Einführung in Information Retrieval
Skriptum zur Vorlesung im SS 10

Norbert Fuhr

9. April 2010

Inhaltsverzeichnis

Kapitel 1

Einführung

Information Retrieval (IR) ist ein Teilgebiet der Informatik, das bis Mitte der 1990er Jahre kaum bekannt war und nur in speziellen Anwendungsbereichen (etwa zur Patent- und Literaturdokumentation) eingesetzt wurde, wo primär speziell geschulte Rechercheure die Systeme bedienten. Erst durch den Siegeszug des WWW kamen immer mehr Menschen mit IR-Systemen in Berührung, insbesondere durch die Web-Suchmaschinen, aber auch durch Site-spezifische IR-Funktionen (z.B. bei Büchern oder andere Produkten). Mittlerweile gibt es ein breites Spektrum von IR-Anwendungen, wo den Nutzern selten bewusst ist, dass hier IR-Methoden eingesetzt werden.

1.1 IR-Methoden und -Anwendungen

Zu Information Retrieval zählen hauptsächlich folgende Methoden, soweit sie sich mit Dokumenten (im weitesten Sinne) beschäftigen:

Adhoc-Suche: Dies ist die wichtigste Anwendung, wo ein Benutzer auf eine spontan formulierte Anfrage nach relevanten Dokumenten oder Objekten (s.u.) sucht.

Klassifikation: Hierbei geht es um die automatische Einordnung von Dokumenten in ein vorgegebenes Klassifikationsschema. Im einfachsten Fall können dies nur zwei Klassen sein, wie etwa bei der Klassifikation von Email in Ham/Spam. Komplexere Klassifikationen findet man z.B. im Yahoo! directory[1] oder im Open directory project[2] – (auch wenn die Klassifikation hier zum großen Teil noch manuell erfolgt).

Clustering ist die automatische Gruppierung von Dokumenten nach Ähnlichkeit, wie z.B. beim Clustern von Web-Suchergebnissen in Vivisimode[3].

Informationsextraktion zielt darauf ab, Objekte und Beziehungen aus Texten zu extrahieren bzw. zu markieren, wie z.B. Firmennamen in Wirtschaftsnachrichten bei Yahoo![4].

(Text-)Zusammenfassung ist die automatische Erstellung von Dokumentzusammenfassungen, wie man Sie z.B. in den Ergebnislisten von Web-Suchmaschinen findet, aber auch bei Nachrichten-Überblicksseiten wie z.B. Google News[5].

Frage-Antwort-Systeme versuchen, bei Faktenanfragen anstelle des Nachweises relevanter Dokumente gleich die passenden Fakten zu präsentieren; bekanntestes Beispiel hierfür ist Wolfram Alpha[6].

Recommender-Systeme beobachten Benutzer bei der Auswahl oder dem Bewerten von Objekten und versuchen dann Ihnen andere Objekte zu empfehlen, die sie wahrscheinlich auch mögen werden. Bekannt wurde diese Methode vor allem durch Amazonund zuletzt durch Web 2.0-Anwendungen wie etwa last.fm[7], der Klassiker ist aber Movielens[8].

[1] `http://dir.yahoo.com`
[2] `http://www.dmoz.de`
[3] `http://de.vivisimo.com`
[4] `http://de.finance.yahoo.com/nachrichten`
[5] `http://news.google.de`
[6] `http://www.wolframalpha.com`
[7] `http://www.lastfm.de`
[8] `http://www.movielens.org`

Dabei beschäftigt sich diese Vorlesung primär mit der Adhoc-Suche, während die übrigen Themen in anderen Veranstaltungen wie z.B. „Information Mining“, „Information Engineering“ und „Informationsextraktion aus Texten“ behandelt werden.

Die vorgenannten Methoden können nicht nur für einfache Text- oder Web-Dokumente eingesetzt werden, sondern finden sich in einer Vielzahl von Anwendungen. Diese lassen sich u.a. durch folgende Facetten charakterisieren:

Sprache: Neben der *monolingualen* Suche kann man auch *cross-linguale* Anwendungen betrachten wie etwa bei Google Übersetzer[9], oder es sind sogar multilinguale Suchen möglich (noch im Forschungsstadium).

Struktur: Werden Dokumente meist als *atomare* Einheiten betrachtet, so geht man bei der Literatursuche üblicherweise von einer *Feldstruktur* aus, um zwischen Titel, Autoren und Kurzfassung zu unterscheiden. Daneben berücksichtigt die Web-Suche teilweise die *Graph-Struktur* der Verlinkung, und bei der Suche in XML-Dokumenten geht man üblicherweise von einer *baumartigen Struktur* aus.

Medien: IR-Methoden lassen sich nicht nur auf *Text* anwenden, sondern auch auf *Fakten*, auf *Bilder* (Gazopa[10]), *Audiodaten* wie Sprache oder Musik (Shazam[11]), auf *Videos* oder *3D-Daten* usw.

Objekte: Es gibt spezialisierte Suchmaschinen, die nach bestimmten Objekttypen suchen, wie z.B. die zahlreichen Buch-Suchmaschinen, nach Personen (123people[12], Yasni[13]) oder Firmen (Firmenfinden.[14]).

Statische/dynamische Inhalte: Während die meisten Suchmaschinen von statischen Inhalten ausgehen (die gleichwohl in regelmäßigen Intervallen aktualisiert werden können), gibt es auch Anwendungsbereiche, wo man von einem stetigen Strom neuer Dokumente ausgeht, wie z.B. bei Nachrichten (news.google.de[15]) oder Twitter-Meldungen (search.twitter.com[16]).

Jeder, der eine dieser Anwendungen wiederholt genutzt hat, wird die wesentlichen Unterschiede zwischen IR-Anwendungen und denen klassischer Datenbanksysteme leicht erkennen:

- Die Formulierung einer zum aktuellen Informationsbedürfnis passenden Anfrage bereitet erhebliche Probleme.
- Meistens durchläuft der Prozess der Anfrageformulierung mehrere Iterationen, bis passende Antworten gefunden werden.
- Anfragen liefern potentiell sehr viele Antworten, aber nur wenige davon sind für den Nutzer interessant.
- Das vorgenannte Problem entschärft sich durch die vom System bereitgestellte Rangordnung der Antworten, wodurch potentiell relevante Antworten gehäuft am Anfang der Rangliste auftauchen (z.B. betrachten bei Internet-Suchmaschinen mehr als 90% aller Nutzer nur die ersten 10 Antworten)
- Bei Textdokumenten, aber noch stärker bei Bildern zeigt sich, dass die systemintern verwendete Repräsentation des Inhalts von Dokumenten teilweise inadäquat, auf jeden Fall aber mit Unsicherheit behaftet, ist.

1.2 Was ist Information Retrieval?

Zur Definition des Gebietes legen wir hier die Beschreibung der Aufgaben und Ziele der Fachgruppe „Information Retrieval“ innerhalb der „Gesellschaft für Informatik“ zugrunde:

„Im Information Retrieval (IR) werden Informationssysteme in bezug auf ihre Rolle im Prozess des Wissenstransfers vom menschlichen Wissensproduzenten zum Informations-Nachfragenden betrachtet. Die Fachgruppe „Information Retrieval“ in der Gesellschaft für Informatik beschäftigt sich dabei schwerpunktmäßig mit jenen Fragestellungen, die im Zusammenhang mit vagen Anfragen und unsicherem Wissen entstehen. Vage Anfragen sind dadurch gekennzeichnet, dass die Antwort a priori nicht eindeutig definiert ist. Hierzu zählen neben Fragen mit unscharfen Kriterien insbesondere auch solche, die nur im

[9] `http:///translate.google.de`
[10] `http://www.gazopa.com`
[11] `http://www.shazam.com`
[12] `http://www.123people.de`
[13] `http://www.yasni.de`
[14] `http://www.firmenfinden.de`
[15] `http://news.google.de`
[16] `http://search.twitter.com`

Dialog iterativ durch Reformulierung (in Abhängigkeit von den bisherigen Systemantworten) beantwortet werden können; häufig müssen zudem mehrere Datenbasen zur Beantwortung einer einzelnen Anfrage durchsucht werden. Die Darstellungsform des in einem IR-System gespeicherten Wissens ist im Prinzip nicht beschränkt (z.B. Texte, multimediale Dokumente, Fakten, Regeln, semantische Netze). Die Unsicherheit (oder die Unvollständigkeit) dieses Wissens resultiert meist aus der begrenzten Repräsentation von dessen Semantik (z.B. bei Texten oder multimedialen Dokumenten); darüber hinaus werden auch solche Anwendungen betrachtet, bei denen die gespeicherten Daten selbst unsicher oder unvollständig sind (wie z.B. bei vielen technisch-wissenschaftlichen Datensammlungen). Aus dieser Problematik ergibt sich die Notwendigkeit zur Bewertung der Qualität der Antworten eines Informationssystems, wobei in einem weiteren Sinne die Effektivität des Systems in bezug auf die Unterstützung des Benutzers bei der Lösung seines Anwendungsproblems beurteilt werden sollte.“

Als kennzeichnend für das Gebiet werden somit vage Anfragen und unsicheres Wissen angesehen. Die Art der Darstellung des Wissens ist dabei von untergeordneter Bedeutung.

Oftmals wird IR auch eingeschränkt auf die inhaltsorientierte Suche in (multimedialen) Dokumenten betrachtet. (Tatsächlich behandeln wir in diesem Skriptum fast ausschließlich Modelle und Methoden aus diesem Bereich.) Für diese Art der Suche kann man folgende Abstraktionsstufen unterscheiden:

Syntax: Hierbei wird ein Dokument als Folge von Symbolen aufgefasst. Methoden, die auf dieser Ebene operieren, sind z.B. die Zeichenkettensuche in Texten sowie die Bildretrievalverfahren, die nach Merkmalen wie Farbe, Textur und Kontur suchen.

Semantik beschäftigt sich mit der Bedeutung eines Dokumentes. Methoden zur Repräsentation der Semantik eines Textes haben eine lange Tradition im Bereich der Wissensrepräsentation; semantisches Bildretrieval müsste die Suche nach Bildern unterstützen, die z.B. bestimmte (Klassen von) Objekten enthalten (Menschen, Häuser, Autos,...).

Pragmatik orientiert sich an der Nutzung eines Dokumentes für einen bestimmten Zweck. Zum Beispiel sucht ein Student Literatur zur einem vorgegebenen Seminarthema. Bildarchive werden häufig von Journalisten in Anspruch genommen, um einen Artikel zu illustrieren; dabei ist meist das Thema vorgegeben, aber nicht der semantische Bildinhalt.

Generell lässt sich festhalten, dass Nutzer meistens an einer Suche auf der pragmatischen Ebene interessiert sind. Insbesondere bei nicht-textuellen Dokumenten können dies heutige IR-Systeme aber kaum leisten.

Eigenschaft	Datenbanken	IR
Matching	exakt	partiell, best match
Inferenz	Deduktion	Induktion
Modell	deterministisch	probabilistisch
Klassifikation	monothetisch	polithetisch
Anfragesprache	formal	natürlich
Fragespezifikation	vollständig	unvollständig
gesuchte Objekte	die Fragespezif. erfüllende	relevante
Reaktion auf Datenfehler	sensitiv	insensitiv

Tabelle 1.1: Dimensionen des Information Retrieval(nach Rijsbergen)

Abschließend zu diesen Betrachtungen diskutieren wir hier die in [Rijsbergen 79] skizzierten Dimensionen des IR (Tabelle 1.1). Dabei steht die mittlere Spalte für mehr Datenbank-orientierte Anwendungen, während die rechte Spalte eher klassische IR-Anwendungen charakterisiert. Allerdings kann man die jeweiligen Einträge einer Zeile auch als die zwei Endpunkte einer kontinuierlichen Skala auffassen, auf der es viele mögliche Zwischenlösungen gibt. Solche Lösungen sind insbesondere bei Anwendungen mit semistrukturierten Daten (z.B. XML) gefragt.

Matching: Typisch für Datenbanken (DB) ist der exact match, während wir im IR i.d.R. auch Dinge finden wollen, die nur teilweise die Anfrage matchen, bzw. diejenigen Objekte, die am besten matchen.

Inferenz: DBn berechnen die Antwort auf eine Anfrage durch Deduktion, während im IR eher induktive (auf der Basis zahlreicher Beobachtungen) vorgegangen wird.

Modell: Ein DB-System operiert deterministisch, während IR-Systeme probabilistisch arbeiten und daher anstelle einer ungeordneten Antwortmenge eine nach fallender Wahrscheinlichkeit geordnete Rangliste liefern.

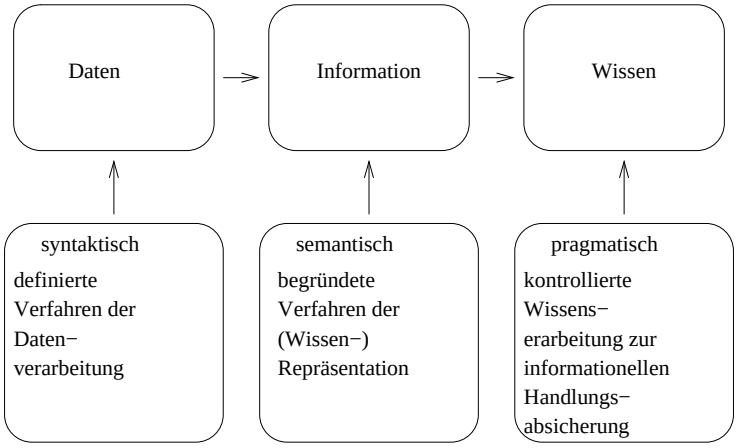

Abbildung 1.1: Daten – Information – Wissen

Klassifikation: Bei einer DB basiert das Schema auf von allen Nutzern akzeptierten Kriterien daher ist die Klassifikation monothethisch. Im IR dagegen müssen bei der Erstellung eines Klassifikationsschemas i.d.R. mehrere Aspekte berücksichtigt werden, so dass die Klassifikation polythetisch ist.

Anfragesprache: DB-Anfragesprachen sind formal, während man im IR natürlichsprachige Anfragen eingeben kann (auch wenn das System nur ein sehr begrenztes Verständnis hiervon hat).

Fragespezifikation: In einem DB-System geht man davon aus, dass Anfragen jeweils vollständig spezifiziert sind, während im IR die Anfrage fast immer unvollständig ist und vom Benutzer sukzessive modifiziert wird.

Gesuchte Objekte: In einer DB werden genau die Objekte gesucht, die die Fragespezifikation erfüllen. Im IR dagegen will der Benutzer relevante Dokumente – auch wenn die nur einen schwachen Bezug zur Fragespezifikation haben.

Reaktion auf Datenfehler: DB-Systeme reagieren hier sehr sensitiv – ist ein Datum falsch eingegeben worden, kann man den betreffenden Datensatz womöglich nie mehr finden. IR-Systeme sind hier toleranter.

1.3 Daten – Information – Wissen

Datenbanksysteme enthalten Daten. IR-Systeme sollen die Suche nach Information[17] unterstützen. Enthalten IR-Systeme also Information? Schließlich ist vor allem in KI (Künstliche Intelligenz)-Publikationen häufig die Rede von Wissensbasen. Was ist denn nun der Unterschied zwischen Daten, Wissen und Information? In der deutschen Informationswissenschaft hat man sich vor einigen Jahren auf eine einheitliche Terminologie geeinigt, die aber leider im Gegensatz zur sonst in der Informatik verwendeten steht. Daher verwenden wir hier die allgemein übliche Begrifflichkeit, allerdings in Kombination mit den Erläuterungen aus der Informationswissenschaft (siehe Abbildung 1.1). Danach sind Daten auf der syntaktischen Ebene anzusiedeln. In diesem Sinne wäre also eine Datenbasis eine nackte Sammlung von Werten ohne jegliche Semantik. Kommt Semantik hinzu, so sprechen wir von Information. Dementsprechend enthalten also Datenbanksysteme nicht nur Daten, sondern auch Information, weil zusätzlich zu den Daten zumindest ein Teil der Semantik des jeweiligen Anwendungsgebietes auch im System modelliert wird. Genauso enthält jedes IR-System Information (im Gegensatz etwa zu dem Fall, wo man Texte einfach in einer Datei abspeichert und mit Hilfe eines Texteditors durchsucht).

Wissen schließlich ist auf der pragmatischen Ebene definiert. In Abwandlung von [Kuhlen 90] lässt sich dies so formulieren: „Wissen ist die Teilmenge von Information, die von jemandem in einer konkreten Situation zur Lösung von Problemen benötigt wird“. Da dieses Wissen häufig nicht vorhanden ist, wird

[17] Da Information keine exakt quantifizierbare Größe ist, gibt es auch den Plural „Informationen“ nicht. Es gibt nur mehr oder weniger Information.

danach in externen Quellen gesucht. Hierbei dient ein Informationssystem dazu, aus der gespeicherten Information das benötigte Wissen zu extrahieren. Wir sprechen auch von Informationsflut, wenn uns große Mengen an Information zugeleitet werden, aus denen wir nur mit Mühe das benötigte Wissen extrahieren können. Daher sind wir auch bereit, für gezielt bereitgestelltes Wissen zu zahlen (z.B. Tageszeitung, werbefreies Fernsehen). Somit kann man die Transformation von Information in Wissen als einen Mehrwert erzeugenden Prozess sehen [Kuhlen 91]. Schlagwortartig lässt sich die Beziehung zwischen Information und Wissen ausdrücken durch die Formulierung „Wissen ist Information in Aktion“.

Als anschauliches Beispiel kann man hierzu die online verfügbaren LINUX-Manuals betrachten. Diese beinhalten Information über LINUX. Wenn nun ein Benutzer eines LINUX-Systems eine bestimmte Aktion ausführen möchte (z.B. ein Dokument drucken), aber nicht weiß, durch welche Kommandos er dies erreicht, so ist das in diesem Fall benötigte Wissen gerade die entsprechende Teilmenge der insgesamt in den Manuals verfügbaren, umfangreichen Information. Da nur ein geringer Teil der gesamten Information benötigt wird, besteht der Mehrwert des Wissens (so sie durch die hierzu verfügbaren Werkzeuge wie z.B. das man-Kommando geliefert wird) gerade in ihrer gezielten Bereitstellung.

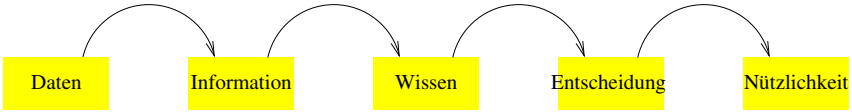

Abbildung 1.2: Wissen zur Entscheidungsunterstützung

Wie oben erwähnt, dient Wissen zur „informationellen Handlungsabsicherung“. Im Kern geht es dabei um die Rolle des Wissens zur Entscheidungsunterstützung. Dieser Zusammenhang wird durch Abbildung 1.2 verdeutlicht. Wissen dient also zur „informationellen Handlungsabsicherung“, und meist stellt sich erst später heraus, wie nützlich die auf dem Wissen basierende Entscheidung war.

1.4 Rahmenarchitektur für IR-Systeme

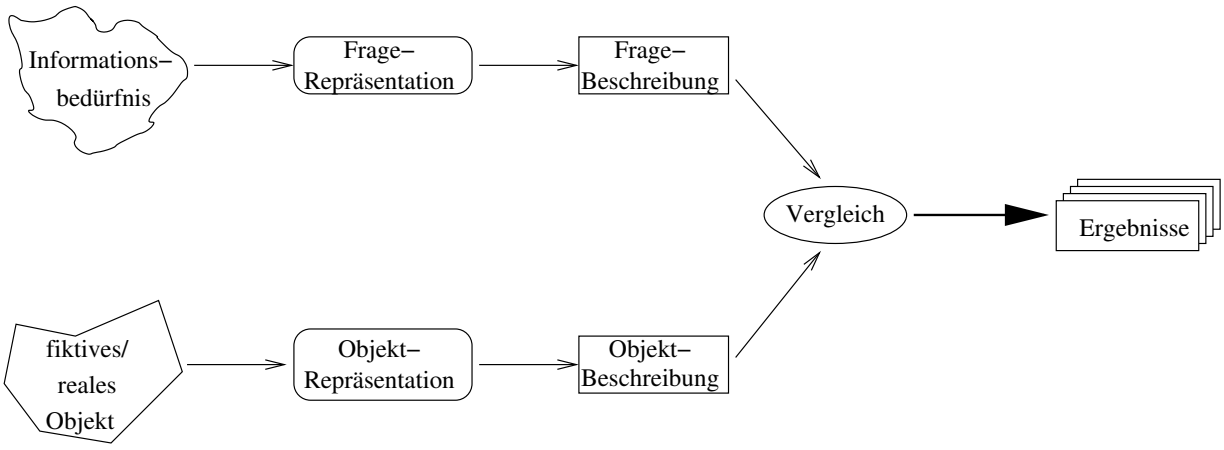

Abbildung 1.3: Konzeptionelles Modell für IR-Systeme

Wir beschreiben hier ein konzeptionelles Modell für IR-Systeme, das wir für die nachfolgenden Ausführungen in diesem Skriptum zugrundelegen wollen (siehe Abb. 1.3). Jedes Objekt einer Datenbasis wird mittels geeigneter Erschließungsverfahren eine entsprechende interne Darstellung (die wir hier Repräsentation nennen wollen) transformiert, in der der Inhalt geeignet repräsentiert wird. Diese wird zu einer Objekt-Beschreibung verdichtet, die für das eigentliche Retrieval benutzt wird.

Am Beispiel des Dokumenten-Retrieval wollen wir diesen Sachverhalt verdeutlichen:

Objekt/Dokumenttext:

Experiments with Indexing Methods. The analysis of 25 indexing algorithms has not produced consistent retrieval performance. The best indexing technique for retrieving documents is not known.

Daraus erstellt ein IR-System z.B. durch Reduktion auf die linguistische Stammform und Eliminierung von Stoppwörtern folgende **Repräsentation:**
(experiment, index, method, analys, index, algorithm, produc, consistent, retriev, perform, best, index, techni, retriev, document, know)

Für das Retrieval wird eine Term-Multimenge als **Beschreibung** verwendet:
{*(experiment,1), (index,3), (method, 1), (analys,1), (algorithm,1), (produc,1), (consistent,1), (retriev,1), (perform,1), (best,1), (techni,1), (retriev,1), (document,1), (know,1)*}

Analog würde eine Suche nach dem besten Indexierungsalgorithmus repräsentiert als
(best index algorithm)
und als Beschreibung könnte die Konjunktion der Frageterme zugrundegelegt werden:
$best \wedge index \wedge algorithm$.

Der Vergleich von Dokumentbeschreibungen mit der Beschreibung der aktuellen Anfrage liefert dann die Retrievalergebnisse.

Anhand dieser Abbildung kann auch der Aspekt der Unsicherheit verdeutlicht werden. Die Ableitung der Repräsentation aus dem eigentlichen Dokument ist eine wesentliche Quelle von Unsicherheit. Speziell bei Texten oder multimedialen Dokumenten kann deren Inhalt nur unzureichend erschlossen werden

Auf der Seite der Fragen ergeben sich die gleichen Probleme der Unsicherheit, insbesondere bei der Abbildung des Informationswunsches auf die formale Anfrage. Zusätzlich spielt hier das für IR-Anwendungen typische Moment der Vagheit eine wichtige Rolle. Daher sollte die Frageformulierung in der Lage sein, diese Vagheit zu repräsentieren. Bei vielen Retrievalmodellen geschieht dies z.B. durch eine Gewichtung der Frageterme.

Die Themen der nun folgenden Kapitel lassen sich ebenfalls anhand von Abbildung 1.3 illustrieren:

- Evaluierung beschäftigt sich mit der Qualität der *Ergebnisse* in Bezug auf das *Informationsbedürfnis*.
- Die Repräsentation von Textinhalten betrachtet die Erstellung der *Repräsentationen* von Fragen und Dokumenten.
- Retrievalmodelle fokussieren auf den Vergleich von *Frage-* und *Dokumentbeschreibung*, wobei bestimmte Formen der Beschreibung zugrundegelegt werden, deren Herleitung aus der *Repräsentation* ebenfalls im Retrievalmodell spezifiziert wird.

Kapitel 2

Evaluierung

Wie in kaum einem anderen Teilgebiet der Informatik spielt die Evaluierung von Verfahren im Information Retrieval eine wichtige Rolle. Aufgrund der Komplexität der Aufgabenstellung sind nicht-experimentelle Methoden zur Beurteilung von Retrievalverfahren wenig geeignet. Zudem ist die Forschungsliteratur im IR reich an Beispielen von plausibel und mächtig erscheinenden Verfahren, die entweder gar nicht praktisch umsetzbar waren oder aber bezüglich der erreichten Retrievalqualität bei weitem nicht an einfachere, aber wirkungsvollere Verfahren heranreichten.

Evaluierungen sollen die Qualität eines Systems beurteilen helfen. Dabei muss man berücksichtigen, dass es unterschiedliche Blickwinkel auf ein IR-System (IRS) gibt, z.B. die von Benutzern, Käufern, Managern, Herstellern oder Entwicklern. Für jede dieser Gruppen sind bestimmte Aspekte eines Systems wichtiger als andere, stehen andere Fragen bei der Evaluierung im Vordergrund. Einige dieser Fragen könnten etwa sein:

- Was kann ich ändern, um die Qualität eines Systems zu verbessern?
- Welche Art der Textrepräsentation ist am besten?
- Welches Retrievalmodell liefert die besten Ergebnisse?
- Welche Qualität weist ein System auf?
- Welches System ist besser?
- Welches System soll ich kaufen?
- Wie kann ich Qualität messen?
- Was bedeutet Qualität für mich?

Um diese Fragen zu beantworten, können jeweils geeignete Evaluierungen konzipiert und durchgeführt werden. Generell sollte jede Evaluierungen – insbesondere, wenn sie wissenschaftlichen Maßstäben genügen will – folgende zwei Eigenschaften erfüllen:

Reliabilität (Zuverlässigkeit) Dieselbe Untersuchung im gleichen Kontext sollte stets dieselben Ergebnisse liefern; das Experiment sollte also *wiederholbar* sein. Dazu ist es notwendig, die Evaluierung ausreichend zu dokumentieren und repräsentative Stichproben von Dokumenten und Nutzern zu verwenden. Ferner müssen Störfaktoren so weit wie möglich ausgeschaltet werden. Im wissenschaftlichen Bereich sollten zudem möglichst Open-Source-Daten verwendet werden bzw. die eigenen Daten Anderen zur Verfügung gestellt werden, damit diese zum Einen die Ergebnisse verifizieren, zum Anderen mit den verwendeten Daten und Methoden weiterarbeiten können

Validität Die Beobachtungen sollten mit den „tatsächlichen“ Verhältnissen übereinstimmen, um die *Gültigkeit* der Ergebnisse zu gewährleisten. Hierbei stellt sich insbesondere die Frage, wie weit man die Ergebnisse verallgemeinern kann, und für welche Gesamtheit denn die Stichproben repräsentativ sind (prädiktive Validität).

Dabei ist zu beachten, dass IR-Experimente stets stochastische Experimente sind, dass also bei Wiederholungen eines Experimentes sich in der Regel nicht genau die gleichen Messwerte wie beim vorherigen Versuch ergeben. Daher muss eine ausreichende Zahl von Versuchen durchgeführt werden (z.B. eine größere Menge von Anfragen betrachtet werden), um sowohl Zuverlässigkeit als auch Validität zu erreichen.

Abhängig von der Entwicklungsphase des zu untersuchenden Systems kann man folgende Arten von Evaluierungen unterscheiden:

- *Formative und iterative Evaluierungen* werden begleitend zur Systementwicklung durchgeführt, um Entwurfsentscheidungen zu treffen oder ggfs. zu revidieren.
- Demgegenüber steht die *summative Evaluierung* am Projektende, die das realisierte System mit den Projektzielen vergleicht.
- Die *komparative Evaluierung* vergleicht mehrere Systeme (bzw. -Komponenten), meist auf der Basis standardisierter Qualitätsmaße.

2.1 Evaluierungskriterien

Wenn man Informationssysteme evaluiert, muss man generell zwischen systemorientierter und benutzerorientierter Evaluierung unterscheiden:

Systemorientierte Evaluierungen fokussieren oft auf die (System-)Effizienz, also die Nutzung der Systemressourcen für eine bestimmte Aufgabe (Zeit/Speicherplatz).

Benutzerorientierte Evaluierungen stellen dagegen den Benutzer in das Zentrum der Betrachtungen. Hier besteht ein enger Zusammenhang mit Usability-Evaluierungen. Im IR-Kontext sind dabei folgende Kriterien von besonderem Interesse:

(Benutzer-)Effizienz bezeichnet die Schnelligkeit, mit der ein Benutzer erfolgreich eine Aufgabe lösen kann.

Effektivität betrachtet die Genauigkeit und Vollständigkeit, mit der ein Ziel erreicht wird.

