Seminararbeit
Suche in digitalen Korpora

Karsten Hütter
huetter@informatik.hu-berlin.de

Björn Schümann
schueman@informatik.hu-berlin.de

Betreuer:
Alexander Karosseit

1. April 2007

Inhaltsverzeichnis

1 Einleitung

Die Vorliegende Arbeit entstand im Rahmen des Seminars *Qualitative und Quantitative Methoden der Textanalyse*. Das Seminar ist ein interdisziplinäres Projekt zwischen dem Institut für Informatik und dem Institut für Deutsche Sprache der HU Berlin.

1.1 Anwendungsgebiete

Die Nutzung digitaler Korpora ist integraler Bestandteil wissenschaftlichen Arbeitens über linguistischen Fragestellungen.

Lexikographie

Als elementares Werkzeug dienen Korpora zur Bestimmung von Kollokationen und Variationen, sowie der Erstellung von Konkordanzen. Auch die Erstellung von Thesauri wird durch korpuslinguistische Verfahren unterstützt.

Lernerkorpora

Lernerkorpora enthalten Lernertexte und dienen der Erforschung von Spracherwerb. Sie können u.A. Fehlerannotationen und korrigierte Transkriptionen enthalten.

Stochastik

Stochastischen Verfahren wie Clustering und Textmining benötigen Textsammlungen zum Trainieren und Evaluieren der zugrunde liegenden Algorithmen. Korpora dienen hier als Goldstandards und Testdaten.

Diachronie

Sprache befindet sich in stetiger Entwicklung. Hypothesen gelten oft nur in einem definierten Zeitkontext und können daher nur mit entsprechenden Daten belegt werden. Zusätzlich kann die Betrachtung geschehener Entwicklungsprozesse Aussagen über Kommende unterstützen.

Für die Diachronie ist Zeitkontext eines Korpus von grosser Bedeutung und muss bei der Pflege von Metadaten berücksichtigt werden.

1.2 Korpustypen

Dieser Abschnitt gibt einen kurzen Überblick über wichtige Korpustypen und stützt sich auf *Computerlinguistik und Sprachtechnologie - Eine Einführung* von Carstensen et. al. [CEJ+04]

Textkorpora

Reine Textkorpora bestehen aus transkribierter oder gesprochener Sprache. Token sind hier die kleinste Einheit.

Sprachkorpora

Sprachkorpora speichern die das urprüngliche Sprachsignal und alignieren diese an einer Zeitachse. Es können sowohl phonetische, als auch linguistische Annotationen hinzugefügt werden. Darunter können Phonemgrenzen, Grundfrequenz und orthografische Transkriptionen sein.

Multimodale Korpora

Multimodale Korpora können als Erweiterung der Sprachkorpora gesehen werden und können neben Sprach- und Videosignalen auch Annotationen für Mimik, Gestik oder Mauszeigerbewegungen enthalten.

Baumbanken

Als spezialisierte Textkorpora halten Baumbanken syntaktisch analysierte Sätze. Diese werden i.A. für die Speicherung und Präsentation als gerichtete azyklische Graphen (Bäume) dargestellt. Das TIGER-Korpus ist eine Baumbank.

2 Boolesche Suche

Die boolesche Suche basiert auf den Grundlagen der Mengenlehre und der boolschen Algebra. Sie ist durch eine strikte Semantik und ein kurzes Regelwerk auch weniger erfahrenen Nutzern leicht zugänglich.

Der Folgende Abschnitt befasst sich mit den theoretischen Grundlagen der booleschen Suche und diskutiert in der Korpuslinguistik gebräuchliche Anfragen und deren technische Umsetzungen.

2.1 Grundlagen der Mengenlehre

Georg Cantor beschrieb sie als

> *"eine Zusammenfassung von bestimmten wohl unterschiedenen Objekten der Anschauung oder des Denkens, welche die Elemente der Menge genannt werden, zu einem Ganzen".*

Aus dieser intuitiv erfassbaren Beschreibung ergeben sich für die Suche in Korpora zwei wiederkehrende Grundmengen.

1. Die Menge alle Dokumente K
2. Die Menge der gesuchten Dokumente R

Abbildung 1 illustriert den Zusammenhang zwischen diesen beiden Grundmengen. Für die Mengen K und R gilt $R \subseteq K$. Die Menge der gesuchten Dokumente R ist also immer eine Teilmenge -oder in einigen Fällen gleich- der Menge aller im Korpus verfügbaren Dokumente K.

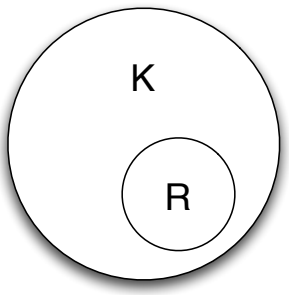

Abbildung 1: Die Grundmengen der Suche in Korpora und ihr Zusammenhang.

Im Gegensatz zu Datenbanksystemen oder der Disziplin des Information Retrieval ist bei der linguistisch motivierten Suche in Korpora der Begriff des Dokuments nicht eindeutig definiert. Hier möchte der Nutzer bei der Formulierung der Suchanfrage entscheiden können, was er unter einem Dokument versteht. Mögliche Definitionen für Dokumente sind:

- Ein Text.
- Ein Satz.
- Ein Token.
- Eine beliebige Folge von Token über Satz oder Textgrenzen hinweg.

