Inhaltsverzeichnis

[1 Einführung 5](#_Toc537948)

[1.1 Mögliche Dienste einer DL 5](#_Toc537949)

[1.1.1 Mögliche weitere Aufgaben einer DL 5](#_Toc537950)

[1.2 Digitalisierung von Papierdokumenten 5](#_Toc537951)

[1.3 Beispiele einer DL 5](#_Toc537952)

[1.3.1 Unterschiede der DL 5](#_Toc537953)

[1.3.2 Deutsche Digitale Bibliothek 5](#_Toc537954)

[1.3.3 Wissenschaftliche Bibliotheken in DE 5](#_Toc537955)

[1.4 Vorteiler einer DL 6](#_Toc537956)

[2 Wissenschaftliches Publizieren 6](#_Toc537957)

[2.1 Publikationshierarchie in der Informatik 6](#_Toc537958)

[2.2 Qualitätssicherung: Peer Review 6](#_Toc537959)

[2.3 Typischer Ablauf für Konferenzen 6](#_Toc537960)

[2.4 Wichtige Fachgesellschaften 6](#_Toc537961)

[2.5 Wichtige Verlage 6](#_Toc537962)

[2.6 Probleme des traditionellen Systems 7](#_Toc537963)

[2.7 Weiteres Problem: Aktualität 7](#_Toc537964)

[2.8 Warum überhaupt Zeitschriften 7](#_Toc537965)

[2.9 Wie man gute Journals/Konferenzen erkennt 8](#_Toc537966)

[2.10 Digital Object Identifiers 8](#_Toc537967)

[2.11 Personen-Identifier 8](#_Toc537968)

[2.12 Mögliche Features zur Autordisambiguierung 8](#_Toc537969)

[2.13 Bewertung von Publikationen 8](#_Toc537970)

[2.14 Bewertung von Journals 9](#_Toc537971)

[2.15 Bewertung von Autoren 9](#_Toc537972)

[3 Einführung in Information Retrieval 10](#_Toc537973)

[3.1 Retrieval-Szenarien 10](#_Toc537974)

[3.2 Herausforderung an IR-Systeme 10](#_Toc537975)

[3.3 Begriffsbildung 10](#_Toc537976)

[3.4 Percision / Recall 10](#_Toc537977)

[4 Boolesches Retrieval – Anfragen und einfache Datenstrukturen 11](#_Toc537978)

[4.1 Dokumente 11](#_Toc537979)

[4.2 Terme 11](#_Toc537980)

[4.3 Informationsbedarf und Ad-hoc-Anfragen 11](#_Toc537981)

[4.4 Term-Dokument Inzidenzmatrix 11](#_Toc537982)

[4.5 Invertierter Index 12](#_Toc537983)

[5 Boolesches Retrieval – Vorverarbeitung von Dokumenten und Indexierung 12](#_Toc537984)

[5.1 Indexierung 12](#_Toc537985)

[5.2 Konstruktion eines invertierten Indexes 12](#_Toc537986)

[5.3 Token und Terme 12](#_Toc537987)

[5.4 Tokenisierung und Normalisierung 13](#_Toc537988)

[5.5 Vorverarbeitung 13](#_Toc537989)

[5.5.1 Normalisierung 13](#_Toc537990)

[5.5.2 Reduktion auf Grundformen 14](#_Toc537991)

[5.5.3 Thesauri 14](#_Toc537992)

[5.5.4 Stoppwörter 14](#_Toc537993)

[6 Boolesches Retrieval – Weitere wichtige Retrievaloperatoren 15](#_Toc537994)

[6.1 Phrasenanfragen 15](#_Toc537995)

[6.2 Positionsindexe 15](#_Toc537996)

[6.3 Proximity-Queries (Nachbarschaftsanfragen) 15](#_Toc537997)

[6.4 Rechtschreibkorrektur 15](#_Toc537998)

[6.4.1 Korrektur isolierter Terme 16](#_Toc537999)

[6.4.2 Kontext-sensitive Rechtschreibkorrektur 16](#_Toc538000)

[6.5 Schwachpunkte des Booleschen IR-Modells 16](#_Toc538001)

[7 Retrievalmodelle – Das Vektorraum-Modell 17](#_Toc538002)

[7.1 Sichten auf ein Dokument 17](#_Toc538003)

[7.2 Modelle 17](#_Toc538004)

[7.3 Taxonomie von Retrieval-Modellen 17](#_Toc538005)

[7.4 Klassisches Retrieval-Modell 18](#_Toc538006)

[7.5 Boolesches Modell 18](#_Toc538007)

