Datenbanken
WS 04/05
(IR-Teil)

Norbert Fuhr

10. Januar 2005

Inhaltsverzeichnis

Kapitel 1

Einführung

Will man den Gegenstand des Information Retrieval (IR) mit wenigen Worten beschreiben, so ist die Formulierung „inhaltliche Suche in Texten“ wohl am treffendsten. Tatsächlich wird damit aber nur ein wesentlicher — wenn auch der wichtigste — Bereich des Information Retrieval umschrieben, den man auch häufig als Textretrieval oder Dokumentenretrieval bezeichnet.

Das klassische Anwendungsgebiet des IR sind Literaturdatenbanken, die heute in Form Digitaler Bibliotheken zunehmend an Bedeutung gewinnen. IR ist besonders populär geworden durch die Anwendung in Internet-Suchmaschinen; dadurch kommt jeder Internet-Nutzer mit IR-Methoden in Berührung. Neben der Suche in Texten werden auch zunehmend IR-Anwendungen für multimediale Daten realisiert, wobei insbesondere Bildretrieval-Methoden eine gewisse Verbreitung erfahren haben.

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 (vgl. die Gesamtzahl der Treffer bei Internet-Suchmaschinen), 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 Unischerheit behaftet ist.

1.1 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 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 Dokumen-

ten können dies heutige IR-Systeme aber kaum leisten.

Abschließend zu diesen Betrachtungen geben wir hier die in [Rijsbergen 01] skizzierten Dimensionen des IR wieder (Tabelle 1.1).

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

Kapitel 2

IR-Konzepte

2.1 Daten — Information — Wissen

Datenbanksysteme enthalten Daten. IR-Systeme sollen die Suche nach Information[1] 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 2.1). 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 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 „Information ist Wissen in Aktion“.

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

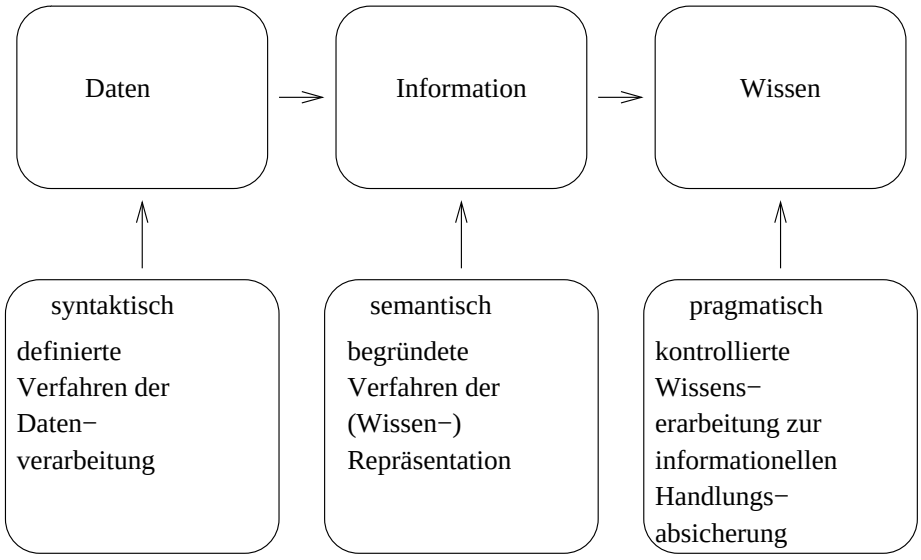


Abbildung 2.1: Daten — Information — Wissen

Als anschauliches Beispiel kann man hierzu die online verfügbaren UNIX-Manuals betrachten. Diese beinhalten Information über UNIX. Wenn nun ein Benutzer eines UNIX-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 Informatio 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.

2.2 Grundmodell des Information Retrieval

Wir stellen hier ein einfaches Grundmodell für Information Retrieval vor. Das allgemeine Modell ist in Abb. 2.2 dargestellt; die linke Seite der Abbildung repräsentiert dabei den Vorgang der Eingabe, während der Retrievalprozess auf der rechten Seite dargestellt ist. Bei der Eingabe werden also Daten analysiert und dann in gespeichertes Wissen überführt. Beim Retrieval wird die benötigte Information durch Transformationen auf diesem gespeicherten Wissen erzeugt.

Die folgenden Abbildungen 2.3–2.5 zeigen die Spezialisierung dieses allgemeinen Modells für verschiedene Arten von IR-Systemen.

2.3 Konzeptionelles Modell für IR-Systeme

Wir beschreiben hier ein konzeptionelles Modell für Informationssysteme, das wir für die nachfolgenden Ausführungen zugrundelegen wollen. Dabei beschränken wir uns auf die Funktion der Informationssuche, während andere Aspekte solcher Systeme (z.B. die Aktualisierung der Datenbank oder zentrale vs. verteilte Datenhaltung) unberücksichtigt bleiben.