Die Definition des Dokumentenbegriffs hat Einfluss auf die Ergebnismenge R und die Darstellung der Ergebniseinheiten für den Nutzer. Viele existierende Systeme legen eine, vom Verwendungszweck bestimmte, feste Definition zugrunde. Eine Festlegung, die

Datenspeicherung und Benutzerschnittstellen vereinfacht aber jeweils nur für bestimmte Abfragezwecke verwendbar ist. Üblicherweise wird man vollständige Sätze oder Texte als Dokumente verstehen, jedoch über Satzgrenzen hinweg suchen können.

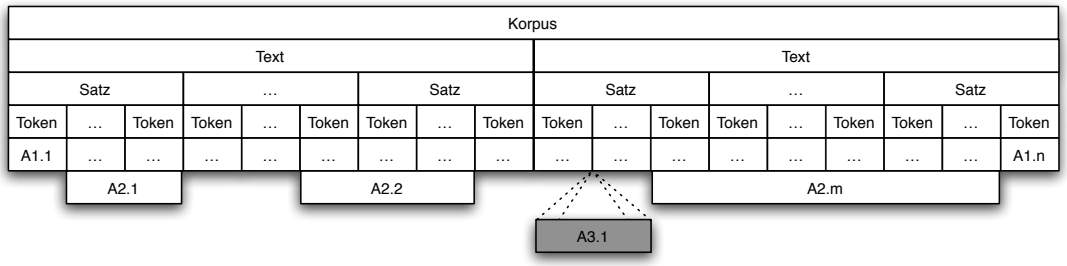

Abbildung 2: Typische Struktur mehrebenenannotierter Korpora.

2.2 Boolesche Algebra

Die boolesche Algebra ist eine algebraische Struktur, die Eigenschaften der logischen Operatoren UND, ODER, NICHT und die Eigenschaften der mengentheoretischen Verknüpfungen Durchschnitt, Vereinigung, Komplement abstrahiert.

Formuliert man eine Suchanfrage als Aussage derart, dass für jedes Dokument in K eindeutig entschieden werden kann ob die Aussage zutrifft oder nicht, so kann die Ergebnismenge R durch einfaches Prüfen der Aussage für jedes Dokument in K berechnet werden. R ist dann die Menge aller Dokumente, die die gemachte Aussage erfüllen.

Beispiele für solch eine Suchanfrage sind z.B.

> "Alle Dokumente, die das Wort *Haus* enthalten."

oder

> "Alle Dokumente, die keine Nomen enthalten.

2.3 Suchanfragen

Betrachtet man die Suche in Korpora im Kontext wissenschaftlicher Problemstellungen, so ist sie hauptsächlich Mittel zur Überprüfung von Hypothesen. Im ersten Schritt zur Erstellung der Suchabfrage muss der Nutzer die Eigenschaften der Gesuchten Dokumente verbalisieren können.

Anschliessend muss die Abfrage in eine für das System spezifische Abfragesprache transformiert- und zur Ausführung gebracht werden. Das Suchsystem wertet die Anfrage aus und präsentiert das Ergebnis in geeigneter -nicht selten in tabellarischer- Form. Der Nutzer

verwendet das Ergebnis zur Überprüfung der Ausgangshypothese oder als Rückmeldung auf die Formulierung der Abfrage. Letzteres ist bei syntaktisch falschen Eingaben oder bei semantisch inkorrekten Ergebnismengen der Fall.

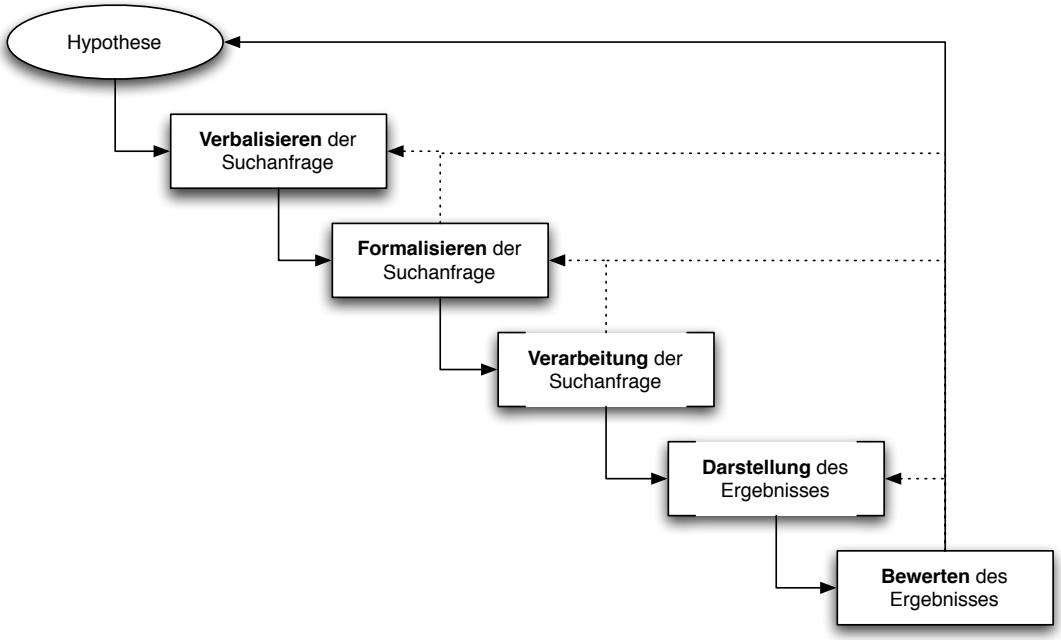

Abbildung 3: Prozessbild der Korpussuche in wissenschaftlichem Kontext.

