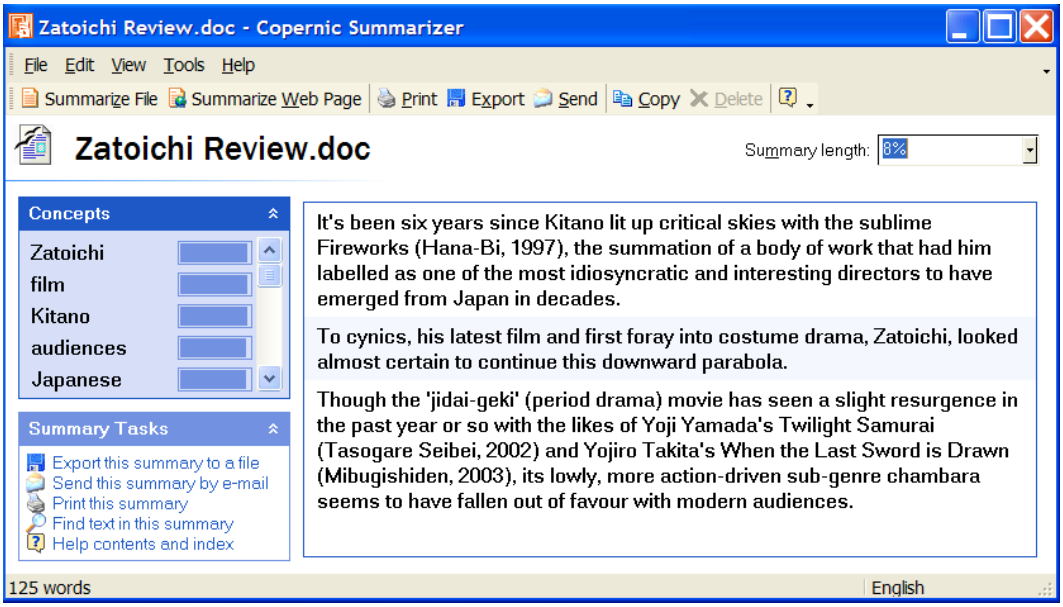


Abbildung 8.1: Betrachten einer Zusammenfassung im Copernic Summarizer

Technologies patentierte Technologie WebEssence filtert dabei alle unnötigen Bestandteile der Seite herraus, dazu gehören Werbeblöcke, sowie Bilder und Texte, die nur zur Navigation der Webseite enthalten sind. Anschliessend wird die Sprache des Dokuments bestimmt, zur Zeit können Englisch, Deutsch, Französisch und Spanisch als Dokumentsprache erkannt werden. Die erkannte Sprache führt zur Selektion von darauf angepassten linguistischen und statistischen Regeln und Heuristiken im weiteren Laufe der Textanalyse. Nun werden die Satzgrenzen identifiziert, zu dieser Aufgabe gehört es auch die einzelnen Punkte von Aufzählungen voneinander zu trennen und wissenschaftliche Formeln am Stück zu lassen. Im Rahmen dieses Arbeitsschrittes werden auch die Tokens eines Satzes identifiziert um die Aktionen, Personen und Orte eines Satzes finden zu können. Im nächsten Schritt werden die wichtigsten Konzepte und Schlüsselwörter des Textes ermittelt, diese stellen die unterste, atomare Ebene der Informationen des Dokuments dar. Danach wird das Dokument in Segmente unterteilt, so das nicht nur Aussagen über den Inhalt des ganzen Dokuments, sondern auch über Teile des Dokuments gemacht werden können. Abschliessend werden die Wichtigkeit und die Relevanz der Sätze bewertet. Sätze mit wichtigen Konzepten ermöglichen es mit hoher Wahrscheinlichkeit, das wichtige Ideen des Textes in ihnen wiedergegeben werden, wohingegen Sätze mit unklaren Referenzen und Konzepten nicht für die Zusammenfassung betrachtet werden. Die Länge der Zusammenfassung kann flexibel vom Benutzer angegeben werden und ohne erneutes Verarbeiten des Dokuments angepasst werden.

In Abbildung 8.1 sieht man die Zusammenfassung einer englischen Filmkritik des Films Zatoichi. Das Lesen des Originaltextes der Kritik unter [Sha04] erleichtert eventuell den Vergleich der Zusammenfassungsergebnisse des Copernic Summarizers und der Word AutoSummarize Funktion. Im Interface des Copernic Summarizers findet sich oben rechts die Vorgabe der Zusammenfassungslänge, hier wurde 8 Prozent eingestellt. Links am Rand befindet sich eine Liste der wichtigsten 20 Konzepte des Textes,

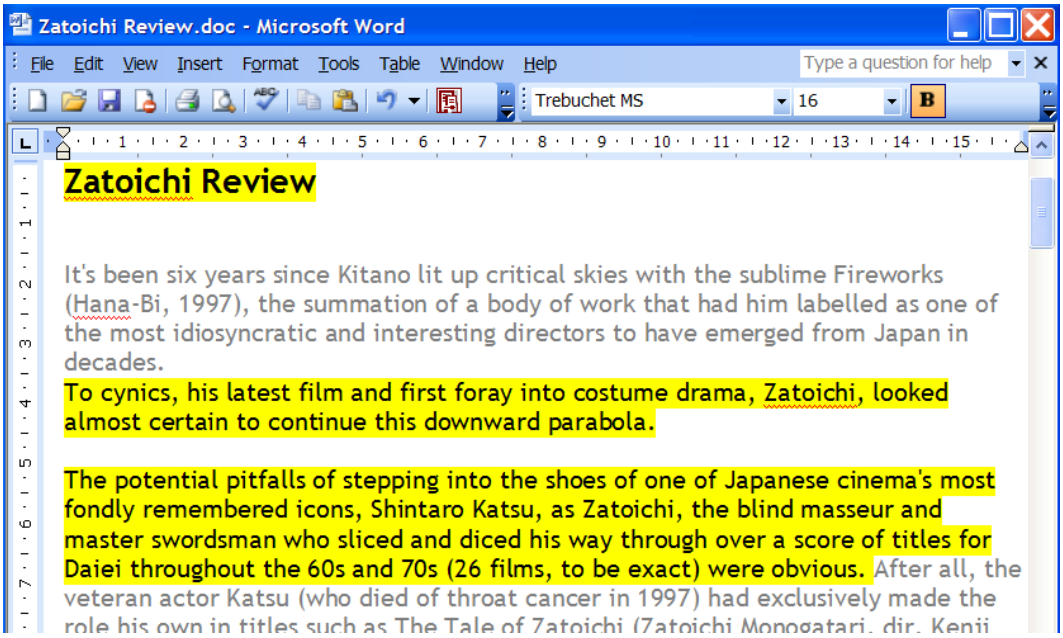


Abbildung 8.2: Ein Text mit gelben Hervorhebungen von Word AutoSummarize

besonders bei langen Texten können mit der, sich darunter befindenden Task-Liste, die Konzepte, welche den Anwender explizit nicht interessieren entfernt werden. Diese Konzepte werden dann auch nicht in der Zusammenfassung berücksichtigt. Den größten Teil des Interfaces nimmt natürlich die Zusammenfassung selbst ein, hier werden anhand des Rankings, die für den Inhalt als repräsentativ bewerteten Sätze aus dem Text unverändert übernommen.

Hervorhebungen von AutoSummarize

Microsoft Word hat seit 1997 die Funktion AutoSummarize. Mit dieser Funktion, welche über das Tools Menü erreichbar ist, kann der aktuelle Text von Word zusammengefasst werden. Dabei werden vermutlich nur statistische Verfahren verwendet, das genaue Vorgehen scheint undokumentiert zu sein. Die Zusammenfassung kann in mehreren Formen präsentiert werden, sie kann als eigenständiges neues Dokument erzeugt werden, am Anfang des Dokuments eingeschoben werden, oder Word kann die als repräsentativ bewerteten Sätze in neongelber Farbe hervorheben. Das Hervorheben der wichtigen Sätze hat in der Praxis den enormen Vorteil, das Fehlentscheidungen des Zusammenfassungsalorithmus vom Menschen ausgeglichen werden, da der Rest vom Dokument immer noch zusätzlich gelesen werden kann, zu sehen in Abbildung 8.2. So kann etwa für einen gelb hinterlegten Satz der Kontext vor und nach dem Satz konsultiert werden, oder es lässt sich das Wissen des Menschen über die Art des Textes verwenden, etwa ein kurzer Blick auf das Ende der Filmkritik um das Fazit des Kritikers zu sehen.

Schwächen der statistischen Zusammenfassung

An der Zusammenfassungsausgabe in einem neuen Dokument (Abbildung 8.3), lässt sich sehr eindrucksvoll eine der wesentlichen Schwächen der reinen statistischen Analyse eines einzelnen Textes zeigen. Im letzten Satz der Zusammenfassung lesen wir, daß der Autor der Filmkritik den Film für einen Kinobesuch empfiehlt. Im Bild sieht man auch den Schieberegler für die Länge der Zusammenfassung. Dieser ist eben so wie bei unserem Test des Copernic Summarizers auf 8 Prozent gestellt, um ei-

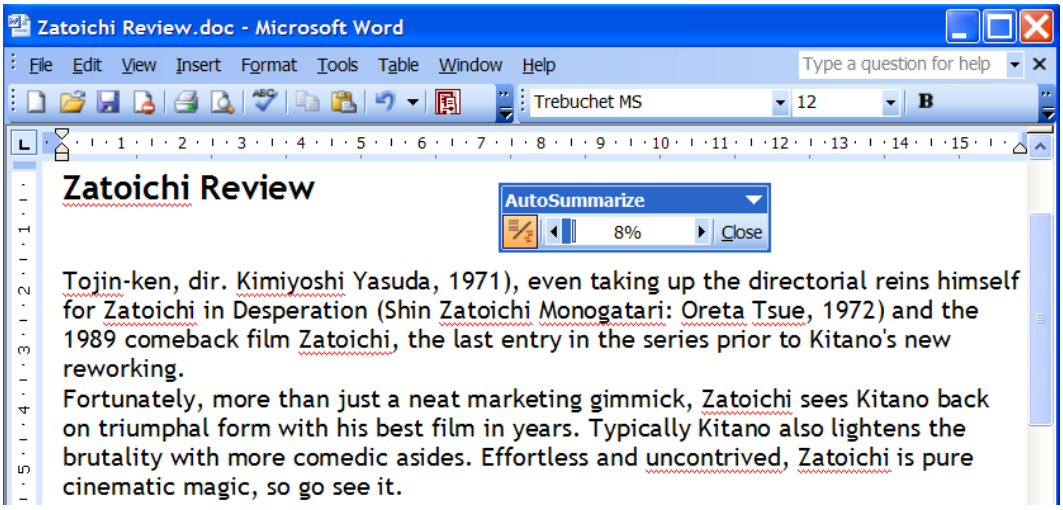


Abbildung 8.3: Eine Word AutoSummarize Zusammenfassung

ne Vergleichbarkeit der Ausgaben der beiden Programme zu gewährleisten. Wird der Schiebereglerr nun um nur ein Prozent geringer eingestellt, also auf 7 Prozent, so wird dieser sehr entscheidende Satz aus der Zusammenfassung gestrichen, obwohl von der reinen Semantik und der intuitiven menschlichen Auffassung der Wichtigkeit, dieser Satz das entscheidende Fazit der Filmkritik bildet. Eine reine statistische Auswertung kann also sehr wohl zu nicht ausreichenden Ergebnissen führen.

8.2.2 Multi Document Clustering

Später betrachten wir den im World Wide Web zugänglichen Output des NewsBlaster Systems. Der erste Einwand gegen das NewsBlaster System, ist oft der Hinweis auf Google News, welches jedoch weder von der Intention noch vom Ergebnis her mit NewsBlaster vergleichbar ist. Google News gruppiert die gesammelten Nachrichtenmeldungen nur in die groben thematischen Kategorien und sammelt zusätzlich noch die Artikel aus verschiedenen Quellen zum gleichen Ereignis. Ausserdem werden zu jedem Ereignis Bilder aus den Quellen extrahiert. Die neuesten Ereignisse werden durch eine repräsentative Überschrift und den ersten Paragraphen eines Artikels der jeweiligen Gruppe, zusammen mit Links anderen Quellen zum gleichen Ereignis, auf der Frontseite gezeigt. Jeder Link führt dann von dort aus zu einer original Quelle. NewsBlaster hingegen gruppiert die Quell-Artikel nicht nur, sondern erzeugt auch eine Zusammenfassung mit einer durchschnittlichen Länge von 10 Sätzen, welche aus unterschiedlichen Artikeln extrahiert oder umgeformt wurden. Im großen Unterschied zu NewsBlaster steht aber die Anzahl der verarbeiteten Quell-Webseiten. Während NewsBlaster nur circa Hundert Webseiten täglich nach neuen Nachrichten durchsucht, und das Ergebnis ein bis zweimal pro Tag im Web zusammenfasst, sammelt Google News Nachrichten von ungefähr 4500 Quellen ein, und stellt kontinuierlich erneuerte Gruppierungen der Nachrichten online. Es kann also davon ausgegangen werden, das der Fokus bei der Konzipierung von Google News sehr stark auf der Bewältigung möglichst vieler Quellen lag.