Zufriedenheit heißt einfach, der Benutzer ist mit dem System zufrieden.

IR-Evaluierungen konzentrieren sich dabei in erster Linie auf die Messung der Effektivität. Grund hierfür ist, dass IR-Systeme wegen der immanenten Vagheit und Unsicherheit weder korrekte (alle gefundenen Dokumente relevant) noch vollständige (alle relevanten Dokumente) Antworten liefern können, so dass man misst, wie nahe denn ein System dem Ideal kommt.

Schaut man sich publizierte Effektivitätsevaluierungen an, so stellt man fest, dass es zwei Arten hiervon gibt: *Systemorientierte* Evaluierungen der Effektivität verwenden vorliegende Relevanzurteile der Benutzer, die eigentlichen Experimente werden ohne weitere Benutzerbeteiligung durchgeführt. *Benutzerorientierte* Experimente beobachten dagegen Benutzer vor einem laufendem IR-System (in der Regel in einer Laborsituation) und berücksichtigen daher auch die Interaktion zwischen Benutzer und System.

Im Folgenden betrachten wir zunächst den ersten Fall und gehen dann in Abschnitt 2.7 auf die zweite Variante ein.

2.2 Relevanz

Um die Qualität der Antworten eines IR-Systems zu beurteilen, legt man meist das Konzept der Relevanz zugrunde: Relevanz bezeichnet dabei eine Eigenschaft der Beziehung zwischen der Anfrage und einem einzelnen Element der Antwortmenge. Hierbei werden folgende Annahmen gemacht:

- Die Systemantwort ist eine Menge von Objekten (z. B. Dokumente). Damit werden stärker strukturierte Antworten nicht berücksichtigt. Wie unten gezeigt wird, lassen sich die hier diskutierten Evaluierungsmethoden aber leicht auf lineare Anordnungen (Rangordnungen) ausdehnen.
- Die Qualität des Objekts, also seine Relevanz bezüglich der Anfrage, hängt nur von der Anfrage ab. Wechselseitige Abhängigkeiten zwischen Objekten bleiben dagegen unberücksichtigt (wenn z. B. die Bedeutung eines bestimmten Dokumentes erst nach der Lektüre eines anderen Dokumentes erkannt wird).

Ebenso unberücksichtigt bleibt die Tatsache, dass die Beziehung zwischen Informationsbedürfnis und Anfrage relativ komplex sein kann und sich nur schlecht auf eine lineare Skala abbilden läßt.

In der Literatur werden meist vier Arten von Relevanz unterschieden:

Situative Relevanz beschreibt die (tatsächliche) Nützlichkeit des Dokumentes in Bezug auf die Aufgabe, aus der heraus das Informationsbedürfnis entstanden ist. Diese Auffassung von Relevanz orientiert sich also an unserer Definition des Informationsbegriffs. Allerdings kann man die situative Relevanz praktisch kaum erfassen, es handelt sich also eher um ein theoretisches Konstrukt.

Pertinenz ist die subjektiv vom Benutzer empfundene Nützlichkeit des Dokumentes in Bezug auf das Informationsbedürfnis. Wenn also der Anfragende selbst Relevanzurteile abgibt, so handelt es sich genau genommen um Pertinenzurteile.

Objektive Relevanz ist die von einem oder mehreren neutralen Beobachtern beurteilte Beziehung zwischen dem geäußerten Informationswunsch und dem Dokument. Der Relevanzbegriff wird häufig bei Systemevaluierungen zugrunde gelegt.

Systemrelevanz bezeichnet die von einem automatischen System geschätzte Relevanz des Dokumentes in Bezug auf die formale Anfrage. In diesem Skript verwenden wir hierfür die Bezeichnung Retrievalwert (englisch: *Retrieval Status Value (RSV)*), der durch die so genannte Retrievalfunktion berechnet wird.

Ein Beispiel soll die Unterschiede verdeutlichen: Ein Benutzer überlegt, ob er das brandneue Handy XYZ kaufen soll. Seine Web-Suche findet unter `ciao.de` einen sehr positiven Erfahrungsbericht zu diesem Gerät. Die situative Relevanz dieses Dokumentes bezieht sich auf die Erfahrungen des Benutzers im Vergleich zu denen des Dokumentes, wenn er das Handy kauft. Pertinenz bezeichnet dagegen die subjektive, momentane Einschätzung der Nützlichkeit dieses Dokumentes. Die objektive Relevanz würde dagegen die Einschätzung eines neutralen Beobachters beschreiben (der z.B. weiß, dass solche Rezensionen häufig vom Hersteller selbst in Auftrag gegeben werden). Die Systemrelevanz bezeichnet stets die Bewertung des Dokumentes durch das Retrievalsystem.

Im Folgenden wird zwischen Pertinenz und objektiver Relevanz nicht mehr unterschieden. Zudem machen wir die Einschränkung, dass die Relevanzskala zweistufig ist, also aus den beiden Werten „relevant“ und „nicht relevant“ besteht.

2.3 Distributionen

Distributionen sind abstrakte Darstellungen von Retrievalantworten, die als Grundlage für Bewertungsmaße dienen. Wir illustrieren dieses Konzept anhand eines Beispiels: Als Antwort auf eine Anfrage berechne ein System folgende Retrievalwerte für die Dokumente in der Datenbasis:

$$\{(d_1, 0.3), (d_2, 0.8), (d_3, 0.1), (d_4, 0.8), (d_5, 0.8), (d_6, 0.6), (d_7, 0.3), (d_8, 0.1)\}$$

Daraus ergibt sich folgende Rangordnung bzw. *Distribution von Dokumenten*:

$$(\{d_2, d_4, d_5\}, \{d_6\}, \{d_1, d_7\}, \{d_3, d_8\})$$

Die Relevanzbeurteilung des Benutzers sei nun folgende (R – relevant, $\bar{R}$ – nicht relevant):

$$\{(d_1, R), (d_2, R), (d_3, \bar{R}), (d_4, R), (d_5, R), (d_6, \bar{R}), (d_7, R), (d_8, R)\}$$

Durch die Zusammenführung von Rangordnung und Relevanzurteilen erhält man die *Distribution mit Relevanzurteilen*:

$$(\{d_2^+, d_4^+, d_5^+\}, \{d_6^-\}, \{d_1^+, d_7^+\}, \{d_3^-, d_8^+\})$$

Für die Bewertung der Retrievalqualität abstrahiert man nun von spezifischen Dokumenten. Dadurch ergeben sich Äquivalenzklassen von Distributionen mit Relevanzurteilen, die wir im folgenden einfach als *Distributionen* bezeichnen:

$$\Delta = (+ + +| - | + +| + -)$$

Die einzelnen Ränge werden dabei durch „|“ getrennt, „+“ bezeichnet ein relevantes und „−“ ein nichtrelevantes Dokument.

2.4 Standpunkte und Bewertungsmaße

Jedem Bewertungsmaß liegt ein bestimmter Standpunkt bezüglich des „Besserseins“ einer Distribution im Vergleich zu einer anderen zugrunde. Bevor man ein Maß anwendet, sollte man sich daher im Klaren darüber sein, welcher Standpunkt dem gewählten Maß zugrundeliegt und ob dieser für die aktuelle Anwendung adäquat ist.

2.4.1 Benutzerstandpunkte

Wir nehmen an, dass das IRS als Antwort auf eine Anfrage eine Rangordnung von Dokumenten produziert, die der Benutzer sequentiell solange durchsieht, bis ein bestimmtes Abbruchkriterium erfüllt ist. Für jedes Kriterium (= Standpunkt) kann man dann ein entsprechendes Bewertungsmaß definieren, das die Präferenzen des Benutzers widerspiegelt. Beispiele für mögliche Abbruchkriterien und zugehörige Bewertungsmaße sind:

- n Dokumente gesehen: # gesehene relevante Dokumente
- n relevante Dokumente gesehen: # gesehene Dokumente
- n nicht relevante Dokumente gesehen: # gesehene / # gesehene relevante Dokumente
- n nicht relevante Dokumente in Folge gesehen: # gesehene / # gesehene relevante Dokumente

2.4.2 Benutzer- vs. Systemstandpunkte

Man kann grob zwischen Benutzer- und Systemstandpunkten unterscheiden. Erstere spiegeln dabei die Sicht eines einzelnen Benutzers wider, während letzteren eine globale Sicht (die des Systembetreibers) zugrundeliegt. Dementsprechend beziehen sich *benutzerorientierte Maße* auf das mögliche Verhalten und die Präferenzen der Benutzer. *Systemorientierte Maße* entsprechen dagegen einer systemorientierten Sicht, die unabhängig von speziellen Benutzerstandpunkten ist. Daher wird eine „globale“ Bewertung der Distribution angestrebt. Im Gegensatz dazu werden etwa bei den obigen benutzerorientierten Maßen jeweils nur die ersten Dokumente der Rangordnung betrachtet. Ein einfaches systemorientiertes Maß wäre daher die Korrelation zwischen Systemantwort Δ und idealer Distribution $\bar{\Delta}$. Bezeichne S^+ die Anzahl richtig angeordnete Paare und S^- die Anzahl falsch angeordnete Paare sowie $S_{\max}$ die Anzahl richtig angeordnete Paare der optimalen Lösung, dann könnten wir z.B. die systemorientierte Güte der Antwort $\Delta = (+++|-|++|+-)$ im Vergleich zur idealen Rangordnung $\bar{\Delta} = (++++++|--)$ berechnen als

$$\varrho = \frac{S^+ - S^-}{S_{\max}} = \frac{8-3}{12} = \frac{5}{12}.$$

2.5 Maße für Ergebnismengen

2.5.1 Recall, Precision und Fallout

Wir betrachten zunächst den Fall der Retrievalbewertung für eine Ergebnismenge, da die Maße für Rangordnungen Erweiterungen der mengenbezogenen Maße sind.

Als Benutzerstandpunkt wird hier angenommen, dass der Benutzer sich stets alle gefundenen Dokumente anschaut. Im Folgenden bezeichne GEF die Menge der gefundenen Antwortobjekte, REL die Menge der relevanten Objekte in der Datenbank und ALL die Gesamtzahl der Dokumente in der Datenbank (Abbildung 2.1).

Basierend auf diesen Mengen lassen sich dann die Maße *Precision*, *Recall* und *Fallout* wie folgt definieren:

$$\text{Precision:} \quad p := \frac{|REL \cap GEF|}{|GEF|}$$

$$\text{Recall:} \quad r := \frac{|REL \cap GEF|}{|REL|}$$

$$\text{Fallout:} \quad f := \frac{|GEF - REL|}{|ALL - REL|}$$

Hierbei gibt die Precision den Anteil der relevanten an den gefundenen Dokumenten wieder. Recall dagegen bezeichnet den Anteil der relevanten Dokumente, die tatsächlich gefunden wurden. Schließlich misst Fallout den Anteil der gefundenen irrelevanten an allen irrelevanten Dokumenten der Kollektion; hiermit wird also die Fähigeit des Systems bewertet, irrelevante Dokumente vom Benutzer fernzuhalten.

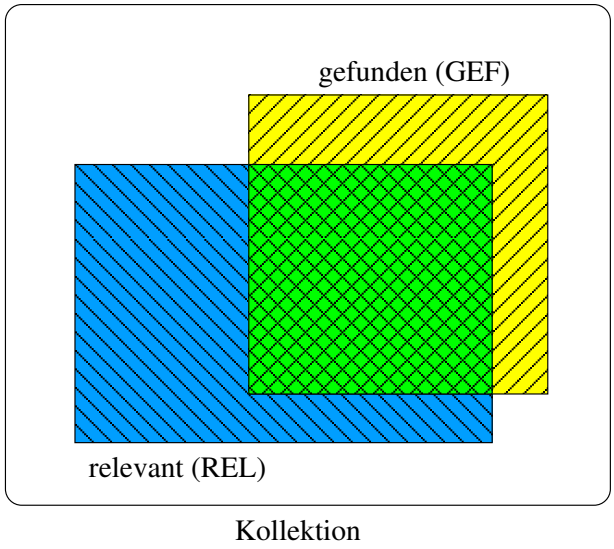

Abbildung 2.1: Mengen der relevanten und gefundenen Dokumente

Als Beispiel nehmen wir an, dass eine Kollektion von 1000 Dokumenten 20 relevante Dokumente zur aktuellen Anfrage enthält. Ein System liefert 10 Dokumente, von denen 8 relevant sind. Dann erhält man folgende Werte:

$$\begin{aligned} p &= \frac{|REL \cap GEF|}{|GEF|} = \frac{8}{10} = 0.8 \\ r &= \frac{|REL \cap GEF|}{|REL|} = \frac{8}{20} = 0.4 \\ f &= \frac{|GEF - REL|}{|ALL - REL|} = \frac{2}{980} \approx 0.002 \end{aligned}$$

Da es sich bei Retrievalexperimenten um stochastische Experimente handelt, sollte man die Messwerte auch entsprechend interpretieren. Im Falle der Precision $p = |REL \cap GEF|/|GEF|$ wird damit die Wahrscheinlichkeit approximiert, dass ein (zufällig ausgewähltes) gefundenes Dokument relevant ist. Analog schätzt man mit dem Recall $r = |REL \cap GEF|/|REL|$ die Wahrscheinlichkeit, dass ein (zufällig ausgewähltes) relevantes Dokument gefunden wird. Entsprechendes gilt für den Fallout. Diese probabilistische Interpretation der Retrievalmaße spielt eine wesentliche Rolle bei den Optimalitätsbetrachtungen zum probabilistischen Ranking-Prinzip.

Für konkrete Anwendungen – insbesondere solche, bei denen anstelle einer Ergebnismenge eine Rangliste von Antworten zom System geliefert wird – werden häufig Varianten dieser Maße verwendet, die an den jeweiligen Kontext angepasst wurden.

- Beim Web-Retrieval kann man davon ausgehen, dass die meisten Benutzer (nach empirischen Untersuchungen ca. 90%) sich nur die erste Seite der Ergebnisliste anschauen, die in der Regel 10 Antworten enthält[1]. Ein passendes Maß ist daher die Precision nach 10 Dokumenten, die meist als „Prec@10“ bezeichnet wird. Ein extremer Standpunkt wäre die Precision des ersten Dokumentes (Prec@1). In diesem Kontext wird auch häufig die Click-Through-Rate betrachtet: in Ermangelung von Relevanzurteilen wird angenommen, dass jede angeklickte Antwort relevant sind
- Bei Evaluierungsinitiativen wie TREC, CLEF oder INEX werden in analoger Weise z.B. Prec@5, Prec@10, Prec@30 und Prec@100 parallel betrachtet, um Benutzerklassen zu simulieren, die sich jeweils die entsprechende Anzahl Dokumente anschauen. Als globales Maß wird hier zudem die Mean Average Precision betrachtet, die man erhält, wenn man für jede Frage zunächst die Average Precision bestimmt und dann das arithmetische Mittel über alle Fragen bildet. Letztere wiederum ist als Mittelwert der Precision nach jedem relevanten Dokument definiert. Abbildung 2.2 zeigt, dass

[1] Viele Benutzer stoppen sogar schon früher, wenn nicht alle zehn Antworten ohne Scrollen sichtbar sind!

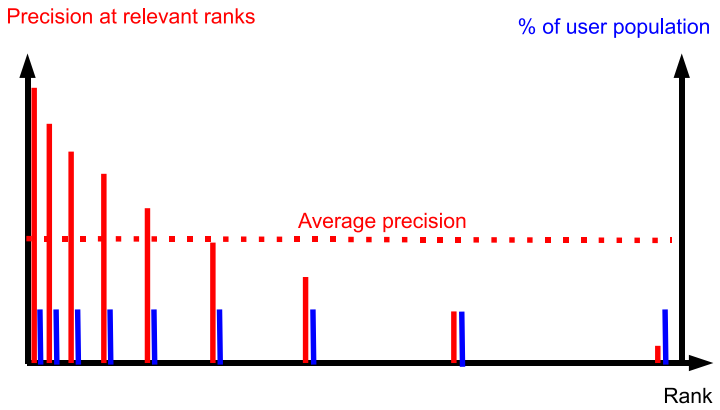

Abbildung 2.2: Berechnung von Average Precision

man dieses Maß als benutzerorientiert interpretieren kann, wenn man annimmt, dass nach jedem relevanten Dokument ein gleich großer Anteil von Benutzern die Suche abbricht. Dies ist natürlich ziemlich unrealistisch – die meisten Benutzer werden schon relativ früh stoppen, so dass man anstelle einer Gleichverteilung eher eine schiefe Verteilung annehmen müsste.

2.5.2 Recall-Abschätzung

Die Größe der Precision ist für jeden Benutzer eines IR-Systems direkt ersichtlich. Die Größe des Recalls ist dagegen für einen Benutzer weder erkennbar, noch kann sie mit vernünftigem Aufwand präzise bestimmt werden. Der Grund hierfür liegt in dem Problem, die Mächtigkeit der Menge REL zu bestimmen. Folgende Näherungsmethoden wurden hierzu vorgeschlagen:

Vollständige Relevanzbeurteilung: einer repräsentativen Stichprobe der gesamten Datenbasis: Da REL sehr viel kleiner als die gesamte Datenbasis ist (z. B. mögen 100 von 10^7 Dokumenten relevant sein), müsste die repräsentative Stichprobe schon einen relativ großen Teil der Datenbasis umfassen, was zu viel Beurteilungsaufwand erfordert.

Source-Dokument-Methode: Hierbei wählt man ein zufälliges Dokument aus der Datenbank und formuliert dann eine Frage, auf die dieses Dokument relevant ist. Anschließend wird geprüft, ob das System das betreffende Dokument als Antwort auf die Frage liefert. Für eine Menge von Fragen schätzt man dann über die relative Häufigkeit die Wahrscheinlichkeit, dass das Source-Dokument gefunden wird, als Näherung des Recalls. Nachteil dieser Methode ist, dass die verwendeten Fragen keine echten Benutzerfragen sind.

Frageerweiterung: Man erweitert die ursprünglichen Anfrage, so dass eine Obermenge der ursprünglichen Antwortmenge gefunden wird, die wesentlich größer ist und weitere relevante Dokumente enthält (z. B. kann man auch mehrere Frageformulierungen von verschiedenen Bearbeitern erstellen lassen und die Vereinigungsmenge der Antwortmengen betrachten). Damit erhält man aber nur eine Teilmenge der Menge REL, somit sind die darauf basierenden Recall-Schätzungen im allgemeinen zu hoch.

Abgleich mit externen Quellen: Man versucht parallel zur Datenbanksuche noch mit davon unabhängigen Methoden, relevante Dokumente zu bestimmen (z. B. indem man den Fragenden oder andere Fachleute fragt, welche relevanten Dokumente sie kennen). Der Anteil der in der Datenbasis vorhandenen Dokumente, die das System als Antwort liefert, ist dann eine gute Näherung für den Recall. Nachteile dieser Methode sind, dass sie zum einen recht aufwendig ist, zum anderen oft nicht anwendbar ist, weil es keine unabhängigen externen Quellen gibt.

Pooling-Methode: (Retrieval mit mehreren Systemen): Man wendet mehrere IR-Systeme auf denselben Dokumentenbestand an und mischt die Ergebnisse verschiedener Systeme zu den gleichen Anfragen. In der Regel gibt es starke Überlappungen in den Antwortmengen der verschiedenen Systeme, so

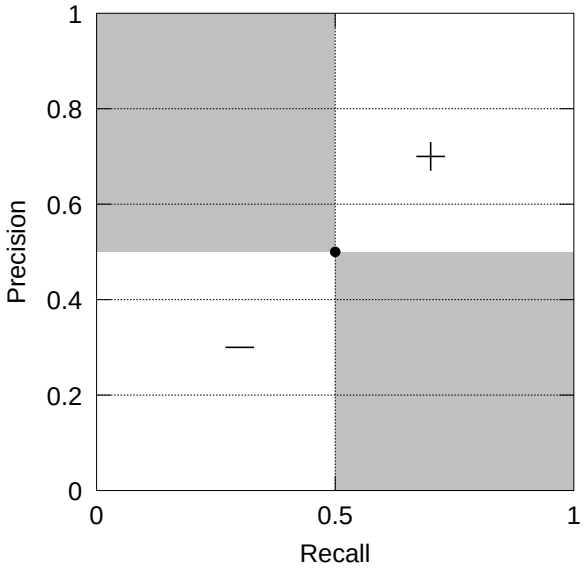

Abbildung 2.3: Darstellung eines Retrievalergebnisses als Punkt im Recall-Precision-Graphen

dass der Aufwand nicht linear mit der Anzahl betrachteter Systeme wächst [Harman 95]. Dieses Verfahren wird derzeit beim Vergleich experimenteller Systeme im Rahmen von Evaluierungsinitiativen angewandt.

Außer den ersten beiden Verfahren liefern alle Methoden nur untere Schranken für REL; die gemessenen Recall-Werte sind daher im Allgemeinen zu optimistisch.

2.5.3 Frageweise Vergleiche

Hat man für eine Frage Recall und Precision bestimmt, so lässt sich dieses Ergebnis als Punkt in einem Recall-Precision-Graphen darstellen. Beim Vergleich zweier Systeme bezüglich einer Frage ist dann dasjenige System besser, das sowohl einen höheren Recall- als auch einen besseren Precision-Wert liefert (einer der beiden Werte darf auch gleich sein). In Abbildung 2.3 sind die Bereiche, in denen bessere bzw. schlechtere Ergebnisse liegen, weiß markiert. Häufig wird allerdings ein System einen höheren Recall, das andere dagegen eine höhere Precision liefern, so dass sich keine Aussage bezüglich einer Überlegenheit eines der beiden Systeme ableiten lässt (die grauen Bereiche in Abbildung 2.3).

Als eine gängige Methode, (r, p)-Paare durch eine einzige Zahl auszudrücken, hat sich das F-Maß durchgesetzt. Abhängig von einem zu wählenden Parameter β berechnet sich dieses Maß zu

$$F_\beta = \frac{(\beta^2 + 1) \cdot p \cdot r}{\beta^2 \cdot p + r}$$

Hierbei gibt β die relative Gewichtung des Recalls an ($\beta = 0$: nur Precision zählt; $\beta = \infty$: nur Recall zählt). Üblicherweise setzt man $\beta = 1$, arbeitet also mit dem F_1-Maß. Abbildung 2.4 zeigt die Aufteilung von Recall-Precision-Punkten in bessere und schlechtere Ergebnisse durch das F-Maß: Bezogen auf den F-Wert 0,5 für verschiedene β-Werte finden sich bessere Recall-Precision-Punkte jeweils im rechten oberen Bereich, schlechtere Punkte auf der jeweils anderen Seite der Kurven.

Als Alternative zu diesen kombinierten Maßen kann man auch Kostenmaße betrachten; diese werden insbesondere bei Systemen zur Informationsfilterung häufig eingesetzt. Dabei geht man von folgender Kontingenztafel aus und zählt die Anzahl Dokumente $h_\cdot^\cdot$ für jeden der vier Fälle:

	relevant	$\overline{\text{relevant}}$
gefunden	h_g^R	h_g^I
$\overline{\text{gefunden}}$	h_n^R	h_n^I

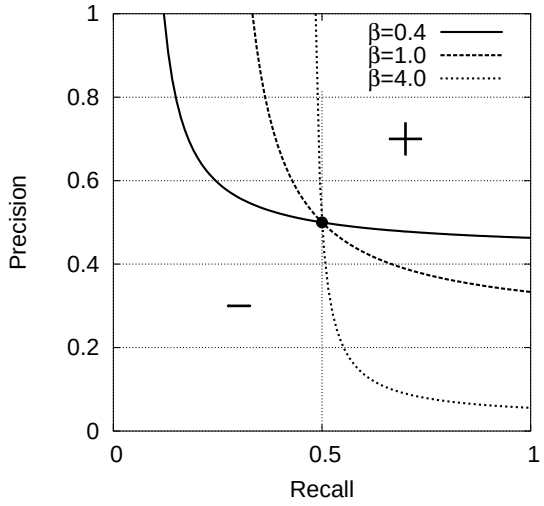

Abbildung 2.4: Aufteilung von Recall-Precision-Punkten durch das F-Maß: Für $F = 0.5$ und verschiedene β-Werte finden sich bessere Recall-Precision-Punkte im rechten oberen Bereich.

Die allgemeine Formel für die Gesamtkosten ergibt sich dann als gewichtete Summe der verschiedenen Anzahlen:

$$C = C_g^R \cdot h_g^R + C_g^I \cdot h_g^I + C_n^R \cdot h_n^R + C_n^I \cdot h_n^I$$

Dabei sind C_g^R, C_g^I, C_n^R und C_n^I die Kostenparameter für die vier Fälle. Im einfachsten Fall könnte man etwa wählen $C_g^R = C_n^I = 0$ und $C_g^I = C_n^R = 1$. $C_g^I = C_n^R = 1$.

Will man dagegen ein System zur Filterung von Spam-Emails bewerten, so sollte zwar das System möglichst viele „relevante“ (d.h. Spam-Mails) identifizieren, aber möglichst keine „irrelevanten“ (nicht-Spam) Mails selektieren. Um also h_g^I (im Vergleich h_n^R, der Anzahl an den Benutzer weitergeleiteten Spam-Mails) zu möglichst klein zu halten, sollten also entsprechende Werte $C_g^I \gg C_n^R$ gewählt werden. Würde man z.B. 20 gesehene Spam-Mails als genauso schlimm wie eine verlorene Ham-Mail ansehen, so könnte man setzen $C_g^I = 20 \cdot C_n^R$ sowie $C_g^R = C_n^I = 0$.

2.5.4 Mittelwertbildung

Wie oben erwähnt, muss man eine Menge von Fragen betrachten, um fundierte Aussagen über die Qualität eines Systems zu erhalten. Dementsprechend müssen Mittelwerte für die Qualitätsmaße berechnet werden. Hierzu werden im IR zwei verschiedene Methoden angewendet (im Folgenden gehen wir von N Fragen aus, wobei REL_i und GEF_i für $i = \{1, \ldots, N\}$ die jeweiligen Mengen gefundener bzw. relevanter Dokumente bezeichnen):

- Bei der *Makrobewertung* wird das arithmetische Mittel der Werte für die einzelnen Fragen gebildet, also z. B. für die Precision:

$$p_M = \frac{1}{N} \sum_{i=1}^{N} \frac{|REL_i \cap GEF_i|}{|GEF_i|}$$

  Probleme ergeben sich bei der Makrobewertung, wenn einzelne Fragen leere Antwortmengen liefern (dies ist z. B. häufig bei Tests der Fall, wo nur eine Stichprobe der Dokumente der gesamten Datenbasis verwendet wird, so dass Fragen mit wenigen Antworten auf der gesamten Datenbasis oft keine Antwort in der Stichprobe liefern). Durch verbesserte probabilistische Schätzmethoden kann dieses Problem unter Umständen behoben werden.
  Aus stochastischer Sicht approximiert die Makro-Methode den Erwartungswert für die Precision zu einer zufällig ausgewählten Anfrage. Somit geht jede Frage gleich stark in den Mittelwert ein, was nicht immer wünschenswert sein mag (wenn man Fragen mit größeren Antwortmengen stärker gewichten will). Daher bezeichnet man diese Methode auch als Frage- oder Benutzer-orientiert.

- Bei der *Mikrobewertung* werden zuerst Zähler und Nenner des Maßes addiert, bevor der Quotient gebildet wird – also bei der Precision:

$$p_\mu = \frac{\sum_{i=1}^{N} |REL_i \cap GEF_i|}{\sum_{i=1}^{N} |GEF_i|}$$

  Dadurch wird das Problem der leeren Antwortmengen umgangen. Da hier jedes Dokument gleich stark in den Mittelwert eingeht, bezeichnet man die Mikrobewertung auch als Dokument- oder System-orientiert. Aus stochastischer Sicht wird hier die Wahrscheinlichkeit approximiert, dass ein (zufällig ausgewähltes) gefundenes Dokument aus einer der N Anfragen relevant ist.

Analoge Betrachtungen gelten für Recall und Fallout.

Ein spezielles Problem der Mikro-Precision ist die fehlende Monotonieeigenschaft: Wir betrachten zwei verschiedene Retrievalergebnisse Δ_1, Δ_2, die von zwei Systemen zur gleichen Frage geliefert worden sind. Ein Maß ist dann monoton, wenn sich durch das Hinzufügen des gleichen Retrievalergebnisses Δ zu beiden Ergebnissen die Aussage über die Überlegenheit eines der beiden Systeme nicht ändert. Seien $\Delta_1 = (+-)$ und $\Delta_2 = (++---)$ Retrievalergebnisse, zu denen später das Retrievalergebnis $\Delta = (++-------)$ hinzugefügt wird.

$$\begin{aligned} \text{Dann ist} \quad p_\mu(\Delta_1) = \frac{1}{2} &> \frac{2}{5} = p_\mu(\Delta_2), \\ \text{aber} \quad p_\mu(\Delta_1, \Delta) = \frac{3}{10} &< \frac{4}{13} = p_\mu(\Delta_2, \Delta). \end{aligned}$$

2.6 Rangordnungen

Fast alle Retrievalverfahren liefern eine Rangordnung von Dokumenten als Antwort (eine Ausnahme bildet nur das boolesche Retrieval, das noch in einigen älteren Systemen im Einsatz ist). Daher müssen die Definitionen der Retrievalmaße entsprechend erweitert werden.