2.4 Abfragesprachen

Digitale Korpora sind im Allgemeinen Werkzeuge für Linguisten zur Beantwortung fachlicher und wissenschaftlicher Fragestellungen. Die Wahl der Nutzerschnittstelle bestimmt die Interaktion zwischen Suchendem und Suchwerkzeug.

Im folgenden werden die 3 Arten von Abfragesprachen vorgestellt.

2.4.1 Textuelle Abfragesprachen

Textuelle Abfragesprachen sind durch ein festes Regelwerk bestimmt. Es legt die erforderliche Syntax der Abfrageausdrücke und die zugrunde liegende Semantik fest. Sie Abfragesprache CQP[Eve] dient in diesem Abschnitt zur Beispiel für die Eigenschaften textueller Abfragesprachen.

CQP dient der Beschreibung der Eigenschaften von Tokensequenzen. Einzelne Token werden über Attributwerte beschrieben.

So liefert z.B. die Anfrage *[pos="NN" & word="Haus"]* alle Token, deren Wortart als Nomen getagged (z.B. STTS[1]) wurde und die das Teilwort 'Haus' enthalten. Das CQP Suchsystem interpretiert die angegebenen Attributwerte für die Auswertung als reguläre Ausdrücke. Attributeigenschaften können durch boolesche Operatoren verbunden werden.

Für die Beschreibung von Tokensequenzen können Beschreibungen einzelner Token verbunden werden. So findet die Suchanfrage *[pos="ADJA"] [pos="NN" & word="Haus"]* alle Vorkommen der Wortsequenz eines attributivem Adjektiv gefolgt von einem Nomen mit dem Teilwort "Haus".

Sequenzbeschreibungen können sowohl durch Platzhalter "." als auch durch die Modifikatoren +, * und Wiederholungsbeschreibungen {i[,j]} erweitert werden.

Die Eigenschaften textueller Abfragesprachen sind

- Sehr gute *Steuerbarkeit* des Suchergebnisses durch umfangreiche Abfragemöglichkeiten.
- Geringe *Selbstbeschreibungsfähigkeit* und *Lernförderlichkeit* durch abstrakte Syntax. Meist auch durch Entkopplung von Anfrage und Ergebnis durch geringe Interaktivität.

2.4.2 Visuelle Abfragesprachen

Systeme mit visuellen Abfragesprachen ermöglichen dem Nutzer strukturelle Eigenschaften gesuchter Dokumente auf direktem Weg symbolisch zu charakterisieren. Diese Methode bietet sich insbesondere in Korpora mit strukturell -beispielsweise syntaktisch- annotierten Daten an.

Am Beispiel des Tiger-Korpus und seinem Abfragewerkzeugs TIGERSearch zeigen sich die Möglichkeiten der Suche nach Strukturen in linguistischen Daten. Zur Eingabe der Anfrage steht dem Nutzer eine virtuell Arbeitsoberfläche zur Verfügung auf der er Satzteile als Knoten und deren Beziehungen als Kanten mittels Drag-and-Drop definieren kann.

Die Ausgabe des Ergebnisses folgt dieser Methodologie und stellt die Ergebnissätze inklusive eines sie überspannenden Syntaxbaumes zur Kennzeichnung abstrakten Satzteile und deren Beziehungen dar.

Zu den Eigenschaften Visueller Abfragesprachen zählen

- Sehr hohe *Lernförderlichkeit* und *Selbstbeschreibungsfähigkeit* durch die Visualisierung abstrakter Konzepte. Das Erlernen einer komplexen Abfragesprache entfällt.
- Die Formulierung von Abfragen ist durch das Arbeiten mit der Maus i.A. zeitaufwändiger.

[1]STTS: Stuttgart-Tübingen Tagset

Dem erhöhten Zeitbedarf bei der visuellen Eingabe wird TIGERSearch mit der Bereitstellung einer textbasierten Abfragesprache gerecht. Das System wird durch die Überführbarkeit der Abfragerepräsentationen zu einem Hybrid aus textuellen und visuellen Eingabekonzepten.

Die Verwendung einer graphischen Benutzeroberfläche trägt maßgeblich zur Benutzerfreundlichkeit bei, eignet sich aber hauptsächlich für Gelegenheitsnutzer und Anfragen durchschnittlicher Komplexität.[Lez02]