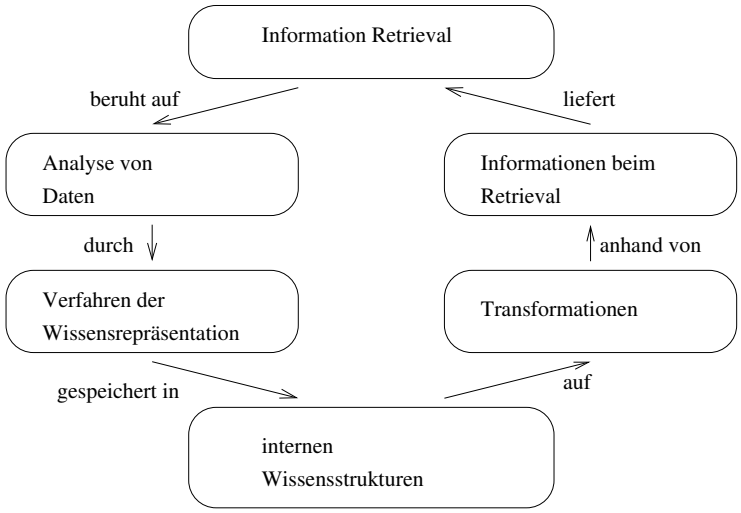


Abbildung 2.2: Grundmodell des Information Retrieval

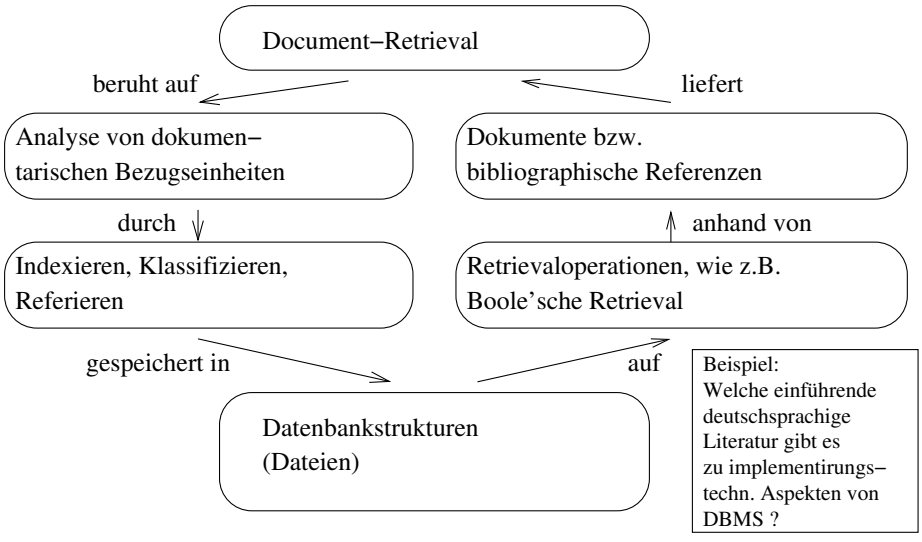


Abbildung 2.3: Grundmodell für Dokumentretrieval

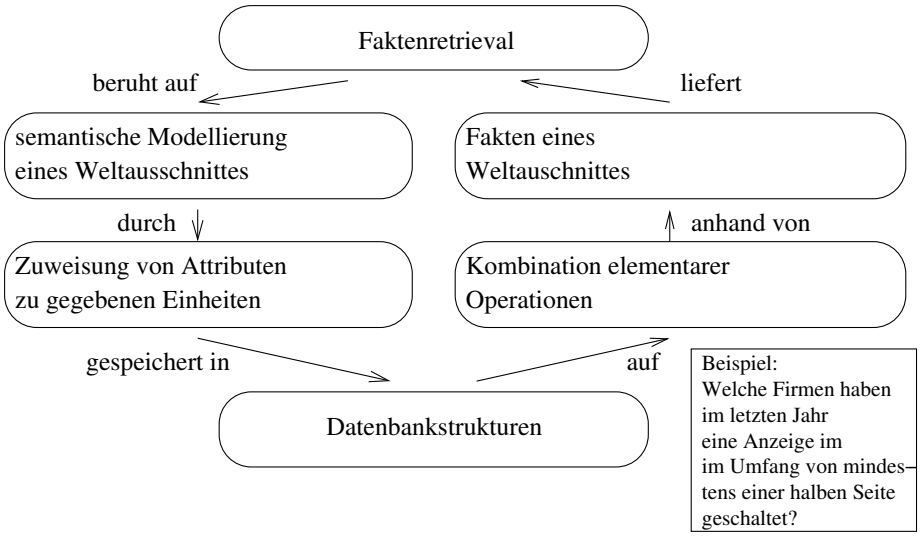


Abbildung 2.4: Grundmodell für Datenretrieval

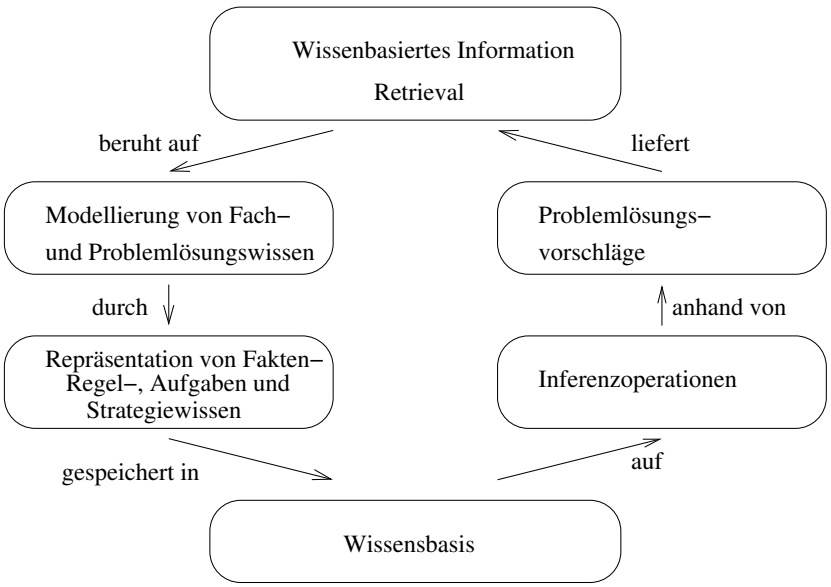


Abbildung 2.5: Grundmodell für wissensbasiertes IR

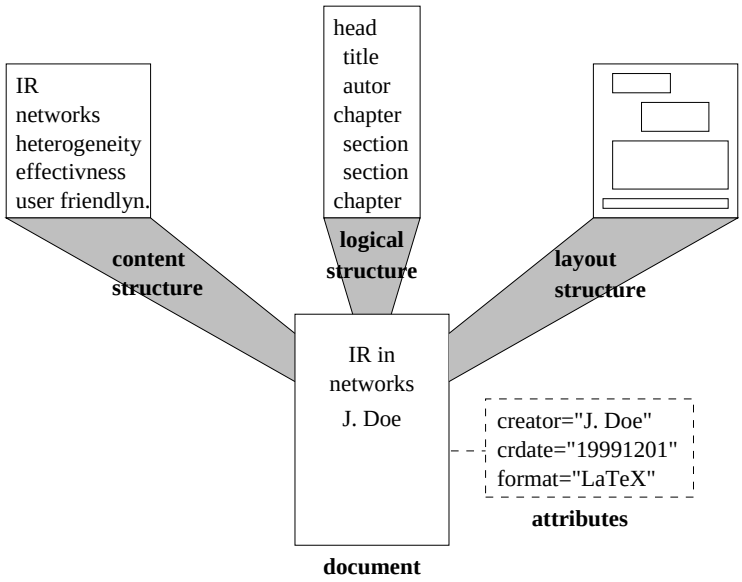


Abbildung 2.6: Sichten auf Dokumente

Das vorgeschlagene konzeptionelle Modell baut auf dem in [Meghini et al. 91] dargestellten Dokumentmodell auf, das wiederum eine Erweiterung der ursprünglich im Electronic Publishing entwickelten Trennung zwischen Layout und logischer Struktur ist. Die wesentliche Idee dieses Dokumentmodells besteht darin, dass es mehrere Sichten auf den Inhalt eines Dokumentes gibt (siehe Abb. 2.6)

- Die Layout-Sicht beschreibt die Darstellung eines Dokumentes auf einem zweidimensionalen Medium.
- Die logische Sicht enthält die logische Struktur eines Dokumentes; diese umfasst die zur Verarbeitung (z.B. Editieren) notwendigen Informationen, also im wesentlichen den Inhalt ohne die Layout-Struktur.
- Die semantische (oder inhaltliche) Sicht bezieht sich auf die Semantik des Inhalts eines Dokumentes. Für IR-Systeme ist diese Sicht essentiell, da ansonsten nur primitive Suchoperationen in der Form der Zeichenkettensuche möglich wären.

Prinzipiell lassen sich diese drei Sichten auf beliebige Objekte in Datenbanken anwenden. Bei herkömmlichen Datenbanken wird nur die logische Sicht unterstützt, da dies für die meisten Anwendungen völlig ausreicht. Solange an die Darstellung der Objekte keine besonderen Anforderungen gestellt werden, reichen meist die generischen Ausgabeformate der interaktiven Anfrageschnittstelle aus. Darüber hinaus werden aber für Standard-Anwendungen spezielle Ausgabemasken erstellt, so dass diese im Prinzip die Layout-Sicht realisieren (allerdings ist die Verwaltung dieser Masken nicht in das Datenbanksystem integriert). Für eine zusätzliche semantische Sicht bestand bislang bei Datenbanksystemen keine Notwendigkeit, da diese bei den vorherrschenden kaufmännischen und administrativen Anwendungen mit der logischen Sicht identisch ist.

Aufbauend auf diesen drei Sichten auf Dokumente und Datenbank-Objekte im allgemeinen lässt sich das in Abbildung 2.7 dargestellte konzeptionelle Modell

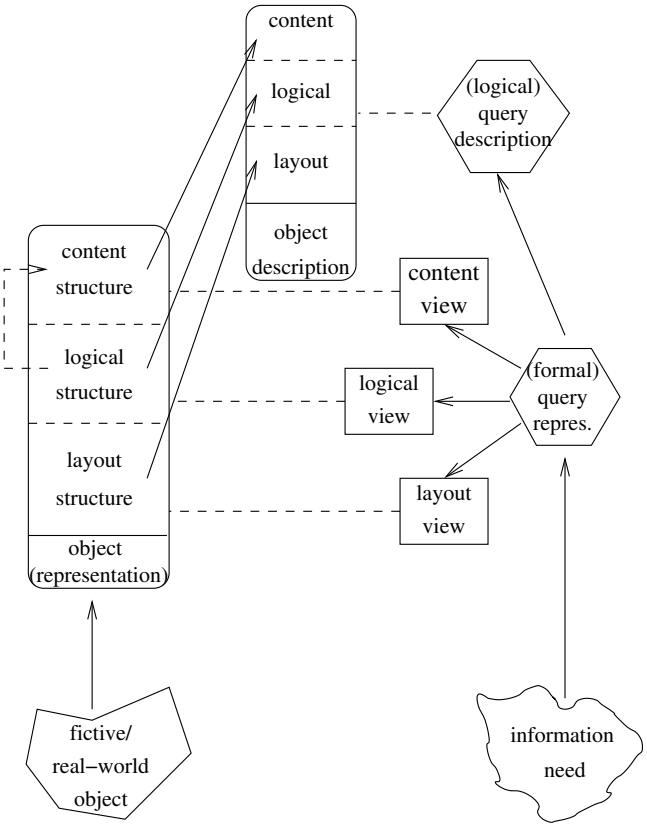


Abbildung 2.7: Konzeptionelles Modell für Informationssysteme

formulieren. In einem Informationssystem werden (fiktive oder reale) Objekte aus der zu modellierenden Anwendung in einer internen Repräsentation in der Datenbank gespeichert, die wir hier als DB-Objekt bezeichnen. Ein solches Objekt kann z.B. ein Textdokument oder eine Menge von Fakten sein, die das reale Objekt beschreiben; im ersteren Falle würde die interne Repräsentation etwa der Form eines Textes entsprechen, wie er z.B. mit einem Texteditor erstellt wurde.

Aus dieser Repräsentation wird sowohl die logische wie auch die semantische und die Layout-Sicht des Objektes abgeleitet. Zum Beispiel wird bei vielen Forschungsansätzen im Bereich des IR für Texte folgende semantische Sicht zugrundegelegt: Der Text besteht aus einzelnen Abschnitten; diese Textabschnitte werden in Worte zerlegt, die wiederum auf Grund- oder Stammform reduziert werden; für diese Terme bestimmt man dann die Vorkommenshäufigkeit im Text und in der gesamten Datenbasis. Bei herkömmlichen kommerziellen IRS werden dagegen die Textabschnitte nur als Folge von Wörtern aufgefasst, wobei ein Wort eine durch Leer- oder Interpunktionszeichen begrenzte Buchstabenfolge ist. Somit existiert hier keine eigentliche semantische Sicht, sondern es wird nur mit der logischen Sicht gearbeitet.

Vom Standpunkt des IR interessiert besonders, welche Arten von Anfragen bezüglich eines Objektes möglich sind. Dieser Aspekt wird im konzeptionellen

Modell durch die Objektattribute dargestellt; Anfragen können sich nur auf diese Attribute beziehen. Die Attribute wiederum können sich aus allen drei Sichten ableiten. Dadurch wird es möglich, sowohl Fragen nach der Semantik von Objekten zu stellen als auch Objekte mit bestimmter logischer oder Layout-Struktur zu suchen. Beispiele für Anfragen mit Bezug zu den verschiedenen Arten von Attributen wären in einem Büro-Informationssystem etwa „Suche alle Informationen über Büromöbel“ (semantisch), „Suche alle Rechnungen der Firma Meier“ (logisch) und „Suche einen Brief, der ein blaues Logo in der rechten oberen Ecke enthält“ (Layout). In der Regel werden in einer Anfrage aber Bedingungen an die verschiedenen Arten der Attribute in Kombination auftreten.

Auf der Seite der Fragen ergibt sich eine ähnliche Struktur wie bei den Objekten. Ein Informationswunsch eines Benutzers muss dem System gemäß der verwendeten Anfragesprache als formalisierte Anfrage übergeben werden. Bei kommerziellen IRS ist dies z.B. häufig ein boolescher Ausdruck oder eine Folge von ausgewählten Menüpunkten, bei experimentellen Systemen werden meist natürlichsprachliche Formulierungen zugelassen. Prinzipiell umfasst die formalisierte Anfrage alle Eingaben eines Benutzers während der Bearbeitung einer Anfrage, also z.B. auch Relevanzbeurteilungen zu einzelnen ausgegebenen Dokumenten. Aus der formalisierten Anfrage wird dreierlei abgeleitet:

- Die Fragelogik enthält die Bedingungen an die Objektattribute, die ein Antwortobjekt erfüllen sollte. Wie oben erläutert, können sich diese Bedingungen aber nur auf die Objektattribute beziehen; auf Aspekte eines Objektes, die nicht durch entsprechende Attribute repräsentiert sind, kann auch nicht in der logischen Frageformulierung bezug genommen werden. Diese Tatsache ist besonders wichtig bei multimedialen Informationssystemen, die zwar häufig die Speicherung von Audio- und Videosequenzen erlauben, aber nur sehr beschränkte Möglichkeiten zur Suche nach solchen Objekten anbieten.
- Die Frage-Sicht definiert, welche Teile des Inhalts eines gefundenen Objektes als Antwort ausgegeben werden sollen. In relationalen Datenbanksystemen wird diese Sicht z.B. mit Hilfe des Projektions-Operators definiert. Für komplexer strukturierte Objekte (wie sie z.B. in objektorientierten Datenbanken zu finden sind) müssen hier natürlich auch entsprechend mächtigere Operationen angeboten werden. Bei der inhaltlichen Suche nach Texten oder multimedialen Dokumenten wäre es wünschenswert, wenn nur die Teile des Objektes, die für die Beantwortung der Frage relevant sind, angezeigt würden. Dies setzt natürlich eine tiefergehende Inhaltserschließung voraus.
- Die Layout-Spezifikation schließlich legt fest, in welcher Form die Sicht auf ein oder mehrere Antwortobjekte dargestellt werden soll.