Bei Rangordnungen muss man zusätzlich unterscheiden, ob eine lineare (totale) Ordnung der Dokumente aus der Datenbasis vorliegt oder nur eine schwache Ordnung (d.h. es können mehrere Dokumente im selben Rang sein). Wir beschränken uns hier auf lineare Ordnungen.

n	Dokumentnr.	$\times$ = rel.	Recall	Precision
1	588	$\times$	0.2	1.00
2	589	$\times$	0.4	1.00
3	576		0.4	0.67
4	590	$\times$	0.6	0.75
5	986		0.6	0.60
6	592	$\times$	0.8	0.67
7	984		0.8	0.57
8	988		0.8	0.50
9	578		0.8	0.44
10	985		0.8	0.40
11	103		0.8	0.36
12	591		0.8	0.33
13	772	$\times$	1.0	0.38
14	990		1.0	0.36

Tabelle 2.1: Recall und Precision für Δ_1 nach dem Nachweis von n Dokumenten bei linearer Ordnung

Retrievalergebnisse werden durch das in Abschnitt 2.3 beschriebene Schema dargestellt. Die Distributionen Δ_1 und Δ_2 dienen im Folgenden als Beispiele für lineare Rangordnungen:

$$\begin{aligned} \Delta_1 &= (+|+|-|+|-|+|-|-|-|-|-|+|-) \\ \Delta_2 &= (+|-|+|+|+|-|-|-|-|-|+|-|+|-) \end{aligned}$$

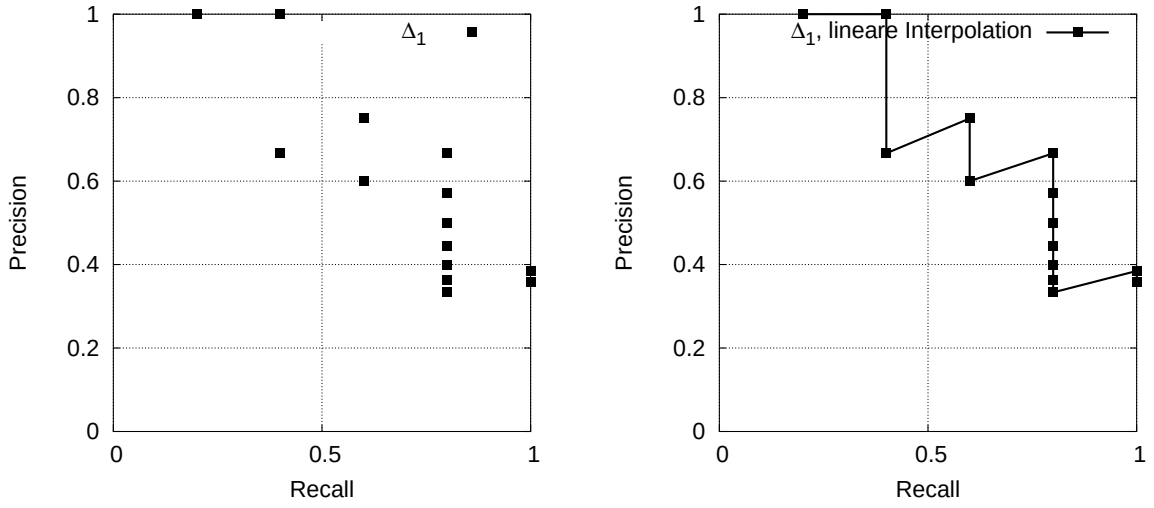

Abbildung 2.5: Graphische Darstellung der Werte aus Tabelle 2.1 (Δ_1), rechts mit linearer Interpolation der Punkte.

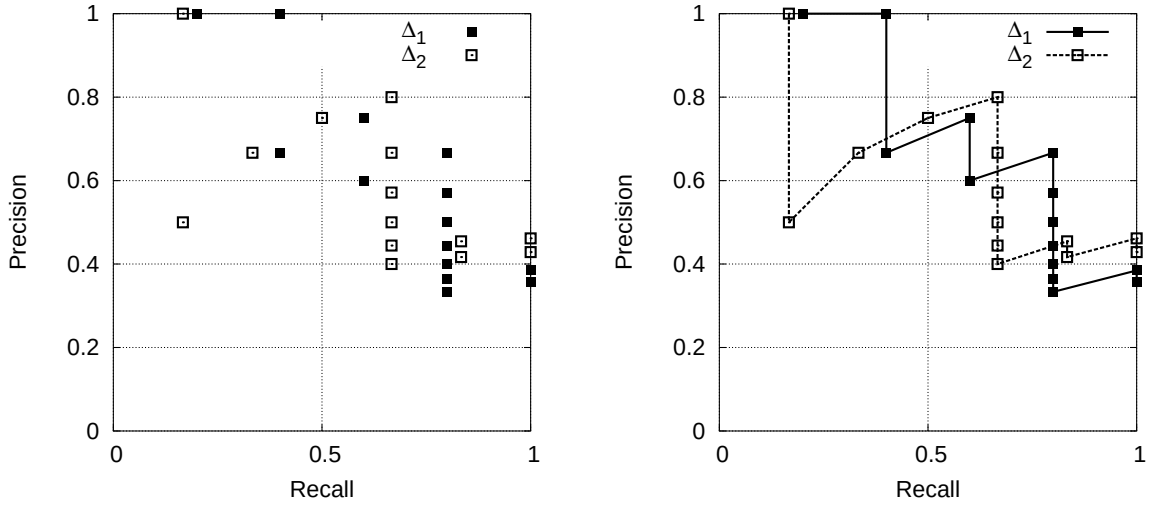

Abbildung 2.6: Graphische Darstellung der Werte für zwei verschiedene Rangordnungen (Δ_1 und Δ_2), rechts mit linearer Interpolation der Punkte.

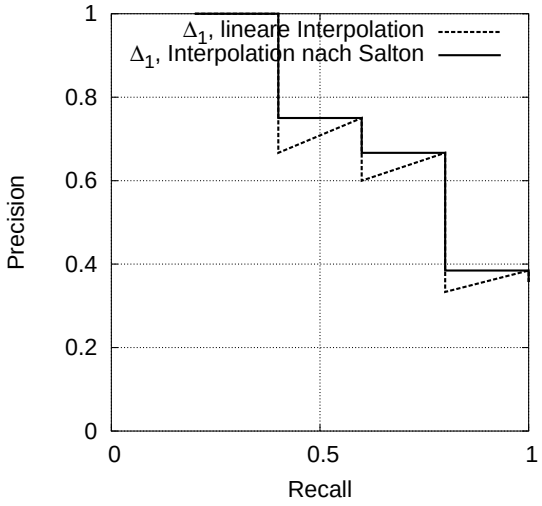

Abbildung 2.7: Interpolation nach Salton

Die zugrundegelegte fiktive Dokumentkollektion enthält also 14 Dokumente, von denen im Fall von Δ_1 5 und im Fall von Δ_2 6 Dokumente als relevant beurteilt wurden.

Bei einer linearen Ordnung können Recall und Precision (r, p) für eine Anfrage in Abhängigkeit von der Mächtigkeit in der Antwortmenge bestimmt werden, wie dies am Beispiel in Tabelle 2.1 gezeigt wird. Δ_1 ist die zugehörige Darstellung des Retrievalergebnisses.

Trägt man die sich für verschiedene n ergebenden (r, p)-Werte in das Recall-Precision-Diagramm ein, so ergibt sich das in Abbildung 2.5 (links) gezeigte Bild. Um die Übersichtlichkeit zu erhöhen, kann man die einzelnen Punkte mit Geradenstücken verbinden (lineare Interpolation, Abbildung 2.5 rechts). Diese Art der Darstellung ist besonders nützlich, wenn man die Qualitätsmaße für mehrere Rangordnungen in einem einzigen Graphen darstellen möchte (siehe Abbildungen 2.6). Allerdings darf man den Zwischenpunkten auf diesen Geradenstücken keine Bedeutung zuordnen, da die lineare Interpolation aus theoretischen Gründen nicht korrekt ist – somit kann man im Beispiel etwa nicht behaupten, die gestrichelte Linie zeige für r=0.5 einen höheren Wert. Ein weiterer Nachteil dieser Methode zeigt sich bei der Mittelung über mehrere Fragen – dann erhält man eine „Zitterkurve“, da sich die Spitzen der „Sägezähne“ nicht ausmitteln.

Um die Kurven im R-P-Graphen interpretieren zu können, wurde von Salton [Salton & McGill 83, S. 167–8] vorgeschlagen, die Originalkurve wie in Abb. 2.7 dargestellt zu interpolieren. Dabei wird jeder einzelne (r, p) Wert durch eine waagerechte Linie bis zu $r = 0$ extrapoliert. Der resultierende Graph ergibt sich dann als das Maximum über diese Geradenstücke. Es wird also angenommen, dass der Benutzer nur nach einem relevanten Dokument stoppt (so dass die Punkte für irrelvante Ränge ignoriert werden können). Zudem stoppt er nur, falls die Precision später nicht noch einmal ansteigt.

Später wurde diese Methode noch etwas weiter entwickelt, indem man aus diesen Kurven die Precision für die 11 Recall-Punkte { 0, 0.1, 0.2, ..., 1 } abliest (oder als Verfeinerung für 101 Punkte 0, 0.01, 0.02...) und diese Werte dann durch Geradenstücke verbindet. Ein Schwaxhpunkt dieses Ansatzes ist die Tatsache, dass für Recall 0 die Precision eigentlich undefiniert ist. Zudem werden schwache Ordnungen einfach in eine zufällige lineare Anordnung überführt, was die Reliabilität der Ergebnisse beeinträchtigt. Abbildung 2.8 zeigt die sich daraus ergebenden Kurven im Vergleich zur Salton-Methode.

2.7 Evaluierung von interaktivem Retrieval

2.7.1 Batch- vs. interaktives Retrieval

Bisher wurde in diesem Kapitel fast ausschließlich die Evaluierung von Batch-artigem Retrieval betrachtet. Dabei wird angenommen, dass der Benutzer eine Anfrage formuliert, und dann wird die Qualität der von den einzelnen Systemen produzierten Ergebnisse bestimmt. Dieser Ansatz hat allerdings eine Reihe von

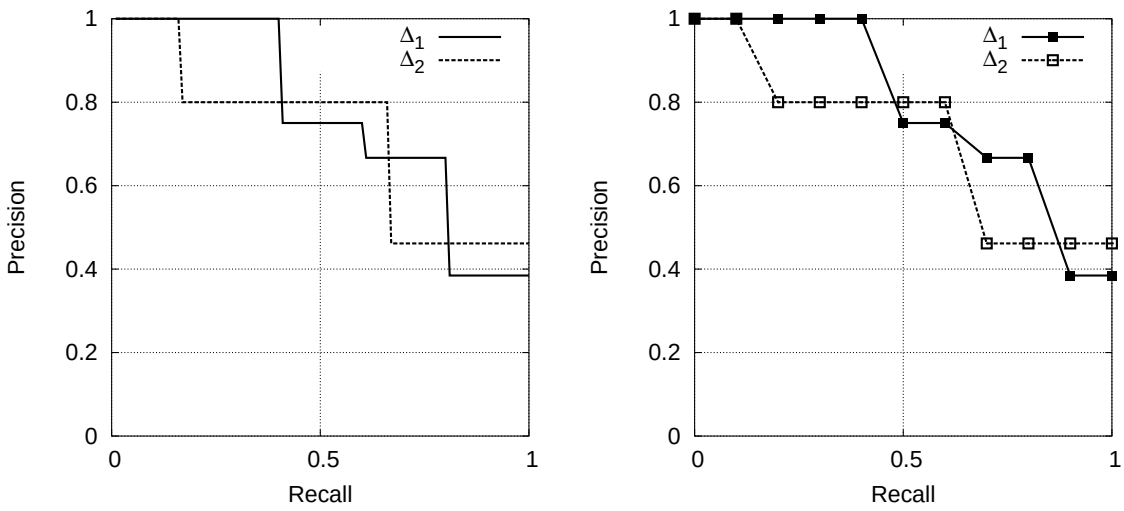

Abbildung 2.8: Salton-Methode (links) im Vergleich zur MIttelung über 11 Punkte (rechts)

Schwächen:

- Es wird nur eine einzige Anfrage betrachtet, eine Reformulierung (wie sie bei interaktiven Systemen üblich ist) wird nicht berücksichtigt.
- Auch bei Relevance Feedback ist die einzig mögliche Interaktion die Relevanzbeurteilung einiger Dokumente, weitergehende Reaktionen des Benutzers (wie etwa Markierung relevanter/irrelevanter Passagen) sind nicht möglich.
- Heutige IR-Systeme bieten oft eine reichhaltige Funktionalität, wie z.B. Highlighting, Clustering, Browsing von Dokumenten oder Termlisten. Diese Funktionalität wird bei der Evaluierung nicht berücksichtigt.
- Ergebnisse aus dem TREC interactive track [Voorhees & Harman 00] zeigen, dass die in herkömmlichen Evaluierungen beobachteten Qualitätsunterschiede zwischen Verfahren beim interaktiven Retrieval verschwinden, da sie durch den Benutzer leicht kompensiert werden können [Turpin & Hersh 01].

Somit ergibt sich der Schluss, dass Ergebnisse aus Batch-Evaluierungen nur sehr beschränkte Aussagekraft auf die viel realistischere Situation des interaktiven Retrieval haben. Daraus ergibt sich die Notwendigkeit für die Evaluierung von interaktivem Retrieval.

Empirische Studien zu interaktivem Retrieval haben immer wieder gezeigt, dass dies ein iterativer Prozess ist, bei dem Nutzer die Anfrage häufig reformulieren. Die eingegebenen Anfragen sind zwar thematisch zusammenhängend, allerdings wandert das Ziel der Suche dabei. Die Idee des Relevance Feedback, die Anfrage für ein feststehendes Informationsbedürfnis zu optimieren, ist also unrealistisch. Als kognitives Modell für diese Situation ist von Marcia Bates das Berrypicking-Modell vorgeschlagen worden: ähnlich einem Beerensucher im Wald sammelt ein Benutzer während der Suche „Beeren“ in Form von relevanten Dokumenten und hilfreichen Suchtermen, und wechselt dabei immer etwas die Richtung (siehe Abbildung 2.9).

2.7.2 Suchaufgaben

Um interaktives Retrieval im Labor zu evaluieren, benötigt man realistische Suchaufgaben. Borlund hat hierzu den Ausdruck “simulated work task” geprägt. anstelle eines vorgegebenen Informationsbedürfnisses holt man also weiter aus und beschreibt den Versuchspersonen eine zu lösende Aufgabe, aus der heraus das Informationsbedürfnis begründet ist.

Einige Beispiele aus dem „Interactive Track“ 2009 der Evaluierungsinitiative INEX sollen dies illustrieren:

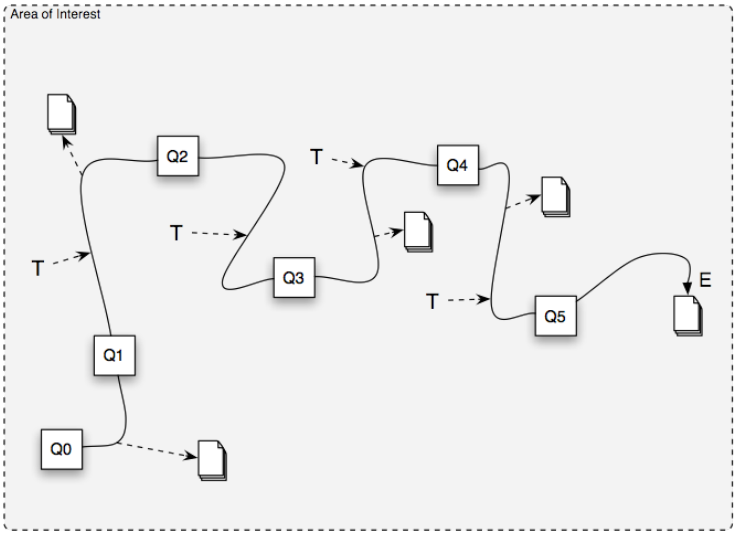

Abbildung 2.9: Berrypicking-Modell nach Bates

Suche nach.../ Vorwissen	ein Objekt	heterogene Objekte	viele heterog. Objekte
Kein Wissen			
Teilwissen			
umfassendes Wissen			

Abbildung 2.10: Information need typology matrix nach Ingwersen

1. Breites Thema: *You are considering to start studying nuclear physics. In order to prepare for the course you would like to get acquainted with some good introductionary texts within the field as well as some of its classics.*
2. Enges Thema: *Find books which present documentation of the specific health and/or beauty effects of consuming olive oil.*
3. Benutzerspezifisches, enges Thema: *For one of the courses you are currently attending, you need an additional textbook. You have only money for one book (assuming they all have about the same price).*

Um solche Aufgaben zu definieren, ist es hilfreich, ein Klassifikationsschema für Suchaufgaben zugrundezulegen, um dann auf die betrachteten Klassen generalisieren zu können. Hierzu gibt es zahlreiche Ansätze, von denen wir zwei kurz skizzieren wollen. Shneiderman unterscheidet zwischen spezifischer Faktensuche, erweiterter Faktensuche, offenem Browsing und der Klärung der Verfügbarkeit von Information. Ingwersen hat die in Abbildung 2.10 dargestellte zweidimensionale Unterteilung definiert, die einerseits nach dem Umfang des Vorwissens, andererseits nach dem Suchziel differenziert.

Mittlerweile gibt es ein reiches Instrumentarium für diese Art der Evaluierung:

- Bei *"think aloud"-Protokollen* soll die Versuchsperson laut denken, um damit mehr Einblick in die bei der Suche ablaufenden kognitiven Prozesse zu bekommen.
- *Beobachtungsdaten* (z.B. Log-Analyse) sind relativ einfach zu erheben, besitzen aber nur eine beschränkte Aussagekraft.
- Durch *Interviews* nach dem Versuch (und evtl. auch schon vorher) lässt sich der subjektive Eindruck der Versuchspersonen erheben und Hinweise auf die subjektiv empfundenen Stärken und Schwächen des Systems sammeln.
- *Fragebögen* können alternativ oder ergänzend zu Interviews eingesetzt werden. Sie erfordern weniger Aufwand für die Versuchsleitung, sind leichter auszuwerten und ermöglichen eine quantitative Beurteilung nach verschiedenen Kriterien.
- *Fehleranalysen* dienen dazu, bei der fehlgeschlagenen Bearbeitung von Aufgaben mit dem System Rückschlüsse auf die Ursachen zu ziehen.

- *Zeitbedarf zur Problembearbeitung* ist eine relative einfach zu erhebende Messgröße: Für eine vorgegebene Menge von Aufgaben misst man jeweils die Zeit, die die Versuchspersonen zu deren Bearbeitung benötigen.
- Die *Kosten-Nutzen-Analyse* versucht, über die reine Retrievalqualität hinaus sowohl den Aufwand des Benutzers als auch den konkreten Nutzen zu quantifizieren.

Mittlerweile wird die Notwendigkeit der Evaluierung von interaktivem Retrieval allgemein anerkannt, allerdings wird der Aufwand zur Durchführung vielfach noch gescheut.

Kapitel 3

Wissensrepräsentation für Texte

3.1 Problemstellung

Da sich IR hauptsächlich mit der inhaltlichen Suche in Texten beschäftigt, stellt sich die Frage nach der geeigneten Repräsentationsform für Textinhalte. Im Gegensatz zu Standard-Datenbanksystemen, wo die Repräsentation mehr oder weniger eindeutig ist, ist die Repräsentation ein zentrales Problem im IR. Dies liegt daran, dass die in einer Frage angesprochenen Konzepte auf unterschiedlichste Weise in Texten formuliert sein können. Eine gewählte Repräsentationsform soll daher zum einen unterschiedliche Formulierungen auf die gleiche Repräsentation abbilden (und damit den Recall erhöhen), zum anderen auch unklare Formulierungen (z.B. Mehrdeutigkeit einzelner Wörter) vereindeutigen, um die Precision zu erhöhen.

Wir werden in diesem Kapitel zwei Arten von Lösungsansätzen für dieses Problem vorstellen:

- semantischer Ansatz:
  Durch die Zuordnung von Deskriptionen zu Texten wird versucht, eine Repräsentation zu erstellen, die weitgehend unabhängig von der konkreten Formulierung im Text ist. Syntax und Semantik solcher Deskriptionen sind in Form sogenannter Dokumentationssprachen festgelegt.
- Freitextsuche:
  Hierbei wird keine zusätzliche Repräsentation erstellt, sondern es werden nur bestimmte Funktionen zur Verbesserung der Suche im Text der Dokumente angeboten.

3.2 Freitextsuche

3.2.1 Grundlagen

Terminologie Wir geben zunächst die Definitionen einiger linguistischer Begriffe wieder, die wir im Folgenden verwenden werden:

- *Token:* einzelnes Wort im laufenden Text
- *Type:* einzelnes Wort des Vokabulars
- *Morphem:* kleinste bedeutungstragende Einheit in einem Wort, z.B. Blend-e, lauf-en,
- *Flexion:* Deklination, Konjugation und Komparation von Wörtern
- *Grundform:* unflektierte Wortform; für Nomen ist es der Nominativ Singular, für Verben der Infinitv, für Adjektive die ungesteigerte Form (Positiv).
- *Derivation:* Wortbildung aus dem Wortstamm mit Hilfe von Präfixen und Suffixen, z.B. haus: Haus – häuslich – aushäusig,
- *Stammform:* (genauer: Derivationsstammform), das der Derivation zugrunde liegende lexikalische Morphem
- *Kompositum:* Bildung eines komplexen Wortes, das aus mindestens zwei Morphemen besteht, die sonst als selbstständige Wörter vorkommen, z.B. Dampfschiff, schreibfaul, strapazierfähig
- *Nominalphrase:* Wortgruppe im Satz, die ein Nomen als Bezugswort hat, z.B. „Wahl des Bundeskanzlers“

Ansätze Bei der Freitextsuche kann man zwischen den beiden folgenden Ansätzen unterscheiden:

- informatischer Ansatz:
  Dieser Ansatz (der in den heute kommerziell angebotenen IR-Systemen fast ausschließlich vertreten ist) fasst Textretrieval als Zeichenkettensuche auf und bietet entsprechende Funktionen auf Zeichenkettenebene.
- computerlinguistischer Ansatz:
  Hier wird mit Hilfe von morphologischen und teilweise auch syntaktischen Verfahren eine Normalisierung von Wortformen angestrebt, so dass sich die Suche auf Wörter bezieht (im Gegensatz zu den Zeichenketten beim informatischen Ansatz).

Vorverarbeitung Bei beiden Ansätzen werden zunächst folgende Verarbeitungsschritte auf den Text der Dokumente angewandt:

1. *Textbereinigung:* Häufig enthält der Text noch Markup oder Trennungszeichen, die vor der weiteren Verarbeitung entfernt werden müssen.
2. *Zerlegung des Textes* in einzelne Wörter (Tokenization): Leer- und Interpunktionszeichen werden hier als Worttrenner aufgefasst. Bei einigen ostasiatischen Sprachen (z.B. chinesisch) gibt es keine expliziten Worttrenner, man kann das Ende eines Wortes nur mit Hilfe eines Lexikons erkennen.
3. Stoppwortbestimmung: Nicht-bedeutungstragende Wörter wie Artikel, Füllwörter oder Konjunktionen werden meist aus Aufwandsgründen von der weiteren Verarbeitung ausgeschlossen. Nur für syntaktische Verfahren müssen die Stoppwörter berücksichtigt werden, um ein korrektes Parsing zu ermöglichen. Stoppwörter machen häufig rund die Hälfte des Textes aus. Allerdings können Stoppwörter auch wichtige Bestandteile von Nominalphrasen sein, z.B. bei “vitamin A” im Englischen.
4. Satzendeerkennung: Für die computerlinguistische Analyse sowie die Freitexbedingung „Suche imselben Satz“ ist es notwendig, die Folge von Wörtern in Sätze zu untergliedern. Wegen der Verwechslungsmöglichkeit des Satzendepunktes mit Abkürzungspunkten kann diese Aufgabe nur approximativ gelöst werden (z.B. mit Hilfe von Abkürzungslisten).

Probleme der Freitextsuche Die eigentliche Freitextsuche bezieht sich dann auf den so reduzierten Text (bzw. die resultierende Folge von Wörtern). Bei dieser Art der Suche nach Wörtern stellen sich folgende Probleme:

- **Homographen** (verschieden gesprochene Wörter mit gleicher Schreibweise)
  Tenor: Sänger / Ausdrucksweise
- **Polyseme** (Wörter mit mehreren Bedeutungen)
  Bank: Sitzgelegenheit / Geldinstitut
- **Flexionsformen**: *Haus – (des) Hauses – Häuser*, *schreiben – schreibt – schrieb – geschrieben*
- **Derivationsformen** (verschiedene Wortformen zu einem Wortstamm)
  Formatierung – Format – formatieren
- **Komposita**: *Donaudampfschiffahrtsgesellschaftskapitän*, *Bundeskanzlerwahl*
- **Nominalphrasen**: *Wahl des Bundeskanzlers*, *information retrieval – retrieval of information – information was retrieved*

Das grundsätzliche Problem der Freitextsuche – die Wortwahl – bleibt aber in jedem Falle ungelöst!

3.2.2 Informatischer Ansatz

Der informatische Ansatz betrachtet Texte als Folgen von Wörtern, wobei ein Wort als eine durch Leer- oder Interpunktionszeichen begrenzte Zeichenfolge definiert ist. Somit wird hier Freitextsuche als eine spezielle Form der Zeichenkettensuche aufgefasst und entsprechende Zeichenketten-Operatoren angeboten. Diese beziehen sich zum einen auf einzelne Wörter, zum anderen auf Folgen von Wörtern. Erstere sind Truncation- und Maskierungs-Operatoren für die Freitextsuche, letztere die Kontextoperatoren. (Wie bei allen IR-Systemen üblich, wird im folgenden nicht zwischen Groß- und Kleinschreibung unterschieden).

- Truncation- und Maskierungs-Operatoren dienen dazu, Flexions- und Derivationsformen von Wörtern zusammenzuführen.

  - Bei der **Truncation** wird einerseits zwischen Front- und End-Truncation unterschieden, wobei die Front-Truncation hauptsächlich benutzt wird, um beliebige Vorsilben bei der Suche zuzulassen. Andererseits kann bei der Truncation entweder eine feste oder eine variable Anzahl von Zeichen zugelassen werden. Bei den folgenden Beispielen verwenden wir das Symbol $ für Truncation für genau ein Zeichen und # für eine beliebig lange Zeichenfolge; im ersten Fall spricht man auch von beschränkter Truncation, im zweiten Fall von unbeschränkter. Wir geben jeweils das Suchmuster an und einige Wörter, die Treffer für dieses Pattern sind:
    schreib#: schreiben, schreibt, schreibst, schreibe
    schreib$$: schreiben, schreibst
    #schreiben: schreiben, beschreiben, anschreiben, verschreiben
    $$schreiben: beschreiben, anschreiben
  - **Maskierung** oder genauer Mitten-Maskierung bezieht sich auf Zeichen in der Mitte eines Wortes; da im Deutschen bei der Konjugation und der Deklination von Wörtern nicht nur die Endung betroffen ist, werden solche Operationen benötigt:
    schr$$b#: schreiben, schrieb / schrauben
    h$$s#: Haus, Häuser / Hanse, hausen, hassen

Der wesentliche Vorteil der Truncation- und Maskierungsoperatoren besteht also darin, dass Flexions- und Derivationsformen von Wörtern zusammengeführt werden und Schreibarbeit gegenüber dem expliziten Aufzählen gespart wird. Möglicherweise werden dadurch aber auch unerwünschte Wörter zugelassen; daher zeigen die meisten Systeme zunächst die verschiedenen Wortformen, die ein Pattern erfüllen, so dass der Benutzer daraus auswählen kann. Das grundsätzliche Problem bei dieser Vorgehensweise ist aber, dass der Benutzer sich zunächst alle möglichen Wortformen vorstellen muss, um eine gute Anfrage zu formulieren.