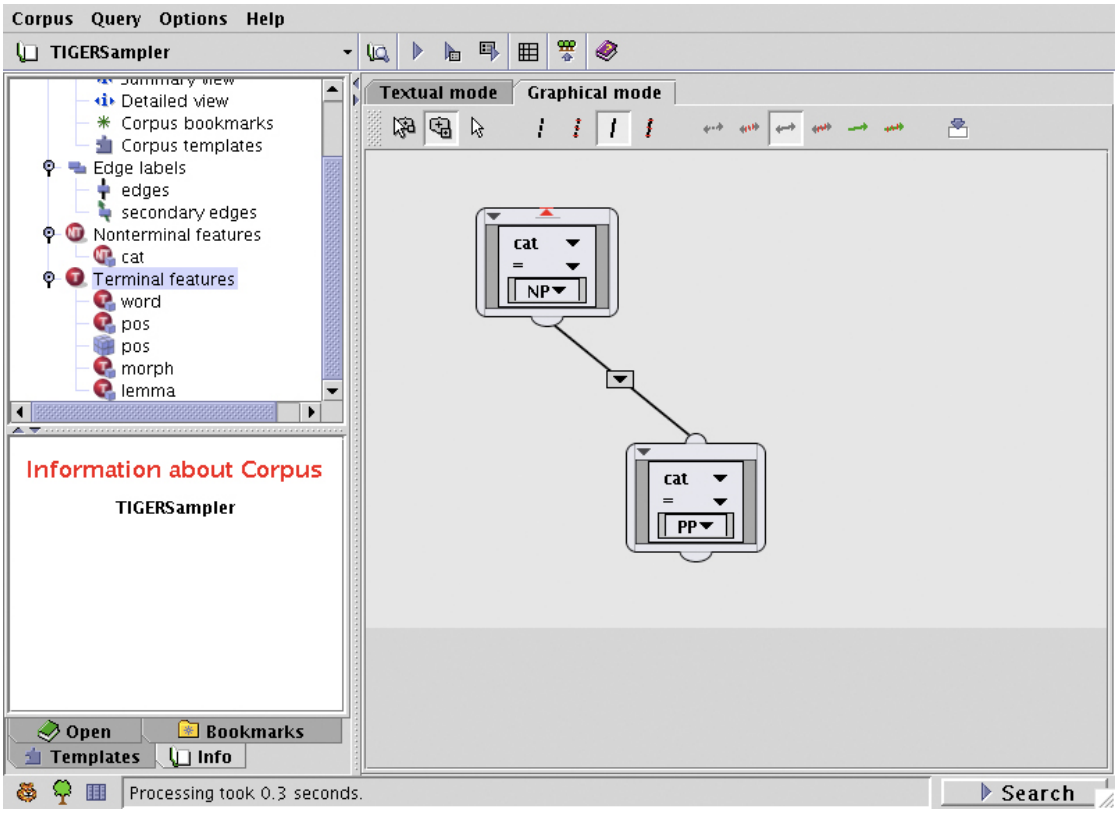

Abbildung 4: Visuelle Abfragebeschreibung mit TIGERSearch.

2.5 Suchtechniken

Die Wahl der Suchtechnik zur Implementierung der Suche ist abhängig von den Eigenschaften des Korpus. Diese Eigenschaften sind

- Physikalische Grösse des Datenbestandes
- Datenhaltung (XML, Relationale Datenbank, Indexstruktur)
- Art der Suchanfragen (Komplexität, Häufigkeit)

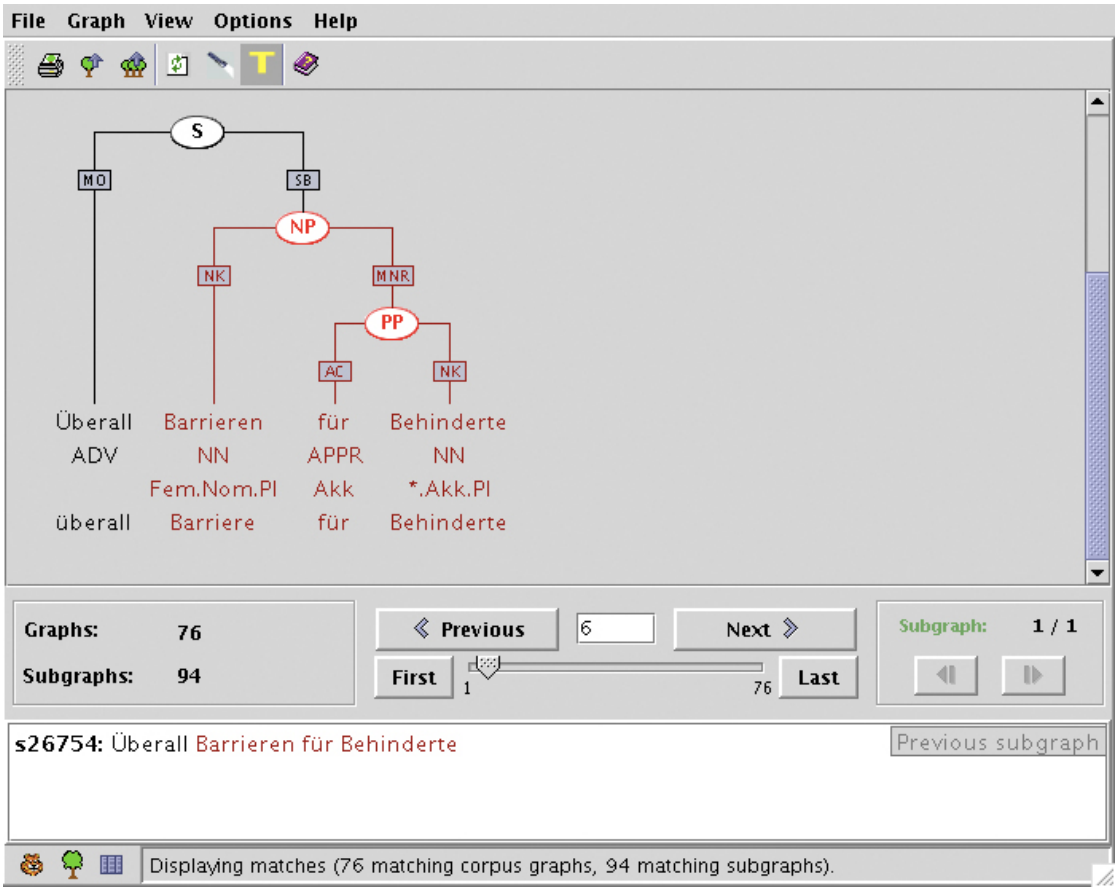

Abbildung 5: Darstellung der TIGERSearch Ergebnissätze als Baumstruktur.