8.2.3 Multi Document Summary

Ein anderes System, welches sehr ähnliche Intentionen und Ergebnisse, wie NewsBlaster hat, ist NewsInEssence [LdUoM05], betreut von der Computational Linguistics And Information Retrieval Gruppe der University of Michigan. Dieses System ist auch schon seit 2001 im Einsatz, es gruppiert die gesammelten Quellen und erzeugt eine Zusammenfassung aus mehreren Quellen. Im Unterschied zu NewsBlaster kann der Anwender aber eigene Begriffe eingeben und sich die Zusammenfassung der zu diesem Begriff thematisch am nächsten stehenden Dokumentgruppe anzeigen lassen. Prinzipiell ist aber der Aufbau von NewsBlaster mit sehr vielen interessanten Eigenheiten und Ideen versehen, so das bei der vorliegenden Arbeit nur das NewsBlaster System betrachtet wurde.

8.3 NewsBlaster in Aktion

NewsBlaster befindet sich seit 2001 im Einsatz. Es erzeugt täglich ein bis zweimal einen Überblick über die Nachrichten des Tages. Damit die Eigenschaften der Zusammenfassungen des Systems und die Art der Präsentation besser nachvollzogen werden können, wird nun stellvertretend ein Zusammenfassungs-Text aus dem System präsentiert.

8.3.1 Die Hauptseite

Auf der NewsBlaster Hauptseite befindet sich das Einstiegsportal zu den Zusammenfassungen des aktuellen Durchlaufs. Oben rechts wird die Zeitspanne der Artikel, welche in den Zusammenfassungen berücksichtigt wurden angegeben, meistens ist dies ein einzelner Tag. Die genaue Uhrzeit des letzten Auswertungsdurchlaufs wird dort auch vermerkt. Links oben ist ein Eingabefeld zum Durchsuchen der Zusammenfassungen des Tages oder der Quell-Artikel des Tages. Die Übersichten der Kategorien „U.S. Politik“, „Globale Politik“, „Finanzen“, „Wissenschaft“, „Entertainment“ und „Sport“ lassen sich direkt auswählen.

Den prominentesten Teil der Hauptseite nimmt das Top-Ereignis des Tages ein, im Beispiel ein Aufruf der Irakischen Übergangsregierung zur Einigkeit nach der ersten Wahl im Irak seit knapp 50 Jahren. Passend zum wichtigsten Ereignis des Tages wird ein Bild ausgewählt. Neben der Schlagzeile des Top-Ereignisses befindet sich die Zahl der Quellen und die nationale Zugehörigkeit der Quellen, in diesem Fall „World“, da es auf der ganzen Welt englisch-sprachige Quellen über dieses Ereignis im Irak gab. Der Text unter der Überschrift stellt schon die eigentliche Zusammenfassung dar, allerdings ohne Quellenangabe nach dem jeweiligen Satz.

Unter der Zusammenfassung listet das System die Haupt-Schlüsselwörter auf, welche in den Quell-Artikeln vorgekommen sind, sowie weitere Ereignisse in denen diese Schlüsselwörter auch eine wichtige Rolle gespielt haben. Links befinden sich dann noch einige weitere Optionen zur Verwendung von NewsBlaster. Die zwei interessantesten sind der „View Archive“ Link, welcher das Betrachten der täglichen Durchläufe seit September 2001 ermöglicht, sowie „About todays run“. Dort findet sich eine Auswertung der Anzahl der Artikel, welche von jeder Quelle geholt wurden.

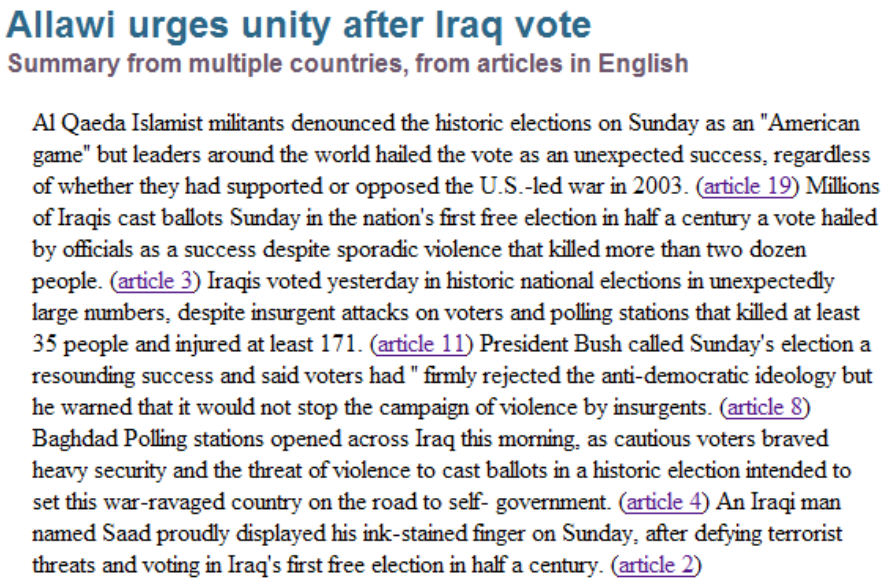


Abbildung 8.4: Eine von NewsBlaster erzeugte Zusammenfassung

8.3.2 Die Detailansicht einer Zusammenfassung

Die Detailansicht eines Ereignisses listet noch einmal die Überschrift, den regionalen Bezug der Quellen, und die eigentliche Zusammenfassung auf. Die Überschrift wird direkt von einem der Artikel, welcher als repräsentativ für das Ereignis gewertet wurde, genommen. Neben jedem Satz befindet sich ein Link auf die original Quelle, aus der der Satz entnommen wurde oder aus dem die Information im Satz stammt, falls der Satz vom NewsBlaster System für die Zusammenfassung umgeformt wurde. Rechts befinden sich alle Photos der zum Ereignis gesammelten Artikel, so das auch ein visueller Überblick des Ereignisses gegeben wird. Speziell bei Themen bei denen die Bildsprache eine wichtige Rolle spielt, spontan drängt sich hier die Bildgewalt der qualmenden Twin-Towers auf, macht dieser visuelle Überblick sehr viel Sinn.

Betrachtet man die Details des Textes (siehe Abbildung 8.4), so fällt auf, das der graduelle Übergang vom Allgemeinen zum Speziellen im gegebenen Beispiel sehr gut funktioniert. Am Anfang der Zusammenfassung befinden sich eher allgemein gehaltene essentielle Aussagen über das Ereignis, gefolgt von detailreicheren Fakten, welche die allgemeinen Informationen in einen konkreten Kontext setzten, und abgeschlossen wird mit passenden aber im grossen Kontext eher ausschmückenden Details.

In der Beispiel-Zusammenfassung steht zu Begin die allgemeine Aussage, das Politiker weltweit die Wahl im Irak als Erfolg bezeichnet hätten, obwohl Vertreter von Al Qaeda die Wahl nicht befürworten. In der Mitte findet sich das Faktum der Anzahl getöteter und verletzter Menschen durch Anschläge auf Wahllokale. Am Schluss wird eher ausschmückend von einem namentlich genannten Iraker berichtet, der sehr stolz seinen Finger hochhält, welcher noch durch die nicht entfernbare Tinte gezeichnet ist, mit der Mehrfachwahlen verhindert werden sollen.

Dieser Aufbau der Zusammenfassung ist für den Leser sehr ansprechend, obwohl der Text maschinell erzeugt wurde. Das genau Verfahren für die Konstruktion der Zusammenfassung, um diesen graduellen Übergang im Informationsgehalt der Bestand-

Abbildung 8.5: Eine Quelle mit einem hervorgehobenem Satz

teile der Zusammenfassung zu erreichen, ist nicht dokumentiert, da das NewsBlaster Projekt seit 2 jahren keine neuen Forschungspublikationen veröffentlicht hat, ausser einem Ausblick auf die in der nahen Zukunft zu erwartenden Weiterentwicklungen. Auch auf eine direkte Anfrage per Mail gab es keine Reaktion vom NewsBlaster Team. Die Weiterentwicklungen werden im Ausblick behandelt. Es wird aber der bisher dokumentierte Stand der Zusammenfassungserzeugung bei der Betrachtung der Architektur des NewsBlaster Systems behandelt.

8.3.3 Ein einzelner Quell-Artikel

Auf der Detailseite eines Ereignisses befindet sich auch eine Auflistung aller für die Zusammenfassung verwendeten Quell-Artikel, sowie direkt neben den Sätzen der Zusammenfassung ein direkter Link nicht nur auf die Quelle, sondern auch auf den spezifischen Satz, welcher aus der Quelle verwendet wurde. Klickt man einen solchen Link an, so wird der original Artikel von der Quelle durch einen NewsBlaster Proxy geladen, und der verwendete Satz wird in Gelb hervorgehoben, falls er unverändert von der Quelle übernommen wurde (siehe Abbildung 8.5). Interessanterweise werden die Sätze für die Zusammenfassung meistens aus den ersten drei Paragraphen eines Artikels entnommen.

8.3.4 Visualisierung der zeitlichen Abfolge von Ereignissen

NewsBlaster ordnet Ereignisse und Dokumentgruppen in mehreren hierarchischen Ebenen an. Die unterschiedliche Granularität der Ebenen ermöglicht es Verbindungen zwischen den Ereignissen zu entdecken, und diese werden in der Timeline des Ereignisses gezeigt (siehe Abbildung 8.6). Diese ist bei der Detailansicht einer Zusammenfassung über den Link „Track this story's development in time“ zu erreichen. Dort können Zusammenhänge zu den verschiedenen anderen Ereignissen, welche zu der Entwicklung des Ereignisses, von welchem ursprünglich die Zusammenfassung betrachtet wurde, visualisiert werden. Das ursprünglich betrachtete Ereigniss ist in einem mintgrünen Kasten.

Abbildung 8.6: Visualisierung der zeitlichen Abfolge verschiedener Ereignisse

8.3.5 Gegenüberstellung von länderspezifischen Quellen

Eine weitere sehr interessante Option ist das Gegenüberstellen der Zusammenfassungen verschiedener Länder. Dabei wird links die Zusammenfassung angezeigt, welche aus Quellartikeln des gesamten englischsprachigen Raums gebildet wurde. Rechts wird die Zusammenfassung angezeigt, welche nur aus Artikeln eines Landes gebildet wurde. Im Beispiel ist dies die Zusammenfassung der Artikel aus England, und dort sind dann auch die diesem Land eigenen Perspektiven zu finden, etwa eine Aussage der englischen Außenministerin. Die Gegenüberstellung erreicht man in der Detailansicht eines Ereignisses, direkt unter der Zusammenfassung, indem man auf einen der „Compare“ Links klickt.

8.4 Aufbau des NewsBlaster Systems

Im Kontext des Internet lässt sich das Hauptproblem für die nahe Zukunft nicht durch Informationsknappheit sondern durch die zunehmende Informationsvielfalt, die Inflation der Qualität der Inhalte und der Bewältigung dieser Informationsvielfalt charakterisieren.

Die Informationsüberflutung im Internet wird verursacht durch die große Anzahl der Quellen, und die unüberschaubare Menge von Artikeln zum gleichen Thema oder zum gleichen Ereignis, zur gleichen Person oder zu einer anders klassifizierten Gruppe ähnlicher Artikel. Da aber im Falle des Internet alle Publikationen das gleiche Informationsmedium verwenden, kann die im traditionellen Sinn als Nachteil begriffene Informationsüberschwemmung vom Nachteil in einen Vorteil verwandelt werden. Durch die zur Verfügung stehende Anzahl der Quellen können ähnliche und wiederholt genannte Informationen als essentiell für eine Zusammenfassung bewertet werden, da sie auch von den menschlichen Redakteuren und Schreibern der Nachrichtenredaktionen für wichtig befunden wurden, wenn sie in jedem Artikel vorkommen. Bei einer großen Anzahl von Quellen lässt sich eine solche Bewertung a priori leichter durchführen. Die Unterschiede in den Quellen können nun anhand ihrer Relevanz für die gefundenen Kernaussagen bewertet werden und in der Zusammenfassung eingesetzt werden. Das Ergebnis sind Zusammenfassungen, welche von Menschen durchschnittlich besser bewertet werden als die Zusammenfassungen von Systemen die nur mit einzelnen Input-Dokumenten arbeiten.