[7.6 Vektorraummodell 19](#_Toc538008)

[7.6.1 Termhäufigkeit 19](#_Toc538009)

[7.6.2 Termgewichte 19](#_Toc538010)

[7.6.3 Term-Dokument Häufigkeitsmatrix 19](#_Toc538011)

[7.6.4 Bag-of-Words-Modell 19](#_Toc538012)

[7.6.5 Absolute Termhäufigkeit 20](#_Toc538013)

[7.6.6 Dokumentenhäufigkeit 20](#_Toc538014)

[7.6.7 Inverse Dokumenthäufigkeit 20](#_Toc538015)

[7.6.8 Kollektionshäufigkeit 20](#_Toc538016)

[7.6.9 tf•idf-Gewichtung 20](#_Toc538017)

[7.6.10 Dokumentvektoren 20](#_Toc538018)

[7.6.11 Anfragen als Vektoren 21](#_Toc538019)

[8 Retrievalmodelle – Probabilistische Modelle 21](#_Toc538020)

[8.1 Probability Ranking Principle 21](#_Toc538021)

[8.2 Binary Independece-Model 21](#_Toc538022)

[8.3 Okapi BM25 21](#_Toc538023)

[9 Retrievalmodelle – Generative Sprachmodelle 22](#_Toc538024)

[9.1 Statistische Sprachmodelle 22](#_Toc538025)

[9.2 Sprachmodelle im Information Retrieval (Query-Likelihood-Modell) 22](#_Toc538026)

[9.3 Abschließende Bemerkungen zu Sprachmodellen 22](#_Toc538027)

[10 Retrievalmodelle – Algebraische Modelle 23](#_Toc538028)

[10.1 Idee 23](#_Toc538029)

[10.2 Vorteile 23](#_Toc538030)

[10.3 Nachteile 23](#_Toc538031)

[11 Retrievalmodelle – Kombination mehrere Modelle 23](#_Toc538032)

[12 Evaluation von IR-Systemen 24](#_Toc538033)

[12.1 Poolbildung 24](#_Toc538034)

[12.2 Anfragelogs 24](#_Toc538035)

[12.3 False Negatives und False Positives 24](#_Toc538036)

[12.4 Percision und Recall 25](#_Toc538037)

[12.4.1 F-Maß 25](#_Toc538038)

[12.4.2 Optionen zum Zusammenfassen eines Rankings 25](#_Toc538039)

[12.4.3 Benutzermodell für AP 25](#_Toc538040)

[12.4.4 Durchschnittsbildung 26](#_Toc538041)

[12.4.5 Interpolation 26](#_Toc538042)

[12.5 Konzentration auf die Top-Dokumente 26](#_Toc538043)

[12.6 Discounted Cumulative Gain (DCG) 26](#_Toc538044)

[12.7 Normalisierter DCG 26](#_Toc538045)

[12.8 BPREF 27](#_Toc538046)

[12.9 Effizienzmaße 27](#_Toc538047)

[13 Evaluierung von IR-Systemen – Tuning von Parametern 27](#_Toc538048)

[13.1 Online-Tests 27](#_Toc538049)

[13.2 Zusammenfassung 28](#_Toc538050)

[14 Websuchmaschinen 29](#_Toc538051)

[14.1 Ansätze für die Informationsfindung 29](#_Toc538052)

[14.2 Herausforderung an Websuchmaschinen 29](#_Toc538053)

[14.3 Crawler 30](#_Toc538054)

[14.3.1 Anforderungen an Crawler 30](#_Toc538055)

[14.3.2 Empfehlungen für Crawler 30](#_Toc538056)

[14.3.3 Aktualisieren von Webseiten 30](#_Toc538057)

[14.4 Indexer 30](#_Toc538058)

[14.5 Searcher 30](#_Toc538059)

[14.6 Google-Crawler 31](#_Toc538060)

[14.7 Google-Indexer 31](#_Toc538061)

[15 Websuchmaschinen – Ranking mit Pagerank 31](#_Toc538062)

[15.1 Webmodell 31](#_Toc538063)

[15.2 Übergangsmatrix A 31](#_Toc538064)

[15.3 Vereinfachter PageRank 32](#_Toc538065)

[15.4 Rangsenken 32](#_Toc538066)

[15.5 Teleport-Operation 32](#_Toc538067)

[15.6 Normaler PageRank 32](#_Toc538068)

[15.7 Suche mit PageRank 32](#_Toc538069)

[16 Websuchmaschinen – Ranking mit HITS 33](#_Toc538070)