Anhand dieser Abbildung kann auch der Aspekt der Unsicherheit verdeutlicht werden. Bei vielen technisch-wissenschaftlichen Anwendungen ist schon die Abbildung eines realen Objektes in die Objektrepräsentation mit Unsicherheit behaftet, weil dessen Eigenschaften (etwa Messwerte) nur mit einer begrenzten Genauigkeit oder nur unvollständig erhoben werden können. Die Ableitung der Objektsemantik aus der internen Repräsentation ist eine weitere Quelle von Unsicherheit. Speziell bei Texten oder multimedialen Dokumenten kann deren Inhalt nur unzureichend erschlossen werden, und durch die anschließende Verdichtung in Form der Objektattribute ergibt sich ein weiterer Informationsverlust.

Auf der Seite der Fragen ergeben sich die gleichen Probleme der Unsicherheit, insbesondere bei der Abbildung auf die interne Repräsentation. Zusätzlich spielt hier das für IR-Anwendungen typische Moment der Vagheit eine wichtige Rolle (Im Prinzip wären auch auf der Seite der Objekte vage Beschreibungen möglich, was wir aber hier nicht weiter betrachten wollen). Daher sollte die Frageformulierung in der Lage sein, diese Vagheit zu repräsentieren. Bei probabilistischen Textretrievalsystemen geschieht dies z.B. durch eine Gewichtung der Frageterme.

Kapitel 3

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 kaum 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.

3.1 Effizienz und Effektivität

Wenn man von Bewertung von IR-Methoden spricht, so muss man zwischen Effizienz und Effektivität unterscheiden. Unter Effizienz versteht man den möglichst sparsamen Umgang mit Systemressourcen für eine bestimmte Aufgabe. Zu diesen Ressourcen zählen hauptsächlich:

- Speicherplatz,
- CPU-Zeit,
- Anzahl I/O-Operationen,
- Antwortzeiten.

In den übrigen Gebieten der Informatik kann man sich meist auf reine Effizienzbetrachtungen beschränken, weil dort die vom System zu lösenden Aufgabenstellungen klar definiert sind und eine korrekte und vollständige Lösung der Aufgaben durch das System unabdingbare Voraussetzung ist. Im Information Retrieval dagegen muss man akzeptieren, dass es praktisch kein System gibt, das die hier betrachteten Aufgabenstellungen perfekt löst. Ein wesentlicher Unterschied zwischen einzelnen Systemen besteht gerade in der Qualität, mit der ein System die gewünschten Leistungen erbringt.

Effektivität bezeichnet das Kosten-Nutzen-Verhältnis bei der Anwendung eines bestimmten Verfahrens. Bei der Nutzung eines IR-System bestehen die „Kosten“ in dem vom Benutzer aufzubringenden Zeitaufwand und seiner mentalen Belastung bei der Lösung seines Problems mithilfe des Systems. Der erzielte Nutzen besteht in der Qualität der erreichten Lösung. (Ein einfaches Beispiel hierfür wäre z.B. die Suche eines Studenten nach Literatur zur Prüfungsvorbereitung in einem bestimmten Fach; der erzielte Nutzen könnte dann z.B. an

der erreichten Note gemessen werden). Wir werden die folgenden Betrachtungen allein auf die resultierende Qualität eines Informationssystems beschränken — nicht zuletzt deshalb, weil die hier betrachteten Verfahren keine oder nur wenige, standardisierte Interaktionsmöglichkeiten zwischen Benutzer und System zulassen.

3.2 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 4 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. Dier 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), der durch die sogenannte Retrievalfunktion berechnet wird.

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.

3.3 Distributionen

Distributionen sind abstrakte Darstellung 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 un 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:

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

3.4 Standpunkte und Bewertungsmaße

Jedem Bewertungsmaß liegt ein bestimmter Standpunkt bzgl. 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.

3.4.1 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

3.4.2 Benutzer- vs. Systemstandpunkte

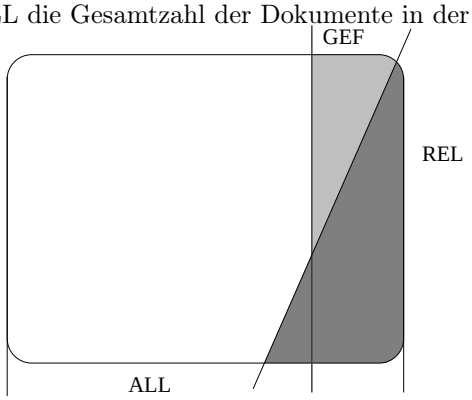
Man kann grob zwischen Benutzer- und Systemstandpunkten unterscheiden. Erstere spiegeln dabei die Sicht eines einzelnen Benutzers wider, während letzteren ein globale Sicht (die des Systembetreibers) zugrundeliegt. Dementsprechend beziehen sich **benutzerorientierte Maße** auf das mögliche Verhalten 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 $\bar{\Delta}$, z.B. $\Delta = (+ + +| - | + +| + -)$ und $\bar{\Delta} = (+ + + + + + | - -)$. Als Beispiel für ein systemorientiertes Maß wird in **??** das Nützlichkeitsmaß vorgestellt.

3.5 Maße für boolesches Retrieval

3.5.1 Recall, Precision und Fallout

Wir betrachten zunächst den Fall der Retrievalbewertung für boolesches Retrieval, da die Maße für Rangordnungen Erweiterungen der Maße für boolesches Retrieval 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.



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 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.

3.5.2 Recall-Abschätzung

Die Größe der Precision ist für jeden Benutzer eines IR-Systems direkt ersichtlich. Die Größe des Recall 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:

1. Vollständige Relevanzbeurteilung einer repräsentativen Stichprobe der gesamten Datenbasis: Da REL sehr viel kleiner als die gesamte Datenbasis ist (z.B. mögen 100 von 10^6 Dokumenten relevant sein), müsste die repräsentative Stichprobe schon einen relativ großen Teil der Datenbasis umfassen, was zuviel Beurteilungsaufwand erfordert.
2. Dokument-Source-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.
3. 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.
4. 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.
5. Retrieval mit mehreren Systemen (Pooling-Methode): 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 dass der Aufwand nicht linear mit der Anzahl betrachteter Systeme wächst [Harman 95]. Dieses Verfahren wird derzeit beim Vergleich experimenteller Systeme im Rahmen der TREC- und CLEF-Konferenzen angewandt.

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

3.5.3 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 3.1 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 3.1).

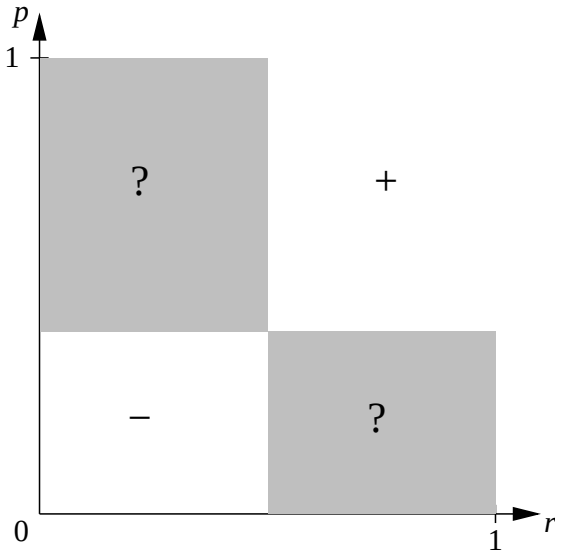


Abbildung 3.1: Darstellung eines Retrievalergebnisses als Punkt im Recall-Precision-Graphen

Als eine gängige Methode, (R,P)-Paare durch eine einzige Zahl auszudrücken (und damit den grauen Bereich aus Abbildung 3.1 in bessere/schlechtere Ergebnisse aufzuteilen), 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 Wichtung des Recall an (β=0: nur Precision zählt; $\beta = \infty$: nur Recall zählt). Üblicherweise setzt man $\beta = 1$, arbeitet also mit dem F_1-Maß.

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.

3.5.4 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:

$$p_M = \frac{1}{N} \sum_{i=1}^{N} \frac{|REL_i \cap GEF_i|}{|GEF_i|} \tag{3.1}$$

  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 diese Problem u.U. 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:

$$p_\mu = \frac{\sum_{i=1}^{N} |REL_i \cap GEF_i|}{\sum_{i=1}^{N} |GEF_i|} \tag{3.2}$$

  Dadurch wird das Problem der leeren Antwortmengen umgangen. Da hier jedes Dokument gleichstark 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.

$$\begin{aligned} \text{Sei } p_\mu(\Delta_1) = \frac{1}{2} \qquad & p_\mu(\Delta_2) = \frac{2}{5} \qquad \text{und } p_\mu(\Delta) = \frac{2}{8} \\ \text{Dann ist } p_\mu(\Delta_1) = \frac{1}{2} \quad & > \quad \frac{2}{5} = p_\mu(\Delta_2), \text{ aber} \\ p_\mu(\Delta_1, \Delta) = \frac{3}{10} \quad & < \quad \frac{4}{13} = p_\mu(\Delta_2, \Delta). \end{aligned}$$

3.6 Rangordnungen

Mit Ausnahme des booleschen Retrieval liefern alle Retrievalverfahren eine Rangordnung von Dokumenten als Antwort. 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 stellen Retrievalergebnisse meist durch folgendes Schema dar:

$$\Delta_1 = (+ - -| + + + - - - - - - -) \tag{3.3}$$

Die einzelnen Ränge werden dabei durch „|“ getrennt, „+“ bezeichnet ein relevantes und „-“ ein nichtrelevantes Dokument. Bei Δ_1 handelt es sich um eine schwache Ordnung. Lineare Ordnungen sind dagegen z.B.

$$\Delta_2 = (+| + | - | + | - | + | - | - | - | - | - | - | + |-). \tag{3.4}$$

$$\Delta_3 = (+| - | + | + | + | - | - | - | - | - | - | - | + | - | + |-) \tag{3.5}$$

3.6.1 Lineare Ordnung

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 3.1 gezeigt wird. Δ_2 (3.4) 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 3.2 gezeigte Bild. Um die Übersichtlichkeit zu erhöhen, kann man die einzelnen Punkte mit Geradenstücken verbinden (siehe Abb. 3.3). Allerdings darf man den Punkten auf diesen Geradenstücken keine Bedeutung zuordnen (um z.B. Zwischenwerte zu interpolieren)! Diese Art der Darstellung ist besonders nützlich, wenn man die die Qualitätsmaße für mehrere Rangordnungen in einem einzigen Graphen darstellen möchte (siehe Abb. 3.4 und 3.5).