- **Kontextoperatoren** dienen zur Suche nach mehrgliedrigen Ausdrücken. Da z.B. der Ausdruck "information retrieval" im Text auch in der Form "information storage and retrieval" oder "retrieval of information" auftreten kann, muss die Anfragesprache Operatoren anbieten, die die einfache Spezifikation solcher Formen ermöglichen. Ohne solche speziellen Operatoren wäre man auf die booleschen Operatoren angewiesen, die sich lediglich auf das Vorkommen der einzelnen Wörter irgendwo im selben Text beziehen. Folgende Kontextoperatoren werden häufig angeboten:
  - genauer Wortabstand ($):
    retrieval $ information: retrieval of information, retrieval with information loss
  - maximaler Wortabstand (#):
    text # # retrieval: text retrieval, text and fact retrieval
  - Wortreihenfolge (,):
    information # , retrieval: information retrieval, retrieval of information
  - gleicher Satz (.):
    information # retrieval. matcht nicht
    ... this information. Retrieval of data ...
    aber auch nicht:
    ... storage of information. Its retrieval ...

3.2.3 Computerlinguistischer Ansatz

Der computerlinguistische Ansatz versucht, Verfahren bereitzustellen, die die verschiedenen Flexions- und Derivationsformen eines Wortes zusammenführen. Analog sollen bei mehrgliedrigen Ausdrücken die verschiedenen möglichen Vorkommensformen erkannt werden. Im Gegensatz zum informatischen Ansatz, der zur Bewältigung dieser Probleme nur recht primitive Hilfsmittel zur Verfügung stellt, werden beim computerlinguistischen Ansatz Algorithmen bereitgestellt, die diese Transformationen automatisch ausführen. Dabei ist allerdings zu beachten, dass diese Aufgabe nicht in perfekter Art und Weise gelöst werden kann.

Es gibt folgende Arten von computerlinguistischen Verfahren:

- **graphematische Verfahren** basieren auf der Analyse von Buchstabenfolgen und werden im Bereich der Morphologie zur Zusammenführung von Flexions- oder Derivationsformen eines Wortes eingesetzt,
- **lexikalische Verfahren** basieren auf einem Wörterbuch, das zum einen mehrgliedrige Ausdrücke enthalten kann, andererseits die verschiedenen Bedeutungen mehrdeutiger Wörter verzeichnet,

- **syntaktische Verfahren** dienen hauptsächlich zur Identifikation von mehrgliedrigen Ausdrücken.

3.2.3.1 Graphematische Verfahren

In diesem Abschnitt sollen graphematische Algorithmen für die englische Sprache vorgestellt werden. Da das Englische im Gegensatz zum Deutschen nicht so stark flektiert ist, erreichen diese Algorithmen eine sehr hohe Genauigkeit und sind daher ohne Probleme praktisch einsetzbar. Es ist zwischen Grundform- und Stammformreduktion zu unterscheiden:

- Bei der **Grundformreduktion** werden Wörter auf ihre Grundform zurückgeführt. Die Grundform ist bei Substantiven der Nominativ Singular und bei Verben deren Infinitv. Je nach Art des Algorithmus wird unterschieden zwischen:
  - **formaler Grundform**, die durch das alleinige Abtrennen der Flexionsendung erzeugt wird, wie z.B.
    activities → activit
  - und **lexikographischer Grundform**, die durch
    Abtrennen der Flexionsendung und ggfs. anschließender Rekodierung entsteht, also z.B.
    applies → appl → apply
- Bei der **Stammformreduktion** werden (nach vorheriger Grundformreduktion) die Wörter auf ihren Wortstamm reduziert, indem die Derivationsendungen entfernt werden, z.B.:
  computer, compute, computation, computerization → comput

Lexikographische Grundformreduktion

Als Beispiel für einen Reduktionsalgorithmus soll hier eine vereinfachte Fassung der in [Kuhlen 77] beschriebenen lexikographischen Grundformreduktion vorgestellt werden. Hierzu verwenden wir folgende Notationen:

%	alle Vokale (einschließlich Y)
∗	alle Konsonanten
/	„oder“
Ø	Leerzeichen
→	„zu“

Die Regeln dieses (vereinfachten) Algorithmus' sind dann folgende:

1) **IES** → **Y**
2) **ES** → **Ø** wenn ∗O / CH / SH / SS / ZZ / X vorangehen
3) **S** → **Ø** wenn ∗ / E / %Y / %O / OA / EA vorangehen
4) **S'** → **Ø**
   IES' → **Y**
   ES' → **Ø**
5) **'S** → **Ø**
   ' → **Ø**
6) **ING** → **Ø** wenn ∗∗ / % / X vorausgehen
   ING → **E** wenn %∗ vorausgehen
7) **IED** → **Y**
8) **ED** → **Ø** wenn ∗∗ / % / X vorausgehen
   ED → **E** wenn %∗ vorausgehen

Der Algorithmus wendet jeweils nur die erste passende Regel an.

Nachfolgend geben wir einige Beispiele zu den einzelnen Regeln.

Regel 1	**IES** → **Y**

Beispiele zu 1:

APPLIES	→	APPLY
IDENTIFIES	→	IDENTIFY
ACTIVITIES	→	ACTIVITY

Regel 2 **ES** → **Ø**, wenn ∗O / CH / SH / SS / ZZ / X vorangehen

Beispiele zu 2:

BREACHES	→	BREACH
PROCESSES	→	PROCESS
FISHES	→	FISH
COMPLEXES	→	COMPLEX
TANGOES	→	TANGO
BUZZES	→	BUZZ

Regel 3 **S** → **Ø**, wenn ∗ / E / %Y / %O / OA / EA vorangehen

Beispiele zu 3:

METHODS	→	METHOD
HOUSES	→	HOUSE
BOYS	→	BOY
RADIOS	→	RADIO
COCOAS	→	COCOA
FLEAS	→	FLEA

Regel 4

S'	→	**Ø**
IES'	→	**Y**
ES'	→	**Ø**

Beispiele zu 4:

MOTHERS'	→	MOTHER
LADIES'	→	LADY
FLAMINGOES	→	FLAMINGO

Regel 5

'S	→	**Ø**
'	→	**Ø**

Beispiele zu 5:

MOTHER'S	→	MOTHER
CHILDREN'S	→	CHILDREN
PETRUS'	→	PETRUS

Regel 6 **ING** → **Ø**, wenn ∗∗ / % / X vorausgehen

ING → **E**, wenn %∗ vorausgehen

Beispiele zu 6:

DISGUSTING	→	DISGUST
GOING	→	GO
MIXING	→	MIX
LOOSING	→	LOOSE
RETRIEVING	→	RETRIEVE

Regel 7 **IED** → **Y**

Beispiel zu 7:

SATISFIED → SATISFY

Regel 8	**ED**	→	**Ø**, wenn ∗∗ / % / X vorausgehen
	ED	→	**E**, wenn %∗ vorausgehen

Beispiel zu 8:

DISGUSTED	→	DISGUST
OBEYED	→	OBEY
MIXED	→	MIX
BELIEVED	→	BELIEVE

3.2.3.2 Lexikalische Verfahren

Graphematische Verfahren sind für stark flektierte Sprachen wie z.B. das Deutsche wenig geeignet. Daher muss man hier lexikalische Verfahren einsetzen. Für den Einsatz im IR sollte ein Lexikon folgende Relationen enthalten (s.a. [Zimmermann 91]):

- Flexionsform (Vollformen) – zugehörige Grundform
  Hauses – Haus, ging – gehen
- Derivationsform – zugehörige Grundformen
  Lieblosigkeit – lieblos, Berechnung – rechnen
- Komposita – zugehörige Dekomposition
  Haustür – Tür, Armbanduhr – Uhr.

Lexikalische Verfahren haben generell den Nachteil, dass hier eine ständige Pflege des Wörterbuches notwendig ist. Für eine neue Anwendung ist zunächst ein hoher Anpassungsaufwand notwendig, um ein Standard-Wörterbuch mit den jeweiligen Fachbegriffen anzureichern. Auch später tauchen ständig neue Begriffe auf, die in das Lexikon aufgenommen werden müssen.

Substantivkomposita (die letzte Komponente ist ein Substantiv) machen im Deutschen weniger als 10% der Token, aber mehr als 50% der Types aus. Eine auch nur annähernd vollständige Auflistung der Komposita im Wörterbuch ist daher schon aus Aufwandsgründen kaum realistisch. Andererseits ist die Kompositazerlegung aber sehr wichtig, um alle Vorkommen eines Suchwortes zu finden, wie z.B. bei *Schweinebraten, Rinderbraten, Hirschbraten, Hühnerbraten, ...* oder *Kernenergie, Solarenergie, Kohleenergie, Windenergie, ...*

Ein Kompositum besteht nicht nur aus einer Reihe von Grundformen, sondern enthält zusätzlich *Fugenelemente* zur Verbindung derselben Fugenelemente können sein: -∅, -e, -en, -ens, -er, -n, -s sowie bei entlehnten Stämmen -i, -o, -al (z.B. in *Elektr-o-motor, Agr-i-kultur*) und natürlich der Bindestrich.

Leider gibt es keine allgemeingültigen Regeln, wann welches Fugenelement verwendet wird, wie die folgenden Beispiele zeigen: *Wind-energie* vs. *Sonne-n-ergie*, *Stadt-mitte* vs. *Städte-partnerschaft*, *Spargelder* vs. *Hilf-s-gelder* und *Schwein-e-braten* vs. *Wildschwein-braten*.

Um die im Text vorkommenden Komposita automatisch zu zerlegen, benötigt man ein Grundformen-Wörterbuch (das man auch weitestgehend vollautomatisch erstellen kann). Mit dessen Hilfe versucht man dann, Kandidaten für Komposita in Folgen von Grundformen mit verbindenden Fugenmorphemen zu zerlegen. Allerdings ist die Zerlegung nicht immer eindeutig, wie folgende Beispiele fehlerhafter Zerlegung zeigen: *Bausch-windel, Hafenbar-kasse, Kopfball-ast, Ster-befall, Tau-sender, Tram-polin.* Man sollte also alle möglichen Zerlegungen generieren und hoffen, dass die fehlerhaften Zerlegungen sich nicht zu negativ auf die Retrievalqualität auswirken.

3.2.3.3 Syntaktische Verfahren

Syntaktische Verfahren werden im IR hauptsächlich zur Identifikation von mehrgliedrigen Ausdrücken (Nominalphrasen) eingesetzt. Hierzu sind zwei Probleme zu lösen:

1. Wortklassenbestimmung: Zuordnung der syntaktischen Kategorie zu einzelnen Wörtern.
2. Parsing: Erkennen der syntaktischen Struktur. Für das Problem der Erkennung von Komposita muss keine vollständige syntaktische Analyse vorgenommen werden; es genügt, ein *partielles Parsing* zur Extraktion der relevanten Teilstrukturen.

Nachfolgend beschreiben wir diese beiden Probleme etwas detaillierter.

AT	article
BEZ	"is"
CONJ	conjunction
IN	preposition
JJ	adjective
JJR	comparative adjective
MD	modal (can, have, may, shall...)
NN	singular or mass noun
NNP	singular proper noun
NNS	plural noun
PERIOD	.:?!
PN	personal pronoun
RB	adverb
RBR	comparative adverb
TO	"to"
VB	verb, base form
VBD	verb, past tense
VBG	verb, present participle, gerund
VBN	verb, past participle
VBP	verb, non 3rd singular present
VBZ	verb, 3rd singular present
WDT	*wh*-determiner (what, which)

Tabelle 3.1: Häufig verwendete Wortklassen (für Englisch)

Wortklassenbestimmung Für die Definition von Wortklassen gibt es keinen Standard. Tabelle 3.1 zeigt jedoch eine häufig verwendetes Schema.

Um die Wortklassen in einem Text zu bestimmen, kann auf dieselben Datenquellen zurückgegriffen werden, die auch bei der morphologischen Analyse verwendet werden:

- Vollformen-Wörterbücher enthalten alle Flexionsformen von Wörtern (üblicherweise durch Angabe der Regelklasse für die möglichen Flexionen bei der Grundform); üblicherweise enthält der Eintrag auch die zugehörige(n) Wortklasse(n).
- graphematische Verfahren versuchen, aus der Wortendung und evtl. Präfixen auf die Wortklasse zu schließen. Wegen des grundsätzlichen Problems der Unvollständigkeit von Wörterbüchern sollten graphematische Verfahren in jedem Fall eingesetzt werden, um auch unbekannte Wörter klassifizieren zu können.

Nr.	Regel	Klasse
1	**IES** → **Y**	NNS/VBP
2	**ES** → **Ø**	NNS/VBP
3	**S** → **Ø**	NNS/VBP
4	**S'** → **Ø**	NNS
	IES' → **Y**	
	ES' → **Ø**	
5	**'S** → **Ø**	NN
	' → **Ø**	
6	**ING** → **Ø**	VBG
	ING → **E**	
7	**IED** → **Y**	VBD/VBN/JJ
8	**ED** → **Ø**	VBD/VBN/JJ
	ED → **E**	

Tabelle 3.2: Wortklassenzuordung basierend auf dem Kuhlen-Algorithmus

Ein einfaches Beispiel für ein graphematisches Verfahren ist die in Tabelle 3.2 dargestellte Zuord-

nung von Wortklassen anhand von Kuhlens Algorithmus zur Grundformreduktion. Leider liefern die meisten Regeln keine eindeutige Wortklassenzuordung.

In wenig flektierten Sprachen haben aber sowohl lexikalische als auch graphematische Verfahren mit einem grundsätzlichen Problem zu kämpfen: Vollformen können zu mehreren Wortklassen gehören, z.B.: *The boys **play** football* vs. *She saw the new **play***. Dieses Problem lässt sich nur durch die zusätzliche Berücksichtigung des Kontextes lösen, etwa in unserem Beispiel: AT NNS VBP/NN NN → VBP und PN VBD AT JJ NN/VBP → NN. Üblicherweise betrachtet man Folgen von zwei oder drei Wörtern (Bigramme, Trigramme) als Kontextinformation.

Allerdings lässt sich auch dadurch keine befriedigende Lösung erreichen. [Greene & Rubin 71] zeigten, dass selbst bei einem vollständigen Wörterbuch die Wortklassenzuordnung mit einem deterministischem Tagger nur 77 % korrekte Zuordungen liefert.

Durch den Übergang zu einem statistischen Ansatz lassen sich jedoch wesentlich bessere Ergebnisse erzielen. Dabei nutzt man die unterschiedliche Häufigkeit des Vorkommens in den verschiedenen Wortklassen aus (die meisten Wörter kommen in einer bevorzugten Wortklasse vor). Z.B. sind folgende Vorkommen eher selten:
*to **flour** a pan*
*to **web** the final report*
Ein einfacher Ansatz besteht daher darin, seltene Verwendungen zu ignorieren. So zeigten [Charniak et al. 93], dass sich durch dieses Vorgehen 90 % korrekte Zuordungen erreichen lassen. Weitere Verbesserungen sind durch statistische Ansätze zur Berücksichtigung der syntaktischen Struktur (z.B. Markov-Modelle) möglich, wodurch sich etwa 95–97 % korrekte Zuordungen erzielen lassen.

S	→	NP VP
NP	→	AT? JJ* NNS+
	→	AT? JJ* NN+
	→	NP PP
VP	→	VB PP
	→	VBZ
	→	VBZ NP
PP	→	IN NP

Tabelle 3.3: Einfache Beispielgrammatik

Parsing Basierend auf den zugeordneten Wortklassen kann man anschließend die syntaktische Struktur eines Textes bestimmen. Tabelle 3.3 zeigt eine einfache Grammatik (? steht für 0/1 Vorkommen, * für beliebig viele und + für mindestens einmaliges Vorkommen). Mit dieser Grammatik lassen sich die nachstehenden Beispielsätze analysieren:

- *The analysis of 25 indexing algorithms shows consistent retrieval performance.*
  AT NN IN JJ NN NNS VBZ JJ NN NN
- *A good indexing technique for Web retrieval is manual classification.*
  AT JJ NN NN IN NN NN VBZ JJ NN

Partielles Parsing Um Nominalphrasen beim Freitextretrieval zu erkennen, reicht in der Regel partielles Parsing aus. Dazu definiert man die relevanten syntaktischen Teilstrukturen. Lassen wir z.B. die Unterscheidung NN/NNP/NNS fallen, so könnte man folgende einfache Muster für Nominalphrasen definieren:

phrase	→	NN NN+
	→	NN+ IN JJ* NN+

Damit kann man folgende Phrasen erkennen:
indexing algorithms
retrieval performance
retrieval of Web documents
retrieval of new documents

Head-Modifier-Strukturen Ein Matching von Nominalphrasen auf der Ebene der syntaktischen Strukturen führt in der Regel zu unbefriedigenden Ergebnissen. Ein besserer Ansatz ist die Transformation der Nominalphrasen in sogenannte Head-Modifier-Strukturen. Für eine zweigliedrige Nominalphrasen bezeichnet dabei *head* das Nomen, das die wesentliche Bedeutung des Kompositums ausdrückt, z.B. information *retrieval*, indexing *algorithm*. Der *modifier* dagegen spezialisiert oder modifiziert die Bedeutung des heads.

Bei mehr als zweigliedrigen Ausdrücken ergeben sich geschachtelte Strukturen, die man in Listen- oder Baum-Form darstellen kann (siehe auch Abbildung3.1). Dabei steht jeweils der Modifier links und der Head rechts: *(((multimedia,document),retrieval),system)*

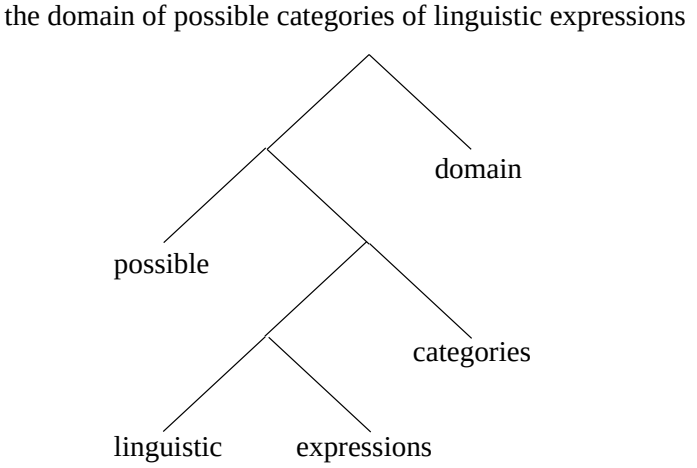

Abbildung 3.1: Beispiel für geschachtelte Head-Modifier-Struktur im Englischen

Analog kann man auch Komposita in Head-Modifier-Strukturen überführen: Bei zweigliedrigen Komposita ist die letzte Komponente der Head, wie etwa bei *(Tür-schloss)* vs. *Schloss-tür*. Bei mehrgliedrigen Komposita kann es dagegen zu Mehrdeutigkeiten kommen, wie etwa bei *(Mädchen(handels-schule)* – und nicht *(Mädchen-handels),schule)*. Hier hilft dann nur ein Wörterbuch.

Matching-Prozess Der Vergleich zwischen einem Kompositum aus der Anfrage und einem im Dokumenttext gefundenen läuft nun wie folgt ab:

1. Nominalphrasen/Komposita in Head-Modifier-Struktur überführen.
   Die Transformationsregeln basieren dabei primär auf der syntaktischen Struktur
2. Test, ob das Anfragewort in der Nominalphrase aus dem Dokument enthalten ist.
   Dabei müssen Head- bzw. Modifier-Rolle bzgl. der gemeinsamen Wurzel übereinstimmen. Ein einzelnes Nomen wird dabei als Head aufgefasst.

Zum Beispiel ist der Dokumentterm *(((semistructured,data), retrieval) system)* ein Treffer bzgl. der Anfrageterme *(retrieval, system)*, *(semistructured, data)* und *(data, retrieval)*, aber nicht für *(retrieval, data)*. Analog liefert im Deutschen die Suche nach *Tür* die *Haustür* und die *Zimmertür*, aber nicht das *Türschloss*, und die Suche nach *Mädchenhandel* würde bei der o.g. Bildungsstätte fehlschlagen.

3.3 Dokumentationssprachen

3.3.1 Allgemeine Eigenschaften

Dokumentationssprachen sollen die im vorangegangenen Abschnitt dargestellten Nachteile der Freitextsuche überwinden helfen. Um sich von der konkreten sprachlichen Formulierung in dem zu indexierenden Dokument zu lösen, wird eine davon unabhängige Repräsentation des Textinhaltes durch Verwendung eines speziellen Vokabulars verwendet. Dieses Vokabular soll alle Mehrdeutigkeiten und die Probleme morphologischer und syntaktischer Art der natürlichen Sprache vermeiden. In den folgenden Abschnitten betrachten wir zunächst zwei „klassische“ Arten von Dokumentationssprachen, nämlich Klassifikationen

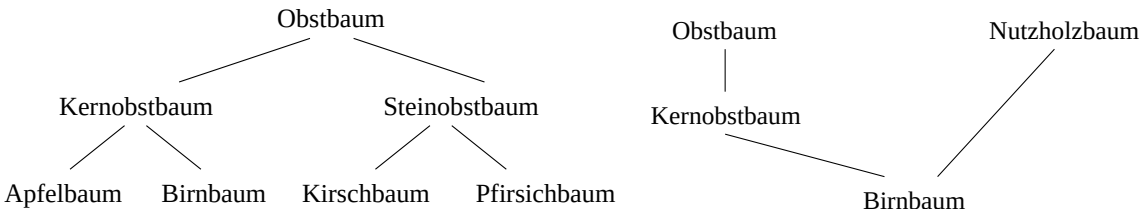

Abbildung 3.2: Monohierarchie (links) und Polyhierarchie (rechts)

und Thesauri. Diese Ausführungen orientieren sich im wesentlichen an der Darstellung in [Burkart 90]. Anschließend wird als Beispiel für einen moderneren Ansatz die Sprache RDF vorgestellt.

3.3.2 Klassifikationen

Klassifikationen dienen als Strukturierung eines Wissensgebietes nach einem vorgegebenen formalen Schema. Einem einzelnen Dokument wird dabei in der Regel nur eine Klasse zugeordnet. Aus dieser Randbedingung ergibt sich bereits eine prinzipielle Schwäche, da viele Dokumente ja gerade versuchen, Brücken zwischen verschiedenen Wissensgebieten zu schlagen, so dass sie zu mehreren Klassen gehören. Andererseits gibt es einige praktische Anwendungen, die gerade eine eindeutige Klassifikation von Dokumenten voraussetzen, z.B. bei der fachsystematischen Aufstellung von Büchern in einer Bibliothek oder bei der Anordnung von Abstracts in der gedruckten Fassung eines Referateorgans.

Die bekanntesten Beispiele für Klassifikationen sind die den Web-Katalogen (wie z.B. Yahoo!) zugrundeliegenden Ordnungssysteme. Daneben gibt es sehr viele fach- oder anwendungsspezifische Klassifikationen, wie z.B.
LCC Library of Congress Classification
DDC Dewey Decimal Classification
UDC Universal Decimal Classification
MSc Mathematics Subject Classification
CCS ACM Computing Classification system

3.3.2.1 Eigenschaften von Klassifikationssystemen

Wir betrachten zunächst einige grundlegende Eigenschaften von Klassifikationssystemen, bevor wir konkrete Beispiele vorstellen.

Monohierarchie – Polyhierarchie

Abbildung 3.2 zeigt links eine monohierarchische Klassifikation; hierbei sind die Klassen in eine Baumstruktur eingeordnet. Häufig reicht aber eine Baumstruktur nicht aus, um die Beziehungen zwischen den Klassen sinnvoll darzustellen. Deswegen geht man zu einer Polyhierarchie über, bei der eine Klasse mehrere Superklassen haben kann.

Monodimensionalität – Polydimensionalität

Bei der Festlegung der Klassenstruktur kann es häufig auf einer Stufe mehrere Merkmale geben, nach denen eine weitere Aufteilung in Unterklassen vorgenommen werden kann, wobei diese Merkmale orthogonal zueinander sind. Eine polydimensionale Klassifikation, wie das Beispiel in Abb. 3.3 links illustriert, erlaubt die Darstellung dieses Sachverhaltes. Erlaubt das Klassifikationsschema keine Polydimensionalität, dann muss diese durch Einführung einer zusätzlichen Hierarchieebene aufgelöst werden (Abb. 3.3 rechts), wodurch das Schema unübersichtlicher wird.

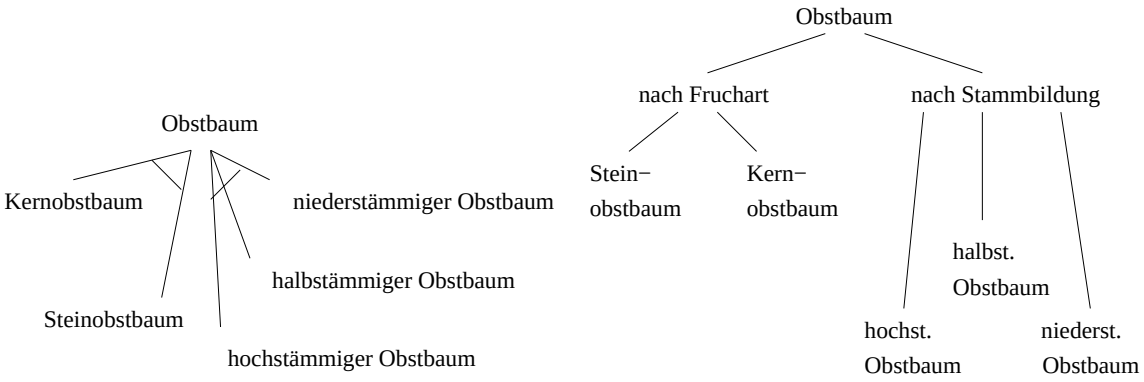

Abbildung 3.3: Polydimensionalität

Analytische vs. synthetische Klassifikation

Beim Entwurf eines Klassifikationsschemas gibt es – ähnlich wie bei der Programmierung – zwei mögliche Vorgehensweisen. Die bisherigen Beispiele illustrieren die analytische Klassifikation, die top-down vorgeht: Ausgehend von der Grundgesamtheit der zu klassifizierenden Objekte sucht man rekursiv jeweils nach dem nächsten Kriterium zur weiteren Aufteilung der Objektmenge.

Facette	Facette	Facette
A Fruchtart	B Stammart	C Erntezeit
A1 Apfel	B1 hochstämmig	C1 früh
A2 Birne	B2 halbstämmig	C2 mittel
A3 Kirsche	B3 niederstämmig	C3 spät
A4 Pfirsich		
A5 Pflaume		

Tabelle 3.4: Beispiel zur Facettenklassifikation

Im Gegensatz dazu geht die synthetische Klassifikation bottom-up vor. Dabei werden zuerst die relevanten Merkmale der zu klassifizierenden Objekte erhoben und im Klassifikationssystem zusammengestellt. Im zweiten Schritt werden dann die Klassen durch Kombination der Merkmale gebildet. Die synthetische Klassifikation bezeichnet man auch als **Facettenklassifikation**. Tabelle 3.4 zeigt eine solche Klassifikation für Obstbäume. In diesem Schema würde ein niederstämmiger Frühapfelbaum mit A1B3C1 klassifiziert. Für die Definition der Facetten gelten folgende Regeln:

1. Die Facetten müssen disjunkt sein.
2. Innerhalb einer Facette muss monodimensional unterteilt werden.

Zusätzlich müssen noch syntaktische Regeln definiert werden, die die Bildung der Klassen aus den Facetten festlegen.

3.3.2.2 Die Yahoo-Klassifikation

Abbildung 3.4 zeigt die Hauptklassen der Yahoo-Klassifikation, und Abbildung 3.5 die weitere Unterteilung der Hauptklasse „Computers & Internet“. Mit „@“ markierte Klassen bezeichen dabei Querverweise in der Klassenhierarchie. Das Ordnungssystem ist somit kein Baum, sondern ein gerichteter Graph. Typisch für Yahoo! ist ferner die variierende Tiefe des Ordnungssystems, die an manchen Stellen nur 3, an anderen bis zu 7 beträgt. Dabei können die zu klassifizierenden (Web-)Dokumente beliebigen Knoten zugeordnet werden. Somit enthält ein Knoten in der Regel die Verweise auf die zugehörigen Dokumente sowie die Liste der Unterklassen.

Arts & Humanities
Literature, Photography...
Business & Economy
B2B, Finance, Shopping, Jobs...
Computers & Internet
Internet, WWW, Software, Games...
Education
College and University, K-12...
Entertainment
Cool Links, Movies, Humor, Music...
Government
Elections, Military, Law, Taxes...
Health
Medicine, Diseases, Drugs, Fitness...
News & Media
Full Coverage, Newspapers, TV...
Recreation & Sports
Sports, Travel, Autos, Outdoors...
Reference
Libraries, Dictionaries, Quotations...
Regional
Countries, Regions, US States...
Science
Animals, Astronomy, Engineering...
Social Science
Archaeology, Economics, Languages...
Society & Culture
People, Environment, Religion...