[16.1 Adjazenzmatrix 33](#_Toc538071)

[16.2 Authorities und Hubs 33](#_Toc538072)

[16.3 HITS (Hyperlink-Induced Topic Search) 33](#_Toc538073)

[16.3.1 Suche mit HITS 33](#_Toc538074)

[16.4 Vergleich PageRank – HITS 34](#_Toc538075)

[16.4.1 PageRank 34](#_Toc538076)

[16.4.2 Hits 34](#_Toc538077)

[17 Personalisierung 35](#_Toc538078)

[17.1 Ziel: Auflösen der inhärenten Ambiguität von Suche 35](#_Toc538079)

[17.2 Dimensionen von Personalisierter Suche 35](#_Toc538080)

[17.3 Einfache Personalisierung: Relevance Feedback 35](#_Toc538081)

[17.3.1 Implizites Feedback durch Clicks 35](#_Toc538082)

[**17.4** Einfacher Einsatz von Feedback: Promoting 36](#_Toc538083)

[17.5 Benutzerprofile 36](#_Toc538084)

[17.6 Persistente vs. Sitzungsprofile 36](#_Toc538085)

[17.7 Personalisierung mit Benutzerprofilen 36](#_Toc538086)

[17.8 Probleme beim Reranking: Ähnliche Ergebnisse 36](#_Toc538087)

[17.9 Diversifizierungsansatz 36](#_Toc538088)

[18 Personalisierung – Empfehlungen 37](#_Toc538089)

[18.1 Drei orthogonale Ansätze 37](#_Toc538090)

[18.1.1 Kollaboratives Filtern 37](#_Toc538091)

[18.2 Content-Based Filtering 37](#_Toc538092)

[18.3 Offline-Evaluation vs. Benutzerexpermimente 37](#_Toc538093)

[18.4 Probleme der Personalisierung 37](#_Toc538094)

# Einführung

## Mögliche Dienste einer DL

* Suche einer bestimmten Publikation
* Suche nach ähnlichen Publikationen
* Suche nach „guten“ Publikationen zu einem Thema

### Mögliche weitere Aufgaben einer DL

* Erschließung von Dokumentbeständen
* Digitalisierung bestehender Dokumentbestände
* Langzeitarchivierung von Dokumentbeständen

## Digitalisierung von Papierdokumenten

* Scannen
* Schrifterkennung (OCR)
  + Buchstabenerkennung: schwierig für „exotische“ Schriftarten
  + Worterkennung: Lexikonbasiert, schwierig für Spezialvokabular

## Beispiele einer DL

* ACM Digital Library
* IEEE Xplore
* DBLP
* Springer Link
* Google Scholar

### Unterschiede der DL

* **Abdeckung** der Publikationen
  + Fokus auf einen Verlag
  + Fokus auf „wichtige“ Publikationen
  + Fokus auf online verfügbare
* Zugriffsrechte
* **Volltext** vs. **Verweis** zur Online-Publikation
* Mächtigkeit des **Suchinterfaces**
* Aufbereitung der Metadaten, Mehrwertdienste
  + Zitate ein- und ausgehend
  + Bibliometrische Maße

### Deutsche Digitale Bibliothek

* Zentrales **nationales Zugangsportal** für Kultur und Wissenschaft in Deutschland
* Verlinkt die digitalen Angebote der deutschen Kultur- und Wissenschaftseinrichtungen miteinander
* Fördert **Aufbau von Kooperationen** und die Entwicklung und gemeinsame Nutzung von Diensten und neuartigen Werkzeugen

### Wissenschaftliche Bibliotheken in DE

* Wissenschaftliche Spezialbibliotheken
* Regionalbibliotheken
* Universitätsbibliotheken
* Hochschulbibliotheken
* Nationalbibliotheken
* Zentrale Fachbibliotheken
* Fachinfromationsdienste

## Vorteiler einer DL

* DL bringt die Bibliothek zum Benutzer.
* Informationen können ausgetauscht werden
* Informationen sind einfacher zu halten, um auf dem neuesten Stand zu bleiben.