- Leistungsfähigkeit der Hardware (Arbeitsspeicher, Rechenleistung)

2.5.1 Naive Suche

Die naive Suche ist die einfachste Form der Suche in Textdaten. Sie beruht darauf, für jede Anfrage alle Dokumente auf die gesuchte Eigenschaft zu überprüfen und sie ggf. der Ergebnismenge zuzuordnen. Trotz der einfachen Implementierung findet dieses Verfahrens durch seine lineare Komplexität praktisch keine Anwendung.

`Pseudocode:`

```
R = {}
FUER ALLE k IN K
ERFUELLT k DAS SUCHKRITERIUM?
'ja':  R = R vereinigt mit {k}
```

```
'nein':  NICHTS
ENDE FUER ALLE
GIB R AUS
```

2.5.2 Suche mit Invertierten Listen

[Dit]

Für die Erstellung einer invertierten Liste muss jedem Dokument k eindeutig identifizierbar sein. Dies geschieht meist über Identifizierungsnummern. Danach wird für jedes vorkommende Schlüsselwort w ein Datensatz der Form $w \rightarrow \{(id(k), \{position\})\}$ mit den Positionen aller Vorkommen dieses Schlüsselwortes in allen Dokumenten aus K erzeugt und der Invertierten Liste hinzugefügt.

```
Beispiel einer invertierten Liste:

"Kopftuch" := {(1,{9,23,42}),(551,{9}),(506,5)}
"Einfluss" := {(1,{10,18}),(217,{5})}
"Baukran" := {1322,{123}}
```

Die Suche nach einem Schlüsselwort läuft linear über die Länge der invertierten Liste. Die Ausgabe der Vorkommen des Schlüsselwortes ist trivial. Für die Suche nach mehreren Schlüsselworten werden die Mengen der Vorkommen der einzelnen Schlüsselworte durch die Anwendung der Mengenalgebra zur Gesamtergebnismenge verknüpft.

Beispielsuche *Baukran "Kopftuch Einfluss"*

Berechne Teilergebnismengen:

$$T_{Baukran} := \{(1, \{123\})\}$$

$$T_{Kopftuch} := \{(1, \{9, 23, 42\}), (551, \{9\}), (506, 5)\}$$

$$T_{Einfluss} := \{(1, \{10, 18\}), (217, \{5\})\}$$

Berechne Gesamtergebnismenge:

$$T = T_{Baukran} \cap \{t \in (T_{Kopftuch} \cap T_{Einfluss}) \vee pos(t, T_{Einfluss}) = pos(t, T_{Kopftuch}) + 1\}$$

Obwohl die Suchzeiten in der invertierten Liste im Durchschnitt deutlich kürzer sind als die der naiven Suche müssen die Kosten der Erstellung des Indexes bei der Auswahl berücksichtigt werden. Darüber hinaus können die Mengenoperationen über grossen Teilmengen hohen Hauptspeicherbedarf aufweisen.

3 Probabilistisches Retrieval

Probabilistisches Retrieval bearbeitet das Suchen von Informationen aus wahrscheinlichkeitstheoretischer Sicht. Vorstellung: Ein Suchproblem teilt die vorhandenen Dokumente in zwei Teilmengen auf. Entweder ist ein Dokument relevant und wird damit der Teilmenge R zuzuordnen, andernfalls gehört es in die Teilmenge $\bar{R}$. Ziel ist es für jedes Dokument einen Wert zu bestimmen, der uns angibt mit welcher Wahrscheinlichkeit es zu R bzw. $\bar{R}$ gehört.

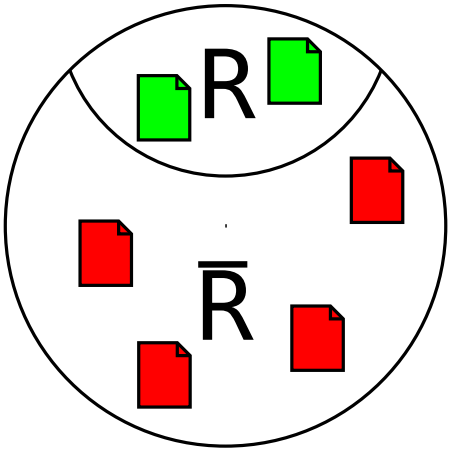

Abbildung 6: Die Grundmengen der Suche in Korpora und ihr Zusammenhang.

Ein optimales Suchverfahren würde nun zu jeder Anfrage stets nur die Dokumente der Menge R zurückgegeben. Allerdings ist dem Nutzer meist nicht genau klar, wie sich Eigenschaften der Dokumente aus R spezifizieren lassen. Er hat zwar ein Konzept des gesuchtem im Kopf, es bereitet aber oft Schwierigkeiten dieses zu verbalisieren und anschließend korrekt zu formalisieren. Dagegen fällt es dem Suchenden meist verhältnismäßig einfach an Hand des Konzeptes zu entscheiden, ob ein gegebenes Dokument dafür relevant ist oder nicht. Alles was man zum Zeitpunkt der Anfrage meist weiß, ist welche Eigenschaften zur Charakterisierung von R beitragen könnten.