Abbildung 3.4: Yahoo!-Hauptklassen

Art@
Bibliographies (6)
Communications and Networking (1146)
Computer Science@
Contests (26)
Conventions and Conferences@
Countries, Cultures, and Groups (38)
Cyberculture@
Data Formats (485)
Desktop Customization@
Desktop Publishing (53)
Dictionaries (24)
Employment@
Ethics (18)
Games@
Graphics (316)
Hardware (2355)
History (106)
Humor@
Industry Information@
Internet (6066)
Magazines@
Mobile Computing (65)
Multimedia (690)
Music@
News and Media (205)
...

Abbildung 3.5: Untergliederung der Hauptklasse Computers & Internet

3.3.2.3 Dezimalklassifikation

Als bekanntestes Beispiel für Klassifikationssysteme gilt sicher die Dezimalklassifikation. Sie geht auf die Dewey Decimal Classification (DDC) zurück, die 1876 von Melvil Dewey in den USA als Universalklassifikation zur Aufstellung von Buchbeständen konzipiert wurde. Daraus entwickelten dann die Belgier Paul Otlet und Henri Lafontaine durch das Hinzufügen von syntaktischen Elementen die **Universelle Dezimalklassifikation** (DK), die zur Inhaltserschließung geeignet ist.

Grundelemente der DK

Wir stellen im folgenden die wesentlichen Grundelemente der DK (Dezimalklassifikation) vor:

- Die Klassen der DK sind hierarchisch gegliedert. Wie der Name schon sagt, ist der maximale Verzweigungsgrad 10. Das gesamte System enthält derzeit über 130000 Klassen.
- Zusätzlich zu diesen Klassen erlauben **Anhängezahlen** die Facettierung.
- Zur Verknüpfung mehrerer DK-Zahlen dienen bestimmte Sonderzeichen.

Klassen der DK

Die DK-Haupttafeln umfassen folgende 10 Hauptabteilungen:

0 Allgemeines
1 Philosophie
2 Religion, Theologie
3 Sozialwissenschaften, Recht, Verwaltung
4 (zur Zeit nicht belegt)
5 Mathematik, Naturwissenschaften
6 Angewandte Wissenschaften, Medizin, Technik
7 Kunst, Kunstgewerbe, Photographie, Musik, Spiel, Sport
8 Sprachwissenschaft, Philologie, Schöne Literatur, Literaturwissenschaft
9 Heimatkunde, Geographie, Biographien, Geschichte

Diese Hauptklasse werden bis hin zu sehr speziellen Sachverhalten weiter untergliedert, wie etwa im folgenden Beispiel:

3 Sozialwissenschaften, Recht, Verwaltung
33 Volkswirtschaft
336 Finanzen. Bank- und Geldwesen
336.7 Geldwesen. Bankwesen. Börsenwesen
336.76 Börsenwesen. Geldmarkt. Kapitalmarkt
336.763 Wertpapiere. Effekten
336.763.3 Obligationen. Schuldverschreibungen
336.763.31 Allgemeines
336.763.311 Verzinsliche Schuldbriefe
336.763.311.1 Langfristig verzinsliche Schuldbriefe

Facettierende Elemente

Zur Facettierung in der DK dienen die Anhängezahlen, die durch spezielle Zeichen eingeleitet werden. Es gibt einerseits allgemeine Anhängezahlen, die überall in der DK verwendet werden dürfen, und andererseits spezielle Anhängezahlen, die nur für bestimmte Klassen innerhalb der DK erlaubt sind. Beispiele für allgemeine Anhängezahlen sind folgende (die jeweils einleitende Zeichenfolge ist vorangestellt):

= Sprache: *=30 ≙ deutsch*
(0...) Form: *(021) ≙ Handbuch,*
53(021)=20=30=40 ≙ Handbuch der Physik in Englisch, Deutsch, Französisch
(...) Ort
(=...) Rassen und Völker
„...“ Zeit
.00 Gesichtspunkt
-05 Person

Verknüpfung von DK-Zahlen

Zur Verknüpfung von DK-Zahlen gibt es als syntaktische Elemente spezielle Sonderzeichen:

+ Aufzählung mehrerer Sachverhalte, *z.B. 178.1+33 ≙ Alkoholismus und Volkswirtschaft*
: Beziehung zwischen zwei Sachverhalten, *z.B. 178.1:33 ≙ Auswirkung von Alkoholismus auf die Volkswirtschaft*
/ Erstreckungszeichen (zur Zusammenfassung mehrerer nebeneinanderstehender DK-Zahlen), *z.B. 592/599 Systematische Zoologie: 592 bis einschließlich 599*
' Zusamenfassungszeichen zur Bildung neuer Sachverhalte aus der Kombination einzelner DK-Komponenten

3.3.2.4 Computing Classification System

Als weiteres Beispiel eines Klassifikationsschemas zeigen wir hier aus dem Bereich der Informatik das von der ACM entwickelte Computing Classification System (CCS). Darüber hinaus wird es auch in vielen anderen Informatik-Zeitschriften verwendet und liegt insbesondere auch der einschlägigen Datenbank *Compuscience* zugrunde.

Das CCS besteht aus folgenden Elementen:

- Die *general terms* sind eine vorgegebene Menge von allgemeinen Begriffen, die zur Facettierung dienen.
- Die *classification codes* stellen eine dreistufige monohierarchische Klassifikation dar.
- Innerhalb einer einzelnen Klasse dienen die *subject headings* zur weiteren Untergliederung. Neben der für jede Klasse vorgegebenen Menge von natürlichsprachlichen Bezeichnungen sind auch alle Eigennamen als subject headings erlaubt.
- Schließlich können jedem Dokument noch *free terms* als zusätzliche, frei wählbare Stichwörter zugeordnet werden.

General terms:
Die general terms des CCS sind in Tabelle 3.5 aufgelistet.

ALGORITHMS	MANAGEMENT
DESIGN	MEASUREMENT
DOCUMENTATION	PERFORMANCE
ECONOMICS	RELIABILITY
EXPERIMENTATION	SECURITY
HUMAN FACTORS	STANDARDIZATION
LANGUAGES	THEORY
LEGAL ASPECTS	VERIFICATION

Tabelle 3.5: General terms der CR Klassifikation

Klassen und subject headings

Die Hauptklassen des CCS sind folgende:

A. GENERAL LITERATURE
B. HARDWARE
C. COMPUTER SYSTEMS ORGANIZATION
D. SOFTWARE
E. DATA
F. THEORY OF COMPUTATION
G. MATHEMATICS OF COMPUTING
H. INFORMATION SYSTEMS

I. COMPUTING METHODOLOGIES
J. COMPUTER APPLICATIONS
K. COMPUTING MILIEUX

Am Beispiel der Klasse H.3 zeigen wir die classification codes und die zugehörigen subject headings:

H.3 INFORMATION STORAGE AND RETRIEVAL
- H.3.0 General
- H.3.1 Content Analysis and Indexing
  - Abstracting methods
  - Dictionaries
  - Indexing methods
  - Linguistic processing
  - Thesauruses
- H.3.2 Information Storage
  - File organization
  - Record classification
- H.3.3 Information Search and Retrieval
  - Clustering
  - Query formulation
  - Retrieval models
  - Search process
  - Selection process
- H.3.4 System and Software
  - Current awareness systems (selective dissemination of information-SDI)
  - Information networks
  - Question-answering (fact retrieval) systems
- H.3.5 Online Information Services
  - Data bank sharing
- H.3.6 Library Automation
  - Large text archives
- H.3.mMiscellaneous

3.3.3 Thesauri

Nach DIN 1463 ist ein Thesaurus eine geordnete Zusammenstellung von Begriffen mit ihren (natürlichsprachlichen) Bezeichnungen. Die wesentlichen Merkmale eines Thesaurus sind folgende:

a) terminologische Kontrolle durch
   - Erfassung von Synonymen,
   - Kennzeichnung von Homographen und Polysemen,
   - Festlegung von Vorzugsbenennungen,

b) Darstellung von Beziehungen zwischen Begriffen.

3.3.3.1 Terminologische Kontrolle

Die terminologische Kontrolle soll zur Reduktion von Mehrdeutigkeiten und Unschärfen der natürlichen Sprache dienen. Hierzu dienen die Synonymkontrolle, die Polysemkontrolle und die Zerlegungskontrolle.

Synonymkontrolle

Bei der Synonymkontrolle werden Bezeichnungen zu Äquivalenzklassen zusammengefasst. Man kann folgende Arten von Synonymie unterscheiden:

- Schreibweisenvarianten
  Friseur – Frisör
  UN – UNO – Vereinte Nationen
- unterschiedliche Konnotationen, Sprachstile, Verbreitung
  Telefon – Fernsprecher
  Pferd – Gaul
  Myopie – Kurzsichtigkeit
- Quasi-Synonyme
  Schauspiel – Theaterstück
  Rundfunk – Hörfunk

Im Thesaurus werden darüber hinaus Begriffe mit geringen oder irrelevanten Bedeutungsdifferenzen zu Äquivalenzklassen zusammengefasst:

- unterschiedliche Spezifität
  Sprachwissenschaft – Linguistik
- Antonyme
  Härte – Weichheit

- zu spezieller Unterbegriff
  Weizen – Winterweizen
- Gleichsetzung von Verb und Substantiv / Tätigkeit und Ergebnis
  Wohnen – Wohnung.

Die Entscheidung, ob zwei Begriffe als Quasisynonyme zu behandeln sind, hängt dabei immer von der jeweiligen Anwendung ab.

Polysemkontrolle
Bei der Polysemkontrolle werden mehrdeutige Bezeichnungen auf mehrere Äquivalenzklassen aufgeteilt. Man kann hierbei noch zwischen Homographen *(Bsp. „Tenor“)* und eigentlichen Polysemen *(Bsp. „Bank“)* unterscheiden.

Zerlegungskontrolle

Bei der Zerlegungskontrolle ist die Frage zu beantworten, wie spezifisch einzelne Begriffe im Thesaurus sein sollen. Gerade im Deutschen mit seiner starken Tendenz zur Kompositabildung *(Bs. Donaudampfschiffahrtsgesellschaftskapitän)* ist die Bildung zu spezieller Begriffe eine große Gefahr. Diese **Präkoordination** führt zu folgenden Nachteilen:

- Der Thesaurus wird zu umfangreich und unübersichtlich.
- Zu einer Äquivalenzklasse gibt es keine oder nur wenige Dokumente in der Datenbank

Den entgegengesetzten Ansatz verfolgt das UNITERM-Verfahren: Hierbei werden nur solche Begriffe (Uniterms) in den Thesaurus aufgenommen, die nicht weiter zerlegbar sind. Zur Wiedergabe eines Sachverhaltes müssen dann mehrere Uniterms verkettet werden. Diese sogenannte **Postkoordination** führt aber zu größerer Unschärfe beim Retrieval *(Beispiel: Baum + Stamm = Baumstamm / Stammbaum).*

Bei der Thesaurusmethode versucht man, durch einen Kompromiss zwischen beiden Ansätzen deren Nachteile zu vermeiden.

3.3.3.2 Äquivalenzklasse – Deskriptor

Die terminologische Kontrolle liefert Äquivalenzklassen von Bezeichnungen. Diese können auf zwei verschiedene Arten dargestellt werden:

1. In einem **Thesaurus ohne Vorzugsbenennung** werden alle Elemente der Äquivalenzklasse gleich behandelt, d.h., jedes Element steht für die Äquivalenzklasse. Diese Vorgehensweise wird wegen des erhöhten Aufwands selten angewandt.
2. Bei einem **Thesaurus mit Vorzugsbenennung** wird ein Element der Äquivalenzklasse zur Benennung ausgewählt. Dieses Element bezeichnet man dann als **Deskriptor**.

Im folgenden betrachten wir nur Thesauri mit Vorzugsbenennung.

3.3.3.3 Beziehungsgefüge des Thesaurus'

Neben der terminologischen Kontrolle ist die Darstellung von Beziehungen zwischen Begriffen die zweite Hauptaufgabe eines Thesaurus. Dabei werden verschiedene Arten von Beziehungen unterschieden.

Äquivalenzrelation

Äquivalenzrelationen verweisen von Nicht-Deskriptoren auf Deskriptoren. Sie werden meist bezeichnet als „**Benutze Synonym**“ (BS) oder im Englischen als USE-Relation. Die Umkehrrelation bezeichnet man als „**Benutzt für**“ (BF, im Englischen “used for” (UF)). Beispiele hierfür sind:
(*Fernsprecher* **BS** *Telefon und Telefon* **BF** *Fernsprecher*)

Hierarchische Relation

Hierarchische Relationen verbinden jeweils zwei Deskriptoren. Man bezeichnet sie als „**Unterbegriff**“ (UB) bzw. „**Oberbegriff**“ (OB), im Englischen “narrower term” (NT) und “broader term” (BT). Beispiele: *Obstbaum* **UB** *Steinobstbaum und Steinobstbaum* **OB** *Obstbaum*

Assoziationsrelation
Die Assoziationsrelation verweist von einem Deskriptor auf einen begriffsverwandten anderen Deskriptor. Im Gegensatz zu den beiden anderen Relationen ist die Assoziationsrelation symmetrisch. Man bezeichnet sie als „**verwandter Begriff**“ (VB, im Englischen “related term” (RT)). Beispiele:
Obstbaum **VB** *Obst und Obst* **VB** *Obstbaum*

Information retrieval	
UF	CD-ROM searching
	Data access
	Document retrieval
	Online literature searching
	Retrieval, information
BT	Information science
NT	Query formulation
	Query processing
	Relevance feedback
RT	Bibliographic systems
	Information analysis
	Information storage
	Query languages

Query processing	
UF	Data querying
	Database querying
	Query optimisation
BT	Information retrieval
RT	Database management systems
	Database theory
	DATALOG
	Query languages

Query formulation	
UF	Search strategies
BT	Information retrieval

Relevance feedback	
BT	Information retrieval

Abbildung 3.6: Auszug aus dem Beziehungsgefüge des INSPEC-Thesaurus’

3.3.3.4 Darstellung des Thesaurus

Deskriptor-Einträge

Ein Deskriptor-Eintrag in einem Thesaurus enthält neben der Vorzugsbenennung häufig noch mehrere der folgenden Angaben:

- Begriffsnummer
- Notation / Deskriptor-Klassifikation
- Scope note / Definition
- Synonyme
- Oberbegriffe / Unterbegriffe
- Verwandte Begriffe
- Einführungs-/ Streichungsdatum

Abbildung 3.7 zeigt ein Beispiel für einen Ausschnitt aus einem Thesaurus.

Gesamtstruktur des Thesaurus

Bei einem IR-System, das zur Recherche in einer Datenbasis mit Thesaurus verwendet wird, sollte auch der Thesaurus zugreifbar sein, wobei spezielle Funktionen zum Suchen im Thesaurus und mit Hilfe des Thesaurus angeboten werden sollten (z.B. wahlweise Einbeziehen von allen Unter-/Oberbegriffen). Daneben ist der Thesaurus aber meistens auch in gedruckter Form verfügbar. Der **Hauptteil** eines Thesaurus enthält dabei die Deskriptor-Einträge, die entweder alphabetisch oder systematisch geordnet sind. Darüber hinaus enthält ein Thesaurus in der Regel noch zusätzliche Register mit Verweisen auf die Deskriptor-Einträge:

0.0058
Magnetband
VB Magnetbandlaufwerk

0,0045
Magnetbandgerät
BS Magnetbandlaufwerk NE7

0. 0046
Magnetbandkassette
NO NE83
BF Kassette
BF MB-Kassette
OB Datenträger
VB Magnetbandkassettenlaufwerk

0.0051
Magnetbandkassettengerät
BS Magnetbandkassettenlaufwerk NE7

0.0050
Magnetbandkassettenlaufwerk
NO NE7
BF Magnetbandkassettengerät
BF MB-Kassettengerät
OB Datenausgabegrät
OB Dateneingabegerät
OB Datenspeichertechnik
VB Magnetbandkassette

0.0044
Magnetbandlaufwerk
NO NE7
BF Magnetbandgerät
OB Bandgerät
OB Datenausgabegerät
OB Dateneingabegerät
OB Datenspeichertechnik
VB Magnetband

0.0059
Magnetfeld
NO WD2
OB Magnetismus

0.0060
Magnetismus
NO WD2
BF Barkhausen-Effek
BF Ferromagnetismus

Magnetismus (Forts.)
BF Halleffekt
BF Induktion
OB Elektrodynamik
UB Magnetfeld
BIK Geophysik
BFK Erdmagnetismus
BIK Optik
BFK Faraday-Effekt

0.0070
Magnetkarte
NO NE87
BF Telefonkärtchen
OB Datenträger
VB Kartensystem

0.0073
Magnetkartensystem
NO ECS
OB Kartensystem

0.0074
Magnetkartentelefon
NO GK72
BF Makatel
OB Kartentelefon

0 0077
Magnetplatte
NO NE82
OB Datenspeicher
OB Datenträger
VB Magnetplattenlaufwerk
BIK Datenspeicher
BFK Plattenspeicher

0.0081
Magnetplattengerät
BS Magnetplattenlaufwerk NE7

0.0079
Magnetplattenlaufwerk
NO NE7
BF Magnetplattengerät
OB Datenausgabegerät
OB Dateneingabegerät
OB Datenspeichertechnik
VB Magnetplatte

Abbildung 3.7: Auszug aus einem Thesaurus

- komplementär zum Hauptteil eine systematische bzw. alphabetische Auflistung der Deskriptoren,
- für mehrgliedriger Bezeichnungen einen speziellen Index für deren Komponenten:
  - KWIC – keyword in context

    computer *system*
    storage *system*
    system *analysis*
    system *design*
  - KWOC – keyword out of context

    system:
    computer ...
    storage ...
    ... *analysis*
    ... *design*

3.3.3.5 Thesauruspflege

Da ein Anwendungsgebiet nie statisch ist und man darüber hinaus auch nicht annehmen kann, dass die erste Version eines Thesaurus bereits alle Ansprüche erfüllt, ist eine ständige Pflege des Thesaurus' notwendig. Insbesondere erfordern folgende Faktoren eine laufende Anpassung des Thesaurus':

- Entwicklung des Fachgebietes,
- Entwicklung der Fachsprache,
- Analyse des Indexierungsverhaltens und der Indexierungsergebnisse,
- Beobachtung des Benutzerverhaltens,
- Analyse der Rechercheergebnisse.

Bei den daraus resultierenden Änderungen muss darauf geachtet werden, dass die Konsistenz des Thesaurus' erhalten bleibt.

3.3.4 Ontologien

Ontologien sind in den letzten Jahren sehr populär geworden. Sie haben ihren Ursprung in den semantische Netzen aus der künstlichen Intelligenz, die zuerst in den 1970er konzipiert wurden, dann stärker formalisiert als terminologische Logiken bzw. Beschreibungslogiken diskutiert wurden und seit einigen Jahren nun im Zusammenhang mit dem "semantic web" wieder aufgelebt sind. Es gibt eine ganze Reihe von Formalismen/Sprachen für Ontologien. Am populärsten sind das vom W3C schon vor mehr als zehn Jahren vorgestellte RDF (Resource Description Framework) und RDF Schema, sowie das deutlich jüngere OWL (Web Ontology Language). Wir gehen hier nicht auf die Besonderheiten der einzelnen Ansätze ein, sondern beschreiben nur die wesentlichen Ideen.

Ontologien vereinigen Konzepte aus Datenbankschemata und Thesauri in sich: Von den Thesauri wurden die Begriffshierarchien sowie die Relationen zwischen den Begriffen übernommen. Von den Datenbank-Schemata stammen die Attribute, Beziehungstypen und insbesondere die Möglichkeit, Instanzen zu Konzepten zu benennen, wobei die möglichen Instanzen durch Bezugnahme auf Datentypen eingeschränkt werden können.

3.3.4.1 Ontologien: Konstrukte

Ontologiesprachen stellen folgende Konstrukte zur Definition einer Ontologie bereit: Konzepte/Klassen, Vererbungsbeziehungen, Eigenschaften/Relationen, sowie Instanzen. Einige Sprachen bieten zudem die Möglichkeit, zusätzlich Regeln zu definieren.

Konzepte/Klassen. Konzepte einer Ontologie werden üblicherweise als Klassen aufgefasst, wobei eine Klasse eine Menge von Instanzen mit gleichen oder ähnlichen Eigenschaften umfasst (analog zu objektorientierter Programmierung). Beispiele wären etwa *Student* als Klasse aller Studenten, *Reiseziel* als Menge aller möglichen Destinationen sowie *Information Retrieval* als Menge aller möglichen IR-Themen.

Vererbung ist ebenfalls wie in der objektorientierten Programmierung als Teilmengenbeziehung zwischen Klassen definiert. So sind etwa *Bachelor-Student* und *Master-Student* Unterklassen von *Student*, *Hiwi* ist sowohl Unterklasse von *Student* als auch von *Mitarbeiter*, und *Klassifikation* und *Indexierung* sind Unterklassen von *Information Retrieval*. Allerdings muss man beachten, dass nicht alle Ontologie-Sprachen Mehrfachvererbung erlauben.

Slots: Eigenschaften/Relationen. Ein Konzept hat i.d.R. mehrere Slots, wobei ein Slot eine Eigenschaft oder eine Beziehung/Relation beschreibt; zwischen diesen Aspekten wird allerdings nicht unterschieden. Die Instanzen eines Konzeptes unterscheiden sich in den Werten für die Slots. Der Wert ist entweder von elementarem Datentyp oder einer Klasse. Für das Konzept *Student* könnten wir etwa als Eigenschaften *Name: string, Matrikelnr: integer, Semester: integer* definieren und als Relationen *studiert → Studiengang, hört → Vorlesung*.

Die zulässigen Werte für einen Slot lassen sich auf verschiedene Arten einschränken. Zunächst unterscheidet man bei einem Slot zwischen *Domain* und *Range*. Dabei bezeichnet Domain die Menge der Konzepte, bei denen dieser Slot vorkommt, und Range ist Klasse bzw. der Datentyp, der als Wert für den Slot zulässig ist. Für obiges Beispiel haben wir etwa *Domain(Name)= {Mitarbeiter,Student}, Range(Name)=string* sowie *Domain(studiert)={Student}, Range(studiert) = Studiengang*.

Die zweite Möglichkeit der Einschränkung betrifft die Kardinalität eines Slots; hier kann man üblicherweise die minimale und die maximale Anzahl von Werten angeben, die ein einzelner Slot haben darf – in unserem Beispiel etwa *card(Name)=(1,1)* und *card(studiert)= (1,2)*.

Als Drittes kann man auch Vererbungsbeziehungen auf Slots definieren. Dann sind die Instanzen des spezielleren Slots auch Instanzen des generelleren Slots. So könnte man etwa *hört_Pflicht* und *hört_Wahlpflicht* als Spezialisierung von *hört* definieren. Der speziellere Slot kann (muss aber nicht) bzgl. Domain, Range, oder Kardinalität eingeschränkt sein.

Schließlich gibt es noch die Möglichkeit, einen Default-Wert für einen Slot anzugeben, der gilt, solange kein expliziter Slot-Wert angegeben wird.

Bei der Vererbung werden die Slots an die Unterklassen vererbt – wie bei der objektorientierten Programmierung. Somit erbt eine Unterklasse alle Slots ihrer Oberklasse, sie kann aber weitere Slots haben. Zudem können die vererbten Slots eingeschränkt werden: entweder bzgl. des Range (zulässige Werte), indem man eine Unterklasse des Range der Oberklasse angibt *(Bs.: Ingenieurstudent studiert → Ingenieurstudiengang)*, bzgl. der Kardinalität *(Bs.: Diplomand: card(hört) = (0,0))*, oder indem man den Slot durch einen spezielleren ersetzt.

Eingabe von Instanzen. Nachdem man die Klassen mit ihren Slots definiert hat, kann man die Ontologie mit Instanzen füllen, zu denen man die jeweiligen Slot-Werte angibt. Dabei muss die Instanz einer Klasse alle Bedingungen der Klasse erfüllen.

3.3.4.2 Retrieval

Nach dem Füllen der Ontologie mit Instanzen kann man Retrieval auf der Datenbasis durchführen. Dabei sucht man nach Instanzen einer Klasse (mit allen Unterklassen), die zusätzlich bestimmte Wertebedingungen erfüllen, z.B. *„Ingenieurstudenten mit Zweitstudiengang“* oder *„Studenten mit mehr als 8 Semestern, die IR hören“*

Ein Beispiel für ein Ontologie-basiertes Retrievalsystem ist YAGO[1], das in automatisch extrahierten Instanzen aus Wikipedia sucht (siehe Abbildung 3.8). Zu YAGO gehört auch der in Abbildung 3.9 gezeigte Ontologie-Browser.

Ein anderes Ontologie-basiertes System ist Freebase[2], das auf einer manuell erstellten Dokumenten- und Faktenbasis basiert. Als Beispiel zeigt Abbildung das erste Antwortdokument auf die Frage nach „Angela Merkel“.

[1] `http://www.mpi-inf.mpg.de/yago-naga/yago/demo.html`
[2] `http://www.freebase.com`

Abbildung 3.8: Retrieval mit YAGO

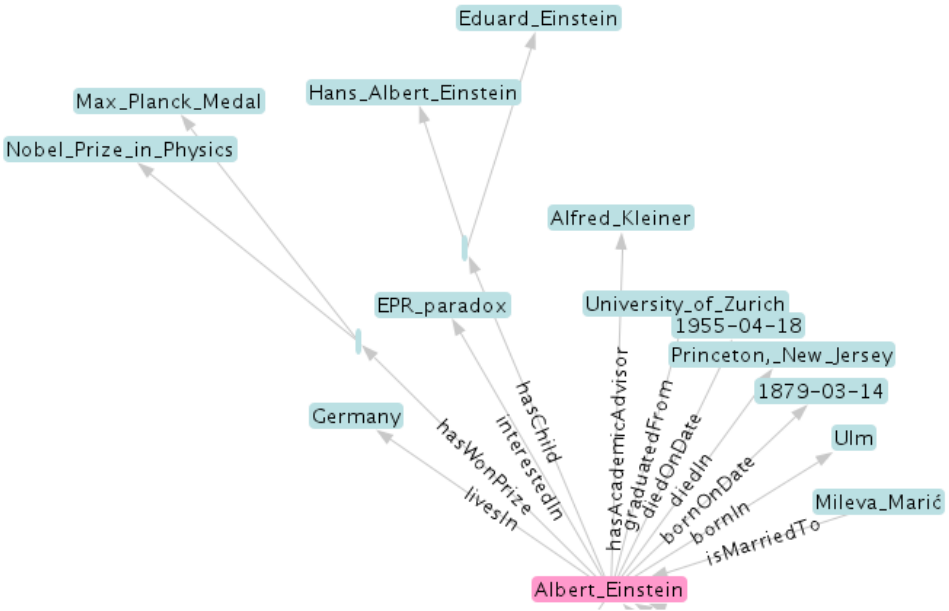

Abbildung 3.9: Ontologie-Browser zu YAGO

3.3.4.3 Werkzeuge und Anwendungen

Für die Entwicklung von Ontologien gibt es spezielle Ontologie-Editoren. Neben der graphischen Darstellung der Ontologie überprüfen diese die Widerspruchsfreiheit der Ontologie sowie die Erfüllung der Konsistenzbedingungen für die eingegebenen Instanzen. Abbildung 3.10 zeigt als Beispiel den Editor Protegé[3] Eine Auflistung weiterer Editoren findet sich z.B. bei Wikipedia[4].