# Wissenschaftliches Publizieren

## Publikationshierarchie in der Informatik

* **Workshops:** Publikation erster Ideen und Ergebnisse ~6 Seiten, informell
* **Konferenzen**: Publikationen aktueller Forschungsergebnisse ­~12 Seiten
* **Zeitschriften**: Erweiterte Fassung von Konferenzbeiträgen ~10-40 Seiten

## Qualitätssicherung: Peer Review

* Fachkompetente **Gutachter** erstellen **Gutachten** über Einreichungen
  + Auswahl der Gutachter durch Editor, unabhängig von Autoren
  + Empfehlung zu Annahme, Überarbeitung oder Ablehnung

## Typischer Ablauf für Konferenzen

1. Call for Papers durch **Organisatoren**
2. Einreichung von fertigen formatierten Beiträgen durch **Autoren**
3. Begutachtung durch **Wissenschaftler**, gesteuert durch **Organisatoren**
4. Zusammenstellung des Tagungsbands durch **Organisatoren**
5. Veröffentlichung
6. Zusammenstellung des Programms durch **Organisatoren**
7. Registrierung für Konferenz durch **Autoren**
8. Vortrag etc. bei Konferenz durch **Autoren**

* Ablauf für Workshops analog, für Zeitschriften bis Schritt 5

## Wichtige Fachgesellschaften

* ACM
* IEEE
* GI (Gesellschaft für Informatik)

## Wichtige Verlage

* Springer
* Elsevier
* MIT Press

## Probleme des traditionellen Systems

* Anzahl der wissenschaftlichen Arbeiten wächst exponentiell
* Gründe:
  + Weltweit **mehr** **Forscher** (Afrika, Asien)
  + **Beurteilung** hängt praktisch immer von Publikationen ab (Beförderung)
  + Oft zählt **Anzahl**, nicht Qualität
* **Preissteigerung** bei Abos wissenschaftlicher Zeitschriften liegt erheblich über der Inflationsrate
  + Größerer Umfang
  + Höherer Seitenpreis
* Etat von Bibliotheken wächst **nur sehr langsam**
  + Abos zu kündigen oder
  + Größeren Anteil des Beschaffungsetats für Zeitschriften auszugeben
* Anzahl der publizierten Zeitschriften wächst
* Neue Zeitschriften haben es sehr schwer
* Konsequenzen
  + Sehr teure Zeitschriften
  + Sehr geringe Auflage und Verfügbarkeit
* **Aktuelles Verfahren**: Lizenzverhandlung auf Ebene großer Konsortien von Bibliotheken/Staatsebene

## Weiteres Problem: Aktualität

* Artikel veröffentlicht, die mehrere Jahre alt sind
  + Langwieriger Begutachtungsprozess
  + Veröffentlichungsstau (backlog), begrenzte Seitenzahl
* Elektronische Zeitschriften
  + Billiger, da kein Druck und keine Lieferung
  + Keine Begrenzung der Seitenzahl
  + Aber: Begutachtung bleibt Engpass
* Nachteil elektronischer Zeitungen
  + Geringere Ansehen (zählen oft nicht als Veröffentlichung)
  + Archivierung (Stabilität von URLs)
  + Henne-Ei-Problem: Werder Verlage, noch Wissenschaftler wollen Wechsel
  + Hybride Zeitschriften
    - Open Access:
      * **Goldener Weg**: Artikel grundsätzlich frei online zugänglich
      * **Grüner Weg**: Artikel auf eigener Homepage
      * **Grauer Weg**: Vorabversionen ohne Peer Review

## Warum überhaupt Zeitschriften

* **Langfristige Archivierung** wissenschaftlicher Ergebnisse
* **Dokumentation**: Wer war der Entdecker?
* Zeitschriftenveröffentlichungen als primäres Kriterium für die **wissenschaftliche Qualifikation**

## Wie man gute Journals/Konferenzen erkennt

* Wer hat da schon mal publiziert?
* Wer sitzt im Editorial Board/ im Programmkomitee?
* Gibt es eine bekannte Trägerorganisation (ACM, IEEE, etc?)
* Achtung: Angaben werden oft gefälscht

## Digital Object Identifiers

* Problem: **Stabile** **Referenzierung** von Online-Objekten, z.B. Publikationen, aber auch Datensätzen, Software, etc.
* Muss unabhängig von Umbaumaßnahmen auf dem Server oder gar Serverumzügen sein
* Lösung: **Digital Object Identifier (DOI)** als eindeutige URI mit festgelegter Struktur zusammen mit **Relokationsdienst** (zb. <http://doi.org>)

## Personen-Identifier

* Eindeutige Identifikation ist auch für Personen nützlich
* **Aktuelle Entwicklung**: Spezielle ID-Dienste für wissenschaftliche Autoren

## Mögliche Features zur Autordisambiguierung

Ähnlichkeiten von zwei (Menge von) Publikationen mit ähnlichen Autornamen kann abhängen von

* Ähnlichkeit der Autornamen
* Ähnlichkeit der Autor-ID
* Ähnlichkeit der Publikationstitel
* Ähnlichkeit der Publikationsorte
* Ähnlichkeit der Publikationszeiten
* Ähnlichkeiten der Co-Autoren