3.1 Grundidee

Die Idee des *Probabilistisches Retrieval* ist nun dem Nutzer eine Auswahl von gefundenen Dokumenten zuzeigen, damit er diese als relevant bzw. nicht relevant makieren kann. An Hand der Eigenschaften dieser markierten Dokumente zieht das Verfahren dann Rückschlüsse auf die übrigen Dokumente.

Die Anfrage an ein *Probabilistisches Retrieval* System besteht dabei nur aus einer Menge von zu analysierenden Eigenschaften. Das Zutreffen einer Eigenschaft bedeutet dabei, direkt das damit das Dokument relevant sein sollte. Eigenschaften aus der Anfrage können

genauso gut auch auf die Nicht-Relevanz eines Dokumentes deuten.

3.2 Beispiel

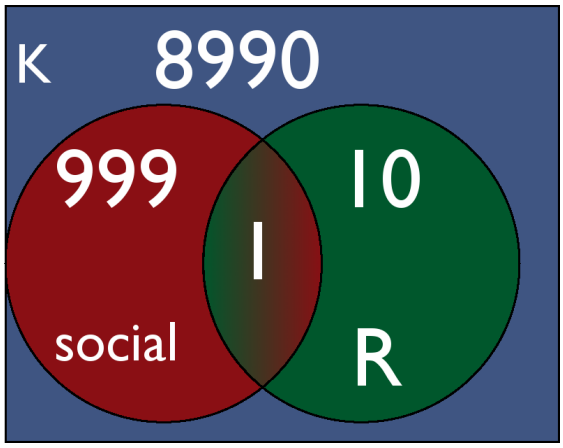

Abbildung 7: Venn-Diagramm eines Beispiel-Korpus zu einer gegebenen Anfrage nach dem Term *social*. (nach [Hie01, S.19])

Angenommen wir haben eine Korpus aus 10.000 Dokumenten aus dem 1.000 den Term *social* enthalten. Der Nutzer hält 11 der 10.000 Dokumente relevant für sein momentanes Suchproblem, aber nur 1 dieser 11 enthält den Term *social*. Bestimmt man sich nun die bedingten Wahrscheinlichkeiten hier für so erhält man, dass $P(R \mid \text{social}) = \frac{1}{1.000} = 0,0010$ kleiner ist als $P(R \mid \overline{\text{social}}) = \frac{10}{9.000} = 0,0011$. Wenn also ein System einzig auf der Eigenschaft „enthält den Term *social*“ ranken soll, so müssten die Dokumente ohne den Term *social* zuerst aufgelistet werden, da sie eine leicht höhere Wahrscheinlichkeit besitzten relevant zu sein.

3.3 Annahmen

Probabilistisches Retrieval versucht zu jedem Dokument die Wahrscheinlichkeiten zu bestimmen, dass es zu R bzw. $\bar{R}$ gehört. Das Verhältnis dieser beiden Wahrscheinlichkeiten dient dann als Ranking auf den Dokumenten und damit als Antwort auf die Anfrage Q.

$$\text{sim}(D_j, Q) = \frac{P(R \mid D_j)}{P(\overline{R} \mid D_j)}$$

Das Verfahren geht dabei davon aus, dass diese Wahrscheinlichkeiten nur von der Anfrage und dem einzelnen Dokument abhängt. Die Relevanz für das einzelne Dokument hängt als ncht davon ab, die Relvanz der anderen Dokumente im Korpus aussieht. Außerdem wird meist angenommen, dass die untersuchten Eigenschaften binär und von einander

unabhängig sind. Im klassischen Information Retrieval wäre eine typische Eigenschaft ob ein bestimmter Indexterm im Dokument vorkommt oder nicht. Diese Annahme der Unabhängigkeit ist eine starke Einschränkung des Verfahrens, da sie wohl so gut wie nie gegeben sein wird. Im klassischen Information Retrieval funktioniert das folgende Vorgehen jedoch trotzallem meist recht gut und es mag für bestimmte Linguistische Verfahren ebenso gelten. Erst die Annahme der Unabhängigkeit macht die einfache Umformung und Lösung der folgenden Gleichungen möglich. Ideen das Verfahren darüber hinauszuerweitern gibt es einige, sollen jedoch in diesem Rahmen erstmal außen vor bleiben.

3.4 Berechnung

Zum Bestimmen der Wahrscheinlichkeit $P(R \mid D)$ wird der Satz von Bayes genutzt, welcher das Vertauschen von Ursache und Wirkung in bedingten Wahrscheinlichkeiten ermöglicht.

$$P(A \mid B) \cdot P(B) = P(A \cap B) = P(B \mid A) \cdot P(A)$$

Angewandt auf unsere Ähnlichkeitsformel, können wir damit Berechnung auf eine Analyse der Eigenschaften auf relevanten und nicht relvanten Dokumenten reduzieren.

$$\mathrm{sim}(D, Q) = \frac{P(R \mid D) \cdot P(D)}{P(\overline{R} \mid D) \cdot P(D)} = \frac{P(D \mid R) \cdot P(R)}{P(D \mid \overline{R}) \cdot P(\overline{R})}$$

Bevor jedoch die Formel benutzt werden kann muß geklärt werden, wie man die Dokumente darstellen möchte. Die binären Eigenschaften lassen sich gut in Binär-Variablen k_i abliegen. Es gilt als $k_i = 1$ genau dann, wenn im Dokument D die Eigenschaft i zutrifft. Alle andere k_i sind 0. Alle k_i zusammenstellen damit einen Vektor dar, der bei richtigen Wahl der Eigenschaften D ausreichend repräsentieren sollte, so dass gilt $P(D \mid R) = P(k \mid R)$ und $P(D \mid \overline{R}) = P(k \mid \overline{R})$. Sind die Eigenschaften des System eien vorher bekannte Menge, so läßt sich dieser Vektor k aus jedem Dokument vorberechen und kann statt dessen abgelegt werden. Für die weitere Verarbeitung wäre der Text nicht mehr nötig. Trifft dies jedoch nicht zu, da eine Eigenschaft zum Beispiel das Zutreffen eines beliebigen Regulären Ausdrucks darstellen soll, so kann der Vektor k erst Ad-Hoc aus der Anfrage erstellt werden.