NewsBlaster verarbeitet die eingegebenen Artikel in einer Pipeline Architektur, siehe [SNM02]. Zuerst werden die Artikel aus den Quell-Webseiten extrahiert, dann in

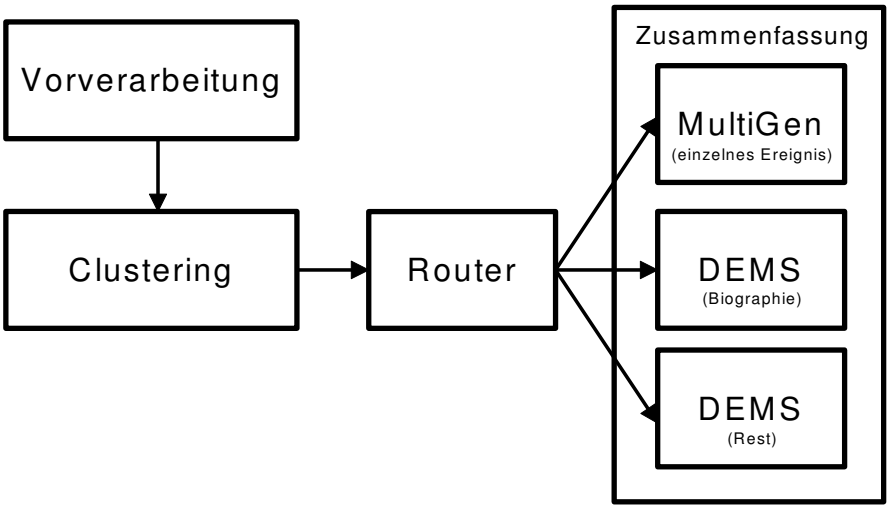


Abbildung 8.7: Der schematische Aufbau des NewsBlaster Systems

Clustern von thematisch verwandten Artikeln gruppiert. Die Artikelgruppen werden dann von einem Router an eine von 3 Zusammenfassungs-Engines weitergereicht. Für Artikelgruppen, welche ein zentrales Ereignis behandeln, ist MultiGen zuständig, für Artikelgruppen mit biographischem Charakter ist DEMS zuständig, und alle Artikelgruppen, welche sich nicht in diese beiden Kategorien einordnen lassen, werden von DEMS in einer allgemeineren Konfiguration verarbeitet. Das Schema der Komponenten nach [MHB+01a] ist in Abbildung 8.7 zu sehen.

8.4.1 Vorverarbeitung

Am Anfang steht das Sammeln der Nachrichten-Artikel, siehe [MBE+02]. Dafür besucht eine Crawler Komponente circa 100 Nachrichtenseiten, darunter auch bekannte Nachrichtenseiten wie CNN, Reuters und USA Today, um nur einige zu nennen. Die Liste der besuchten Nachrichtenseiten wird ständig erweitert. Jede Nachrichten-Quelle wird bis zu einer maximalen Tiefe von zur Zeit vier Seiten besucht. Das bedeutet, das die Frontseite der Nachrichtenquelle zuerst besucht wird, dann werden die Links auf der Frontseite analysiert, und alle Links, welche wiederrum auch auf Nachrichten-Artikel verweisen, werden geladen. Die Artikel von diesen Links werden dann ihrerseits wieder untersucht, bis der Crawler bei der vierten Ebene angekommen ist. Bei jeder Seite, die auf diesem Weg erhalten wurde, wird der HTML Code der Seite untersucht. Alle HTML Eigenheiten werden entfernt, bis auf Tabellenstrukturen und den eigentlichen Text. Da im World Wide Web Tabellenstrukturen auch zur visuellen Formatierung einer HTML Seite eingesetzt werden, orientiert sich der NewsBlaster Crawler an der Anzahl der Zeichen in einer Tabellenzelle. Wenn eine Tabellenzelle mehr als 512 Zeichen enthält, so wird davon ausgegangen, das der Text zu einem Artikel gehört und nicht zu Werbung oder zur Seitennavigation. Außerdem werden Überschriften, Zwischentitel, Bilder und Bildunterschriften markiert. Der Text wird dann aus der HTML Seite extrahiert, und in ein einheitliches XML Format umgewandelt. Dabei werden auch die Metadaten des Artikels umgewandelt, etwa das Veröffentlichungsdatum und die Angabe der Quelle des Artikels. Die Überschriften des Artikels werden hin-

zu gefügt, und es werden Verweise auf die Bilder des Artikels mit abgespeichert. Die auf diese Weise aufbereiteten Artikel im einheitlichen XML Format werden nun an die Clustering Komponente von NewsBlaster einzeln weitergegeben.

8.4.2 Clustering

Um mehrere Artikel mit dem gleichen Schwerpunkt-Ereignis, oder der selben Person im Mittelpunkt oder einem anderen thematischen Schwerpunkt zusammenfassen zu können, muss man überhaupt erst mal eine Gruppe zusammengehöriger Artikel finden.

NewsBlaster klassifiziert die vom Crawler gesammelten Artikel in 3 hierarchischen Ebenen. Auf der obersten Ebene werden die Artikel in eine der übergeordneten Kategorien „U.S. Politik“, „Globale Politik“, „Finanzen“, „Wissenschaft“, „Entertainment“ und „Sport“ eingeordnet. Innerhalb jeder übergeordneten Kategorie von Artikeln werden die Artikel auf zwei hierarchischen Ebenen weiter kategorisiert. Auf der untersten Ebene werden Artikel in Gruppen zum gleichen Ereiginis eingeordnet, während auf der darüberliegenden Ebene Gruppen aus miteinander zu assozierenden Ereignissen gebildet werden.

TF*IDF

Für die Aufgabe des Clustering wurden in der bisherigen Entwicklung von NewsBlaster mehrere Algorithmen verwendet. Der Basis-Algorithmus ist TF*IDF, wobei TF für term frequency steht, das ist die Häufigkeit eines Wortes in allen Input-Dokumenten. IDF ist die inverse document frequency, das ist die Häufigkeit des Terms in einem einzelnen Dokument. Beide Häufigkeiten werden miteinander aufgerechnet, und bilden dann ein Maß für die Relevanz eines Begriffs für ein Dokument. über dieses Maß lässt sich eine Distanz zwischen Dokumenten berechnen. Dokumente mit den gleichen relevanten Begriffen können dann zu Clustern zusammengefasst werden.

SimFinder

Eine aufwendigere Heuristik zur Bestimmung der Relevanz eines Begriffs bietet SimFinder [HKH+01], welches im Rahmen von Forschungsbemühungen zu Topic Detection and Tracking (TDT) der University of Columbia für NewsBlaster und vergleichbare Systeme entwickelt wurde. SimFinder bewertet nicht nur Wörter sondern darüber hinausgehende Eigenschaften des Textes. Dabei wird zwischen „primitive features“, im folgenden Primitiv-Eigenschaften genannt, und „composite features“, im folgenden Komposit-Eigenschaften genannt, unterschieden. Primitiv-Eigenschaften sind auf das Vergleichen einzelner Wörter beschränkt. Es werden aber nicht nur gleiche Wörter, sondern auch Wörter mit dem gleichen Wortstamm beachtet und es werden Wörter mit gleichen WordNet Synonymen gezählt. Nicht nur gleichwertige Synonyme werden gewertet sondern auch Hypernyme (generellere Begriffe) und Hyponyme (spezifischere Begriffe). Die Primitiv-Eigenschaften lassen sich nun zu Komposit-Eigenschaften verknüpfen. Eine Komposit-Eigenschaft setzt sich aus zwei Primitiv-Eigenschaften mit speziellen grammatikalischen Rollen zusammen, wobei die beiden Wörter höchstens 5 Wörter Abstand voneinander haben dürfen. Eine Komposit-Eigenschaft setzt sich dann aus einem Paar eines Satz-Subjekts und einem Verb, oder aus einem Satz-Objekt und einem Verb zusammen. Die Primitiv-Eigenschaften und die Komposit-Eigenschaften der Artikel werden dann miteinander verglichen, und so kann wieder eine Distanz zwischen Dokumenten errechnet werden.

primitive features
composite features

Cluster Algorithmen

Unabhängig von der Methode zur Berechnung der Distanz zwischen Dokumenten, können auf der Basis der berechneten Distanz mit verschiedenen Algorithmen iterativ Cluster über der Menge der eingegebenen Dokumente gebildet werden. Ein Dokument ohne Clusterzugehörigkeit wird beim single link Algorithmus zum Cluster mit dem

nächsten Mitglied zugeordnet, falls die Distanz zu diesem Mitglied kleiner als eine bestimmte Schwellwertkonstante ist. Beim complete link Algorithmus, wird der neue Artikel einem Cluster zugeordnet wenn die Distanz zu allen Mitgliedern des Clusterns kleiner als der Schwellwert ist. Beim groupwise average Algorithmus muss die durchschnittliche Distanz des neuen Artikels zu allen Mitgliedern eines Clusters kleiner als der Schwellwert sein.

Diese drei Varianten sind für das Bilden von Clustern über Mengen sehr bekannt. Sie zeichnen sich auch dadurch aus, das sie iterativ vorgehen, und die Zugehörigkeit eines Artikels zu einem Cluster neu in jeder Iteration des Algorithmus neu berechnet werden kann, bis die Ergebnisse sich nicht mehr verbessern. Der single pass clustering Algorithmus [HGM00], welcher an der University of Columbia entwickelt wurde, legt bei Hinzufügen eines Artikels zur Menge der zu gruppierenden Artikel sofort die Zugehörigkeit des neuen Artikels fest. Ist die Distanz des neuen Artikels zu einem Mitglied eines bestehenden Cluster unterhalb des Schwellwertes so wird er diesem Cluster zugeordnet, wenn nicht, so bildet er einen neuen eigenen Cluster. Jeder Artikel wird nur einmal einem Cluster zugeordnet. Diese Variante eigent sich zur Bearbeitung von Mengen mit sehr vielen Artikeln oder zur Gruppierung von Artikeln die über die Zeitachse verteilt im System eintreffen, wie es besonders beim Sammeln von Nachrichten aus dem Internet der Fall ist.

single pass clustering

Die gefundenen Artikelgruppen werden dann an den Router weitergegeben.

8.4.3 Routing von Artikelgruppen

Die Router Komponente untersucht jede Gruppe von Dokumenten, klassifiziert sie in eine von drei Kategorien, und reicht die Gruppe dann an die zuständige Zusammenfassungsengine weiter [MHB+01a].