Um die Kurven im R-P-Graphen interpretieren zu können, wird in [Salton & McGill 83, S. 167-8] vorgeschlagen, die Originalkurve wie in Abb. 3.6 dargestellt zu interpolieren. Dabei wird jeder einzelne (r, p) Wert durch eine waagerechte Linie bis zu $r = 0$ interpoliert. Der resultierende Graph ergibt sich dann als das Maximum über diese Geradenstücke.

n	Dokumentnr.	x=rel.	Recall	Precision
1	588	x	0.2	1.00
2	589	x	0.4	1.00
3	576		0.4	0.67
4	590	x	0.6	0.75
5	986		0.6	0.60
6	592	x	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	x	1.0	0.38
14	990		1.0	0.36

Tabelle 3.1: Recall-Precision nach n Dokumenten bei linearer Ordnung

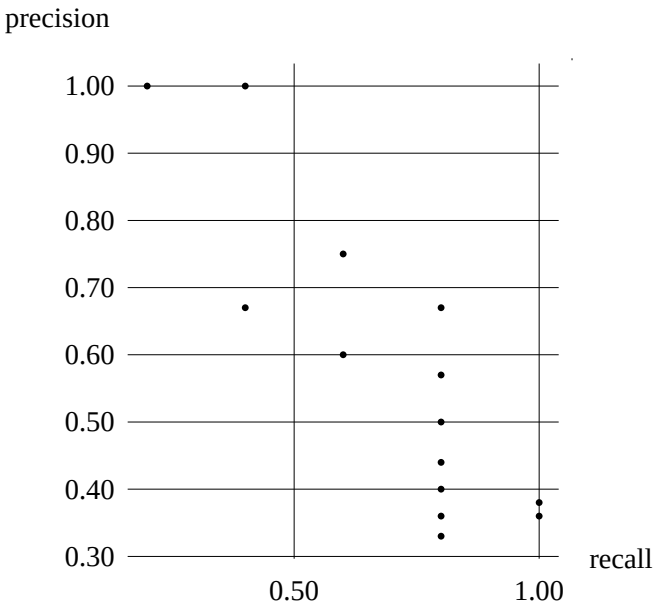


Abbildung 3.2: Graphische Darstellung der Werte aus Tabelle 3.1

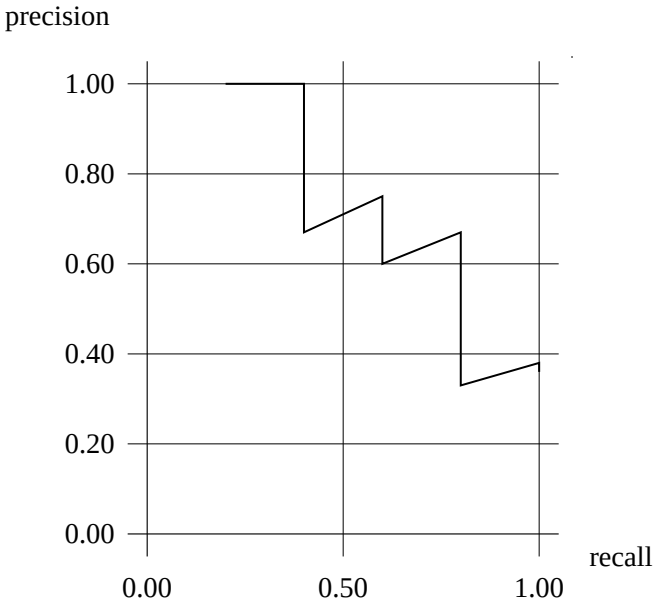


Abbildung 3.3: Darstellung der Werte aus Tabelle 3.1 als Kurve

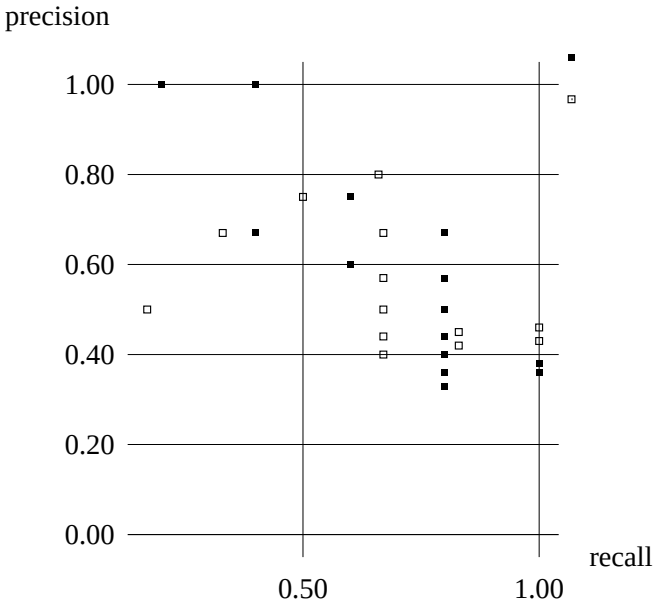


Abbildung 3.4: Graphische Darstellung der Werte für zwei verschiedene Rangordnungen (Δ_2 (3.4) und Δ_3 (3.5))

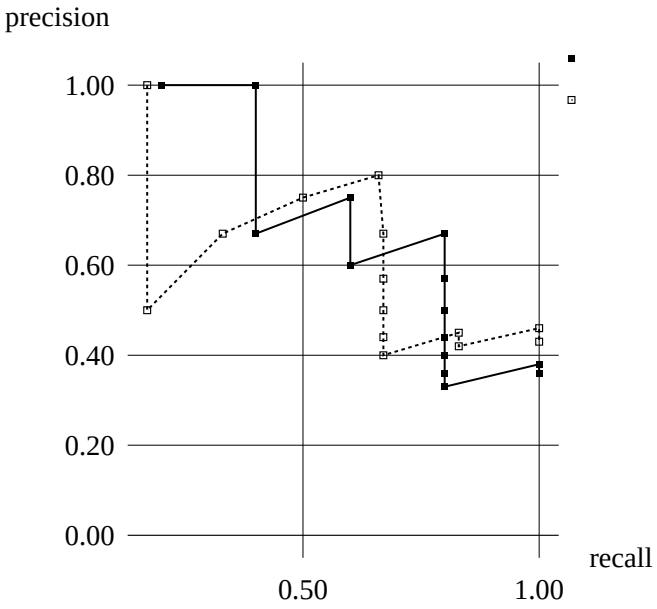


Abbildung 3.5: Darstellung der Werte aus Abb. 3.4 als Kurve

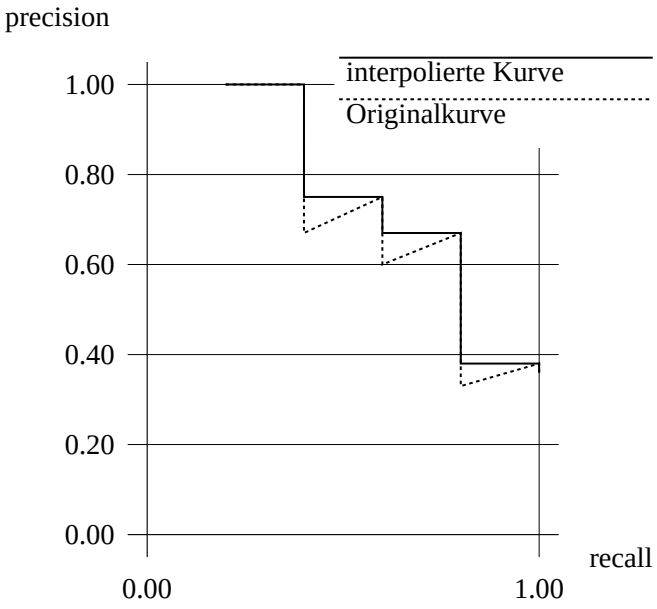


Abbildung 3.6: R-P-Graph nach Salton

Kapitel 4

Wissensrepräsentation für Texte

4.1 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.

4.2 Freitextsuche

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).

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

1. Zerlegung des Textes in einzelne Wörter: Leer- und Interpunktionszeichen werden hier als Worttrenner aufgefasst.
2. Stoppwortbestimmung: Nicht-bedeutungstragende Wörter wie Artikel, Füllwörter oder Konjunktionen werden meist aus Aufwandsgründen von der weiteren Verarbeitung ausgeschlossen. Nur für syntaktische Verfahren müssen die Stoppworte berücksichtigt werden, um ein korrektes Parsing zu ermöglichen. Stoppwörter machen häufig rund die Hälfte des Textes aus.
3. Satzendeerkennung: Einige Freitextfunktionen erlauben den Bezug auf Satzgrenzen, die folglich erst erkannt werden müssen. Wegen der Verwechslungsmöglichkeit mit Abkürzungspunkten kann diese Aufgabe nur approximativ gelöst werden (z.B. mit Hilfe von Abkürzungslisten).

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**, die durch Konjugation und Deklination eines Wortes entstehen
  Haus – (des) Hauses – Häuser,
  schreiben – schreibt – schrieb – geschrieben
- **Derivationsformen** (verschiedene Wortformen zu einem Wortstamm)
  Formatierung – Format – formatieren
- **Komposita** (mehrgliedrige Ausdrücke)
  Bundeskanzlerwahl – Wahl des Bundeskanzlers
  information retrieval – retrieval of information – information was retrieved

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

4.2.1 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 Suchpattern 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 . . .

4.2.2 Computerlinguistischer Ansatz

Der computerlinguistischer 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.

4.2.2.1 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 Infinitiv. 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

4.2.2.1.1 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
J	Länge des Wortes
/	‚oder'
Ø	Leerzeichen
→	'zu'
←	'aus'
¬	'nicht'

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

4.2.2.2 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 allerdings 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.

4.3 Dokumentationssprachen

4.3.1 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 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.

4.3.2 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

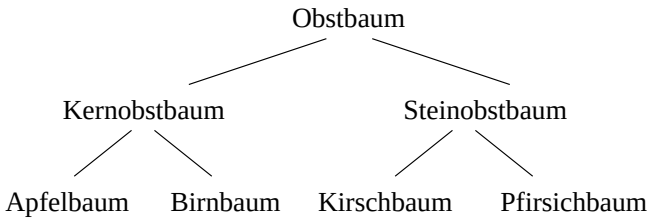


Abbildung 4.1: Monohierarchie

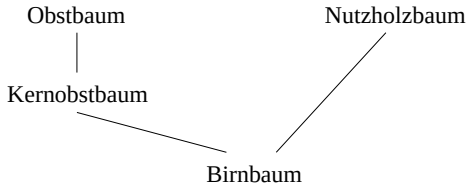


Abbildung 4.2: Polyhierarchie

UDC Universal Decimal Classification
MSc Mathematics Subject Classification
CCS ACM Computing Classification system

4.3.2.1 Eigenschaften von Klassifikationssystemen

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

4.3.2.1.1 Monohierarchie — Polyhierarchie

Abbildung 4.1 zeigt 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 (Abbildung 4.2).