Die Annahme der Unabhängigkeit erlaubt die Faktoriserung der Wahrscheinlichkeiten zu Einzelwahrscheinlichkeiten des Zutreffens der Eigenschaften k_i.

$$P(k \mid R) = \prod_i P(k_i \mid R) \quad P(k \mid \overline{R}) = \prod_i P(k_i \mid \overline{R})$$

Dieses Produkt lässt sich für jedes Dokument D dann aufteilen in das Produkt der jenigen Eigenschaften mit $k_i = 1$ und das Produkt der Eigenschaften mit $k_i = 0$.

$$P(k \mid R) = \prod_{k_i=1} P(k_i \mid R) \cdot \prod_{k_i=0} P(k_i \mid R)$$

Die an das System gestellte Anfrage Q stellt sich dann als eine Teilmenge dieser k_i dar, welche als relevant angesehen werden. Für die für Q nicht relavanten Eigenschaften kann damit angenommen werden, dass sie auf relevanten Dokumenten genauso oft auftauchen wie auf nicht relevaten.

$$P(k_i \mid R) = P(k_i \mid \overline{R})$$

Damit haben wir alle nötigen Vorarbeiten zusammen um zur endgültigen Ähnlichkeitsformel zu kommen. Zunächst führen wir folgende Abkürzungen ein

$$r_i := P(k_i = 1 \mid R) \quad 1 - r_i := P(k_i = 0 \mid R)$$
$$n_i := P(k_i = 1 \mid \overline{R}) \quad 1 - n_i := P(k_i = 0 \mid \overline{R})$$

Einsetzen das bisherige Maß ergibt damit

$$\begin{aligned}
\mathrm{sim}(D, Q) &= \frac{P(R)}{P(\overline{R})} \cdot \frac{P(k \mid R)}{P(k \mid \overline{R})} \\
&= \frac{P(R)}{P(\overline{R})} \cdot \prod_{k_i \notin Q} \frac{P(k \mid R)}{P(k \mid \overline{R})} \cdot \prod_{k_i \in Q} \frac{P(k \mid R)}{P(k \mid \overline{R})} \\
&= \frac{P(R)}{P(\overline{R})} \cdot 1 \cdot \prod_{k_i \in Q, k_i = 0} \frac{1 - r_i}{1 - n_i} \cdot \prod_{k_i \in Q, k_i = 1} \frac{r_i}{n_i} \\
&= \frac{P(R)}{P(\overline{R})} \cdot \prod_{k_i \in Q, k_i = 0} \frac{1 - r_i}{1 - n_i} \cdot \frac{(1 - n_i)(1 - r_i)}{(1 - r_i)(1 - n_i)} \cdot \prod_{k_i \in Q, k_i = 1} \frac{r_i}{n_i} \cdot \frac{(1 - n_i)(1 - r_i)}{(1 - r_i)(1 - n_i)} \\
&= \frac{P(R)}{P(\overline{R})} \cdot \prod_{k_i \in Q} \frac{1 - r_i}{1 - n_i} \cdot \prod_{k_i \in Q, k_i = 0} 1 \cdot \prod_{k_i \in Q, k_i = 1} \frac{r_i(1 - n_i)}{n_i(1 - r_i)} \\
&= \frac{P(R)}{P(\overline{R})} \cdot \prod_{k_i \in Q} \frac{1 - r_i}{1 - n_i} \cdot \prod_{k_i \in Q, k_i = 1} \frac{r_i(1 - n_i)}{n_i(1 - r_i)}
\end{aligned}$$

Da das Ähnlichkeitsmaß nur zum Ranken der Suchergebnisse dienen soll und der genaue Wert nicht von Interresse ist, brauchen alle Faktoren die nicht vom konkreten Dokument abhängen gleich weggelassen werden. Diese Faktoren würde den Ähnlichkeits werd nur skalieren.

$$\mathrm{sim}(D, Q) \sim \prod_{k_i \in Q} \frac{r_i(1 - n_i)}{n_i(1 - r_i)}$$

Zusammengefasst benötigt man also für jede Eigenschaft k_i aus Q, zum einen eine Möglichkeit alle Dokumente zufinden, für die diese Eigschaft zutrifft. Dies entspricht dem klassischen Suchproblem, wie es in den vorangegangen Abschnitt besprochen wurde.

Außerdem werden für jede dieser Eigenschaften die Werte $r_i = P(k_i = 1 \mid R)$ und $n_i = P(k_i = 1 \mid \overline{R})$ benötigt. Also die Wahrscheinlichkeiten, dass diese auf relevanten bzw. nicht relevanten Dokumenten zutreffen.