In single-event Artikelgruppen haben alle Artikel das gleiche Ereignis als Schwerpunkt, die Inhalte der Artikel ereignen sich alle am gleichen Schauplatz zu ziemlich genau dem gleichen Zeitpunkt mit den selben Akteuren und Handlungen. Eine Artikelgruppe über die Oscarpreis-Verleihung fällt in die Kategorie der Single-Event Gruppen. Eine solche Gruppe enthält typischerweise nur Artikel die innerhalb einer sehr kurzen Zeitspanne publiziert wurde, vorgeschlagen wird, das die Artikel innerhalb von 80 Tagen veröffentlich sein müssen, wobei auch eine kürzere Zeitspanne als Parameter realistisch wäre. Als zweites Indiz muss ein hoher Prozentsatz an Artikeln aus der Gruppe, etwa 50 Prozent, am gleichen Tag veröffentlich worden sein. Diese Artikelgruppen werden an die MultiGen Zusammenfassungsengine weitergereicht.

single-event Artikelgruppen

Als person-centered Artikelgruppen bzw. als biographische Artikelgruppen, werden Artikel bezeichnet in denen es um eine einzelne Person und ein Ereignis geht, welches dieser Person wiederfahren ist. Meistens enthalten die Artikel dann noch zusätzliche Informationen über die Person, in der Form von Hintergrundfakten und zeitlich davor oder danach eingeordneten Ereignisse, jedoch immer mit der gleichen zentralen Person. Eine Gruppe von Berichten über die Karriere eines Politikers fällt in diese Kategorie. Auf den biographischen Charakter einer Dokumentengruppe weist die hohe Häufigkeit eines einzelnen großgeschriebenen Namens hin, und die hohe Häufigkeit der Personalpronomen „he“ und „she“, wobei das erste Kriterium nur in der englischen Sprache seine Berechtigung hat, da dies in der deutschen Sprache auf alle Substantive zutrifft. Diese Artikelgruppen werden an die DEMS Zusammenfassungsengine weitergereicht.

biographische Artikelgruppen

Alle anderen Artikelgruppen werden als „other“ eingestuft. Dies bedeutet nicht das

andere Artikelgruppen

ihnen keine Struktur oder keine Gemeinsamkeit zugrunde liegt, sondern das nach dem aktuellen Stand der Forschung eine besonderte Behandlung zu komplex wäre. Nicht gesondert behandelt werden multi-event Gruppen, in denen über verschiedene Ereignisse zu unterschiedlichen Zeitpunkten und mit unterschiedlichen Protagonisten gemeinsam berichtet wird, etwa Artikel über mehrere Brandstiftungen in einem Sommer oder über mehrere Sonnenfinsternisse. Dokumentgruppen über noch schwächer miteinander verbundene Ereignisse, etwa Artikel über die Forschung in der Antarktis und die Konflikte im Zusammenhang mit der Verwertung der Forschungsergebnisse, bieten eine so kleine Angriffsfläche, das NewsBlaster hier den Ansatz verfolgt, diese Sonderfälle alle zusammen mit einer einzigen möglichst robusten Konfiguration der DEMS Zusammenfassungsengine abzudecken.

8.4.4 Single Event Zusammenfassungen mit MultiGen

MultiGen erzeugt Zusammenfassungen aus Gruppen von Dokumenten, in welchen das gleiche Ereignis aus verschiedenen Perspektiven behandelt wird. MultiGen war mehrere Jahre ein intensiver Forschungsschwerpunkt an der Natural Language Processing Group der University of Columbia, deswegen verwendet es mehrere neue Ansätze im Bereich der Textzusammenfassung mit mehreren Quelldokumenten.

Um eine Zusammenfassung zu erstellen, werden von MultiGen zuerst „Themes“ im Text identifiziert, danach wird die Information eines „Themes“ verschmolzen und durch einen Textgenerator als neuer Satz formuliert [MKH+99].

Textanalyse Komponente

Artikelgruppen, welche die MultiGen Zusammenfassungsengine erhält, gelangen zuerst an die Textanalyse Komponente. Aus allen Artikeln der Gruppe werden die Paragraphen herausgelöst, und anhand von Primitiv-Eigenschaften und Komposit-Eigenschaften der Paragraphen, werden Paragraphen, welche die gleiche Information ausdrücken unabhängig von ihrem Ursprungsdokument identifiziert. Dabei werden gleiche Wort, Wortstämme oder WordNet Synonyme sowie Wortpaare mit einer bestimmten Funktion zwischen den Paragraphen verglichen, in einer ähnlichen Weise wie beim Clustering. Das Zeil ist aber nicht das Bilden von Clustern über der Menge der Artikel, sondern das Identifizieren von „Themes“ über der Menge der Paragraphen einer vorher schon identifizierten Gruppe von Artikeln. Dabei bezeichnet man eine Gruppe von ähnlichen Sätzen oder Paragraphen als Theme. Zu beachten ist hier, das in englischen Nachrichtenartikeln in der Regel nur ein Satz pro Paragraph vorkommt, so das im folgenden Paragraphen und Sätze gleichzusetzten sind.

Informationsfusion

Nachdem alle Themes in der Artikelgruppe identifiziert worden sind, werden sie von der Analyse Komponente an die Neuformulierungskomponente von MultiGen weitergegeben. Dort wird zuerst eine „Informationsfusion“ vorgenommen [BME99]. Das bedeutet, das die Information, welche die Sätze in einem Theme wiederspiegeln verschmolzen wird. Jeder Satz im Theme wird dazu als Abhängigkeitsbaum dargestellt. An der Wurzel des Baums steht das Hauptverb des Satzes, alle anderen davon abhängigen Satzteile sind die Kindknoten des Wurzelknotens. Nun werden die Abhängigkeitsbäume des Themes miteinandern verglichen, wobei gleichwertige Knoten von zwei Bäumen miteinander verschmolzen werden, angefangen bei der Würzel. Um zu ermitteln ob sich zwei Knoten verschmelzen lassen werden wieder Wortstämme und WordNet Synonyme verglichen. Oder wenn ein Nomen das Subjekt eines Nebensatzes ist, so sind das Nomen und der Nebensatz gleich zu setzten. Auf diese Weise wird versucht möglichst viele Abhängigkeitsbäume des Themes zu verschmelzen.

Die Abhängigkeitsbäume eines Themes bei denen Informationen miteinander ver-

einbart werden konnten, werden dann zusammen mit Bezeichnern für die grammatikalische Rolle, welche jeder Knoten in einem Baum spielt an das Satzerzeugungssystem FUF/SURGE weitergegeben. Dieses Satzerzeugungssystem wurde ausserhalb des NewsBlaster Forschungsprojektes entwickelt, deswegen müssen die Abhängigkeitsbäume um die grammatischen Informationen erweitert werden und in das Eingabe-Format von FUF/SURGE konvertiert werden. Dort wird dann entschieden ob sich mehrere kleinere Sätze zu einem großen Satz mit mehreren Nebensätzen zusammenfassen lassen, und es werden relative Zeitangaben, wie „last year“ in absolute Zeitangaben umgewandelt. So entsteht für jedes Theme ein Ausgabesatz in der Zusammenfassung der Artikelgruppe.

Erzeugung neuer Sätze

Bevor die Zusammenfassung ausgegeben werden kann, müssen die Sätze der Zusammenfassung noch in eine sinnvolle Reihenfolge gebracht werden, da die Reihenfolge der Sätze die Lesbarkeit der Zusammenfassung durch einen Menschen stark beeinflussen kann. Zuerst wird dafür ein Timestamp für jedes Theme ermittelt, das ist das Veröffentlichungsdatum, des Artikels im Theme, welcher am frühesten veröffentlicht wurde. Dann werden die Themes in Blöcke mit ähnlichen Ideen und Worten gruppiert, um die Kohärenz zu verbessern. So das nicht etwa drei Sätze mit Hintergrundinformationen von einem Satz über den Ausgang des Ereignisses unterbrochen werden. Die Sätze in jedem Block werden chronologisch anhand der Timestamps der Themes geordnet, Sätze aus den frühesten Themes kommen zuerst. Dann werden die Blöcke geordnet, wobei hier auch wieder die Blöcke mit den frühesten Timestamps zuerst in der Zusammenfassung gebracht werden [BEM02].

chronologische Ausgabe-Sortierung

Im Gegensatz zu bisherigen Ansätzen und im Gegensatz zu DEMS werden die Sätze in den Zusammenfassungen, welche von MultiGen erzeugt werden, nicht aus den Ursprungsdokumenten extrahiert, sondern neu generiert. Dies hat den Vorteil, das nur die Informationen aus den verschiedenen Quell-Artikeln übernommen werden, und nicht die Satzbaustrukturen, welche bei Dokumenten mit unterschiedlichen Stilen meistens sehr verschieden sind.

8.4.5 Zusammenfassungen von biographischen Dokumenten mit DEMS

Innerhalb des NewsBlaster System ist die DEMS Zusammenfassungsengine zuständig für alle Dokumentgruppen, welche sich nicht mit einem Schwerpunkt-Ereignis beschäftigen. DEMS steht für Dissimilarity Engine for Multidocument Summarization.

DEMS geht zum Erstellen der Zusammenfassung in drei Schritten vor. Zuerst werden die relevanten und informativen Sätze in allen Artikeln der Gruppe identifiziert, dann werden die gefundenen Sätze bewertet und zuletzt werden die am besten bewerteten Sätze aus den Artikeln extrahiert [MHB+01b].

Zum finden der relevanten und informativen Sätze werden vier verschiedene Kriterien von Texteigenschaften verwendet. Die erste Klasse bewertet linguistische Eigenschaften des Textes. In dieser Klasse befindet sich das Kriterium der „lead values“. Dies ist eine Sammlung von Wörtern, welche mit hoher Wahrscheinlichkeit von Journalisten in den Aufmachern der Artikel verwendet werden, also im ersten Paragraphen eines Artikels. Diese Sammlung beinhaltet etwa 5000 Wörter, wie „bloody“ oder „gigantic“. Sätze, die solche lead values Wörter enthalten, werden mit einer höheren Relevanz bewertet. Eine weitere Eigenschaft ist die Spezifizität der Verben eines Satzes. Während manche Verben, wie „to be“ und „to do“ in sehr allgemeiner Weise verwendet werden können, lässt sich „to arrest“ nur ein einer sehr spezifischen Weise in einem Satz verwenden, das Verb hat also einen sehr hohen Informationsgrad.

linguistische Eigenschaften

semantische Konzepte

Die zweite Klasse von Kriterien bewertet die Häufigkeit der semantischen Konzepte der gesamten Dokumentgruppe im betrachteten Satz. Im Gegensatz zu bisherigen Ansätzen, welche nur die Häufigkeit von Wörtern bewerten, werden hier auch Wortstämme und WordNet Synonyme mit weniger als fünf Bedeutungen verglichen.

Aufbau von Nachrichtenartikeln

Die dritte Klasse von Kriterien basiert auf dem typischen Aufbau von Nachrichtenartikeln. Zwei Kriterien aus dieser Klasse bewerten das Veröffentlichungsdatum des Ursprungsartikels eines Satzes und die Position des Satzes im Ursprungsartikel. Dabei werden neuer Artikel besser bewertet und Sätze die näher am Anfang eines Artikels stehen erhalten eine höhere Bewertung als Sätze am Ende des Artikels.

Syntax und Stil

Die vierte Klasse von Kriterien basiert auf den syntaktischen und stilistischen Gegebenheiten eines Nachrichtenartikels, unter anderem die Länge eines Satzes und das Vorhandensein von Pronomen. Die ideale Länge eines Satzes liegt demnach zwischen 5 und 20 Wörtern, da kürzere Sätze zu wenig Information enthalten, und längere Sätze die Gefahr besitzen, zu viele ablenkende Information einzuführen. Zu viele Pronomen werden negativ bewertet, da sie die Gefahr unaufgelöster Referenzen auf Subjekte in anderen Teilen des Ursprungsdokuments erhöhen.

Handelt es sich ausserdem um eine biographische Dokumentengruppe, so werden Sätze die das biographische Subjekt enthalten hoch bewertet. Das häufigste großgeschriebene Subjekt wird dabei als das biographische Subjekt behandelt.

Satzextraktion

Die Sätze aller Artikel der Artikelgruppe, werden dann bewertet, und die am besten bewerteten Sätze werden aus den Ursprungsartikeln extrahiert. Ähnlich wie bei der Ausgabe der Zusammenfassung von MultiGen, werden alle Sätze mit den Timestamps der Ursprungsartikel versehen, und die Sätze werden chronologisch in der Ausgabe sortiert, wobei die neuesten Sätze zuerst ausgegeben werden.

8.5 Ausblick

Wie wir gesehen haben, ist NewsBlaster ein sehr robustes System zum Zusammenfassen von Nachrichten, welches durch die Verwendung von zwei verschieden konzipierten Zusammenfassungsengines, MultiGen und DEMS, sehr flexibel auf verschiedene Arten von Dokumentengruppen reagieren kann, und Zusammenfassungen erzeugt, die schon jetzt für den interessierten Anwender eine Erleichterung bei der täglichen Orientierung im weltweiten Nachrichtenjungel darstellen können.

Für die nahe Zukunft werden verschiedene Weiterentwicklungen des Systems, neben der reinen Verbesserung der Qualität der Zusammenfassungen in Aussicht gestellt [MBC+03]. So wurden schon Testläufe zur Zusammenfassung von Artikeln aus mehreren Sprachen gestartet. Dabei werden die Artikel aus so unterschiedlichen Sprachen, wie Spanisch, Französisch, Deutsch, Russisch, Japanisch, Vietnamesisch und Arabisch, zunächst maschinell übersetzt, und dann wie die anderen englisch-sprachigen Artikel dem NewsBlaster System zur Verfügung gestellt. Es hat sich aber dabei schon herrausgestellt, das die Qulität der maschinellen Übersetzung sehr stark schwanken kann, so das die Bedeutung der fremdsprachigen Artikel von vorne herein stark herabgesetzt wurde.