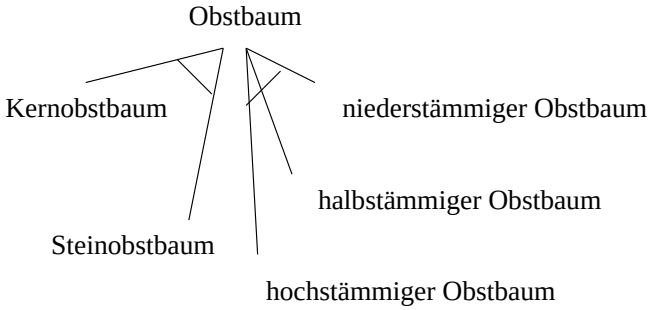


Abbildung 4.3: Polydimensionalität

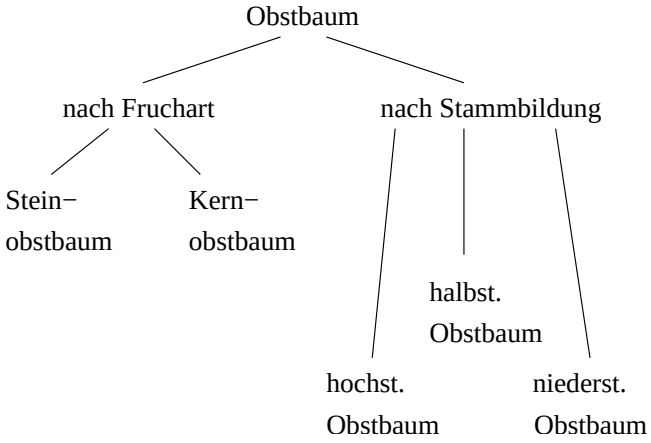


Abbildung 4.4: Aufgelöste Polydimensionalität

4.3.2.1.2 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 am Beispiel in Abb. 4.3 illustriert, erlaubt die Darstellung dieses Sachverhaltes. Erlaubt das Klassifikationsschema keine Polydimensionalität, dann muss diese durch Einführung einer zusätzlichen Hierarchieebene (s. Abb. 4.4) aufgelöst werden, wodurch das Schema unübersichtlicher wird.

4.3.2.1.3 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 4.1: 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 4.1 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.

4.3.2.2 Die Yahoo-Klassifikation

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 4.5: 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 4.6: Untergliederung der Hauptklasse Computers & Internet

Abbildung 4.5 zeigt die Hauptklassen der Yahoo-Klassifikation, und Abbildung 4.6 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.

4.3.2.3 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.

4.3.2.3.1 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.

4.3.2.3.2 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

4.3.2.3.3 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
(0...) Form
(...) Ort
(=...) Rassen und Völker
„...“ Zeit
.00 Gesichtspunkt
-05 Person

4.3.2.3.4 Verknüpfung von DK-Zahlen

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

\+ Aufzählung mehrerer Sachverhalte,
: symmetrische Beziehung zwischen zwei Sachverhalten
:: asymmetrische Beziehung zwischen zwei Sachverhalten,
/ Erstreckungszeichen (zur Zusammenfassung mehrerer nebeneinanderstehender DK-Zahlen),
' Zusamenfassungszeichen zur Bildung neuer Sachverhalte aus der Kombination einzelner DK-Komponenten.

4.3.2.4 Computing Reviews Classification

Als weiteres Beispiel eines Klassifikationsschemas zeigen wir hier aus dem Bereich der Informatik die Computing Reviews (CR) Classification, die zur Anordnung der Artikel in der Zeitschrift *ACM Computing Reviews* entworfen wurde. Darüber hinaus wird sie auch in vielen anderen Informatik-Zeitschriften verwendet und liegt insbesondere auch der einschlägigen Datenbank *Compuscience* zugrunde.

Die CR Classification 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.

4.3.2.4.1 General terms:

Die general terms der CR Klassifikation sind in Tabelle 4.2 aufgelistet.

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

Tabelle 4.2: General terms der CR Klassifikation

4.3.2.4.2 Klassen und subject headings

Die Hauptklassen der CR Klassifikation 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
- H.3.2 Information Storage

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.m Miscellaneous

4.3.3 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.

4.3.3.1 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.

4.3.3.1.1 Synonymkontrolle

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

- Schreibweisenvarianten
  Friseur — Frisör
  UN — UNO — Vereinte Nationen
- unterschiedlichen 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.

4.3.3.1.2 Polysemkontrolle

Bei der Polysemkontrollewerden mehrdeutige Bezeichnungen auf mehrere Äquivalenzklassen aufgeteilt. Man kann hierbei noch zwischen Homographen *(Bs. Tenor)* und eigentlichen Polysemen *(Bs. Bank)* unterscheiden.

4.3.3.1.3 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.

4.3.3.2 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.

4.3.3.3 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.

4.3.3.3.1 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*

4.3.3.3.2 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*

4.3.3.3.3 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 “see also” (SEE)). 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 4.7: Auszug aus dem Beziehungsgefüge des INSPEC-Thesaurus’

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 4.8: Auszug aus einem Thesaurus

4.3.3.4 Darstellung des Thesaurus

4.3.3.4.1 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 4.8 zeigt ein Beispiel für einen Ausschnitt aus einem Thesaurus.

4.3.3.4.2 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:

- 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*

4.3.3.5 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.

4.3.4 RDF (Resource Description Framework)

RDF ist eine vom W3C im Rahmen der 'Semantic Web'-Initiative geförderte Beschreibungssprache. Diese Initiative verfolgt die Vision einer weltweit verteilten Wissensbasis. Jede Web-Site kann dabei einen bestimmten Wissensausschnitt modellieren, und durch Kombination der für eine bestimmte Anwendung relevanten Ausschnitte erhält man mit geringem Aufwand eine sehr detaillierte Modellierung, mit deren Hilfe sich anspruchsvolle Aufgaben realisieren lassen.

Die grundlegende Idee ist dabei, eine im Vergleich zu Thesauri und Klassifikationen ausdrucksstärkere Beschreibungssprache zu entwickeln. Diese soll folgende Eigenschaften besitzen:

- Während bei der Deskribierung mittels Thesauri nur Konzepte zugeordnet werden können, erlaubt RDF auch die Zuordnung von Instanzen zu Konzepten (z.B. „Schröder" als Instanz von „Bundeskanzler")
- Neben den in Thesauri üblichen Beziehungen können zusätzlich beliebige Beziehungen eingeführt und zur Deskribierung verwendet werden.
- Anstelle der einfachen Zuordnung von Konzepten zu Dokumenten besteht die Beschreibungssprache aus Statements der Art Subjekt-Prädikat-Objekt.

4.3.4.1 RDF: Grundlegende Konzepte

RDF beinhaltet folgende Grundkonstrukte:

Resource Ressourcen bezeichnen beliebige Objekte im WWW, wie z.B. eine einzelne Web-Seite, oder auch ein umfangreiche Datenbasis. Ressourcen werden in der Regel durch einen Uniform Resource Identifier (URI) identifiziert.

Literal Literale sind spezielle Ausprägungen von Resourcen, die eine Zeichenkette als Wert haben, aber keine explizite URI.

Property Eigenschaften bezeichnen eine gerichtete Beziehung zwischen zwei Ressourcen (Attribut, Relation, Rolle o.ä.).

Statement Aussagen sind ein Tripel bestehend aus Ressource, benannter Eigenschaften und einer weiteren Ressource, die den Wert für die Eigenschaft angibt. Somit entspricht die Syntax einer Aussage der einfacher Sätze der Form Subjekt – Prädikat – Objekt.

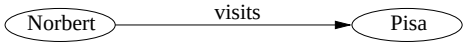


Abbildung 4.9: Eine einfache Aussage in RDF

Diese Konstrukte kann man graphisch darstellen, wobei Ressourcen als Ellipsen, Literale als Rechtecke und Eigenschaften als gerichtete Kanten dargestellt werden (siehe z.B. Abbildung 4.9).

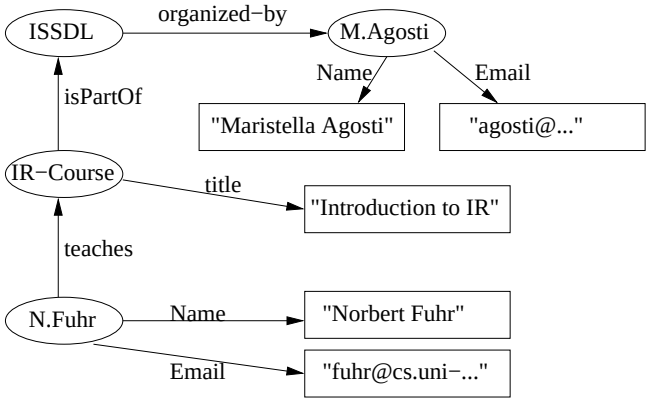


Abbildung 4.10: Ein RDF-Graph

Nehmen verschiedene Aussagen auf die selben Ressourcen Bezug, so erhält man einen RDF-Graphen wie z.B. in Abbildung 4.10. So intuitiv diese Darstellung auch sein mag, so ist doch offensichtlich, dass durch die uneingeschränkte Verwendung von Ressourcen und Eigenschaften das Ziel der Interoperabilität von RDF-Wissensbanken nicht erreicht werden kann. Man benötigt daher Mechanismen zur Deklaration der verwendeten Ressourcen und Eigenschaften, analog zu einem Datenbankschema.

4.3.4.2 RDF Schemas

RDF Schemas sind ähnlich zu denen objektorientierter Datenbanken. Der wesentliche Unterschied besteht darin, dass RDF Schemas nur selten Informationen über Datentypen enthalten, dagegen aber über stärkere Abstraktionsmechanismen verfügen. Sie stehen damit in einer langen KI-Tradition, die ihren Ursprung in den semantischen Netzen (z.B. KL-ONE) hat und über terminologische bzw. Beschreibungs-Logiken schließlich zu RDF geführt hat.