3.5 Initiales Ranking

Für ein erstes initiales Ranking bleibt einem nichts anderes übrig, als mittels einer Heuristik eine grobe Schätzung der Werte r_i und n_i zu bestimmen. Da es am Anfang kein Wissen über die Verteilung der Bedingungen in der Relevanten Dokumentenmenge vorliegt, beginnt man mit einer 50/50 Chance für jede Eigenschaft.

$$r_i = P(k_i \mid R) = 0.5$$

Der Großteil der Dokumente aus dem Korpus sind meist nicht relevant. Selten will man mehr als die Hälfte der aller Dokumente im Korpuse als Antwort haben. Als erste Heuristik läßt sich also die Verteilung der Eigenschaften auf den nicht relevanten Dokumenten mit der Verteilung auf allen Dokumenten gleichsetzen. Diese läßt sich anschließend durch einfaches Auszählen bestimmen.

$$n_i = P(k_i \mid \overline{R}) \approx P(k_i) = \frac{\sharp(D \text{ mit } k_i = 1)}{\sharp D}$$

Aus diesem ersten Ranking wird dann ein erste Teilung in R und $\bar{R}$ gewonnen werden. Dies kann entweder durch einfache Zuordnung der besten r Dokumente als relevante Dokumente erfolgen.[BYRN99, S.33] Bessere Ergebnisse erhält man jedoch, wenn man vom Nutzer Feedback einholt. Dazu lässt man ihn auf einer Teilmenge $M \subset K$ der Ergebnisse, die für ihn relevanten L und nicht relevanten M
L markieren.

3.6 Feedback-Schritt

Aus der nun gefunden Aufteilung in relevante und nicht relevante Dokumente auf einer Teilmenge lassen sich nun bessere Rückschlüsse auf die Verteilung der betrachteten Eigenschaften auf beiden Teilmengen schließen. Dazu schätz man die Eigenschaftswahrscheinlichkeiten auf den relevanten Dokumenten mit Hilfe der Verteilung auf L ab

$$r_i = P(k_i \mid R) \approx P(k_i) = \frac{\sharp(L \text{ mit } k_i = 1)}{\sharp L}$$

Die Verteilung der Eigenschaften auf den restlichen Dokumenten dient dann als Schätzer für n_i

$$n_i = P(k_i \mid R) \approx P(k_i) = \frac{\sharp(M \text{ mit } k_i = 1) - \sharp(L \text{ mit } k_i = 1)}{\sharp M - \sharp L}$$

Die Verwendung der Formel in dieser Form bereitet jedoch Probleme, falls nur wenige Dokumente als relevant markiert werden und somit bestimmten Eigenschaften schnelle sehr geringe Wahrscheinlichkeiten zugeordnet bekommen, obwohl sie außerhalb der betrachten Menge M möglicherweise doch vorkommen. Um dies zu verhindern werden die

Formeln oft noch um einen kleinen konstanten Faktor entschärft.

$$r_i = P(k_i \mid R) \approx P(k_i) = \frac{\sharp(L \text{ mit } k_i = 1) + 0.5}{\sharp L + 1}$$

$$n_i = P(k_i \mid R) \approx P(k_i) = \frac{\sharp(M \text{ mit } k_i = 1) - \sharp(L \text{ mit } k_i = 1) + 0.5}{\sharp M - \sharp L + 1}$$

Mit diesen Werten läßt sich dann ein neues, besseres Ranking bestimmen. Durch rekursives Anwenden der Vorschrift versucht man dem Idealen Ergebniss imme näher zukommen mit dem der Nutzer zufrieden ist.

3.7 Bewertung

Der Nutzer kann also mit eine unklaren Vorstellung wie er die Anfrage stellen soll an das System herantreten und muss nur die zubetrachtenden Eigenschaften auswählen. Es kann dabei dann sehr wohl herauskommen, dass das Vorkommen einer Eigenschaft ein Zeichen dafür ist, dass das Dokument nicht relevant ist. Die Anfrage alleine ist nicht mehr allein entscheidend für die Qualität des Ergebnisses. Die Wahrscheinlichkeiten ermöglichen eine Sortierung der Ergebnisse nach der Relevanz für den Nutzer, welches oft ein wichtiger Vorteil gegenüber den einfach boolschen Verfahren ist.

In der hier vorgestellten Variante ist das Verfahren jedoch an recht strikte Annahmen gebunden. Die Verwendung von binären Bedingungen verhindet die verwendung von komplexeren Eigenschaften, wie zum Beispiel der Termhäufigkeit. Die Unabhängigkeit der Bedinungen ist so gut wie nie gegeben. Gerade dies ist für Problemstellungen in der Linguistik oft ein großes Problem.

Literatur

[BYRN99] Baeza-Yates, R. ; Ribeiro-Neto, B.: *Modern Information Retrieval*. New York, ACM Press et al., 1999

[CEJ+04] Carstensen, K.-U. ; Ebert, Ch ; Jekat, S. ; Klabude, R. ; Langer, H.: *Computerlinguistik und Sprachtechnologie - Eine Einführung*. Bd. 2. ELSEVIER, 2004

[Dit] Dittrich, Dr. J.-P.: *LV: Architektur und Implementierung von Datenbanksystemen*. ETH Zürich WS05/06,

[Eve] Evert, Stefan: *The CQP Query Language Tutorial*

[Hie01] Hiemstra, D.: *Using language models for information retrieval*. Enschede: Neslia Paniculata, 2001

[Lez02] Lezius, Wolfgang: *Ein Suchwerkzeug für syntaktisch annotierte Textkorpora*. AIMS, 2002