Zusammenfassungen aus der Perspektive fremdsprachiger Länder

Auf dieser Entwicklung aufbauend, soll es dann auch möglich sein, Zusammenfassungen zu einem bestimmten Ereignis zu sehen, welche nur aus Quell-Artikeln aus einem bestimmten Land oder aus einer bestimmten Sprache erzeugt wurden. So liesse sich auch Perspektiven zu bestimmten Ereignissen von exotischen Regionen der Erde mit den Perspektiven der westlichen Presse vergleichen. Etwa die Meinung der arabischen Medien zur ersten Wahl im Irak. Dies war bisher nur mit Dolmetschern und

hohem finanziellem Aufwand möglich.

Sehr vielversprechend und auch schon zum größten Teil in NewsBlaster implementiert, jedoch nicht im Rahmen der Forschungspublikationen dokumentiert, ist das Verfolgen der Entwicklung mehrerer Ereignisse. Dies lässt sich schon jetzt bei der Visualisierung der Zusammenhänge zwischen den Ereignisse nachvollziehen. Darüber hinausgehend soll es später auch möglich sein, sich alle neuen Aspekte eines Ereignisses oder einer Themengruppe ab einem bestimmten Zeitpunkt anzeigen zu lassen. Etwa alle Entwicklungen der letzten sieben Tage zum Kyoto Protokoll.

Mit diesen Entwicklungen, welche von keinem kommerziellen, öffentlichen System geboten werden, könnte es das NewsBlaster System in der nahen Zukunft schaffen eine breite Bekanntheit zu erlangen, und die Erforschung der Zusammenfassung mehrerer Dokumente im Nachrichtenbereich und auch auf allgemeine Texte bezogen vor das Licht eines breiteren Publikums zu bringen.

Literaturverzeichnis

[BEM02] Regina Barzilay, Noemie Elhadad, and Kathleen R. McKeown. Inferring Strategies for Sentence Ordering in Multidocument News Summarization. In *Journal of Artifical Intelligence Research*, volume 17, pages 35–55, 2002.

[BME99] Regina Barzilay, Kathleen R. McKeown, and Michael Elhadad. Information Fusion in the Context of Multi-Document Summarization. Technical report, Association for Computational Linguistics, 1999.

[dUoC05] Natural Language Processing Group der University of Columbia. NewsBlaster Webseite. http://newsblaster.cs.columbia.edu/, 2005.

[HGM00] Vasileios Hatzivassiloglou, Luis Gravano, and Ankineedu Maganti. An Investigation of Linguistic Features and Clustering Algorithms for Topical Document Clustering. In *Proceedings of the 23rd ACM SIGIR Conference on Research and Development in Information Retrieval*, 2000.

[HKH+01] Vasileios Hatzivassiloglou, Judith L. Klavans, Melissa L. Holcombe, Regina Barzilay, Min-Yen Kan, and Kathleen R. McKeown. SimFinder: A Flexible Clustering Tool for Summarization. Technical report, North American Association of Computational Linguistics, 2001.

[LdUoM05] Computational Linguistics and Information Retrieval Group der University of Michigan. NewsInEssence Webseite. http://www.newsinessence.com/, 2005.

[MBC+03] Kathleen McKeown, Regina Barzilay, John Chen, David Elson, David Evans, Judith Klavans, Ani Nenkova, Barry Schiffman, and Sergey Sigelman. Columbias Newsblaster: New Features and Future Directions. Technical report, Natural Language Processing Group der University of Columbia, 2003.

[MBE+02] Kathleen R. McKeown, Regina Barzilay, David Evans, Vasileios Hatzivassiloglou, Judith L. Klavans, Ani Nenkova, Carl Sable, Barry Schiffman, and Sergey Sigelman. Tracking and Summarizing News on a Daily Basis with

Columbias Newsblaster. Technical report, Human Language Technology Conference, 2002.

[MHB+01a] Kathleen R. McKeown, Vasileios Hatzivassiloglou, Regine Barzilay, Barry Schiffman, David Evans, and Simone Teufel. Columbia Multi-document Summarization: Approach and Evaluation. Technical report, National Institute of Standards and Technology, 2001.

[MHB+01b] Kathleen R. McKeown, Vasileios Hatzivassiloglou, Regine Barzilay, Barry Schiffman, David Evans, and Simone Teufel. Producing Biographical Summaries: Combining Linguistic Knowledge with Corpus Statistics. Technical report, European Association for Computational Linguistics (ACL/EACL), 2001.

[MKH+99] Kathleen R. McKeown, Judith L. Klavans, Vasileios Hatzivassiloglou, Regina Barzilay, and Eleazar Eskin. Towards Multidocument Summarization by Reformulation: Progress and Prospects. Technical report, American Association for Artificial Intelligence, 1999.

[Sha04] Jasper Sharp. Zatoichi Review. http://midnighteye.com/reviews/zatoichi.shtml, 2004.

[SNM02] Barry Schiffman, Ani Nenkova, and Kathleen McKeown. Experiments in Multidocument Summarization. Technical report, Human Language Technology Conference, 2002.

[Tec03] Copernic Technologies. Copernic Summarization Technologies White Paper. http://www.copernic.com/data/pdf/summarization-whitepaper-eng.pdf, 2003.

Florian Wild

9 Extraktion von Argumentationsprofilen aus Zeitungsartikeln

In diesem Kapitel geht es um die Untersuchung amerikanischer Zeitungsartikel anhand einer rein lexikalisch-linguistischen Analyse. Die Informationen des Textes, genauer gesagt Meinungen und Äusserungen von bestimmten Personen oder allgemeiner Entitäten, werden extrahiert und zur weiteren Verwendung in Profilen gespeichert. Im letzten Schritt, der Perkolation, wird die Informationsaufnahme durch einen virtuellen Leser simuliert, das heißt es wird untersucht, wie glaubwürdig die gefundenen Aussagen auf ihn wirken.

9.1 Einleitung

Amerikanische Zeitungsartikel unterscheiden sich von anderen schriftlichen Medien vor allem durch die häufige Verwendung der Reported Speech. Insbesondere durch den Online-Zugang von Zeitungsarchiven wie zum Beispiel dem des Wall Street Journals steht der breiten Öffentlichkeit eine gewaltige Masse an potentieller Information zur Verfügung, die jedoch von Hand niemals in akzeptabler Zeit untersucht werden könnte. Solch eine automatisierte Untersuchung wäre interessant, wenn man beispielsweise die Äußerungen bestimmter Politiker über mehrere Jahre hinweg verfolgen und deren Trends oder Meinungsänderungen aufzeigen wollte. Eine ähnliche Anwendungsmöglichkeit wäre die Untersuchung von Zeitungsartikeln politisch verschieden gerichteter Zeitungen zu einem bestimmten Thema, um so die Beeinflussung eines neutralen oder auch schon vorbelasteten Lesers zu simulieren.

Im Gegensatz zu einer statistischen Analyse soll im Ansatz von Bergler [Ber95] die Information genauer und besser durch eine rein lexikalische Analyse gewonnen werden. So wird ein Zeitungsartikel einer gezielten linguistischen Vorverarbeitung unterzogen, um die Information des Textes in einer aufgabenunabhängigen und wiederverwertbaren Repräsentation zu verwahren.

In amerikanischen Zeitungsartikeln ist, wie schon erwähnt, die *Reported Speech*, Reported Speech

die die direkte und indirekte Rede im Deutschen vereint, ein sehr häufig auftretendes Phänomen. Mit ihrer Hilfe werden in einer standardisierten Weise Meinungen, Blickpunkte und Aussagen dargestellt. In Einzelfällen finden sich Zeitungsartikel, die bis zu 90% aus Reported Speech bestehen. Um also einen Zeitungsartikel verstehen zu können, wird man um eine genaue Untersuchung der Reported Speech nicht herumkommen. Verben der Verlässlichkeit beziehungsweise der Kognition zum Beispiel werden als Hauptverb der Reported Speech benutzt, um unter anderem klarzumachen, dass der überlieferten Information eventuell nicht zu glauben ist. Damit ist man auch schon beim Problem der Reported Speech angekommen. Es wird von etwas berichtet, das ein anderer von sich gegeben hat. Der Reporter könnte eine versteckte Meinung durch Wahl des Kontexts mit ins Spiel bringen oder die ursprüngliche Quelle könnte sogar gelogen haben.

Eine genaue Analyse der Reported Speech in Zeitungsartikeln macht klar, dass sie vor allem deshalb benutzt wird, um einen Beweis der eingebetteten Information zu liefern. Es wird eine bestimmte Quelle zitiert und damit eine Umgebung für die Interpretation der Information geliefert. Der einleitende Satz der Reported Speech liefert dabei die nötige Hintergrundinformation.

Wie in Abbildung 9.1 zu sehen, wird in den folgenden Abschnitten die Reported Speech genauer betrachtet. Der erste Schritt, die Vorverarbeitung des Zeitungsartikels, durch welche Ergebnisse wie Tokens und Part-of-Speech-Tags gewonnen werden, wurde schon in früheren Kapiteln behandelt. In einer genauen Analyse wird nun ein Reported Speech Satz in seine Einzelteile zerlegt und diese in einem sogenannten Profil (vergleiche Doandes [Doa03]) wie im folgenden Beispiel abgelegt:

> (S1) Yesterday, Senator Packwood acknowledged, „We don’t have the votes for cloture today.“

```
Profile S1:
reported speech: We don’t have the votes for cloture today.
textsource: Senator Packwood
textverb: acknowledged
circumstantial: Yesterday
semantic dimensions:
     EXPLICIT:yes,
     POLARITY:positive,
     SPEECHACT:inform,
     STRENGTH:high
```

Da ein Zeitungsartikel aus vielen Quellen und Zitaten besteht, werden so nach und nach aus allen Aussagen Profile erstellt. Diese Menge an Informationen gliedert das System nun mittels sogenannter Belief Diagramme, mit deren Hilfe die Zusammenhänge und Verschachtelungen der Aussagen genau strukturiert werden können. Im letzten Abschnitt wird dann schlußendlich untersucht, welche Aussagen und Zitate ein möglicher Leser als glaubhaft ansehen könnte, um sich mit ihrer Hilfe eine Meinung zur Thematik des Zeitungsartikels zu bilden. In die Berechnung der Glaubwürdigkeit einer Aussage, Perkolation genannt (vergleiche Gerard [Ger00]), fließen dabei verschiedene Faktoren wie die Glaubwürdigkeit der Quelle, des Reporters und der Information selber mit ein. Am Ende der Untersuchung erhält man so eine Simulation des Eindrucks, den ein Leser von einem bestimmten Artikel haben könnte.

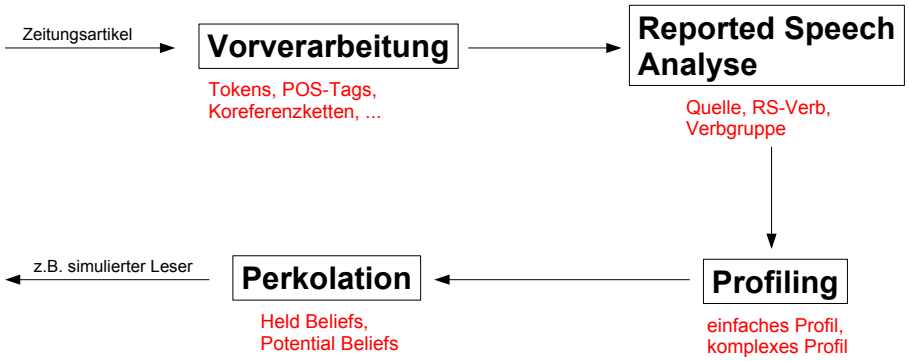


Abbildung 9.1: Übersicht über die einzelnen Schritte vom Zeitungsartikel über die lexikalische Auswertung hin zum simulierten Leser

9.2 Reported Speech

Wie schon in der Einleitung angesprochen, zeichnen sich amerikanische Zeitungsartikel durch einen besonderen Stil aus: Der häufigen Verwendung der Reported Speech. Diese kann leider nicht mit einigen wenigen syntaktischen Regeln zusammengefasst werden. Charakteristisch jedoch sind die verschachtelten Sätze und vor allem die überladenen Nominalphrasen, mit deren Hilfe der Reporter viel zusätzliche Information über bestimmte Entitäten in einen Satz stecken kann. Desweiteren sind die Verben, die für die Einleitung der Reported Speech verwendet werden, eine relativ deutlich abzugrenzende Untergruppe aller Verben, was für die automatisierte Erkennung der Reported Speech natürlich sehr wichtig ist.