RDF Schemas beschreiben die Beziehungen zwischen Typen von Ressourcen und/oder Eigenschaften. Dabei werden die Schemas selbst auch wieder mittels RDF beschrieben. Hierzu stehen folgende Grundkonstrukte zur Verfügung:

- Grundlegende Klassen:
  - rdfs:Resource bezeichnet die Klasse aller Ressourcen,
  - rdf:Property die Klasse aller Eiegenschaften und
  - rdfs:Class die Klasse aller Klassen.
- Beziehungen zwischen verschiedenen Ressourcen, Eigenschaften oder Klassen können durch folgende Eigenschaften beschrieben werden:
  - rdf:type gibt den Typ eines Konstrukts an.

– rdfs:subClassOf bezeichnet die Teilklassen-Beziehung.
– rdfs:subPropertyOf erlaubt die Spezialisierung von Eigenschaften.
– rdfs:seeAlso beschreibt Querverweise.
– rdfs:isDefinedBy verweist von einer Ressource oder Eigenschaft auf die zugehörige Definition.

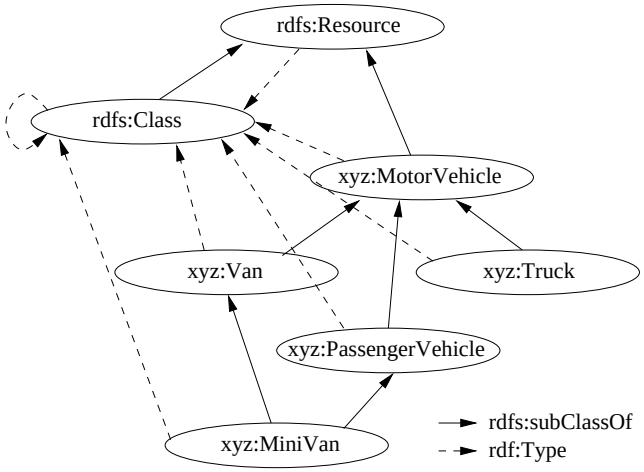


Abbildung 4.11: Eine Hierarchie von RDF-Ressourcen

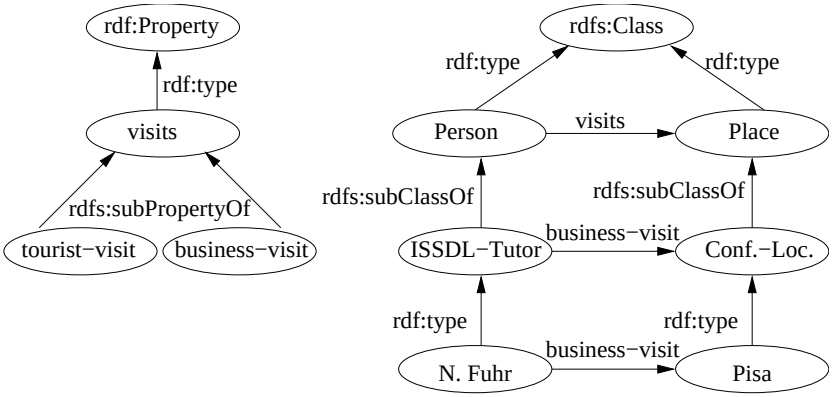


Abbildung 4.12: Hierarchien auf Ressourcen und Eigenschaften

Die Abbildungen 4.3.4.2 und 4.3.4.2 zeigen kleine Beispiele von RDF-Schemata.

4.3.5 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.
– 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.

4.4 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 halbformale 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. Wie auch die Studie [Krause 92] zeigt, gibt es kaum erfolgversprechenden Ansätze in diesem Bereich. Das Hauptproblem ist das Fehlen entsprechender Wissensbasen, die nicht nur sehr umfangreich sein müssen, sondern auch das für die jeweilige Anwendung pragmatische Wissen bereitstellen sollten. Letzteres ist wohl der Hauptgrund, warum selbst so

umfangreiche Wissensbasen wie CYC [Lenat et al. 90] bislang nicht erfolgreich im IR eingesetzt werden konnten.
- 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.

4.5 Zusammenhang zwischen Modellen und Repräsentationen

4.5.1 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.

4.5.2 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 5

Nicht-probabilistische IR-Modelle

5.1 Notationen

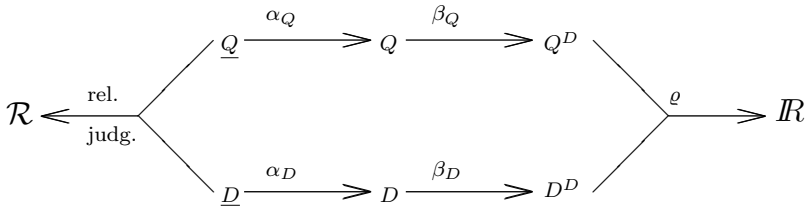


Abbildung 5.1: Konzeptionelles Modell für Textretrieval

Als grundlegendes konzeptionelles Modell für alle Arten von Modellen für (Text-)Retrieval verwenden wir das in Abb. 5.1 dargestellte Modell, das eine Vereinfachung des in Abschnitt 2.3 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 Retrieval 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.

5.2 Überblick über die Modelle

	Bool.	Fuzzy	Vektor	Prob.	Cluster.
theoretische Basis:					
– boolesche Logik	x				
– Fuzzy-Logik		x			
– Vektoralgebra			x		x
– Wahrsch.-Theorie				x	
Bezug zur Retrievalqual.		(x)		x	
gewichtete Indexierung		x	x	x	x
gewichtete Frageterms		(x)	x	x	
Fragestruktur:					
– linear			x	x	
– boolesch	x	x	(x)	(x)	
Suchmodus:					
– Suchen	x	x	x	x	
– Browsen					x

Abbildung 5.2: IR-Modelle

Abbildung 5.2 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.

5.3 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.

$$d_m^D = \vec{d}_m \quad \text{mit} \quad d_{m_i} \epsilon \{0,1\} \quad \text{für} \quad i = 1, \ldots, n \tag{5.1}$$

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.

5.3.1 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.

5.3.2 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 Ordung 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).

5.4 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 Retrieval entfallen. Theoretische Überlegungen wie auch experimentelle Untersuchungen zeigen aber, dass die Definition der Retrievalfunktion

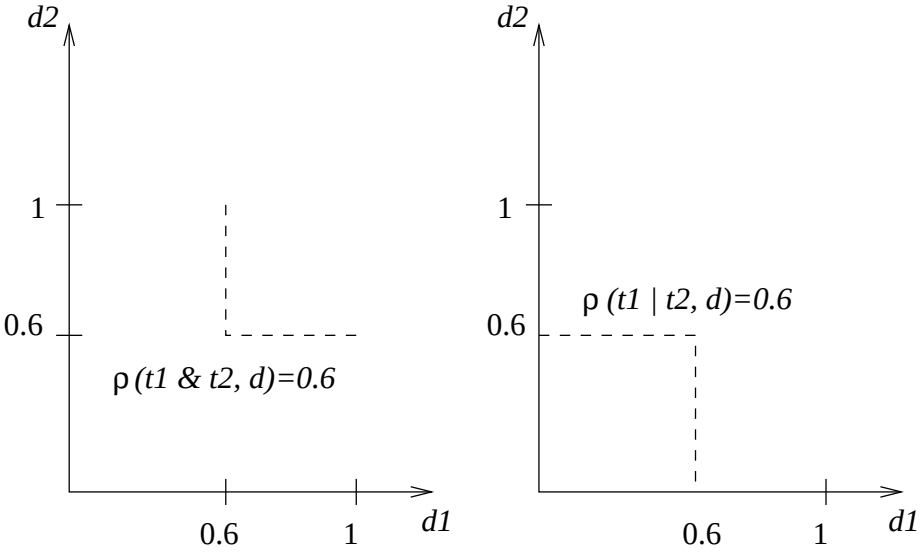


Abbildung 5.3: Punkte mit gleichem Retrievalgewicht beim Fuzzy-Retrieval

ungünstig ist. Wir illustrieren dies zunächst an einem Beispiel:

$$\begin{aligned} T &= \{t_1, t_2\} \\ q &= t_1 \wedge t_2 \\ \vec{d_1} = (0.4, 0.4) \quad &, \quad \vec{d_2} = (0.39, 0.99) \\ \varrho(q, \vec{d_1}) = 0.4 \quad &, \quad \varrho(q, \vec{d_2}) = 0.39 \end{aligned}$$

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. 5.4 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. Offensichtlich wäre es wünschenswert, wenn man zumindest eine teilweise Kompensation der Gewichte für die verschiedenen Terme aus der Anfrage zulassen würde. 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 5.1 zeigt diese Ergebnisse in Form mittlerer Precision-Werte für die Recall-Punkte 0.25, 0.5 und 0.75. (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.)

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 5.1: Experimenteller Vergleich von Booleschem Retrieval, Fuzzy-Retrieval und Vektorraummodel

5.4.1 Beurteilung des Fuzzy-Retrieval

Zusammengefasst bietet Fuzzy-Retrieval folgende Vor- und Nachteile:

+ 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.

5.5 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):

$$d_m^D = \vec{d}_m \quad \text{mit} \quad d_{m_i} \epsilon I\!R \quad \text{für } i = 1, \ldots, n \tag{5.2}$$

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

$$q_k^Q = \vec{q}_k \quad \text{mit} \quad q_{k_i} \epsilon I\!R \quad \text{für } i = 1, \ldots, n \tag{5.3}$$

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

$$\varrho(\vec{q}_k, \vec{d}_m) = \vec{q}_k \cdot \vec{d}_m \tag{5.4}$$

Das folgende Beispiel illustriert die Anwendung des VRM:

Beispiel-Frage: „side effects of drugs on memory and cognitive abilities, not aging“

t_i	q_{k_i}	d_{1_i}	d_{2_i}	d_{3_i}	d_{4_i}
side effect	2	1	0.5	1	1
drugs	2	1	1	1	1
memory	1	1		1	
cognitive ability	1		1	1	0.5
¬ aging	-2		1		
Retrievalgewicht		5	2	6	4.5

Entsprechend den Retrievalgewichten werden die Dokumente in der Reihenfolge d_3, d_1, d_4, d_2 ausgegeben.