Wiederverwendung von Ontologien. Um nicht für jede neue Anwendung eine eigene Ontologie definieren zu müssen, sollte man zunächst versuchen, vorhandene Ontologien wiederzuverwenden. Hierzu gibt es zum einen Ontologie-Bibliotheken, die frei verfügbare Ontologien sammeln, und zum anderen gibt es generelle Ontologien, die man für eigene Zwecke geeignet verfeinern kann. Populäre Beispiele für letzter

[3] `http://protege.stanford.edu`
[4] `http://en.wikipedia.org/wiki/Ontology_editor`

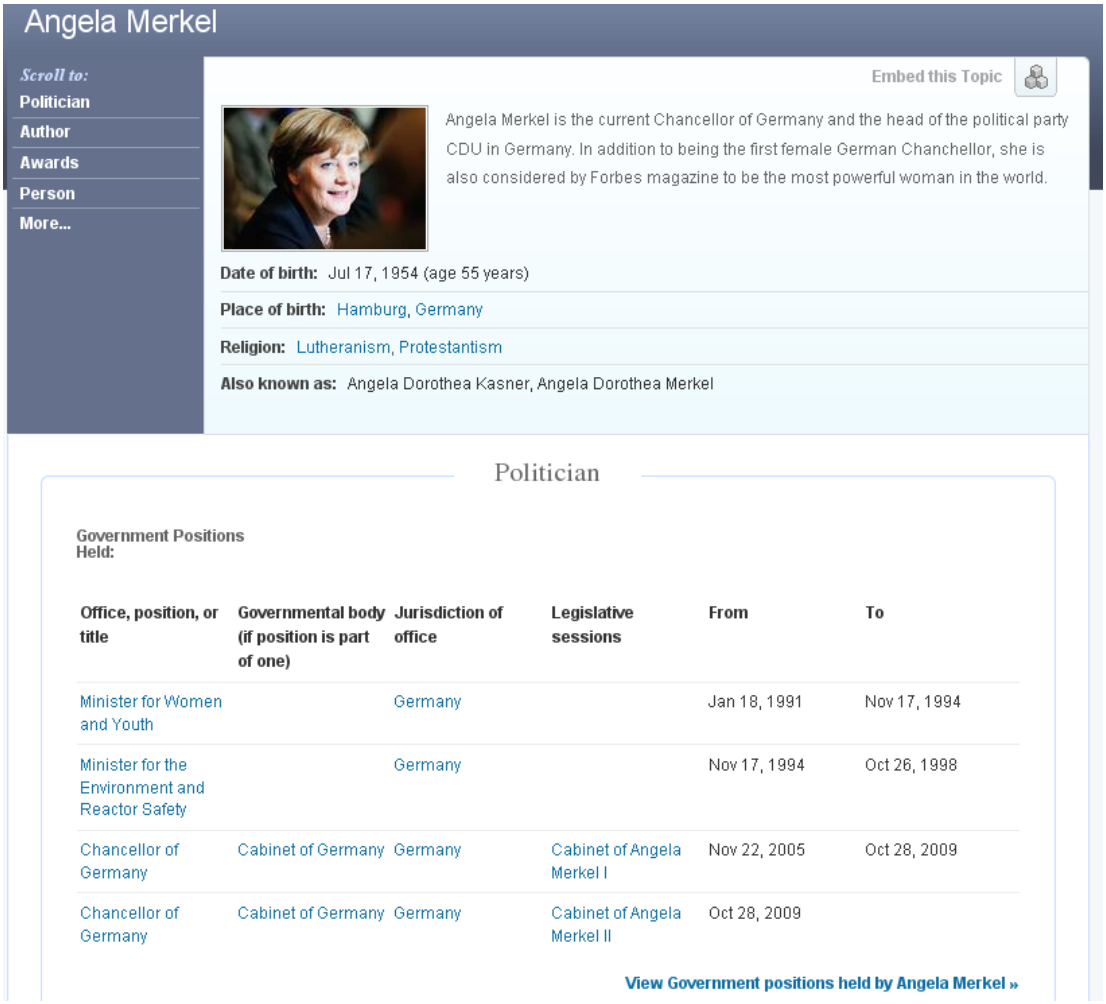

Abbildung 3.10: Freebase: Erste Antwort zu „Angela Merkel“

sind etwa die DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering)[5], Cyc[6], DMOZ[7] und WordNet[8].

3.3.5 Dokumentationssprachen vs. Freitext

Beim Vergleich mit der Freitextsuche sind folgende Vor- und Nachteile von Dokumentationssprachen zu nennen:

+ Durch die Abbildung verschiedener Textformulierungen auf eine einzige Bezeichnung kann ein höherer Recall erreicht werden.
+ Da das kontrollierte Vokabular keine mehrdeutigen Begriffe zulässt, kann auch eine höhere Precision erreicht werden.
+ Da ein Benutzer ein gesuchtes Konzept nur auf die entsprechende Benennung in der Dokumentationssprache abbilden muss, ergibt sich eine größere Benutzerfreundlichkeit.
– Die Benutzung des Systems setzt die Kenntnis der Dokumentationssprache voraus; für gelegentliche Benutzer ist diese Hürde zu hoch.

[5] http://www.loa-cnr.it/DOLCE.html
[6] http://www.cyc.com
[7] http://www.dmoz.org
[8] http://www.cogsci.princeton.edu/~wn/

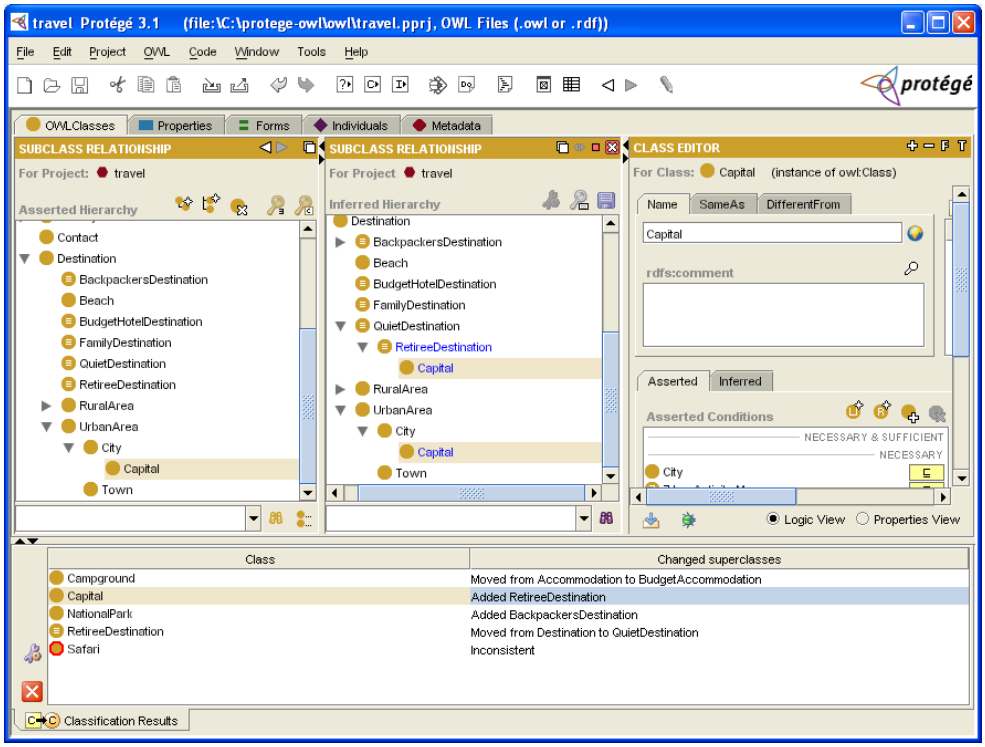

Abbildung 3.11: Oberfläche des Ontologie-Editors Protegé

– Aufgrund der i.a. groben Granularität des kontrollierten Vokabulars kann bei spezifischen Anfragen die Precision im Vergleich zur Freitextsuche sinken.
– Bei der Eingabe neuer Dokumente in die Datenbasis erhöht sich der Erschließungsaufwand deutlich, weil die Klassifikation bzw. Indexierung meist manuell erfolgt. Allerdings verringert sich durch diese Maßnahme der Aufwand bei den Recherchen, so dass die Gesamtbilanz wohl eher positiv ist.

Um die Nachteile des kontrollierten Vokabulars bei der Recherche zu kompensieren, kombinieren heutige kommerziell angebotenen Datenbasen beide Suchmöglichkeiten, so dass die Dokumentationssprache die Freitextsuche ergänzt.

3.4 Beurteilung der Verfahren zur Repräsentation von Textinhalten

- Obwohl rein intuitiv die Vorteile von Dokumentationssprachen überzeugen, ist deren Nutzen jedoch wissenschaftlich sehr umstritten. Der Grund hierfür ist die unzureichende experimentelle Basis für diesen Vergleich. Seit den Anfang der 60er Jahre von Cyril Cleverdon geleiteten Cranfield-Experimenten [Cleverdon 91], wo alle Dokumentationssprachen deutlich schlechter abschnitten als eine Freitextsuche mit Terms in Stammform, neigt die Mehrzahl der IR-Forscher zu der Ansicht, dass Dokumentationssprachen überflüssig sind. Allerdings wurden die damaligen Experimente mit nur 1400 Dokumenten durchgeführt, so dass die Gültigkeit der Resultate für heutige Datenbasen in der Größenordnung von 10^6 Dokumenten mit Recht bezweifelt werden muss. Auch einige wenige neuere Vergleiche [Salton 86] lassen keine endgültige Aussage zu dieser Problematik zu.
- Im Rahmen der TREC-Initiative werden verschiedene IR-Verfahren auf Datenbasen mit mehreren GB Text angewendet und die Ergebnisse miteinander verglichen. Die auf den TREC-Konferenzen [Voorhees & Harman 00] präsentierten Ergebnisse zeigen, dass halb-formale Konzepte (wie z.B. geographische oder Datumsangaben) durch eine reine Freitextsuche nicht abzudecken sind, so dass zumindest für diesem Bereich Dokumentationssprachen notwendig sind.

- Es liegt nahe, nach dem Einsatz von wissensbasierten Verfahren im IR zu fragen. Frühere Studien (z.B. [Krause 92]) haben den anfänglichen Optimismus stark gedämpft. Die seit einigen Jahren populären Semantic-Web-Ansätze sind bislang ebenfalls den Nachweis schuldig geblieben, dass sie für Datenbanken realistischer Größenordnung traditionellen Ansätzen (wie z.B. Thesauri) überlegen sind.
- Syntaktische Verfahren sind wohl hauptsächlich für die Identifikation von Nominalphrasen einsetzbar.
- Maschinenlesbare Wörterbücher sind in immer größerem Maße verfügbar. Sie unterstützen die morphologische Analyse bei stark flektierten Sprachen und die Erkennung von Nominalphrasen. Einige Forschungsgruppen untersuchen auch deren Einsatz für die Disambiguierung von Begriffen.

3.5 Zusammenhang zwischen Modellen und Repräsentationen

3.5.1 Textrepräsentation für IR-Modelle

Abschließend zu diesem Kapitel soll eine Einordnung der verschiedenen vorgestellten Ansätze zur Repräsentation von Textinhalten im Hinblick auf ihre Kombination mit IR-Modellen versucht werden.

3.5.2 Einfache statistische Modelle

Zunächst illustrieren wir die Vorgehensweise bei der Freitextindexierung an einem Beispieltext:

Experiments with Indexing Methods.
The analysis of 25 indexing algorithms has not produced consistent retrieval performance. The best indexing technique for retrieving documents is not known.

Zunächst werden die (oben unterstrichenen) Stoppwörter entfernt:

experiments indexing methods analysis indexing algorithms produced consistent retrieval performance best indexing technique retrieving documents known.

Die anschließende Stammformreduktion liefert folgendes Ergebnis:

experiment index method analys index algorithm produc consistent retriev perform best index techni retriev document.

Die einfachsten IR-Modelle betrachten Dokumente als Mengen von Terms, so dass die zugehörige Repräsentation eines Dokumentes wie folgt aussieht:

algorithm analys best consistent document experiment index method perform produc retriev techni.

Wir nehmen nun an, dass wir ein Dokument durch einen Beschreibungsvektor $\vec{x} = (x_1, \ldots, x_n)$ repräsentieren, wobei die Komponente x_i jeweils das Vorkommen des Terms $t_i \in T = \{t_1, \ldots, t_n\}$ in dem aktuellen Dokument beschreibt.

Im Falle einer **Term-Menge** sind die Vektor-Komponenten binär, also $x_i = 1$, falls t_i im Dokument vorkommt, und $x_i = 0$ sonst.

Als eine Verbesserung dieser Repräsentationsform kann man die Vorkommenshäufigkeit des Terms im Dokument berücksichtigen. Somit haben wir jetzt eine **Multi-Menge von Terms**, repräsentiert durch $x_i \in \{0, 1, 2, \ldots\}$.

Die semantische Sicht auf Texte besteht hier also aus dieser Multimenge von Terms. Die eigentliche Semantik (z.B. die Unterscheidung zwischen wichtigen und unwichtigen Wörtern) kommt jedoch durch das auf diese Sicht aufbauende Retrievalmodell zustande, und zwar bei der Abbildung auf die Objektattribute mit Hilfe von statistischen Verfahren!

Kapitel 4

Nicht-probabilistische IR-Modelle

4.1 Notationen

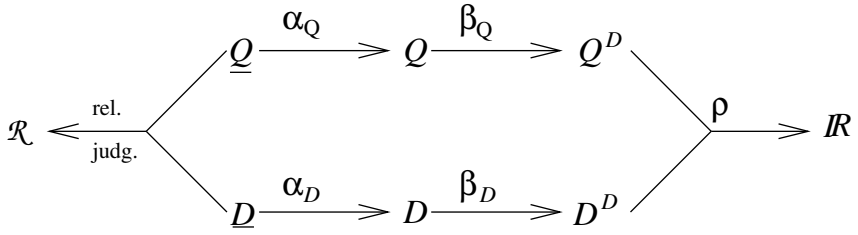

Abbildung 4.1: Konzeptionelles Modell für Textretrieval

Als grundlegendes konzeptionelles Modell für alle Arten von Modellen für (Text-)Retrieval verwenden wir das in Abb. 4.1 dargestellte Modell, das eine Vereinfachung des in Abschnitt 1.4 vorgestellten allgemeinen Modells ist. Dabei steht $\underline{D}$ für die Menge der Dokumente in der Datenbasis und $\underline{Q}$ für die Menge der Anfragen an das IRS. Zwischen den Dokumenten und den Anfragen besteht die Relevanzbeziehung, die hier als Abbildung in die Menge $\mathcal{R}$ der möglichen Relevanzurteile aufgefasst wird. Die in dem IRS repräsentierte semantische Sicht von Dokumenten bezeichnen wir im folgenden einfach als Dokumentrepräsentationen D und die formalisierten Anfragen als Frage-Repräsentationen Q. Diese entstehen aus den ursprünglichen Objekten durch die Abbildungen α_D und α_Q. Eine Dokumentrepräsentation kann z.B. eine Menge von Terms mit zugehörigen Vorkommenshäufigkeiten sein, eine Frage-Repräsentation ein boolescher Ausdruck mit Terms als Operanden.

Die Repräsentationen werden für die Zwecke des Retrievals in Dokumentbeschreibungen (Objektattribute) D^D und Fragebeschreibungen (logische Frageformulierung) Q^D überführt. Die Retrievalfunktion ϱ vergleicht für Frage-Dokument-Paare diese Beschreibungen und berechnet daraus das Retrievalgewicht, das i.a. eine reelle Zahl ist. Die Erstellung der Beschreibungen aus den Repräsentationen und die (mehr oder weniger begründete) Definition einer Retrievalfunktion hängt von dem jeweils zugrundegelegten Retrievalmodell ab. In diesem und dem folgenden Kapitel werden verschiedene solcher Retrievalmodelle beschrieben, die nicht nur in der Retrievalfunktion, sondern auch schon bzgl. der zugrundegelegten Repräsentationen und den daraus abgeleiteten Beschreibungen, differieren.

Nachstehend verwenden wir außerdem folgende Abkürzungen:

$T = \{t_1, \ldots, t_n\}$: Indexierungsvokabular
$\underline{q}_k$: Frage
q_k: Frage-Repräsentation (*formalisierte Anfrage*)
q_k^D: Frage-Beschreibung (*Fragelogik*)
$\underline{d}_m$: Dokument
d_m: Dokument-Repräsentation (*semantische Sicht*)
d_m^D: Dokument-Beschreibung (*Objektattribute*)

$\vec{d_m} = \{d_{m_1}, \ldots, d_{m_n}\}$: Dokument-Beschreibung als Menge von Indexierungsgewichten.

4.2 Überblick über die Modelle

	Boolesch	Fuzzy	Vektor	Probabilistisch	Sprachmodelle
theoretische Basis	Boolesche Logik	Fuzzy-Logik	Vektorraum-Modell	Wahrscheinlichkeits-Theorie	Statistische Sprachmodelle
Bezug zur Retrievalqualität		(x)		x	(x)
gewichtete Indexierung		x	x	x	x
gewichtete Frageterme		(x)	x	x	x
Fragestruktur:					
– linear			x	x	x
– boolesch	x	x	(x)	(x)	

Abbildung 4.2: IR-Modelle

In diesem und dem folgenden Kapitel behandeln wir die wichtigsten IR-Modelle: Boolesches und Fuzzy Retrieval, das Vektorraummodell, das probabilistische (Relevanz-orientierte) Modell sowie das statistische Sprachmodell. Abbildung 4.2 gibt eine Einordnung der hier und im folgenden Kapitel behandelten IR-Modelle. Eingeklammerte Markierungen bedeuten dabei, dass dieses Merkmal im Prinzip zutrifft, diese Variante des Modells allerdings hier nicht behandelt wird.

4.3 Boolesches Retrieval

Boolesches Retrieval ist historisch als erstes Retrievalmodell entwickelt und eingesetzt worden. Vermutlich hat Taube als erster dieses Modell zugrundegelegt, um Retrieval mit Hilfe von Schlitzlochkarten durchzuführen. Auch als man später die Dokumente auf Magnetbändern speicherte, war boolesches Retrieval das einzig anwendbare Modell: aufgrund der geringen Speicherkapazität damaliger Rechner musste direkt nach dem Einlesen des Dokumentes entschieden werden, ob es als Antwort ausgedruckt werden sollte oder nicht. Obwohl sich die Rechnerhardware seitdem rasant weiterentwickelt hat, hat man in der Praxis dieses Modell bis heute nicht grundlegend in Frage gestellt, sondern sich nur mit einigen funktionalen Erweiterungen begnügt.

Beim booleschen Retrieval sind die Dokumenten-Beschreibungen D^D ungewichtete Indexierungen, d.h.

$$d_m^D = \vec{d_m} \quad \text{mit} \quad d_{m_i} \epsilon \{0,1\} \quad \text{für} \quad i = 1, \ldots, n \tag{4.1}$$

Die Frage-Beschreibungen Q^D sind boolesche Ausdrücke, die nach folgenden Regeln gebildet werden:

1. $t_i \, \epsilon T \Rightarrow t_i \, \epsilon Q^D$
2. $q_1, q_2 \, \epsilon \, Q^D \Rightarrow q_1 \wedge q_2 \, \epsilon \, Q^D$
3. $q_1, q_2 \, \epsilon \, Q^D \Rightarrow q_1 \vee q_2 \, \epsilon \, Q^D$
4. $q \epsilon Q^D \Rightarrow \neg q \, \epsilon \, Q^D$

Die Retrievalfunktion ϱ kann man analog zu diesen Regeln ebenso rekursiv definieren:

1. $t_i \, \epsilon T \Rightarrow \varrho(t_i, \vec{d_m}) = d_{m_i}$
2. $\varrho(q_1 \wedge q_2, \vec{d_m}) = \min(\varrho(q_1, \vec{d_m}), \varrho(q_2, \vec{d_m}))$
3. $\varrho(q_1 \vee q_2, \vec{d_m}) = \max(\varrho(q_1, \vec{d_m}), \varrho(q_2, \vec{d_m}))$
4. $\varrho(\neg q, \vec{d_m}) = 1 - \varrho(q, \vec{d_m})$

Aufgrund der binären Gewichtung der Terme in der Dokumentbeschreibung kann die Retrievalfunktion ebenfalls nur die Retrievalgewichte 0 und 1 liefern. Daraus resultiert als Antwort auf eine Anfrage eine Zweiteilung der Dokumente der Datenbasis in gefundene ($\varrho = 1$) und nicht gefundene ($\varrho = 0$) Dokumente.

In realen IR-Systemen ist boolesches Retrieval meist nur in einer etwas modifizierten Form implementiert: Gegenüber der Darstellung hier ist die Verwendung der Negation derart eingeschränkt, dass diese nur in Kombination mit der Konjunktion verwendet werden darf, also z.B. in der Form $a \wedge \neg b$; eine Anfrage der Form $\neg b$ oder $a \vee \neg b$ ist hingegen nicht zulässig. Die Gründe für diese Einschränkung sind implementierungstechnischer Art.

4.3.1 Mächtigkeit der booleschen Anfragesprache

Ein wesentlicher (theoretischer) Vorteil der booleschen Anfragesprache besteht in ihrer Mächtigkeit. Man kann zeigen, dass mit einer booleschen Anfrage jede beliebige Teilmenge von Dokumenten aus einer Datenbasis selektiert werden kann. Voraussetzung ist dabei, dass alle Dokumente unterschiedliche Indexierungen (Beschreibungen) besitzen.

Zu einer vorgegebenen Dokumentenmenge $\underline{D}_k \subseteq \underline{D}$ konstruiert man dann die Frageformulierung q_k, die genau diese Dokumente selektiert, wie folgt: Zunächst wird für jedes Dokument eine Frage d_m^Q konstruiert, die nur dieses Dokument selektiert; anschließend werden diese Teilfragen für alle Dokumente $\underline{d}_m \in \underline{D}_k$ disjunktiv miteinander verknüpft.

$$
\begin{aligned}
d_m^Q &= x_{m_1} \wedge \ldots \wedge x_{m_n} \quad \text{mit} \\
x_{m_i} &= \begin{cases} t_i & \text{falls } d_{m_i} = 1 \\ \neg t_i & \text{sonst} \end{cases} \\
q_k &= \bigvee_{\underline{d}_j \epsilon \underline{D}_k} d_j^Q
\end{aligned}
$$

Dieser theoretische Vorteil ist aber (im Gegensatz zu Datenbanksystemen) von geringer praktischer Bedeutung; da ein Benutzer in der Regel nicht genau weiß, wie die zu seiner Frage relevanten Dokumente aussehen, kann er auch die Anfrage nicht entsprechend der hier skizzierten Vorgehensweise formulieren.

4.3.2 Nachteile des booleschen Retrieval

In der IR-Forschung ist man sich seit langem darüber einig, dass das boolesche Modell ziemlich ungeeignet für die Anwendung im IR ist [Verhoeff et al. 61]. In [Salton et al. 83] werden folgende Nachteile für boolesches Retrieval genannt:

1. Die Größe der Antwortmenge ist schwierig zu kontrollieren.
2. Es erfolgt keine Ordnung der Antwortmenge nach mehr oder weniger relevanten Dokumenten.
3. Es gibt keine Möglichkeit zur Gewichtung von Fragetermen oder zur Berücksichtigung von gewichteter Indexierung.
4. Die Trennung in gefundene und nicht gefundene Dokumente ist oftmals zu streng:
   Zu $q = t_1 \wedge t_2 \wedge t_3$ werden Dokumente mit zwei gefundenen Termen genauso zurückgewiesen wie solche mit 0 gefundenen Termen.
   Analog erfolgt für $q = t_1 \vee t_2 \vee t_3$ keine Unterteilung der gefundenen Dokumente
5. Die Erstellung der Frageformulierung ist sehr umständlich und überfordert daher gelegentliche Benutzer.
6. Die Retrievalqualität von booleschem Retrieval ist wesentlich schlechter als die von anderen Retrievalmodellen (s. nächster Abschnitt).

Trotz dieser Nachteile wird das boolesche Retrieval heute immer noch in bestimmten Bereichen wie z.B. dem Patentretrieval eingesetzt, wo erfahrene Rechercheure davon überzeugt sind, dadurch eine bessere Kontrolle über die vom System gelieferten Antworten zu haben. Ferner spielen boolesche Anfragen bei Rechtsstreitigkeiten (z.B. in den USA) eine Rolle, wo eine beklagte Firma genau jene Dokumente herausgeben muss, die eine zuvor ausgehandelte boolesche Frageformulierung erfüllen.

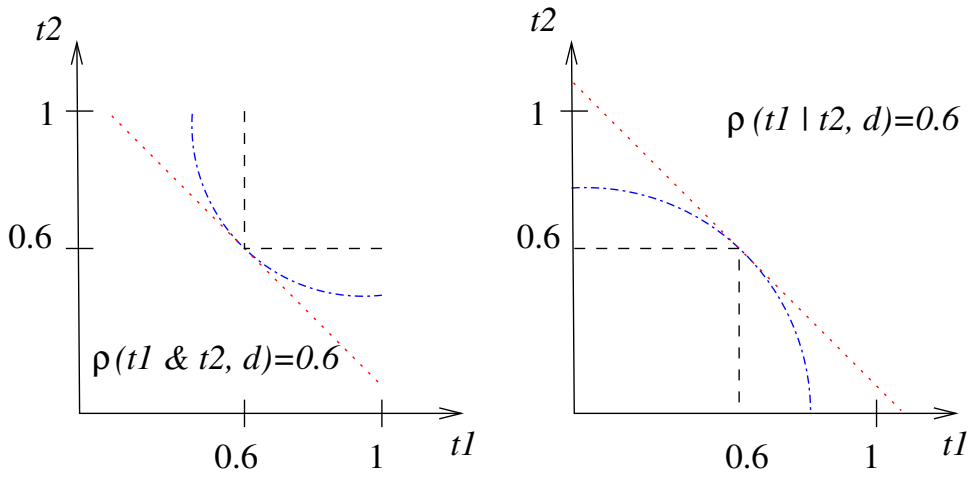

Abbildung 4.3: Punkte mit gleichem Retrievalgewicht beim Fuzzy-Retrieval und Varianten

4.4 Fuzzy-Retrieval

Als ein Ansatz, um einige der Nachteile von booleschem Retrieval zu überwinden, wurde basierend auf der Theorie der Fuzzy-Logik [Zadeh 65] Fuzzy-Retrieval vorgeschlagen. Im Unterschied zum booleschen Modell werden hier bei den Dokumenten-Beschreibungen nun auch gewichtete Indexierungen zugelassen, d.h. $d_{m_i} \epsilon [0,1]$. Frage-Beschreibungen und Retrievalfunktion sind wie beim booleschen Retrieval definiert.

Durch die gewichtete Indexierung liefert die Retrievalfunktion jetzt Werte $\varrho(q_k^D, \vec{d}_m) \epsilon [0,1]$. Damit ergibt sich im Gegensatz zum booleschen Modell nun eine Rangordnung der Antwortdokumente, und die diesbezüglichen Nachteile des booleschen Retrievals entfallen. Theoretische Überlegungen wie auch experimentelle Untersuchungen zeigen aber, dass die Definition der Retrievalfunktion ungünstig ist. Wir illustrieren dies zunächst an einem Beispiel:

$$
\begin{aligned}
T &= \{t_1, t_2\} \\
q &= t_1 \wedge t_2 \\
\vec{d}_1 = (0.6, 0.6) \quad &, \quad \vec{d}_2 = (0.59, 1.00) \\
\varrho(q, \vec{d}_1) = 0.6 \quad &, \quad \varrho(q, \vec{d}_2) = 0.59
\end{aligned}
$$

Obwohl hier d_2 bezüglich t_2 ein deutlich höheres Indexierungsgewicht als d_1 hat, gibt das um 0.01 niedrigere Gewicht bzgl. t_1 den Ausschlag für das insgesamt höhere Retrievalgewicht von d_1. Der Grund hierfür ist die Verwendung der Minimum-Funktion bei der konjunktiven Verknüpfung. In der Abb. 4.4 ist jeweils für Konjunktion und Disjunktion die Menge aller Paare von Gewichten (d_{m_1}, d_{m_2}) markiert, für die sich ein Retrievalgewicht von 0.6 ergibt (schwarz gestrichelte, rechtwinklige Linien). Offensichtlich wäre es wünschenswert, wenn man zumindest eine teilweise Kompensation der Gewichte für die verschiedenen Terme aus der Anfrage zulassen würde, wie dies die anderen beiden Kurven andeuten. In [Lee et al. 93] werden die hierzu aus der Fuzzy-Theorie bekannten T-Normen sowie eigene Erweiterungsvorschläge evaluiert; dabei zeigt sich dass die hier vortgestellte Standarddefinition der Fuzzy-Operatore relative schlecht abschneidet. Ein alternatives Modell ist unter dem Namen “Extended Boolean Retrieval” in [Salton et al. 83] beschrieben worden.

In der gleichen Veröffentlichung werden auch experimentelle Ergebnisse zum Vergleich von booleschen und Fuzzy-Retrieval mit dem Vektorraummodell präsentiert. Tabelle 4.1 zeigt diese Ergebnisse in Form mittlerer Precision-Werte (für die Recall-Punkte 0.25, 0.5 und 0.75)[1].

4.4.1 Beurteilung des Fuzzy-Retrieval

Zusammengefasst bietet Fuzzy-Retrieval folgende Vor- und Nachteile:

[1]Das teilweise schlechtere Abschneiden von Fuzzy- gegenüber booleschem Retrieval ist dabei wohl auf die verwendete Evaluierungsmethode zurückzuführen, die für mehrere Dokumente im gleichen Rang ungeeignet ist.

+ Durch Generalisierung des booleschen Retrieval für gewichtete Indexierung ergibt sich eine Rangordnung der Dokumente.
– Der Ansatz erlaubt zunächst keine Fragetermgewichtung. Es wurden zwar einige Vorschläge hierzu gemacht (siehe den Überblick in [Bookstein 85]), die aber allesamt wenig überzeugen; zudem wurde keiner dieser Ansätze evaluiert. Den besten Vorschlag zur Behandlung dieser Problematik stellt das oben erwähnte "Extended Boolean Retrieval" dar.
– Die Retrievalqualität ist immer noch schlecht im Vergleich z.B. zum Vektorraummodell.
– Da die Frageformulierungen die gleichen wie beim booleschen Retrieval sind, bleibt der Nachteil der umständlichen Formulierung bestehen.