Eine Validierung der Genauigkeit oder der Verlässlichkeit der gegebenen Information ist nun sowohl mit Hilfe der Quelle der Reported Speech, als auch des Verbs bzw. der ganzen Verbgruppe möglich. Im folgenden wird genauer auf die einzelnen Komponenten der Reported Speech eingegangen.

9.2.1 Syntax

Doandes beschreibt in [Doa03] die zwei verschiedenen Informationskomponenten, die ein Satz eines Zeitungsartikels grundsätzlich enthält: Die Primärinformation, die normalerweise von einem Experten oder einem Zeugen übermittelt wird, der entweder interviewt wurde oder sich geäussert hat, und die nebenläufige Information, die normalerweise vom Reporter miteingebracht wird. Letztere enthält Details über die Umstände bzw. die Situation, in der die Aussage getätigt wurde. Damit ist es möglich, die Primärinformation unter einem bestimmten Blickwinkel zu betrachten. In Zeitungsartikeln werden diese beiden Komponenten syntaktisch getrennt.

Reported Speech ist in vielen verschiedenen syntaktischen Ausführungen möglich, wobei sich die Struktur oft über mehrere Sätze erstreckt. Es kommen dabei sowohl Varianten des direkten Zitats als auch der Paraphrasierung vor. Zusätzlich variiert die Position der Quelle und des Reporting Verbs.

Anbei einige typische Strukturen mit Beispielen[1], wobei mit *RS-Verb* im Folgenden das Verb des einleitenden Satzes der Reported Speech abgekürzt wird:

[1] aus dem Seattle Post-Intelligencer vom 16. und 17. Februar 1999

a) <Quelle> <RS-Verb> „<Aussage>“

b) „<Aussage>,“ <RS-Verb> <Quelle>

c) „<Aussage>,“ <Quelle> <RS-Verb>
wie in:

„She would be terrific,“ Clinton said.

d) „<Teil der Aussage>,“ <Quelle> <RS-Verb>, „<Rest der Aussage>“
wie in:

„In Turkey,“ Ocalan insisted, „the only avenue open to the Kurds is to take ups arms.“

e) „<Teil der Aussage>,“ <RS-Verb> <Quelle>, „<Rest der Aussage>“

f) <Quelle> <RS-Verb>, „<Teil der Aussage>“ <Paraphrase>

g) „<Teil der Aussage>,“ <Quelle> <RS-Verb>, <Paraphrase>

h) <Quelle> <RS-Verb> (that) <Paraphrase>

Die nebenläufige Information, also die Information, die durch den Reporter noch zusätzlich zur Primärinformation mitgeliefert wird, setzt sich zusammen aus der Quelle, dem Hauptverb der Reported Speech und anderen Sekundärinformationen. Auch wenn man meinen könnte, solch eine nebenläufige Information liesse sich leicht finden, so ist es doch in seltenen Fällen sehr schwierig zu unterscheiden, ob eine Zusatzinformation oder eine Paraphrase vorliegt. Die Lösung dieses Problem ist für ein System nur sehr schwer möglich, glücklicherweise kommt das aber so gut wie nie vor. Gut behelfen kann man sich mit einer Heuristik, die die häufigsten Vorkommen von Zusatzinformationen in Zeitungsartikeln abdeckt.

Folgende zwei Haupttypen treten laut Doandes [Doa03] in bis zu 60% der Fälle auf:[2]

a) <Quelle>, <Zusatzinformation>, <RS-Verb>
wie in:

Paul Keough, *acting regional director for the Environmental Protection Agency in Boston*, says, „. . . “

b) <RS-Verb> <Quelle>, <Zusatzinformation>
wie in:

„. . . “, says John Skinner, *a University of Virginia economist*.

Der Hauptsatz der Reported Speech besteht aus vor allem zwei Komponenten: Der Quelle der Information und dem Hauptverb, dem sogenannten Reporting Verb. Diese beiden reichen aus, um die Information in einer Umgebung darzustellen, die ihre Glaubwürdigkeit untersuchbar macht.

9.2.2 Quelle

Das primäre Ziel der *Quelle* ist es, eine Glaubwürdigkeit beim Leser aufzubauen. Der Reporter macht also dem Leser durch die Beschreibung der Quelle klar, wie er die Reported Speech interpretieren soll. Grundsätzlich nimmt der Leser im Bereich des Journalismus an, dass der Reporter kompetente Quellen ausgewählt hat. Um nun

[2] Beispiele aus dem Wall Street Journal vom 27.10.89

dem Leser einen Eindruck von dieser Kompetenz zu verschaffen, beinhaltet die Beschreibung der Quelle oft zusätzliche Information über die Autoriät oder Position der Quelle.

Folgende drei Aspekte sprechen dabei für die Glaubwürdigkeit einer Quelle (vergleiche [Ber95]):

- die Identität oder die persönlichen Eigenschaften der Quelle
- die offizielle Position oder Rolle der Quelle
- den Bezug oder die Relevanz der Quelle zum Thema

Normalerweise sind diese drei Aspekte dem Leser nicht unbedingt alle bekannt, sondern er muss sich auf die Beschreibung verlassen, die er vom Reporter bekommt.

Es kann dabei durchaus vorkommen, dass ein Satz alle Aspekte enthält, wie das folgende Beispiel zeigt:

> *The country's hard-line Communist leadership reaffirmed the nation's commitment to socialism, but said demands for the types of changes sweeping much of Eastern Europe „are open to discussions".* (WSJ, 10.05.89)

Bergler [Ber95] stellte fest, dass diese Aspekte nach ihrer Wichtigkeit geordnet werden können: Dabei gilt, dass der Bezug zur Thematik am wichtigsten ist, die Position und die Identität eher weniger wichtig. Sollte natürlich bekannt sein, dass eine Person mit Bezug zum Thema grundsätzlich die Unwahrheit von sich gibt, gilt diese Ordnung nicht.

9.2.3 Verben

Die Untersuchung des *Hauptverbs* der Reported Speech ist wichtig, da die Auswahl durch den Reporter dessen Einschätzung des Kontexts der Information widerspiegelt. Man kann die Verben der Reported Speech, wie in Tabelle 9.1 zu sehen, nach folgendem Paradigma einteilen (vergleiche [Doa03]): vokal (vocal), neutral (neutral), kontextbezogen (contextual) und mit pragmatischer Absicht (pragmatic intent).

Hauptverb

vocal	neutral	contextual	pragmatic intent
cry	say	allude	accuse
call	tell	argue	admit
mumble	report	contend	claim
mutter	relate	insist	deny
scream	announce	reiterate	promise
shout	release	reply	joke
stammer	state	dispute	assure
stutter	add	ask	pledge

Tabelle 9.1: Einordnung von Reported-Speech Verben nach Art ihrer Benutzung

Neutral bedeutet dabei, dass man der Wahl des Verbs keine besondere Bedeutung zuschreiben kann. Vokal dagegen ist als Spezifizierung in Bezug auf die physischen und emotionalen Aspekte des Verbs zu verstehen. Ist die Aussage in einem leisen Ton oder klar und deutlich getroffen worden? Kontextbezogen zeigt eine Kohärenz der ursprünglichen Aussage im ursprünglichen Kontext. Die pragmatische Absicht

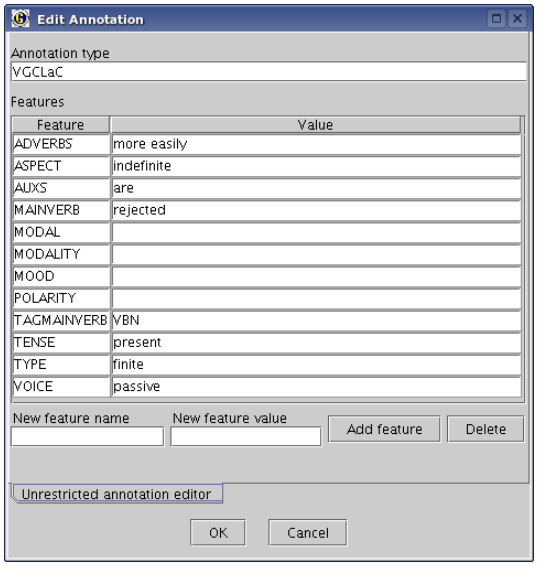


Abbildung 9.2: Ergebnisse des CLaC Verbgroup Chunkers für die Verbgruppe „. . . are rejected more easily . . .“

dagegen geht über die textuelle Beschreibung hinaus, indem eine bestimmte Intention des Zitierten dargestellt wird. Zum Beispiel bei „joke“ kann man davon ausgehen, dass die ursprüngliche Quelle die Aussage nicht ernst gemeint hat, der Informationsgehalt beziehungsweise die Glaubwürdigkeit also gering ist.

Zusätzlich zur eben genannten Einordnung können die Verben mehrere verschiedene semantische Dimensionen annehmen, die für jedes Verb, das in der Reported Speech verwendet wird, in einer Tabelle abgelegt sind. So wird zum Beispiel bei der Lautstärke oder der Formalität der Aussage unterschieden, ob sie in der Öffentlichkeit oder im privaten Umfeld getätigt wurde und ob das Verb positiv oder negativ verwendet wurde (zum Beispiel „The man *didn't* pretend that . . . “). Ein Teil der Dimensionen bezieht sich also auf die Umgebung der Aussage („During the press conference, President Bush declared that . . . “), der andere klärt die Beziehung zwischen Quelle und Aussage. Auf diese Art und Weise kann alleine durch die Analyse des Verbs sehr viel Information gewonnen werden.

9.2.4 Verbgruppe

Neben der Untersuchung des Hauptverbs alleine darf nach Doandes [Doa03] natürlich der Rest der *Verbgruppe* nicht ausser Acht gelassen werden, denn diese liefert einige zusätzliche Informationen. Grundsätzlich setzt sich die Verbgruppe aus einem Hauptverb und bis zu vier Hilfsverben zusammen. Hilfsverben können modal (can, could, would, might, . . .) oder anderer Art sein (have, can, do). Mit ihrer Hilfe wird der Aussage der Charakter einer Möglichkeit, Wahrscheinlichkeit oder ähnlichem gegeben. Das Hauptverb letztendlich enthält die lexikalische Information und beschreibt, was dem Subjekt widerfährt beziehungsweise was es tut. Mit Hilfe der Analyse seiner semantischen Dimensionen (vergleiche Abbildung 9.2) kann man sich ein genaueres Bild über die Glaubwürdigkeit der Aussage machen.

Es wird unter anderem der Tempus der Verbgruppe analysiert, also die Unterschei-

dung zwischen Gegenwart, Vergangenheit und Zukunft. Mithilfe des Tempus wird die Position des Ereignisses auf der Zeitachse festgelegt. Zusätzlich zum Tempus wird der Modus untersucht, ob also Indikativ, Konjunktiv oder Imperativ vorliegt. Auf diese Art und Weise kann festgestellt werden, wie der Zitierte die Situation bezüglich ihrer Wahrscheinlichkeit beurteilt oder ganz einfach erkannt werden, ob er der folgenden Aussage einen Wahrheitsgehalt zuordnet. Weiterhin wird untersucht, ob grammatikalisches Aktiv oder Passiv verwendet wurde und ob die Aussage positiv oder verneint vorliegt. Diese Dimensionen helfen dabei, verschiedene Aussagen gegeneinander abzugrenzen oder miteinander in Beziehung zu setzen.

9.3 Profilerstellung

Nach der lexikalischen Analyse der Reported Speech Sätze, wie im vorherigen Kapitel beschrieben, müssen die Informationen nun in eine verarbeitungsfreundliche Struktur gebracht werden. Bergler [Ber92] definiert *Profilerstellung (Profiling)* als den Prozess der Umwandlung von Sätzen, die Reported Speech enthalten, in eine strukturierte Repräsentation, genannt Profil. Ein Profil enthält eine Liste aller Eigenschaften, die ein Text von einer bestimmten Entität preisgibt. Das können nun im Falle eines CEOs dessen blonde Haarfarbe sein, seine politische Einstellung oder seine geäußerte Meinung zu einem bestimmmten Thema. Mehrere Entitäten eines Textes haben jeweils getrennte Profile. Diese Sammlung von Informationen kann dazu verwendet werden, die Entität in das Gesamtbild des Textes einzuordnen, sei es, um sie von anderen Entitäten abzugrenzen oder gemeinsame Gruppen von Entitäten zu bilden. So könnte man zum Beispiel Profile gruppieren, deren Meinungen zu einem bestimmten Thema übereinstimmen.