5.5.1 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:

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

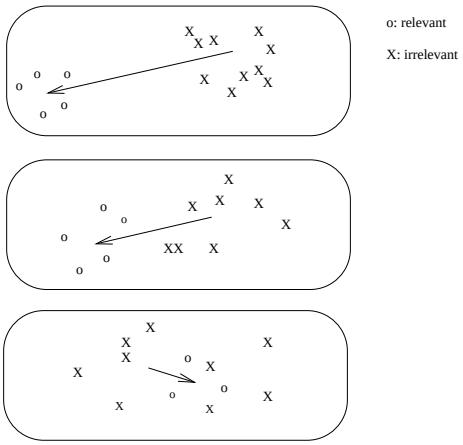


Abbildung 5.4: Trennung von relevanten und nichtrelevanten Dokumenten im VRM

5.5.2 Relevance Feedback

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 Frageterm-gewichte verändert, wodurch sich ein anderer Fragevektor ergibt. Abb. 5.4 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:

$$\sum_{(d_k,d_l)\in D^R\times D^N} \vec{q}\vec{d_k} - \vec{q}\vec{d_l} \stackrel{!}{=} \max \tag{5.5}$$

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

$$\sum_{i=1}^{n} q_i^2 = c \tag{5.6}$$

Somit liegt ein Extremwertproblem mit Randbedingung vor, das man mit

Hilfe eines Lagrange-Multiplikators lösen kann:

$$F = \lambda \left(\sum_{i=1}^{n} q_i^2 - c \right) + \sum_{(d_k,d_l) \in D^R \times D^N} \sum_{i=1}^{n} q_i d_{k_i} - q_i d_{l_i} \tag{5.7}$$

Zur Lösung muss man nun alle partiellen Ableitungen von F nach den Komponenten q_i des Fragevektors 0 setzen; zusätzlich muss auch die Nebenbedingung 5.6 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

$$\vec{q} = \frac{1}{|D^R|} \sum_{d_k \in D^R} \vec{d_k} - \frac{1}{|D^N|} \sum_{d_l \in D^N} \vec{d_l} \tag{5.8}$$

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

Abbildung 5.5.2 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 5.5.2 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

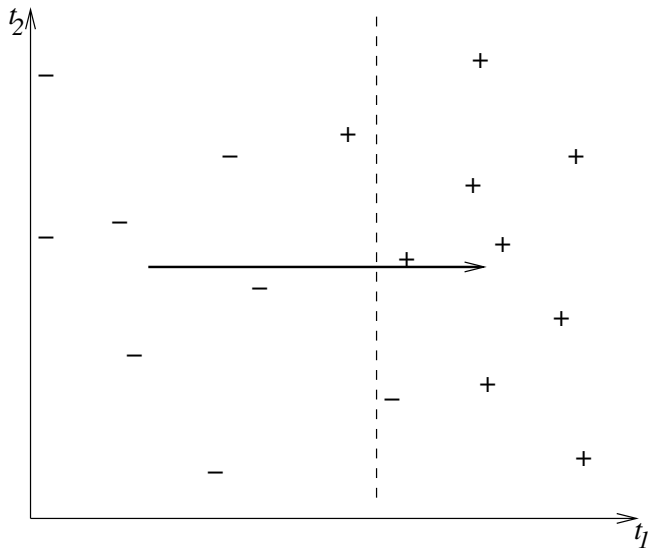


Abbildung 5.5: Optimaler Fragevektor als Verbindungsvektor der Zentroiden

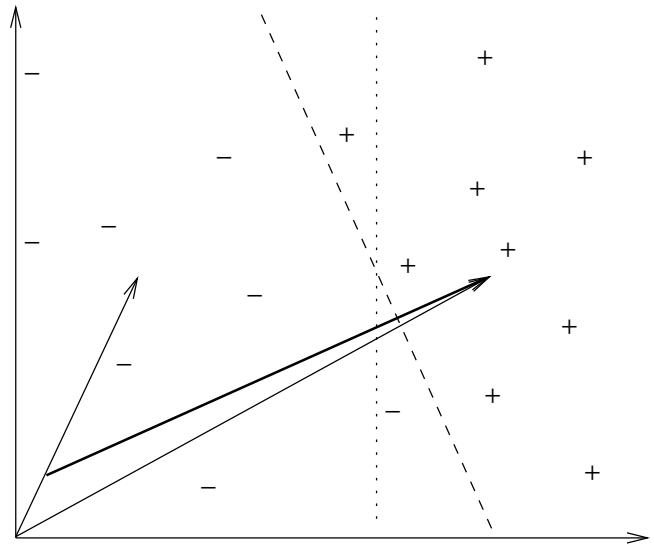


Abbildung 5.6: Unterschiedliche Gewichtung positiver und negativer Beispiele

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 5.2: Experimentelle Ergebnisse zu Relevance Feedback

Hilfe der Relevance-Feedback-Daten zu verbessern. Prinzipiell ergibt sich also folgende Vorgehensweise:

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:

$$\vec{q}' = \vec{q} + \alpha \frac{1}{|D^R|} \sum_{d_j \epsilon D^R} \vec{d}_j - \beta \frac{1}{|D^N|} \sum_{d_j \epsilon D^N} \vec{d}_j \tag{5.9}$$

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

Tabelle 5.2 zeigt experimentelle Ergebnisse, die durch Anwendung der Formel 5.9 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 5.9 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 5.5 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).

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 5.3: Binäre Gewichte vs. SMART-Gewichtung

5.5.3 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.
$|\underline{D}|$: # 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:

$$idf_i = \frac{\log \frac{|\underline{D}|}{n_i}}{|\underline{D}| + 1} \tag{5.10}$$

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 durchschnittlicehn Dokumentlänge in der Kollektion:

$$ntf_i = \frac{tf_{mi}}{tf_{mi} + 0.5 + 1.5\frac{l_m}{al}} \tag{5.11}$$

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

$$w_{mi} = ntf_i \cdot idf_i \tag{5.12}$$

.

Tabelle 5.3 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 5.10–5.12 sowohl zur Dokumentindexierung als auch zur Bestimmung des Fragevektors angewendet.

5.5.4 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.

5.6 Dokumenten-Clustering

Eine von allen anderen Retrievalmodellen gänzlich unterschiedliche Retrievalmethode ist das Cluster-Retrieval. Während andere Retrievalmodelle stets von einer expliziten Frageformulierung ausgehen, nutzt man beim Cluster-Retrieval hauptsächlich die Ähnlichkeit von Dokumenten, um von einem relevanten Dokument zu weiteren (potentiell) relevanten zu gelangen. Diese Art der Suche wird durch die vorherige Bestimmung von (Dokumenten-)Clustern, also Mengen von ähnlichen Dokumenten, unterstützt.

Ausgangspunkt für diese Art der Suche ist die sogenannte „Cluster-Hypothese“: Man kann nämlich zeigen, dass die Ähnlichkeit der relevanten Dokumente untereinander und der irrelevanten Dokumente untereinander größer ist als die zwischen anderen (zufälligen) Teilmengen der Dokumentenkollektion. Diese Hypothese wurde auch experimentell in [Rijsbergen & Jones 73] nachgewiesen. Beim Dokumenten-Clustering wird versucht, diese Cluster unabhängig von den Fragen schon beim Aufbau der Kollektion zu berechnen. Dabei geht man prinzipiell wie folgt vor:

1. Festlegung eines Ähnlichkeitsmaßes (z.B. Skalarprodukt oder Cosinus-Maß) .
2. Berechnung der Ähnlichkeitmatrix für alle möglichen Dokumentenpaare aus $|\underline{D}|$.
3. Berechnung der Cluster.

4. Physisch gemeinsame Abspeicherung der Dokumente eines Clusters. (Durch diese Form der Speicherung werden I/O-Zugriffe beim Retrieval gespart.)

Zur Berechnung der Cluster aus der Ähnlichkeitsmatrix gibt es eine ganze Reihe von Cluster-Algorithmen (siehe die einschlägige Literatur). Prinzipiell gibt es zwei wesentliche Strategien: agglomeratives Clustering und partitionierendes Clustering.

Agglomeratives Clustering geht von einem vorgegebenen Schwellenwert α für die Ähnlichkeit aus. Es erfolgt ein einmaliger Durchgang durch alle Dokumente mit dem Ziel, diese zu (anderen) Clustern hinzuzufügen. Entsprechend der jeweiligen Bedingung für die Aufnahme eines Dokumentes d_k in ein Cluster C_l kan man unter anderem folgende Verfahren unterscheiden:

a) single link-Clustering:

$$\alpha \geq \arg \min_{d_i \in C_l} sim(d_k, d_i)$$

b) complete link-Clustering:

$$\alpha \geq \arg \max_{d_i \in C_l} sim(d_k, d_i)$$

c) average link-Clustering:

$$\alpha \geq \frac{1}{|C_l|} \sum_{d_i \in C_l} sim(d_k, d_i)$$

Gibt es kein solches Cluster, wird für d_k ein neues Cluster gebildet. Der Aufwand für all diese Verfahren beträgt (aufgrund der vollständigen Berechnung der Ähnlichkeitsmatrix) $O(n^2)$.

Partitionierendes Clustering begnügt sich dagegen mit einem Aufwand von $O(kn)$. Dabei ist k die vorzugebende Anzahl zu bildender Cluster. Zu Beginn bestimmt das Verfahren k "seed"-Dokumente, die hinreichend unterschiedlich sind. Diese bilden jeweils den Kern eines der Cluster $C_1, \ldots, C_k$. Alle übrigen Dokumente werden dann jeweils dem ähnlichsten Cluster hinzugefügt. Die Qualität dieses Verfahrens hängt stark von der Wahl der seed-Dokumente ab, wofür es wiederum verschiedene Strategien gibt.

Mit beiden Arten von Clustering-Verfahren kann hierarchisches Clustering realisiert werden, das als Ergebnis eine baumförmige Clustering-Struktur liefert. Hierzu werden die o.g. Clustering-Verfahren iterativ angewendet, um die in der vorangegangenen Stufe gebildeten Cluster weiter zu zerlegen.

Die so berechneten Cluster können auf zwei Arten genutzt werden:

1. Zur Suche ähnlicher Dokumente zu einem bereits bekannten relevanten Dokument.
2. Wenn man noch kein relevantes Dokument kennt, wird zunächst Cluster-Retrieval durchgeführt. Hierbei wird ein anderes Retrievalmodell (üblicherweise VRM) angewendet, um Cluster mit potentiell relevanten Dokumenten zu lokalisieren.

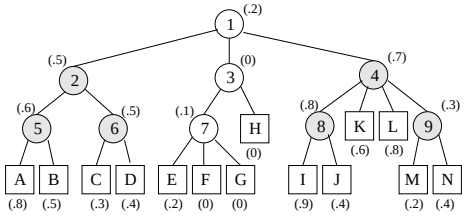


Abbildung 5.7: Beispiel zum Cluster-Retrieval

5.6.1 Cluster-Retrieval

Beim Cluster-Retrieval wird bei der Berechnung der Cluster zu jedem Cluster ein Zentroid berechnet. Dieser Zentroid ist ein virtuelles Dokument mit minimalem Abstand zu allen Dokumenten des Clusters. Alle Zentroiden werden gemeinsam abgespeichert, und zwar getrennt von den eigentlichen Clustern. Dadurch erspart man sich I/O-Zugriffe beim Durchsuchen der Zentroiden.