4.5 Das Vektorraummodell

Das Vektorraummodell (VRM) ist wahrscheinlich das bekannteste Modell aus der IR-Forschung. Es wurde ursprünglich im Rahmen der Arbeiten am SMART-Projekt entwickelt [Salton 71]. SMART ist ein experimentelles Retrievalsystem, das von Gerard Salton und seinen Mitarbeitern seit 1961 zunächst in Harvard und später in Cornell entwickelt wurde. In den 80er Jahren wurde das Modell nochmals von Wong und Raghavan überarbeitet [Raghavan & Wong 86].

Im VRM werden Dokumente und Fragen (bzw. deren Beschreibungen) als Punkte in einem Vektorraum aufgefasst, der durch die Terme der Datenbasis aufgespannt wird. Beim Retrieval wird dann nach solchen Dokumenten gesucht, deren Vektoren ähnlich (im Sinne einer vorgegebenen Metrik) zum Fragevektor sind. Durch diese geometrische Interpretation ergibt sich ein sehr anschauliches Modell.

Der zugrundeliegende Vektorraum wird als orthonormal angenommen, d.h.
- alle Term-Vektoren sind orthogonal (und damit auch linear unabhängig), und
- alle Term-Vektoren sind normiert.

Diese Annahmen stellen natürlich eine starke Vereinfachung gegenüber den realen Verhältnissen dar. (In [Wong et al. 87] wird alternativ hierzu versucht, explizit einen solchen orthonormalen Vektorraum zu konstruieren, dessen Dimensionalität deutlich niedriger als $|T|$ ist.)

Die im VRM zugrundegelegte Dokument-Beschreibung ist ähnlich der des Fuzzy-Retrieval eine gewichtete Indexierung; allerdings sind hier neben Gewichten größer als 1 prinzipiell auch negative Gewichte zulässig (obwohl negative Gewichte in SMART nie verwendet werden):

$$d_m^D = \vec{d}_m \quad \text{mit} \quad d_{m_i} \epsilon \mathbb{R} \quad \text{für } i = 1, \ldots, n \tag{4.2}$$

Die Frage-Beschreibungen haben die gleiche Struktur wie die Dokument-Beschreibungen:

$$q_k^Q = \vec{q}_k \quad \text{mit} \quad q_{k_i} \epsilon \mathbb{R} \quad \text{für } i = 1, \ldots, n \tag{4.3}$$

Als Retrievalfunktion werden verschiedene Vektor-Ähnlichkeitsmaße (z.B. das Kosinus-Maß) angewendet. Meistens wird mit dem Skalarprodukt gearbeitet:

$$\varrho(\vec{q}_k, \vec{d}_m) = \vec{q}_k \cdot \vec{d}_m \tag{4.4}$$

Das folgende Beispiel illustriert die Anwendung des VRM:
"retrieval experiments with weighted indexing"
Entsprechend den Retrievalgewichten werden die Dokumente in der Reihenfolge $d_4, d_3, (d_1, d_2)$ ausgegeben.

Kollektion	MEDLARS	ISI	INSPEC	CACM
#Dok.	1033	1460	12684	3204
#Fragen	30	35	77	52
Bool.	0.2065	0.1118	0.1159	0.1789
Fuzzy	0.2368	0.1000	0.1314	0.1551
Vektor	0.5473	0.1569	0.2325	0.3027

Tabelle 4.1: Mittlere Precision für Boolesches Retrieval, Fuzzy-Retrieval und Vektorraummodel

term	q_{k_i}	d_{1_i}	d_{2_i}	d_{3_i}	d_{4_i}
retrieval	1	0.33	0.33	0.25	0.25
experiment	1	0.33	0.33	0.25	0.25
weight	1				0.25
index	1			0.25	0.25
XML		0.33			
method			0.33		
binary				0.25	
RSV		0.66	0.66	0.75	1.00

4.5.1 Coordination Level Match

Eine vereinfachte Variante des Vektorraummodells ist der Coordination Level Match. Dabei sind sowohl für Frage- als auch für Dokumenttermgewichtung nur die binären Werte 0 und 1 zugelassen. Die *Dokument-Beschreibung* ist somit die gleiche wie beim Booleschen Retrieval:

$$d_m^D = \vec{d}_m \quad \text{mit} \quad d_{m_i} \varepsilon \{0,1\} \quad \text{für} \quad i = 1, \ldots, n.$$

Die *Frage-Beschreibung* ist ebenfalls ein binärer Vektor:

$$q_k^Q = \vec{q}_k \quad \text{mit} \quad q_{k_i} \varepsilon \{0,1\} \quad \text{für} \quad i = 1, \ldots, n.$$

Als *Retrievalfunktion* verwendet man meist das Skalarprodukt; dadurch zählt die Retrievalfunktion die Anzahl der Frageterme, die im jeweiligen Dokument vorkommen:

$$\varrho(\vec{q}_k, \vec{d}_m) = \vec{q}_k \cdot \vec{d}_m = |q_k^T \cap d_m^T|$$

4.5.2 Dokumentindexierung

Das VRM macht keine Aussagen darüber, wie die Dokumentenbeschreibung zu erstellen ist. Bei den Arbeiten am SMART-Projekt wurden heuristische Formeln zur Berechnung der Indexierungsgewichte für Dokumente (und Fragen) entwickelt, die sich als besonders leistungsfähig erwiesen haben. Diese Formeln wurden später im Rahmen der Arbeiten zu den experimentellen Systemen Inquery (U. Massachusetts / Bruce Croft) und OKAPI (MS Research Lab Cambridge / Stephen Robertson) weiterentwickelt. Wir stellen hier eine relativ neue Variante der Gewichtungsformel vor.

Die der Indexierung zugrundeliegende Dokumenten-Repräsentation ist eine Multi-Menge (Bag) von Terms. Darauf aufbauend werden zunächst folgende Parameter definiert:

- d_m^T Menge der in d_m vorkommenden Terms
- l_m Dokumentlänge (# Anzahl laufende Wörter in d_m)
- al durchschnittliche Dokumentlänge in $\underline{D}$
- tf_{mi}: Vorkommenshäufigkeit (Vkh) von t_i in d_m.
- n_i: # Dokumente, in denen t_i vorkommt.
- N: # Dokumente in der Kollektion

Eine Komponente der Gewichtung ist die inverse Dokumenthäufigkeit idf_i, die umso höher ist, je seltener ein Term in der Kollektion vorkommt:

$$idf_i = \frac{\log \frac{N}{n_i}}{N+1} \tag{4.5}$$

Die zweite Komponente ist die normalisierte Vorkommenshäufigkeit ntf_i. Hierbei sollen die Terms entsprechend ihrer Vorkommenshäufigkeit im Dokument gewichtet werden. Um den Einfluss der Dokumentlänge auszugleichen, geht diese ebenfalls mit ein, und zwar als Verhältnis zur durchschnittlichen Dokumentlänge in der Kollektion:

$$ntf_i = \frac{tf_{mi}}{tf_{mi} + 0.5 + 1.5\frac{l_m}{al}} \tag{4.6}$$

Das endgültige Indexierungsgewicht ergibt sich als Produkt der beiden Komponenten und wird daher meist als tfidf-Gewichtung bezeichnet:

$$w_{mi} = ntf_i \cdot idf_i \tag{4.7}$$

Kollektion	CACM	CISI	CRAN	INSPEC	MED
Coord.	0.185	0.103	0.241	0.094	0.413
SMART	0.363	0.219	0.384	0.263	0.562

Tabelle 4.2: Mittlere Precision für binäre Gewichte vs. SMART-Gewichtung

.

Tabelle 4.2 zeigt einige experimentelle Ergebnisse (aus [Salton & Buckley 88] mit einer früheren Version der tfidf-Formel aus dem SMART-Projekt) zu dieser Art der Gewichtung im Vergleich zu einer rein binären Gewichtung (Coordination Level Match). Dabei wurden die Gewichtungsformeln 4.5–4.7 sowohl zur Dokumentindexierung als auch zur Bestimmung des Fragevektors angewendet.

4.5.3 Relevance Feedback

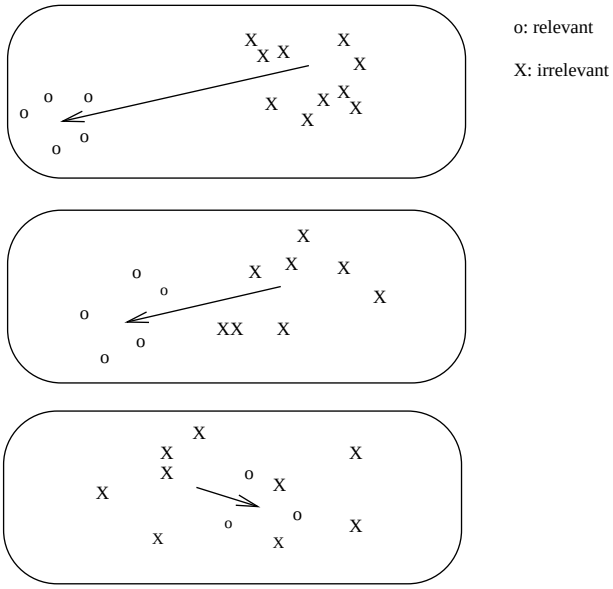

Abbildung 4.4: Beispiele zur Trennung von relevanten und nichtrelevanten Dokumenten im VRM

Ein wesentlicher Vorteil des VRM insbesondere auch gegenüber Fuzzy-Retrieval ist die Möglichkeit, Relevance-Feedback-Daten zur Verbesserung der Retrievalqualität auszunutzen. Dabei wird versucht, Angaben über die Relevanz bzw. Nicht-Relevanz einiger Dokumente zur Modifikation des ursprünglichen Fragevektors zu verwenden. Genauer gesagt, werden die ursprünglichen Fragetermgewichte verändert, wodurch sich ein anderer Fragevektor ergibt. Abb. 4.4 illustriert verschiedene mögliche Verteilungen von relevanten und nichtrelevanten Dokumenten im Vektorraum. Außerdem ist jeweils der Vektor eingezeichnet, der vom Zentroiden der nichtrelevanten Dokumente zum Zentroiden der relevanten Dokumente führt. Dieser Vektor eignet sich offensichtlich als Fragevektor, um relevante und nichtrelevante Dokumente möglichst gut zu trennen. Nimmt man nämlich das Skalarprodukt als Retrievalfunktion an, dann werden die Dokumente auf eine Gerade entlang des Fragevektors projiziert, wobei der Vektor die Richtung höherer Retrievalgewichte anzeigt.

In [Rocchio 66] wird eine optimale Lösung für die Bestimmung eines Fragevektors aus Relevance-Feedback-Daten vorgestellt. Die Grundidee ist dabei die, einen Fragevektor $\vec{q}$ zu bestimmen, der die Differenz der RSVs zwischen relevanten und irrelevanten Dokumenten maximiert. Sei D^R die Menge der relevanten Dokumente zu q und D^N die Menge der nichtrelevanten Dokumente zu q, dann lautet das

Optimierungskriterium:

$$\sum_{(d_k,d_l)\in D^R\times D^N} \vec{q}\vec{d_k} - \vec{q}\vec{d_l} \stackrel{!}{=} \max \tag{4.8}$$

Zusätzlich muss man noch als Nebenbedingung den Betrag des Fragevektors beschränken:

$$\sum_{i=1}^{n} q_i^2 = c \tag{4.9}$$

Somit liegt ein Extremwertproblem mit Randbedingung vor, das man mit Hilfe eines Lagrange-Multiplikators lösen kann:

$$F = \lambda\left(\sum_{i=1}^{n} q_i^2 - c\right) + \sum_{(d_k,d_l)\in D^R\times D^N} \sum_{i=1}^{n} q_i d_{k_i} - q_i d_{l_i} \tag{4.10}$$

Zur Lösung muss man nun alle partiellen Ableitungen von F nach den Komponenten q_i des Fragevektors 0 setzen; zusätzlich muss auch die Nebenbedingung 4.9 gelten.

$$\begin{aligned}
\frac{\partial F}{\partial q_i} &= 2\lambda q_i + \sum_{(d_k,d_l)\in D^R\times D^N} d_{k_i} - d_{l_i} \stackrel{!}{=} 0 \\
q_i &= -\frac{1}{2\lambda} \sum_{(d_k,d_l)\in D^R\times D^N} d_{k_i} - d_{l_i} \\
\vec{q} &= -\frac{1}{2\lambda} \sum_{(d_k,d_l)\in D^R\times D^N} \vec{d_k} - \vec{d_l} \\
&= -\frac{1}{2\lambda}|D^N| \sum_{d_k\in D^R} \vec{d_k} - |D^R| \sum_{d_l\in D^N} \vec{d_l} \\
&= -\frac{|D^N||D^R|}{2\lambda} \frac{1}{|D^R|} \sum_{d_k\in D^R} \vec{d_k} - \frac{1}{|D^N|} \sum_{d_l\in D^N} \vec{d_l}
\end{aligned}$$

Zur Vereinfachung wählen wir c (den Betrag des Fragevektors) so, dass $|D^N||D^R|/2\lambda = -1$. Damit ergibt sich der optimale Fragevektor zu

$$\vec{q} = \frac{1}{|D^R|} \sum_{d_k\in D^R} \vec{d_k} - \frac{1}{|D^N|} \sum_{d_l\in D^N} \vec{d_l} \tag{4.11}$$

Der optimale Fragevektor ist somit der Verbindungsvektor der beiden Zentroiden der relevanten bzw. irrelevanten Dokumente.

Abbildung 4.5 illustriert diese Lösung. Gleichzeitig wird deutlich, dass der optimale Fragevektor nicht immer die bestmögliche Lösung (bezogen auf die Retrievalqualität) darstellt. (Ein wesentlich besseres, allerdings auch aufwändigeres Verfahren ist die Support Vector Machine [Joachims 01].) Als heuristische Verbesserung, die sich in zahlreichen Experimenten bewährt hat, hat Rocchio vorgeschlagen, relevante und irrelevante Dokumente unterschiedlich stark zu gewichten, konkret: den Vektor zum Zentroiden der irrelevanten Dokumente weniger stark in die Lösung einfließen zu lassen. Abbildung 4.5.3 verdeutlicht diese Vorgehensweise für unser Beispiel. Intuitiv kann man sich diese Verbesserung dadurch erklären, dass in der Regel die relevanten Dokumente höhere Indexierungsgewichte als die irrelevanten aufweisen, so dass diese Modifikation den Fragevektor in die richtige Richtung „dreht“.

Weitere Experimente haben gezeigt, dass man den neuen Fragevektor nie allein aus den Relevance-Feedback-Daten ohne Berücksichtigung des ursprünglichen Vektors bilden sollte; Es gibt ja noch weitere Dokumente, über die noch keine Relevanzinformation verfügbar ist, weil das System diese dem Benutzer noch nicht vorgelegt hat. Gerade diese Dokumente sollen aber möglichst gut in relevante und nichtrelevante aufgeteilt werden – das ist ja die eigentliche Aufgabe beim Retrieval. Also geht es darum, den ursprünglichen Vektor mit Hilfe der Relevance-Feedback-Daten zu verbessern. Prinzipiell ergibt sich also folgende Vorgehensweise:

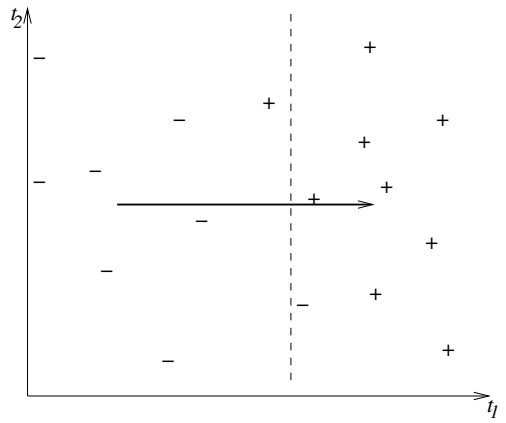

Abbildung 4.5: Optimaler Fragevektor als Verbindungsvektor der Zentroiden

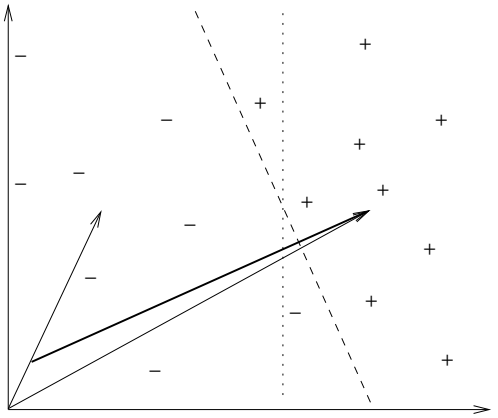

Abbildung 4.6: Unterschiedliche Gewichtung positiver und negativer Beispiele

1. Retrieval mit dem Fragevektor $\vec{q}_k$ vom Benutzer.
2. Relevanzbeurteilung der obersten Dokumente der Rangordnung.
3. Berechnung eines verbesserten Fragevektors $\vec{q}'$ aufgrund der Feedback-Daten.
4. Retrieval mit dem verbesserten Vektor.
5. Evtl. Wiederholung der Schritte 2-4.

Als Iterationsvorschrift zur Berechnung eines verbesserten Fragevektors $\vec{q}'$ wird in [Rocchio 66] folgende Kombination aus ursprünglichem Vektor $\vec{q}$ und den Zentroiden der relevanten und der nichtrelevanten Dokumente vorgeschlagen:

$$\vec{q}' = \vec{q} + \alpha \frac{1}{|D^R|} \sum_{d_j \epsilon D^R} \vec{d}_j - \beta \frac{1}{|D^N|} \sum_{d_j \epsilon D^N} \vec{d}_j \tag{4.12}$$

Dabei sind α und β nichtnegative Konstanten, die heuristisch festzulegen sind (z.B. $\alpha = 0.75$, $\beta = 0.25$).

Kollektion	CACM	CISI	CRAN	INSPEC	MED
ohne RF	0.1459	0.1184	0.1156	0.1368	0.3346
Feedback	0.2552	0.1404	0.2955	0.1821	0.5630
Feedback*	0.2491	0.1623	0.2534	0.1861	0.5279

Tabelle 4.3: Experimentelle Ergebnisse zu Relevance Feedback (mittlere Precision)

Tabelle 4.3 zeigt experimentelle Ergebnisse, die durch Anwendung der Formel 4.12 gewonnen wurden (aus [Salton & Buckley 90]). Hier wurde Feedback-Information von den obersten 15 Dokumenten des

Retrievallaufs mit dem initialen Fragevektor verwendet. Zur Bewertung wurde die sogenannte "residual collection"-Methode angewendet: dabei bleiben die Dokumente, deren Feedback-Daten benutzt wurden, bei der Bewertung unberücksichtigt. Dadurch ergibt sich ein fairer Vergleich mit der Retrievalfunktion ohne Relevance Feedback. Die Ergebnisse zeigen hier sehr deutliche Verbesserungen durch die Relevance-Feedback-Methode. Die letzte Tabellenzeile (Feedback*) zeigt die Ergebnisse für eine modifizierte Anwendung der obigen Formel, bei der nur die häufigsten Terme zur Frageerweiterung benutzt werden, d.h., bei den Termen, deren Fragetermgewicht ursprünglich 0 war (weil sie in der Fragerepräsentation nicht vorkamen), wird die Formel nicht generell in der beschriebenen Weise angewandt; es werden nur die n häufigsten Terme in der vorgeschriebenen Weise berücksichtigt, die übrigen Terme behalten das Gewicht 0. Es zeigt sich, dass diese Methode bei einigen Kollektionen noch zu besseren Ergebnissen führt, während bei anderen Kollektionen schlechtere Ergebnisse produziert werden.

Auch wenn die Formel 4.12 erwiesenermaßen gute Ergebnisse liefert, so sind die heuristischen Komponenten in diesem Ansatz doch unbefriedigend. Letzten Endes liegt die grundlegende Schwäche des VRM in dem fehlenden Bezug zur Retrievalqualität. Auch die o.g. Optimierungsbedingung 4.8 nimmt nicht auf die Retrievalqualität Bezug, und man kann zeigen, dass es tatsächlich in manchen Fällen bessere Vektoren zur Trennung in relevante und nichtrelevante Dokumente gibt, als sie durch diese Bedingung geliefert werden (näheres siehe Übung).

4.5.4 Beurteilung des VRM

Zusammenfassend ergeben sich folgende Vor- und Nachteile für das VRM:

+ Das VRM ist ein relativ einfaches, anschauliches Modell, das insbesondere wegen der einfachen Art der Frageformulierung auch benutzerfreundlich ist.
+ Das Modell ist unmittelbar auf neue Kollektionen anwendbar; probabilistische Modelle erfordern dagegen teilweise zuerst das Sammeln von Relevance-Feedback-Daten für eine Menge von Fragen, bevor sie sinnvoll eingesetzt werden können.
+ Das Modell liefert in Kombination mit den SMART-Gewichtungsformeln eine sehr gute Retrievalqualität.
– Leider enthält das Modell, so wie es letztendlich angewendet wird, sehr viele heuristische Komponenten; dabei stellt sich insbesondere die Frage, inwieweit diese Heuristiken auch noch beim Übergang auf wesentlich andere Kollektionen (z.B. Volltexte statt Kurzfassungen) gültig bleiben.
– Der heuristische Ansatz zur Berechnung der Indexierungsgewichte hat zur Folge, dass die Dokumentrepräsentation nur schlecht erweitert werden kann. Wenn man z.B. Terms aus dem Titel stärker gewichten möchte als solche, die nur im Abstract vorkommen, dann müssen hierfür erst umfangreiche Experimente durchgeführt werden, um eine geeignete Gewichtungsformel zu finden.
– In dem Modell wird keinerlei Bezug auf die Retrievalqualität genommen; es ist theoretisch nicht zu begründen, warum die zu einer Frage ähnlichen Dokumente auch relevant sein sollen.

Kapitel 5

Probabilistische IR-Modelle

5.1 Einführung

Ein wesentlicher Unterschied zwischen IR-Systemen und vielen anderen klassischen Informationssystemen besteht in der intrinsischen Unsicherheit des IR. Während man etwa bei Datenbankanwendungen ein Informationsbedürfnis typischerweise eindeutig auf eine präzise Anfrage abbilden kann, wofür wiederum die Antwort eindeutig definiert ist, haben wir im IR eine viel schwierigere Situation: hier repräsentiert die Anfrageformulierung das Informationsbedürfnis nur approximativ, und zudem existiert keine eindeutige Vorschrift, wie die Antwort auf eine Anfrage definiert ist. (Boolesches Retrieval stellt keine Ausnahme zu dieser Aussage dar: hier werden nur die Probleme mit Unsicherheit und Vagheit dem Benutzer aufgebürdet.) Als der erfolgreichste Ansatz, um mit Unsicherheit im IR umzugehen, haben sich die probabilistischen Modelle erwiesen.

Wir betrachten in diesem Kapitel zwei Arten von probabilistischen Modellen: Zunächst gehen wir auf die älteren, relevanzorientierten Modelle ein, deren Vorläufer bis in die 1960er Jahre zurückreichen. Anschließend gehen wir auf die wesentlich jüngeren statistischen Sprachmodelle ein, die 1998 erstmals im IR auftauchten und seitdem sehr populär sind.

5.2 Das Binary-Independence-Retrieval-Modell

Als Vertreter der klassischen IR Modelle präsentieren wir hier das populärste dieser Modelle, das sogenannte BIR-Modell. Wir geben zunächst eine eher informelle Einführung, und gehen dann im nächsten Abschnitt auf die theoretischen Grundlagen ein.

5.2.1 Herleitung

Wie in anderen relevanzorientierten Modellen auch, versucht man im BIR-Modell die Wahrscheinlichkeit zu schätzen, dass ein gegebenes Dokument d_m bezüglich der aktuellen Anfrage q als relevant beurteilt wird. Um diese als $P(R|d_m)$ bezeichnete Wahrscheinlichkeit zu schätzen, betrachten wir die Verteilung der Terme in der Kollektion; dabei nehmen wir an, dass die Verteilung in den relevanten und den irrelevanten Dokumenten unterschiedlich ist. (Wie wir weiter unten sehen werden, sollte die Anfrage idealerweise aus solchen Termen bestehen, deren Verteilung in den relevanten und irrelevanten Dokumenten sich stark unterscheidet.) Bezeichne $T = \{t_1, \ldots, t_n\}$ wie üblich die Menge der in der Kollektion vorkommenden Terme. Dann können wir die Menge d_m^T der im Dokument d_m vorkommenden Terme als binären Vektor repräsentieren: $\vec{x} = (x_1, \ldots, x_n)$ mit $x_i = 1$, falls $t_i \epsilon d_m^T$ und $x_i = 0$ sonst.

Im Folgenden unterscheiden wir nur zwischen Dokumenten, die unterschiedliche Mengen von Termen beinhalten. Anstelle der Relevanzwahrscheinlichkeit $P(R|d_m)$ für ein spezifisches Dokument d_m schätzen wir dann die Wahrscheinlichkeit $P(R|\vec{x})$; somit wird für unterschiedliche Dokumente, die aber die gleiche Termmenge beinhalten, die gleiche Relevanzwahrscheinlichkeit berechnet. Weiterhin nehmen wir an, dass eine Anfrage q in Form einer Termmenge $q^T \subset T$ gegeben ist.

Um nun die gesuchte Relevanzwahrscheinlichkeit zu berechnen, wenden wir zwei Arten von Transformationen an, die häufig im Kontext probabilistischer IR-Modelle eingesetzt werden

1. Anwendung des Bayes'schen Theorems (in der Form $P(a|b) = P(b|a) \cdot P(a)/P(b)$),
2. Benutzung von Odds (Chancen) anstelle von Wahrscheinlichkeiten, wobei $O(y) = P(y)/P(\bar{y}) = P(y)/[1 - P(y)]$.

Damit können wir die Chancen berechnen, dass ein Dokument relevant zur Anfrage q ist, basierend auf seiner Beschreibung durch einen binären Vektor $\vec{x}$:

$$O(R|\vec{x}) = \frac{P(R|\vec{x})}{P(\bar{R}|\vec{x})} = \frac{P(R)}{P(\bar{R})} \cdot \frac{P(\vec{x}|R)}{P(\vec{x}|\bar{R})} \cdot \frac{P(\vec{x})}{P(\vec{x})} \tag{5.1}$$

Hier bezeichnet $P(R)$ die Wahrscheinlichkeit, dass ein zufälliges Dokument relevant ist – dies bezeichnet man auch als die Generality der Anfrage (und $P(\bar{R})$ ist die Gegenwahrscheinlichkeit hiervon); dieser Parameter ist offensichtlich konstant für alle Dokumente zu einer Frage. Wichtiger für das Ranking ist die Wahrscheinlichkeit $P(R|\vec{x})$, dass ein zufälliges relevantes Dokument die Beschreibung $\vec{x}$ besitzt (und $P(\bar{R}|\vec{x})$ das Entsprechende für die irrelevanten Dokumente). Da dieser Parameter in der vorliegenden Form kaum geschätzt werden kann, benötigen wir zusätzliche Unabhängigkeitsannahmen, um das Problem zu vereinfachen. Wie in [Cooper 91] gezeigt wurde, sind es genau genommen, keine Unabhängigkeitsannahmen (weshalb der Name des Modells nicht ganz korrekt ist), sondern dem BIR liegt tatsächlich eine verbundene Abhängigkeitsannahme zugrunde, die folgende Form hat:

$$\frac{P(\vec{x}|R)}{P(\vec{x}|\bar{R})} = \prod_{i=1}^{n} \frac{P(x_i|R)}{P(x_i|\bar{R})} \tag{5.2}$$

Würde man Zähler und Nenner getrennt betrachten, so hätte man zwei Unabhängigkeitsannahmen, die besagen, dass sowohl in den relevanten als auch den irrelevanten Dokumenten die Terme unabhängig voneinander verteilt sind, dass also die Wahrscheinlichkeit, einen bestimmten Vektor $\vec{x}$ zu beobachten, gleich dem Produkt der entsprechenden Wahrscheinlichkeiten für die einzelnen Terme ist. Die verbundene Abhängigkeitsannahme ist hingegen weniger streng, aber etwas komplizierter: Sie besagt, dass der Quotient der beiden Wahrscheinlichkeiten für das Vorkommen von $\vec{x}$ in relevanten bzw. irrelevanten Dokumenten, gleich dem Produkt der Quotienten für die einzelnen Terme ist. Natürlich ist auch die Annahme der verbundenen Abhängigkeit nur eine Annäherung an die Realität – die aber erstaunlich gut funktioniert.