9.3.1 Einfaches Profil

Die Komponenten eines *einfachen Profils* (vergleiche [Ber92]) sind identisch mit den syntaktischen Komponenten eines einzelnen typischen Reported Speech Satzes. Die vorangegangene lexikalische Analyse macht es möglich, die einzelnen Satzteile zu extrahieren und aus ihnen ein Profil zu erstellen. So werden neben Quelle, Verbgruppe und der Primärinformation auch Informationen einsortiert, die das Ganze zeitlich oder räumlich einordnen, um auch im Nachhinein einzelne Profile interpretieren zu können oder Beziehungen zwischen einzelnen Profilen besser verarbeiten zu können. Das folgende Beispiel zeigt einen Satz, aus dem zwei Profile erstellt werden, da sowohl ein Teil in direkter als auch ein Teil in indirekter Rede vorkommt:

> (S1) „The editorial side is complicated," Ms. Salembier added during the briefing, (S2) saying that editorial layoffs will be decided later. (WSJ, 27.10.89)

Die beiden *einfachen Profile* dazu könnten folgendermassen aussehen, wie in Abbildung 9.3 gezeigt. Man sieht, dass das zweite Profil keine Quelle enthält, da diese Quelle schon für das erste Profil verwendet wurde. Daher ist es notwendig, mit Hilfe komplexerer Profile nicht nur einzelne Sätze, sondern einen ganzen Zeitungsartikel abzudecken und alle darin vorkommenden Informationen miteinander zu verbinden.

```
Profile S1:
text: „The editorial side is complicated, Ms. Salembier added
during the briefing“
textsource „Ms. Salembier“
textverb „added“
reportedspeech „The editorial side is complicated“
textcircumstance „during the briefing“
textdirectquote „The editorial side is complicated“
textindirectquote „ “

Profile S2:
text: „saying that editorial layoffs will be decided later“
textsource „“
textverb „saying“
reportedspeech „that editorial layoffs will be decided later“
textcircumstance „“
textdirectquote „“
textindirectquote „that editorial layoffs will be decided
later“
```

Abbildung 9.3: Beispiel für einfache Profile

9.3.2 Komplexes Profil

Im Gegensatz zu einfachen Profilen können *komplexe Profile* ineinander verschachtelt werden, um so zum Beispiel Quellen-Beziehungen wie Firmenzugehörigkeit oder ähnliches auszudrücken. Mit Hilfe von Koreferenz-Analysen (siehe Kapitel 5) können so verschiedene Entitäten zu einem einzigen Profil mit mehreren Informationsfeldern zusammengefasst werden. In einem solchen Fall wird das Koreferenzfeld mit einem entsprechenden Verweis zu anderen einfachen oder komplexen Profilen versehen.

Nachfolgend ein schlichtes Beispiel aus dem Wall Street Journal vom 27.10.89 mit einer entsprechenden graphischen Repräsentation (Abbildung 9.4, vergleiche [Doa03]):

> *In late September, the Post, a well established newspaper, announced it was canceling its Sunday edition. Valerie Salembier, president of the Post, said the Sunday circulation has reached only about 250,000. „In any other city, 250,000 would be considered great, but it just wasn't enough in New York," added Peter Kalikow, owner and publisher of the Post.*
> *Ms. Salembier said about 30 people in circulation, ad sales and other business departments would lose their jobs. „What we don't know about is the number of layoffs on the editorial side," she said. „The editorial side is complicated," Ms. Salembier added during the briefing, saying that editorial layoffs will be dedided later.*

Man kann leicht erkennen, dass es sich um drei Quellen in einer verschachtelten Struktur handelt: The Post, Valerie Salembier und Peter Kalikow, wobei die letzteren der Post angehören.

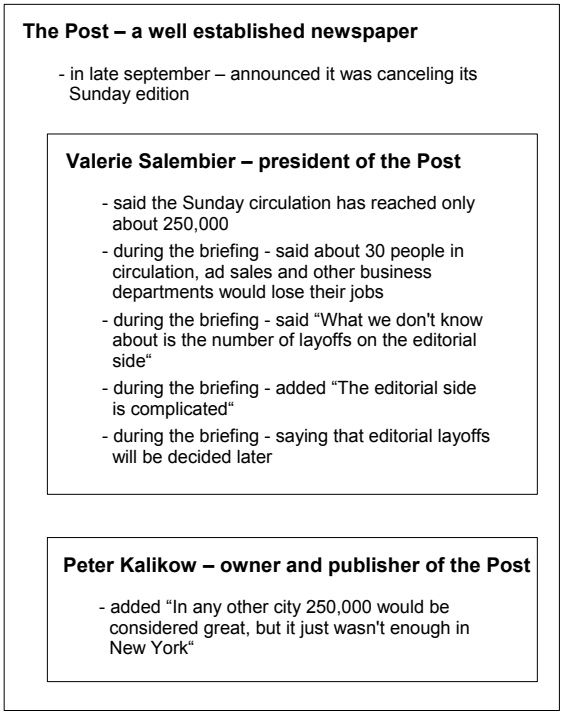


Abbildung 9.4: Beispiel eines komplexen Profils

9.3.3 Opposing und Supporting Groups

Wenn ein Autor einen Artikel verfasst, hat er ein bestimmtes Ziel vor Augen. Er will eine spezielle Thematik vermitteln. Darüber hinaus kann er sie entweder unterstützen oder ihr entgegentreten. In letzterem Fall wird also die Hauptquelle des Artikels der Absicht des Reporters entsprechen und durch eventuell mehrere zusätzliche Aussagen von anderen Quellen gestärkt werden. Alle zusammen bilden nach Bergler eine sogenannte *Supporting Group*. Dem entgegen stehen dann meist mehrere Aussagen von verschieden Quellen, die in einer *Opposing Group* zusammengefasst werden. Solche Gruppierungen sind sehr wichtig für das Verständnis eines Textes. Sie bilden einen sogenannten *lokalen Kontext* (local context), in dem unter anderem eine Kohärenzanalyse oder andere Fragen der Zugehörigkeit von Aussagen untereinander erleichtert werden. Beobachtet man eine solche Gruppe über mehrere Texte, kann man beispielsweise untersuchen, wie sich die Aussagen von bestimmten Gruppierungen über eine längere Zeitdauer verändert haben.

Supporting Group
Opposing Group

Im folgenden ist ein Beispiel[3] angeführt, das zwei gegensätzliche Meinungsgruppen beinhaltet (vergleiche Abbildung 9.5, Bergler [Ber95]):

> *(S1) Analyst Marc Spencer said the naming of Mr. Ukropina represented a conservative move by an usually conservative utility concern.*
> *(S2) Unlike some companies, Pacific Enterprises has „made no major errors*

[3] aus dem Wall Street Journal vom 10.05.89

moving outside their area of expertise," said Craig Hill, an analyst with Seidler Amdec Securities Inc. in Los Angeles.
(S3) „None of the company's businesses are positioned to do well in the coming year," insisted Paul Milbauer, an analyst with C.J. Lawrence in New York.

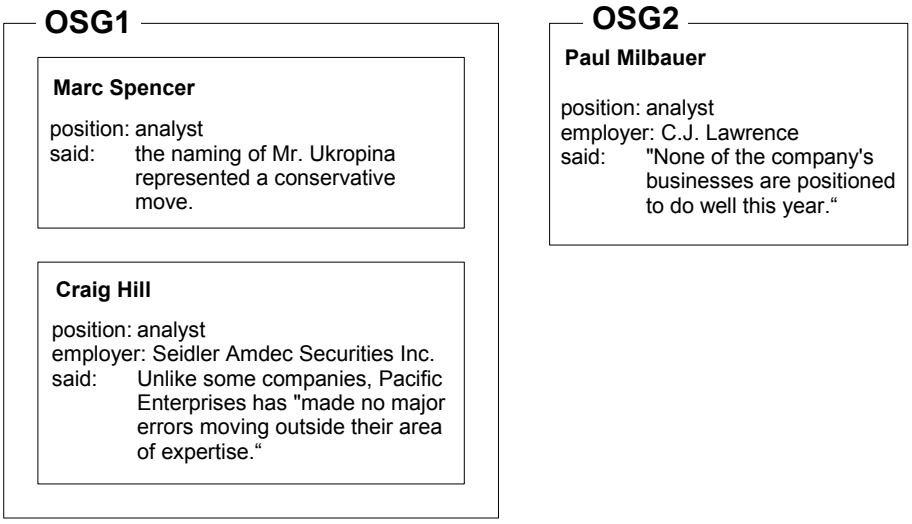


Abbildung 9.5: Beispiel zweier Opposing Groups

Im obigen Beispiel wurden die Profile nach ihrer Haltung gegenüber dem Unternehmen *Pacific Enterprises* angeordnet. Profil S1 und S2 haben dabei eine neutrale oder auch positive Aussage gegenüber dem Konzern gemein und bilden somit eine Supporting Group. Dem gegenüber steht die negative Analyse von Profil S3. Ein eindeutiges Indiz der konträren Beziehung der beiden resultierenden Opposing Supporting Groups ist unter anderem die Wahl des Reporting Verbs *insisted* im Profil S3.

9.4 Perkolation

In den vorigen Unterkapiteln wurde ein Zeitungsartikel auf seine Reported Speech Sätze hin untersucht und die gesammelten Informationen in eine verarbeitungsfreundliche Repräsentation, die Profile, überführt. In diesem Abschnitt geht es nun darum zu modellieren, wie ein Leser Informationen aufnimmt und sich mithilfe der größeren oder geringeren Glaubwürdigkeit letzterer eine Meinung bildet. Gerard [Ger00] betrachtet in diesem Zusammenhang drei verschiedene Sichtweisen (viewpoints): Die des Systems, das von aussen den Vorgang betrachtet, die des Reporters, der über bestimmte Quellen die Informationen zur Verfügung stellt und die des Lesers, der Schritt für Schritt die Informationen, die er vom Reporter bekommt, evaluiert. Für diesen komplexen Prozess muss eine umfangreiche Logik aufgestellt werden, denn nicht oft kommt es vor, dass sich Reporter oder Leser in der Thematik des Artikels schon gut auskennen und die genannten Informationen in einem besonderen Licht betrachten. So kann es gut vorkommen, dass zwei Leute einen Text lesen und bei dessen Diskussion ein völlig unterschiedliches Verständnis an den Tag legen. Im Folgenden wird zu

Beginn eine Strukurierungsmöglichkeit der verschiedenen Meinungen (Beliefs) aufgezeigt, auf welcher dann in einem weiteren Schritt ein Evaluierungsprozess, genannt Perkolation, arbeiten wird.

9.4.1 Belief Diagramme

Für eine Modellierung des Systems eignen sich sogenannte *Belief Diagramme* (vergleiche Ballim und Wilks [BW92]). In diesem Modell ist es möglich, Meinungen (sogenannte *Beliefs*) in Unterstrukturen und Umgebungen zu verschachteln und so miteinander in Beziehung zu setzen oder abzugrenzen, in jedem Fall aber einer Person oder Entität zuzuordnen. In solch einer Umgebung können beispielsweise die Meinungen zweier Personen gut verglichen werden, da nur deren Unterumgebungen untersucht werden müssen. Diese Belief Diagramme bestehen aus verschachtelten Strukturen aus Themen (Topics) und Blickpunkten (Viewpoints). In Abbildung 9.6 sind beispielsweise drei Viewpoints vertreten. Das sind im Einzelnen das System, das die Thematik von aussen betrachtet, der Leser, der in diesem System simuliert wird und schlußendlich der Reporter, der eine bestimmte Information übermittelt. In diesem Stadium des Diagramms gibt es ein Thema, nämlich das Gras, zu dem den einzelnen Protagonisten unterschiedliche Informationen vorliegen. So hat der Reporter den Belief „ist_Grün()“, der Leser jedoch „ist_nass()“. In dieser Art und Weise existieren Belief Diagramme direkt nach dem Profiling Prozess. Momentan kann man das Diagramm mit folgendem Wortlaut interpretieren: „Das System glaubt, dass der Leser glaubt, dass das Gras nass ist und dass der Reporter glaubt, dass das Gras grün ist.“

Beliefs

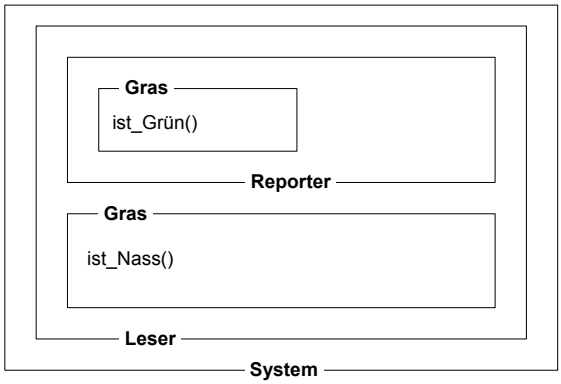


Abbildung 9.6: Belief Diagramm vor dem Perkolationsprozess

Mit dieser Art Diagramme können ganze Zeitungsartikel in ihre Quellen und Meinungen zerlegt werden und dienen dann als Grundlage für die nachfolgende Untersuchung, bei der Schritt für Schritt überprüft wird, ob der Leser im System vorhandene Beliefs als seine eigenen übernehmen kann.