Beim eigentlichen Retrieval geht man dann wie folgt vor:

1. Bestimmung der Zentroiden mit den höchsten Retrievalgewichten.
2. Ranking der Dokumente in den zugehörigen Clustern.

Abbildung 5.7 illustriert diese Vorgehensweise. Die Zentroiden sind als Kreise dargestellt, die Dokumente als Quadrate. Bei Dokumenten und Zentroiden ist das Retrievalgewicht jeweils in Klammern notiert. Nur die Cluster zu den grau markierten Zentroiden werden in das abschließende Ranking einbezogen.

Insgesamt ergibt sich folgende Beurteilung für das Cluster-Retrieval:

+ Die Abhängigkeiten zwischen Dokumenten werden berücksichtigt. Fast alle anderen IR-Modelle nehmen dagegen die Dokumente als voneinander unabhängig an, was natürlich in der Realität nicht stimmt.
+ Im Vergleich zu anderen Retrievalverfahren reduziert sich der I/O-Aufwand.
– Soweit experimentelle Ergebnisse zum Cluster-Retrieval vorliegen, zeigen diese eine deutlich schlechtere Retrievalqualität im Vergleich zu anderen Verfahren.

5.6.2 Ähnlichkeitssuche von Dokumenten

Wenn bereits ein relevantes Dokument bekannt ist, dann können die Dokumenten-Cluster zur Ähnlichkeitssuche ausgenutzt werden. Dies ist bei realen Benutzungen von IR-Systemen häufig der Fall. Statt nun eine entsprechende Anfrage zu formulieren, kann man dann eine Ähnlichkeitssuche durchführen. (Heutige kommerzielle IR-Systeme bieten diese Funktion leider nicht.) Auch wenn keine Dokumenten-Cluster vorliegen, kann man natürlich nach ähnlichen Dokumenten suchen: Beim Vektorraummodell fasst man dann einfach den Dokumentvektor des bekannten Dokumentes als Fragevektor auf und führt damit Retrieval durch. Vergleicht man nun den Berechnungsaufwand für diese beiden Vorgehensweisen, so zeigt sich, dass eine Vorprozessierung der Cluster nicht lohnt [Willett 88]. Bezüglich der resultierenden Retrievalqualität gibt es zwar kaum aussagekräftige Resultate; man ist aber allgemein der Auffassung, dass

die Ähnlichkeitssuche sinnvoll ist als Ergänzung zu den anderen Retrievalmodellen. Es zeigt sich nämlich, dass hiermit andere relevante Dokumente gefunden werden.

5.6.3 Cluster-Browsing

Wenn Clustering für normales Retrieval gemäß dem oben gesagten wenig Vorteile bietet, so ermöglicht es aber eine alternative Suchstrategie: Während sonst beim Retrieval zunächst der Benutzer eine möglichst konkrete Anfrage formulieren muss, kann man bei einem geclusterten Dokumentenbestand auch einfach browsen. Eine einfache Möglichkeit hierzu bietet hierarchisches Clustern (wie in Abb. 5.7), wo der Benutzer ausgehend von der Wurzel Pfade zu den einzelnen Dokumenten verfolgt. Für jedes Cluster müssen dann geeignete Repräsentanten angezeigt werden (im Gegensatz zum Cluster-Retrieval wären die berechneten Zentroiden für den Benutzer wenig hilfreich), z.B. indem man das dem Zentroiden ähnlichste Dokument des Clusters nimmt. Dem Benutzer werden dann jeweils die Repräsentanten der Teilcluster des vorher ausgewählten Clusters angezeigt.

5.6.4 Scatter/Gather-Browsing

Eine andere mögliche Anwendung des Clustering wird in [Cutting et al. 92] vorgeschlagen. Die wesentliche Idee hierbei ist, dass Cluster nicht statisch berechnet werden, sondern dynamisch während der interaktiven Suche. Jeder Suchschritt besteht dabei aus zwei Phasen, einer Scatter-Phase und einer Gather-Phase. In der Scatter-Phase wird die Dokumentmenge in eine vorgegebene Anzahl von Clustern zerlegt (mittels partitionierendem Clustering) und dem Benutzer Repräsentanten für die resultierenden Cluster angezeigt. Diese Repräsentanrten bestehen dabei einerseits aus Titeln von Dokumenten in der Nähe des Cluster-Zentroiden, andererseits aus häufig im Cluster vorkommenden Wörtern.

In der Gather-Phase wählt der Benutzer einen oder mehrere ihn interessierende Cluster aus, die dann zusammengemischt die Ausgangsmenge für den nächsten Suchschritt bilden. Abbildung 5.8 zeigt eine Folge von drei Suchschritten. Scatter/Gather-Browsing kann somit als Kombination von Browsing in (statischen) Clustern und Relevance Feedback aufgefasst werden.

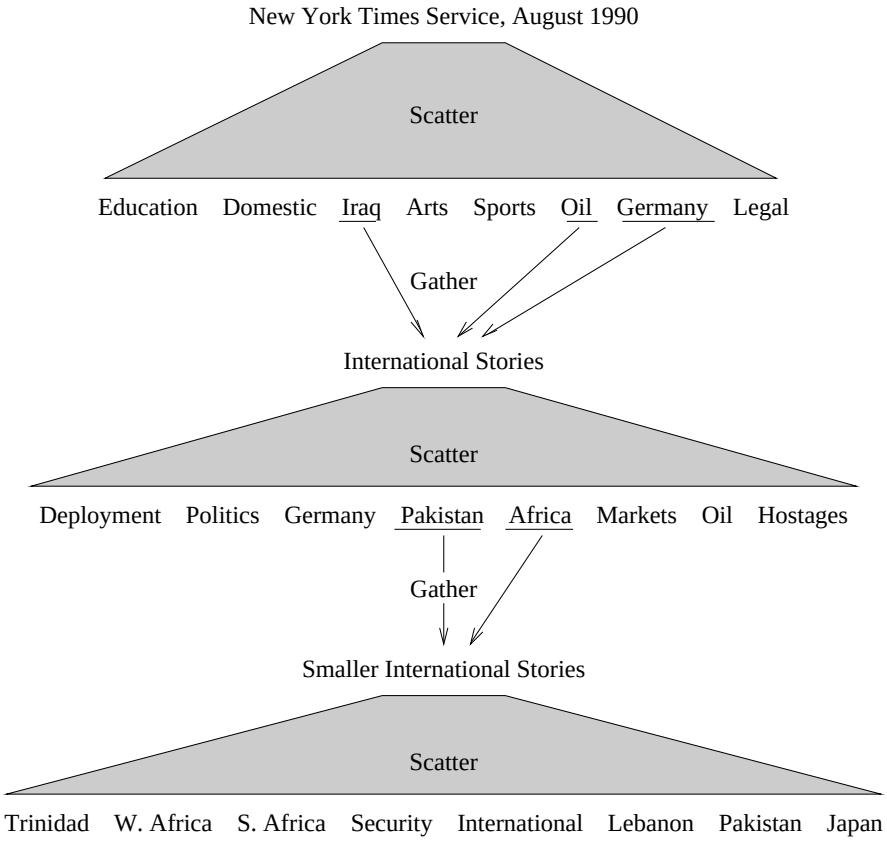


Abbildung 5.8: Beispiel zu Scatter/Gather

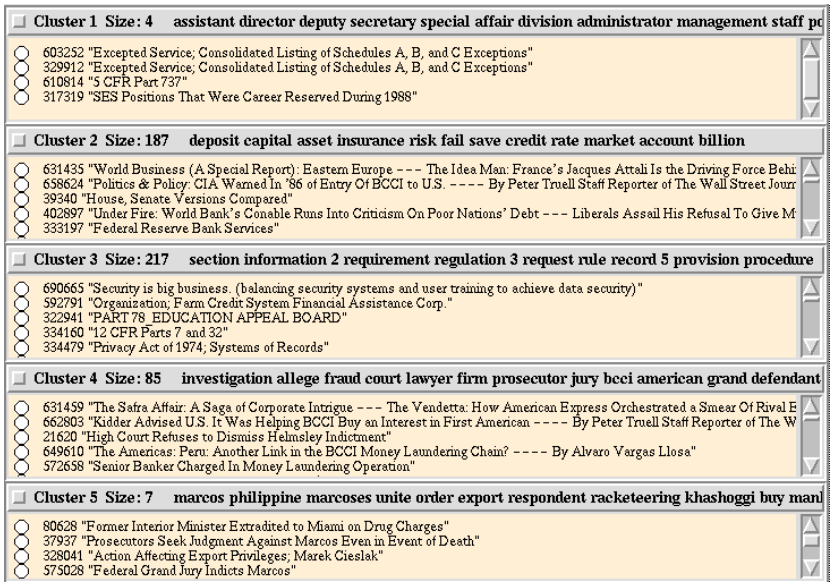


Abbildung 5.9: Scatter/Gather: Bildschirmausgabe

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.

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.

Cutting, D. R.; Pedersen, J. O.; Karger, D.; Tukey, J. W. (1992). Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections. In: *Proceedings of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, S. 318–329. ACM, New York. http://citeseer.nj.nec.com/cutting92scattergather.html.

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 [SIG93], S. 291–297.

Lenat, D. B.; Guha, R. V.; Pittman, K. (1990). Cyc: Toward Programs With Common Sense. *Communications of the ACM 33(8)*, S. 30–49.

Meghini, C.; Rabitti, F.; Thanos, C. (1991). Conceptual Modeling of Multimedia Documents. *IEEE Computer 24(10)*, S. 23–30.

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.; Jones, K. S. (1973). A Test for the Separation of Relevant and Non-relevant Documents in Experimental Retrieval Collections. *Journal of Documentation 29*, S. 251–257.

Rijsbergen, C. J. (2001). Getting into Information Retrieval. In: Agosti, M.; Crestani, F.; Pasi, G. (Hrsg.): *Lectures in Information Retrieval*, S. 1–20. Springer, Heidelberg et al.

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.

(1993). *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York. ACM.

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.

Willett, P. (1988). Recent Trends in Hierarchic Document Clustering: A Critical Review. *Information Processing and Management 24(5)*, S. 577–597.

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.