Mit der Annahme (5.2) können wir Gleichung (5.1) überführen in

$$O(R|\vec{x}) \quad = \quad O(R) \prod_{i=1}^{n} \frac{P(x_i|R)}{P(x_i|\bar{R})}$$

Das Produkt kann nun aufgeteilt werden in ein erstes Produkt über alle im Dokument vorkommenden Terme, und ein zweites für die nicht vorkommenden Terme:

$$O(R|\vec{x}) = O(R) \prod_{x_i=1} \frac{P(x_i{=}1|R)}{P(x_i{=}1|\bar{R})} \cdot \prod_{x_i=0} \frac{P(x_i{=}0|R)}{P(x_i{=}0|\bar{R})}.$$

Im Folgenden bezeichne $p_i = P(x_i{=}1|R)$ die Wahrscheinlichkeit, dass der Term t_i in einem zufällig gewählten relevanten Dokument vorkommt, und $s_i = P(x_i{=}1|\bar{R})$ die entsprechende Wahrscheinlichkeit für die irrelevanten Dokumente. Zusätzlich nehmen wir an, dass $p_i = s_i$ gilt für alle Terme, die nicht in der Menge q^T der Frageterme vorkommt. Mit dieser Vereinfachung und den eingeführten Notationen erhalten wir dann

$$O(R|\vec{x}) \quad = \quad O(R) \prod_{t_i \epsilon d_m^T \cap q^T} \frac{p_i}{s_i} \prod_{t_i \epsilon q^T \setminus d_m^T} \frac{1-p_i}{1-s_i} \tag{5.3}$$

$$\quad = \quad O(R) \prod_{t_i \epsilon d_m^T \cap q^T} \frac{p_i(1-s_i)}{s_i(1-p_i)} \prod_{t_i \epsilon q^T} \frac{1-p_i}{1-s_i} \tag{5.4}$$

Bei der Anwendung dieser Formel ist man primär am Ranking der Dokumente zu der gegebenen Anfrage interessiert, während die tatsächliche Relevanzwahrscheinlichkeit eher nebensächlich ist. Wenn wir also nur eine Rangordnung erzeugen wollen, dann ist das zweite Produkt in Gleichung (5.4) ebenso wie der Wert von $O(R)$ konstant für alle Dokumente zu einer Anfrage. Daher können wir dieser beiden Faktoren ignorieren und brauchen nur noch das erste Produkt zu betrachten, um eine Rangordnung für die aktuelle Anfrage zu erzeugen. Zur Vereinfachung der Rechnung betrachten wir den Logarithmus dieses Produkts, so dass sich der Retrievalwert (RSV) des Dokumentes d_m für die Frage q berechnen lässt durch die Summe

$$\varrho_{BIR}(q, d_m) = \sum_{t_i \epsilon d_m^T \cap q^T} c_i \qquad \text{mit} \qquad c_i = \log \frac{p_i(1-s_i)}{s_i(1-p_i)}.$$

Dann werden die Dokumente nach fallenden Retrievalwerten geordnet.

5.2.2 Parameterschätzung

Zur Anwendung des BIR-Modells müssen die Parameter p_i und s_i für alle in der Frage vorkommenden Terme ($t_i \epsilon q^T$) geschätzt werden.

Wir betrachten zunächst den Parameter $s_i = P(x_i{=}1|\bar{R})$, also die Wahrscheinlichkeit, dass t_i in einem arbiträren nicht-relevanten Dokument vorkommt. Da in der Regel nur ein kleiner Bruchteil einer Kollektion relevant ist auf eine Anfrage, nehmen wir nun vereinfachend an, dass die Anzahl der nicht-relevanten Dokumente durch die Größe der Kollektion approximiert werden kann. Bezeichne N diesen Wert (Anzahl der Dokumente in der Kollektion) und n_i die Anzahl der Dokumente, in denen der Term t_i vorkommt, dann kann man s_i einfach durch die relative Häufigkeit n_i/N schätzen.

Der Parameter $p_i = P(x_i{=}1|R)$ bezeichnet die Wahrscheinlichkeit, dass t_i in einem arbiträren relevanten Dokument vorkommt. Zu seiner Schätzung benötigt man eigentlich Relevance-Feedback-Daten (s.u.). Durch vereinfachende Annahmen können wir aber auch ohne diese Information auskommen. Hierzu nehmen wir einen globalen Wert p für alle p_i an. Damit erhalten wir

$$\begin{aligned} c_i &= \log \frac{p}{1-p} + \log \frac{1-s_i}{s_i} \\ &= c_p + \log \frac{N-n_i}{n_i} \end{aligned}$$

Häufig wird $p = 0.5$ angenommen, so dass $c_p = 0$ wird. Damit erhält man dann die Termgewichtung nach inverser Dokumentenhäufigkeit (IDF) gemäß folgender Formel:

$$\varrho_{IDF}(q, d_m) = \sum_{t_i \in q^T \cap d_m^T} \log \frac{N-n_i}{n_i} \tag{5.5}$$

In der Regel lässt man als weitere Vereinfachung n_i im Zähler weg, so dass man die IDF-Gewichtung in der Form bekommt, wie wir sie schon beim Vektorraummodell kennengelernt haben. Während dort aber heuristisch vorgegangen wurde, haben wir hier jetzt eine theoretische Begründung für diese Art der Termgewichtung.

Nun kommen wir zu dem Fall, dass wir Relevance-Feedback-Daten haben. Dies könnte z.B. dadurch geschehen dass wir zuerst Retrieval mit der IDF-Formel durchführen, und dann den Benutzer die obersten Antwortdokumente beurteilen lassen. Bezeichne r die Anzahl der insgesamt vom Benutzer als relevant beurteilten Dokumente und r_i die Mächtigkeit der Teilmenge hiervon, in denen der Term t_i vorkommt, dann kann man $p_i \approx \frac{r_i}{r}$ durch die entsprechende relative Häufigkeit schätzen. Wegen der geringen Anzahl an Beobachtungen sind diese Werte aber systematisch falsch; eine bessere Schätzung liefert die Formel $p_i \approx \frac{r_i+0.5}{r+1}$.

5.2.3 Beispiel

Wir geben nun ein umfangreicheres Beispiel zum BIR-Modell. Hierzu nehmen wir an, dass wir eine Frage q mit nur zwei Termen haben, also $q^T = \{t_1, t_2\}$. Tabelle 5.1 zeigt die Relevanzurteile sowie die Verteilung der Terme in diesen Dokumenten.

d_i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
x_1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
x_2	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0
$r(q, d_i)$	R	R	R	R	$\bar{R}$	R	R	R	R	$\bar{R}$	$\bar{R}$	R	R	R	$\bar{R}$	$\bar{R}$	$\bar{R}$	R	$\bar{R}$	$\bar{R}$

Tabelle 5.1: Beispiel zum BIR-Modell

	$P(R\|q, \vec{x})$	
$\vec{x}$	BIR	actual
(1,1)	0.76	0.8
(1,0)	0.69	0.67
(0,1)	0.48	0.5
(0,0)	0.4	0.33

Tabelle 5.2: Geschätzte Relevanzwahrscheinlichkeiten für unser Beispiel

Für die Schätzung der BIR-Parameter verwenden wir hier nicht die o.g. Näherungsformeln, sondern berechnen die Wahrscheinlichkeiten gemäß Definition über die zugehörigen relativen Häufigkeiten. So kommt etwa t_1 in 8 von 12 relevanten Dokumenten vor, was $p_1 = 8/12 = 2/3$ ergibt, und in 3 von 8 irrelevanten Dokumenten, was $s_1 = 3/8$ liefert. Analog erhalten wir $p_2 = 7/12$ und $s_2 = 4/8$. Ferner haben wir $O(R) = 12/8$. Zur Schätzung der Relevanzwahrscheinlichkeit (bzw. -chancen) wenden wir Gleichung (5.3) an, was uns die in Tabelle 5.2 dargestellten Werte liefert. Hier zeigt sich, dass z.B. für die Beschreibung $\vec{x} = (1, 1)$ das BIR-Modell eine Relevanzwahrscheinlichkeit von rund 0.76 liefert, tatsächlich aber 4 von 5 Dokumenten mit dieser Beschreibung relevant sind. Diese Abweichung rührt von den Näherungsannahmen des Modells her. Anderseits zeigt sich aber, dass die Rangordnung für die verschiedenen Beschreibungen korrekt ist – was ja das Hauptziel des Modells ist.

5.3 BM25

Das obige Beispiel zeigt eine wesentliche Beschränkung des BIR-Modells auf: Es ist nicht möglich, zwischen den verschiedenen Dokumenten mit gleichem Dokumentvektor $\vec{x}$ weiter zu differenzieren, da das Modell nur mit binärer Indexierung arbeitet. Das BM25-Modell von Robertson stellt eine heuristische Erweiterung des BIR-Modells auf gewichtete Indexierung dar. Hierzu wird die Vorkommenshäufigkeit der Terme im Dokument berücksichtigt. Statt also nur zwischen Vorkommen ($x_i = 1$) und Nicht-Vorkommen ($x_i = 0$) eines Terms t_i zu unterschieden, sollen nun auch Gewichte zwischen 0 und 1 zugelassen werden. In Anlehnung an die Heuristiken des Vektorraum-Modells hat Robertson eine ähnliche Gewichtungsformel entwickelt: Für einen Term t_i bezeiche tf_{mi} dessen Vorkommenshäufigkeit im Dokument d_m, das insgesamt l_m laufende Wörter enthält. Die durchschnittliche Dokumentlänge der Kollektion sei al. Zusätzlich beinhaltet die Formel noch zwei Parameter, die an die jeweilige Kollektion angepasst werden müssen: b steuert den Einfluss der Längennormalisierung (mit $0 \leq b \leq 1$), und k kontrolliert die Gewichtung der Vorkommenshäufigkeit.

Mit diesen Parametern berechnet man zunächst die Längennormalisierung $B = \left((1 - b) + b\frac{l_m}{al}\right)$, woraus sich die normalisierte Vorkommenshäufigkeit zu $ntf_{mi} = tf_{mi}/B$ ergibt. Das BM25-Gewicht berechnet sich dann zu

$$u_{mi} = \frac{ntf_{mi}}{k + ntf_{mi}} = \frac{tf_{mi}}{k\left((1 - b) + b\frac{l_m}{al}\right) + tf_{mi}} \tag{5.6}$$

Den Einfluss des Parameters k illustriert die Abbildung 5.1, in der tf auf der Abszisse abgetragen ist und die Ordinate das resultierende Indexierungsgewicht für unterschiedliche Werte von k zeigt. Die Auswirkung der Längennormalisierung ist in Abbildung 5.2 dargestellt: Die Achsen sind wie vor, nur zeigen die einzelnen Kurven, wie das Indexierungsgewicht von der Dokumentlänge abhängt.

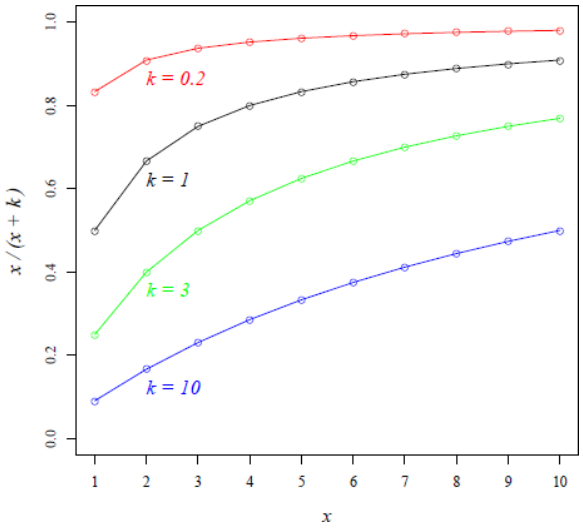

Abbildung 5.1: Einfluss von k

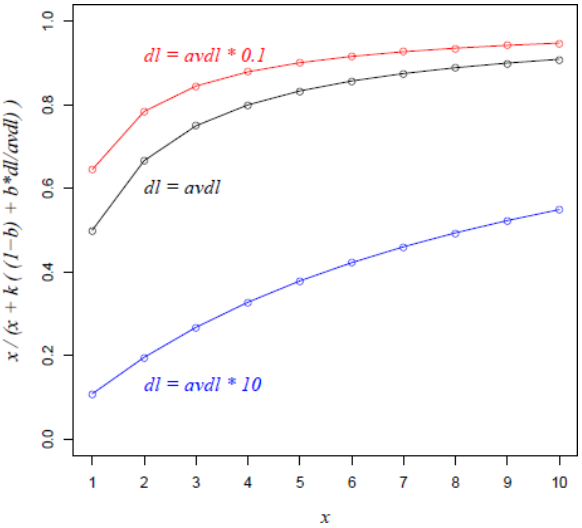

Abbildung 5.2: Einfluss der Dokumentlänge

Die vollständige Retrievalfunktion lautet dann

$$\varrho_{BM25}(q, d_m) = \sum_{t_i \epsilon d_m^T \cap q^T} u_{mi} c_i \tag{5.7}$$

$$= \sum_{t_i \epsilon d_m^T \cap q^T} \frac{tf_{mi}}{k((1-b) + b\frac{l_m}{al}) + tf_{mi}} \log \frac{p_i(1-s_i)}{s_i(1-p_i)} \tag{5.8}$$

5.4 Statistische Sprachmodelle

Obwohl die BM25-Formel gute Retrievalergebnisse liefert und sehr häufig angewendet wird, hat sie doch den Nachteil, dass ihre Indexierungsgewichte keine theoretische Fundierung besitzen. Statistische Sprachmodelle können diesen Nachteil überwinden. Solche Modelle betrachten Sprache als Folge von Wörtern, die durch einen stochastischen Prozess erzeugt wird. Ein Sprachmodell θ ist somit definiert als eine Wahrscheinlichkeitsverteilung über die Terme des Vokabulars:

$$\theta = \{(t_i, P(t_i|\theta)|t_i \in T)\} \quad \text{mit} \sum_{t_i \in T} P(t_i|\theta) = 1$$

Damit lässt sich dann z.B. für einen gegebenen Text $d = t_1 t_2 t_3 \ldots t_m$ die Wahrscheinlichkeit berechnen, dass dieser von θ generiert wurde: $P(d|\theta) = \prod_{j=1}^{m} P(t_j|\theta)$.

Statistische Sprachmodelle wurden zuerst in der automatischen Spracherkennung verwendet, später allgemein in der quantitativen Linguistik, bevor sie dann 1998 auch im IR Einzug hielten. Die Grundidee beim Retrieval ist dabei, die Wahrscheinlichkeit zu betrachten, dass Frage und Dokument von demselben Sprachmodell generiert wurden. Genau genommen wird damit etwas anderes als die Relevanzwahrscheinlichkeit berechnet; man nimmt daher an. dass die beiden Wahrscheinlichkeiten proportional zueinander sind, so dass es für ein Ranking ausreicht, die vom Sprachmodell gelieferte Wahrscheinlichkeit zu betrachten.

5.4.1 Sprachmodell von Zhai und Lafferty

Als ein populäres Beispiel für ein Sprachmodell betrachten wir den Ansatz von Zhai und Lafferty. Als Variante des o.g. allgemeinen Ansatzes betrachten diese die Wahrscheinichkeit, dass die Anfrage q vom Sprachmodell des Dokumentes d generiert wurde:

$$\begin{aligned} P(q|d) &\approx \prod_{t_i \subseteq q^T} P(t_i|d) \\ &= \prod_{t_i \in q^T \cap d^T} P_s(t_i|d) \prod_{t_i \in q^T - d^T} P_u(t_i|d) \\ &= \prod_{t_i \in q^T \cap d^T} \frac{P_s(t_i|d)}{P_u(t_i|d)} \prod_{t_i \in q^T} P_u(t_i|d) \end{aligned}$$

Hierbei bezeichnet $P_s(t_i|d)]$ die Wahrscheinlichkeit dass das Dokument über t_i ist, falls t_i im Dokumenttext vorkommt. Analog steht $P_u(t_i|d)$ für den Fall, dass das Dokument über t_i ist, obwohl der Term selbst im Dokument nicht vorkommt. Zur Schätzung dieser beiden Parameter benötigt man nun spezielle Verfahren, da der direkte Weg über die relative Häufigkeit angesichts der spärlichen Daten zu systematisch verfälschten Werten führen würde.

Im Folgenden bezeichne N die Anzahl Token (fortlaufende Wörter) der Kollektion, $tf(t,d)$ die Vorkommenshäufigkeit von t in d, $l(d)$ die Dokumentlänge (Anzahl Token) von d sowie $cf(t)$ die Kollektionshäufigkeit (Gesamtzahl Vorkommen) von t. Damit berechnet man zunächst die beiden Parameter

$$P_{avg}(t) = \frac{cf(t)}{N} \qquad \text{und} \qquad P_{ML}(t|d) = \frac{tf(t,d)}{l(d)} \tag{5.9}$$

Hierbei steht $P_{avg}(t)$ für die relative (mittlere) Vorkommenshäufigkeit von T in der Kollektion, und $P_{ML}(t|d)$ bezeichnet den Maximum-Likelihood-Schätzer (durch die relative Häufigkeit) für die Vorkommenshäufigkeit von t in d. Nun schätzt man

$$\begin{aligned} P_u(t_i|d) &= \alpha_d P_{avg}(t) \\ P_s(t_i|d) &= \lambda P_{ML}(t|d) + (1-\lambda) P_{avg}(t) \end{aligned}$$

Hierbei bezeichnet λ (mit $0 < \lambda < 1$) den Glättungsfaktor nach der Jelinek-Mercer-Methode, und α_d ist eine Dokument-spezifische Konstante, die wie folgt definiert ist:

$$\alpha_d = \frac{1 - \sum_{t_i \in q^T \cap d^T} P_{avg}(t)}{1 - \sum_{t_i \in q^T \cap d^T} P_{ML}(t|d)}$$

Alternativ gibt es noch eine Reihe weiterer Glättungsverfahren, die $P_s(t_i|d)$ auf unterschiedliche Weise aus den beiden Parametern $P_{avg}(t)$ und $P_{ML}(t|d)$ berechnen.

5.4.2 Ähnlichkeit von Wahrscheinlichkeitsverteilungen

Ein alternativer Ansatz zur Definition einer Retrievalfunktion für Sprachmodelle besteht in der Betrachtung der Ähnlichkeit der Sprachmodelle von Frage und Dokument. Wir gehen also davon aus, dass wir ein Dokument-Sprachmodell θ_d haben, dessen Parameter wir z.B. wie oben berechnen können. Zusätzlich berechnen wir noch das Sprachmodell θ_q der Anfrage, das wir z.B. als $P_{ML}(t|q)$ abschätzen können.

Für die Quantifizierung der Ähnlichkeit der beiden Wahrscheinlichkeitsverteilungen kann man nun die *Kullback-Leibler Divergence* verwenden, die ein Maß für die Unähnlichkeit der beiden Verteilungen darstellt. Die Grundidee dieses Maßes besteht darin, die relative Information zu messen. Gemäß der Informationstheorie kann man Information quantifizieren als den Logarithmus der zugehörigen Wahrscheinlichkeit, so dass die Information eines Terms in einem Sprachmodell sich zu $-\log P(t|\theta)$ ergibt. Dann ist die Differenz der Information dieses Terms in den zwei Sprachmodellen: $\log P(t|\theta_q) - \log P(t|\theta_d) = \log \frac{P(t|\theta_q)}{P(t|\theta_d)}$. Nun summiert man diese Differenzen über alle Terme, wobei man zusätzlich entsprechend der relativen Häufigkeit der Terme gewichtet:

$$D(\theta_q||\theta_d) = \sum_{i=1}^{n} P(t_i|\theta_q) \log \frac{P(t_i|\theta_q)}{P(t_i|\theta_d)}$$

Die Dokumente werden dann nach steigenden Divergenz-Werten angeordnet.

5.5 Das Probabilistische Ranking-Prinzip

Das Probabilistische Ranking-Prinzip (PRP) stellt die theoretische Rechtfertigung für probabilistische IR-Modelle dar. Eine solche Begründung gibt es allein für die probabilistischen Modelle, während alle anderen Modelle nur empirisch (durch ihre mehr oder weniger guten Retrievalergebnisse) begründet werden können. Das PRP zeigt, wie man optimales Retrieval erreichen kann. Optimales Retrieval wird dabei in Bezug auf die Repräsentationen definiert (z.B. liegt bei dem in den Tabellen 5.1 und 5.2 dargestellten Beispiel optimales Retrieval vor). Im Gegensatz dazu sprechen wir von perfektem Retrieval, wenn wir uns auf die Objekte selbst (und nicht auf die Repräsentationen) beziehen. Perfektes Retrieval ordnet alle relevanten Dokumente vor dem ersten irrelevanten Dokument an. Da aber IR-Systeme immer mit Repräsentationen arbeiten, ist perfektes Retrieval kein realistisches Ziel.

Das in [Robertson 77] ausführlich beschriebene PRP besagt, dass man optimales Retrieval erhält, wenn man die Dokumente nach fallender Relevanzwahrscheinlichkeit anordnet. Als Optimierungskriterium werden hierzu verschiedene Aspekte Maße betrachtet, insbesondere die gängigen Retrievalmaße. Wir betrachten hier nur die einfachste Rechtfertigung in Form eines entscheidungstheoretischen Ansatzes, die auf einem Kostenmaß basiert.

Bezeichne $\bar{C}$ die Kosten für das Retrieval eines irrelevanten Dokumentes, und C seinen die entsprechenden Kosten im relevanten Fall. Diese Kosten sind dabei abstrakte Größen – neben monetären Aspekten können sie etwa auch den Aufwand eines Benutzers (z.B. in Form von Arbeitzeit) messen. Die einzige Bedingung ist, dass $\bar{C} > C$, also relevante Dokumente weniger Kosten (oder höheren Nutzen) bedeuten. Mit Hilfe der vom System geschätzten Relevanzwahrscheinlichkeit kann man nun die erwarteten Kosten für das Retrieval eines bestimmten Dokumentes d_j abschätzen:

$$EC(q, d_j) = C \cdot P(R|q, d_j) + \bar{C}(1 - P(R|q, d_j)) \tag{5.10}$$

Nun nehmen wir an, dass der Benutzer die ausgegebenen Dokumente in der Reihenfolge der Rangordnung anschaut und nach l Dokumenten stoppt, wobei l nicht im Voraus bekannt ist. Die durch das System erzeugte Rangordnung beschreiben wir durch eine Ranking-Funktion $r(i)$, die den Index des Dokumentes für den Rang i angibt. Die erwarteten Gesamtkosten für die vom Benutzer betrachteten Dokumente berechnen sich dann als Summe der erwarteten Kosten der einzelnen Dokumente:

$$EC(q, l) = EC(q, d_{r(1)}, d_{r(2)}, \ldots, d_{r(l)}) = \sum_{i=1}^{l} EC(q, d_{r(i)}) \tag{5.11}$$

Um diese Kosten zu minimieren, müssen wir die Dokumente einfach nach aufsteigenden Kosten anordnen:

$$EC(q, d_{r(i)}) \leq EC(q, d_{r(i+1)})$$

Dann haben wir für jeden beliebigen Abbruchpunkt minimale Kosten. Setzen wir nun die Formel (5.10) in diese Bedingung ein, so erhalten wir

$$C \cdot P(R|q, d_{r(i)}) + \bar{C}(1 - P(R|q, d_{r(i)})) \leq C \cdot P(R|q, d_{r(i+1)}) + \bar{C}\left(1 - P(R|q, d_{r(i+1)})\right) \tag{5.12}$$

Da $C < \bar{C}$, ist dies äquivalent zu :

$$P(R|q, d_{r(i)}) \geq P(R|q, d_{r(i+1)}).$$

Damit haben wir nun die Kernaussage des PRP bewiesen: *Optimales Retrieval erhält man, indem man die Dokumente nach absteigender Relevanzwahrscheinlichkeit anordnet.*

Literaturverzeichnis

Bookstein, A. (1985). Probability and Fuzzy-Set Applications to Information Retrieval. *Annual Review of Information Science and Technology 20*, S. 117–151.

Burkart, M. (1990). Dokumentationssprachen. In: *Grundlagen der praktischen Information und Dokumentation*, S. 143–182. K.G. Saur, München et al.

Charniak, E.; Hendrickson, C.; Jacobson, N.; Perkowitz, N. (1993). Equations for Part-of-speech Tagging. In: *Proceedings of the Elenth National Conference on Artificial Intelligence*, S. 784–789. Morgan Kaufman, Menlo Park, CA.

Cleverdon, C. W. (1991). The Significance of the Cranfield Tests on Index Languages. In: *Proceedings of the Fourteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, S. 3–11. ACM, New York.

Cooper, W. S. (1991). Some Inconsistencies and Misnomers in Probabilistic IR. In: *Proceedings of the Fourteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, S. 57–61. ACM, New York.

Greene, B. B.; Rubin, G. M. (1971). *Automatic Grammatical Tagging of English.* Technical report, Brown University, Providence, RI.

Harman, D. (1995). Overview of the Second Text Retrieval Conference (TREC-2). *Information Processing and Management 31(03)*, S. 271–290.

Joachims, T. (2001). *The Maximum-Margin Approach to Learning Text Classifiers. Methods, Theory, and Algorithms.* PhD thesis, Fachbereich Informatik, Universität Dortmund.

Krause, J. (1992). Intelligentes Information Retrieval. Rückblick, Bestandsaufnahme und Realisierungschancen. In: *Experimentelles und praktisches Information Retrieval*, S. 35–58. Universitätsverlag Konstanz, Konstanz.

Kuhlen, R. (1977). *Experimentelle Morphologie in der Informationswissenschaft.* Verlag Dokumentation, München.

Kuhlen, R. (1990). Zum Stand pragmatischer Forschung in der Informationswissenschaft. In: *Pragmatische Aspekte beim Entwurf und Betrieb von Informationssystemen. Proceedings des 1. Internationalen Symposiums für Informationswissenschaft*, S. 13–18. Universitätsverlag Konstanz, Konstanz.

Kuhlen, R. (1991). Zur Theorie informationeller Mehrwerte. In: *Wissensbasierte Informationssysteme und Informationsmanagement*, S. 26–39. Universitätsverlag Konstanz.

Lee, J. H.; Kim, W. Y.; Kim, M. H.; Lee, Y. J. (1993). On the Evaluation of Boolean Operators in the Extended Boolean Retrieval Framework. In: *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, S. 291–297. ACM, New York.

Raghavan, V. V.; Wong, S. K. M. (1986). A Critical Analysis of Vector Space Model for Information Retrieval. *Journal of the American Society for Information Science 37(5)*, S. 279–287.

van Rijsbergen, C. J. (1979). *Information Retrieval.* Butterworths, London, 2. Auflage.

Robertson, S. E. (1977). The Probability Ranking Principle in IR. *Journal of Documentation 33*, S. 294–304.

Rocchio, J. J. (1966). *Document Retrieval Systems - Optimization and Evaluation.* Report ISR-10 to the NSF, Computation Laboratory, Harvard University.

Salton, G.; Buckley, C. (1988). Term Weighting Approaches in Automatic Text Retrieval. *Information Processing and Management 24(5)*, S. 513–523.

Salton, G.; Buckley, C. (1990). Improving Retrieval Performance by Relevance Feedback. *Journal of the American Society for Information Science 41(4)*, S. 288–297.

Salton, G.; McGill, M. J. (1983). *Introduction to Modern Information Retrieval.* McGraw-Hill, New York.

Salton, G. (Hrsg.) (1971). *The SMART Retrieval System - Experiments in Automatic Document Processing.* Prentice Hall, Englewood, Cliffs, New Jersey.

Salton, G. (1986). Another Look at Automatic Text-Retrieval Systems. *Communications of the ACM 29(7)*, S. 648–656.

Salton, G.; Fox, E.; Wu, H. (1983). Extended Boolean Information Retrieval. *Communications of the ACM 26*, S. 1022–1036.

Turpin, A. H.; Hersh, W. (2001). Why batch and user evaluations do not give the same results. In: Croft, W. B.; Harper, D.; Kraft, D. H.; Zobel, J. (Hrsg.): *Proceedings of the 24th Annual International Conference on Research and development in Information Retrieval*, S. 225–231. ACM Press, New York.

Verhoeff, J.; Goffmann, W.; Belzer, J. (1961). Inefficiency of the Use of Boolean Functions for Information Retrieval Systems. *Communications of the ACM 4*, S. 557–558.

Voorhees, E.; Harman, D. (2000). Overview of the Eighth Text REtrieval Conference (TREC-8). In: *The Eighth Text REtrieval Conference (TREC-8)*, S. 1–24. NIST, Gaithersburg, MD, USA.

Wong, S. K. M.; Ziarko, W.; Raghavan, V. V.; Wong, P. C. N. (1987). On Modeling of Information Retrieval Concepts in Vector Spaces. *ACM Transactions on Database Systems 12(2)*, S. 299–321.

Zadeh, L. A. (1965). Fuzzy Sets. *Information and Control 8*, S. 338–353.

Zimmermann, H. (1991). Ein Verfahren zur automatischen Trunkierung beim Zugang zu textbezogenen Informationsbanken. In: *Wissensbasierte Informationssysteme und Informationsmanagement*, S. 125–144. Universitätsverlag Konstanz.