9.4.2 Perkolationsprozess

Um nun die verschiedenen Blickpunkte auf andere Entitäten zu übertragen, die diesen Blickpunkten im Diagramm übergeordnet sind, kommt folgende Regel zum Einsatz (vergleiche [Ger00]):

Definition 9.4.1 *Der Blickpunkt von X bezüglich Thema Y kann als eigene Ansichtsweise übernommen werden, wenn nicht ausdrücklich ein Beweis existiert, der das verbietet.*

So kann der Leser zum Beispiel annehmen, dass die Sicht des Reporters auf ein bestimmtes Themengebiet die gleiche wie die eigene ist, genauso wie der Reporter annehmen kann, dass die Sicht der zitierten Quelle dieselbe ist wie seine eigene, solange nichts grundsätzlich dagegen spricht. Abbildung 9.7 soll dies verdeutlichen. Damit auf dem Weg der Perkolation nicht verloren geht, von wem die Aussage ursprünglich stammt, wird parallel zur Weiterreichung eine Quellenliste (Source List) mitgeführt.

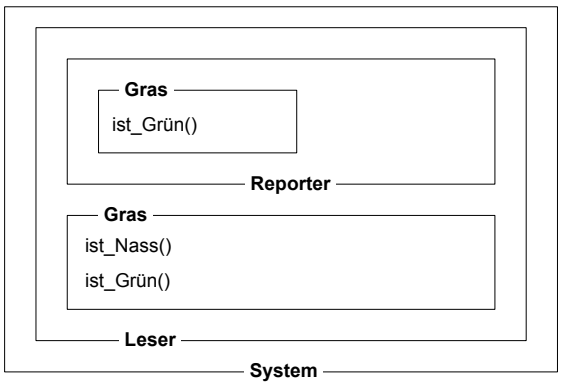


Abbildung 9.7: Belief Diagramm nach dem Perkolationsprozess

9.4.3 Erweiterte Belief Diagramme

Liest ein Leser einen Artikel, so kommen darin viele unterschiedliche, oft gegensätzliche Meinungen vor, die er nicht ohne weiteres als Tatsachen glauben kann. Er wird daher auf bestätigende oder widerlegende Meinungen warten, um sich einen sicheren Eindruck der Situation erstellen zu können. Man muss also unterscheiden zwischen Beliefs, die der Leser sofort akzeptieren kann oder schon ausreichend verifiziert hat (sogenannte *Held Beliefs*) und solchen, die weiterer Information bedürfen (sogenannte *Potential Beliefs*). Grundsätzlich wird im Falle von Zeitungsartikeln bei der Wissensgewinnung und Perkolation eine neue Information im Viewpoint des Lesers als potentielle Information eingestuft, um ihn so gegen unverlässliche oder gegensätzliche Information zu schützen. Ein einfaches Belief Diagramm reicht dafür nicht mehr aus, es werden *erweiterte Belief Diagramme* (vergleiche [Ger00]) benötigt, die sowohl Held als auch Potential Beliefs unterbringen. Im Diagramm geschieht dies durch Abgrenzung mittels einer senkrechten Linie (vergleiche Beispiel in Abbildung 9.8). Dadurch, dass hier nur die Simulation des Lesers von Bedeutung ist, wird auch nur in seiner Ansichtsweise nach Potential und Held Beliefs differenziert.

Held Belief
Potential Belief
Erweitertes Belief Diagramm

9.4.4 Belief Promotion

Durch die Unterscheidung in die beiden Arten von Beliefs ist es dem simulierten Leser möglich, neue Meinungen als Potential Beliefs zu halten und sich eventuell später gegensätzlich zu entscheiden, je nachdem, wie er die Glaubwürdigkeit der einzelnen

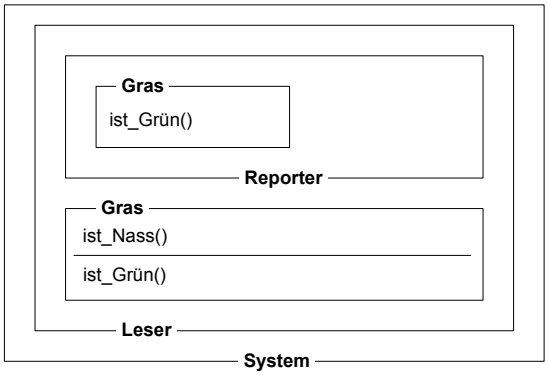


Abbildung 9.8: Erweitertes Belief Diagramm

Informationen bewertet. Den Prozess der Umwandlung eines Potential Beliefs in einen Held Belief nennt man *Belief Promotion*. Durch diesen zweiten Schritt des Perkolationsprozesses wird mittels einer Heuristik evaluiert, ob die Glaubwürdigkeit eines Potential Beliefs ausreicht, um ihn zum Held Belief zu befördern.

Belief Promotion

Bei dieser Untersuchung spielen nach Gerard [Ger00] vier Faktoren eine Rolle:

- die Glaubwürdigkeit der Quelle für den Leser (S1)
- die Glaubwürdigkeit der Information für den Reporter (I)
- die Glaubwürdigkeit der Quelle für den Reporter (S2)
- die Glaubwürdigkeit des Reporters für den Leser (R)

Bei der Untersuchung der Glaubwürdigkeit der Quelle für den Reporter muss man beachten, dass der Reporter durch die Wahl der Worte die Möglichkeit besitzt, sein Vertrauen in die Quelle auszudrücken. Die Identität, Expertise, Erfahrung und Relevanz der Quelle können eine wichtige Einsicht in die Situation der originalen Aussage bedeuten und dem Leser dabei helfen, die Glaubwürdigkeit der Quelle festzulegen. Der Reporter kann seine eigene Einschätzung der Quelle mit einbringen und dem Leser so Hinweise geben, wie er den Text zu verstehen hat. Das ist besonders dann wichtig, wenn der Leser noch keine eigene Meinung zur zitierten Entität mitbringt, sondern sich voll auf den Reporter verlassen muss. Die entscheidenden Informationen zur Glaubwürdigkeit der Quelle werden hierbei durch den einleitenden Satzes geliefert.

Wenn es darum geht, die Glaubwürdigkeit der Information für den Reporter zu untersuchen, muss das Reporting Verb genau analysiert werden. Es wird verwendet, um die Art und Weise, die Intention und Ausdruckskraft der originalen Äusserung auszudrücken. Das Verb an sich zeigt an, wie der Reporter die Aussage einschätzt und wie er sie dem Leser vermitteln will.

Für die beiden übrigen Faktoren, also die Einschätzung der Quelle und des Reporters durch den Leser, kann die lexikalische Analyse nicht zur Hand genommen werden. Hierfür ist Wissen oder ein Glaubwürdigkeitsmass aus früheren Zeitpunkten nötig, auf das zurückgegriffen werden kann.

Nachdem die verschiedenen Glaubwürdigkeitsfaktoren untersucht wurden, stellt sich die Frage, wie sie zu einer Heuristik im Sinne einer quantitativen Evaluierung

kombiniert werden können. Die Konfidenzkriterien könnten dabei einen numerischen Wert zugewiesen bekommen und in einer mathematischen Formel zu einem Ergebnis verrechnet werden. Sollte dieses Ergebnis über einem vorher festgelegten sinnvollen Schwellwert liegen, würde der Potential Belief zum Held Belief.

Im Folgenden ist ein solches Beispiel aufgeführt, wobei S1, S2, R und I die Werte HIGH (1.5), NEUTRAL (1) und LOW (0.5) annehmen können (vergleiche [Ger00]):

Glaubwürdigkeit = (S1 + (R * I) + S1 * (R * S2))

7.125 ist in dieser Formel der größte Wert, der erreicht werden kann, wenn alle Werte auf HIGH stehen, der kleinste ist 0.875. Die Differenz dieser beiden Werte beträgt 6.25 und kann in drei gleichgroße Abschnitte aufgeteilt werden, um so dem Ergebnis wieder eine LOW (0.875 bis 2.954), NEUTRAL (2.955 bis 5.034) oder HIGH (5.035 bis 7.125) Wertung zuzuordnen. Nimmt man nun das Zahlenbeispiel 9.9, käme für die Glaubwürdigkeit ein Wert von 5.5 heraus, was eine Einstufung in den Bereich HIGH bedeutet, also eine Hochstufung dieses Potential Beliefs zum Held Belief.

```
POTENTIAL Beliefs about President
  sexual_relationship(president,intern)
    Source 1: Intern
    Reporter: P. Jennings
      Reporter's confidence in the Source: HIGH
      Reporter's confidence in the information: HIGH
      Reader's confidence in the source: NEUTRAL
      Reader's confidence in the reporter: HIGH
```

Abbildung 9.9: Zahlenbeispiel eines Potential Beliefs

9.5 Zusammenfassung

Um von einem Zeitungsartikel über ein bestimmtes Thema zu einem simulierten Leser zu kommen, benötigt man zu Beginn eine relativ aufwändige Vorverarbeitung, die die einzelnen Satzbaukomponenten herausfiltert, insbesondere jene, die für die Reported Speech so wichtig sind, wie beispielsweise Quelle und Verbgruppe. Die so gewonnene Strukturinformation kann in verschieden komplexen Profilen sinnvoll dargestellt und zur weiteren Verwendung gespeichert werden.

Aufbauend auf den Profilen können mit Hilfe von Belief Diagrammen die verschiedenen Sichtweisen und Meinungen der Entitäten des Textes voneinander abgegrenzt werden und Schritt für Schritt durch den Perkolationsprozess vom Reporter beziehungsweise der Quelle auf den virtuellen Leser übertragen werden. Die Glaubwürdigkeitsheuristik legt dabei fest, welche der genannten Äußerungen der Leser als neue eigene Meinung übernehmen kann und welche nicht. Die dafür benötigten Glaubwürdigkeitskriterien werden aus den vom Reporter gewählten Kontextinformationen beziehungsweise der Wahl seiner Worte und dem schon vorhandenen Meinungsbild des Lesers zum Thema gewonnen. Diese bestimmen letztendlich das Bild, das der Leser sich vom vorliegenden Zeitungsartikel gebildet hat.

Dieses hier vorgestellte System könnte zum Beispiel dafür verwendet werden, einen speziell trainierten virtuellen Leser aus einer grossen Menge an über eine Suchmaschine gefundenen Texten automatisiert diejenigen heraussuchen zu lassen, die für

den Benutzer von Interesse sein könnten. Die Zeitersparnis könnte dabei gewaltig sein, bräuchte ein Mensch mit blossem Auge doch sehr viel länger, alle zur Verfügung gestellten Links zu durchforsten.

Literaturverzeichnis

[BDGW04] S. Bergler, M. Doandes, C. Gerard, and R. Witte. Attributions. In Yan Qu, James G. Shanahan, and Janyce Wiebe, editors, *Proceedings of AAAI Spring Symposium on Exploring Attitude and Affect in Text*, Stanford, California, USA, March 22–25 2004.

[Ber92] S. Bergler. *Evidential Analysis of Reported Speach*. PhD thesis, Brandeis University, Massachusetts, 1992.

[Ber95] S. Bergler. From Lexical Semantics to Text Analysis. In P. Saint-Dizier and E. Viegas, editors, *Computational Lexical Semantics*, Cambridge, 1995. UK: Cambridge University Press.

[BW92] A. Ballim and Y. Wilks. *Artificial Believers*. Lawrence Earlbaum Associates, Hillsdale, New Jersey, 1992.

[Doa03] M. Doandes. Profiling For Belief Acquisition From Reported Speech. Master's thesis, Concordia University, Montreal, Quebec, Canada, 2003.

[Ger00] C. Gerard. Modelling Readers Of News Articles Using Nested Beliefs. Master's thesis, Concordia University, Montreal, Quebec, Canada, 2000.