## Bewertung von Publikationen

* Ideales Maß: **Wissenschaftlicher Beitrag**, Nützlichkeit, Einfluss, …
* Approximatives Maß: **Zitationshäufigkeit**
* Aber die Zitierhäufigkeit alleine ist nicht ausreichend
  + Anzahl Publikationen insgesamt, Alter einer Publikation
  + Rolle des Zitats: Weiterverwendung vs. Erwähnung vs. Widerlegung vs. Selbstzitat

## Bewertung von Journals

* Journal Impact Factor JIF
  + Durchschnittliche Anzahl von Publikationen, die Artikel aus den letzten zwei Jahren in diesem Jahr erhalten haben
* Eigenfactor Ranking EF
  + Ein Journal ist das gut, wenn seine Artikel oft von Artikeln in anderen guten Journalen zitiert werden
* Manuelle Rankings

## Bewertung von Autoren

* Anzahl von Publikationen
* Anzahl von Zitaten
* Hirsch-Index (h-Index)
  + Größte Zahl h, so dass mindestens h Publikationen des Autors mindestens h mal zitiert wurden
* Hirsch-Index mit Zeitconstraint
  + Zb. H5: wie Hirsch-Index, aber zeitliche Beschränkung der betrachteten Publikationen (z.B. bei h5 auf die letzten 5 Hare)
* Werte hängen stark von Datenbasis ab
* Hirsch-Index analog für Journals definierbar
* I10-Index: Publikationen, die mindestens 10mal zitiert wurden

# Einführung in Information Retrieval

Information Retrieval beschäftigt sich mit der **Repräsentation**, **Speicherung** und **Organisation** von **Informationen** und dem **Zugriff** auf Informationen.

## Retrieval-Szenarien

* Adhoc-Suche
* Ortsabhängige Suche
* Desktop Suche
* Question Answering
* Bildsuche
* Bildersuche
* Empfehlung
* Recherche

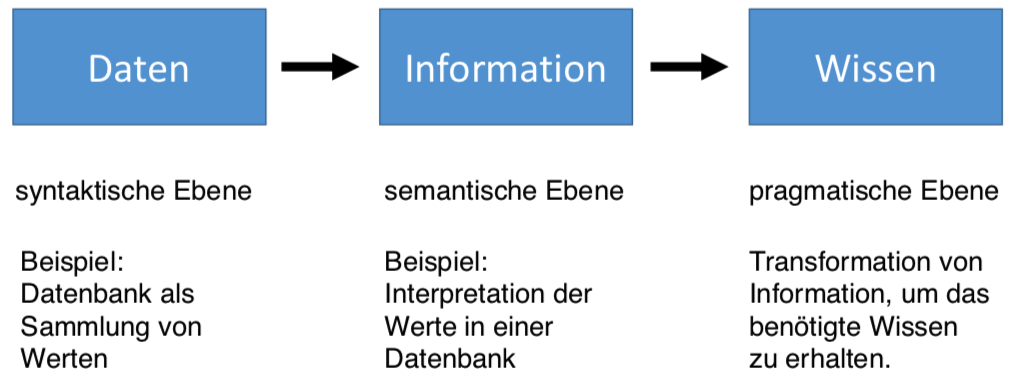
## Herausforderung an IR-Systeme

* Speicherung und effizienter Zugriff auf riesige Datenmenge
* Effiziente und effektive Suche
* Komplexe Suchanfragen (Queries)

## Begriffsbildung

Suche in Dokumentkollektionen kann auf verschiedenen Abstraktionsstufen stattfinden. Vergleiche hierzu die Ebenen der Semiotik:

* **Syntax**: Ein Dokument wird als Folge von Symbolen betrachtet
  + Zeichenkette in Texten
* **Semantik**: Ein Dokument wird auf der Ebene seiner Bedeutung betrachtet. Semantik hat immer etwas mit Interpretation zu tun
* **Pragmatik**: Ein Dokument wird hinsichtlich seines Verwendungszusammenhangs betrachtet
  + Enthält ein Dokument eine Lösung meines Problems



## Percision / Recall

* Percision
  + Erfordert nur die Analyse des Retrieval-Results
  + Kann vom Endbenutzer eingeschätzt werden
  + Ist ein subjektives Maß
* Recall
  + Erfordert die Analyse der gesamten Dokumentenbasis
  + Ist dem Endbenutzer nicht zugänglich
  + Ist ein subjektives Maß

# Boolesches Retrieval – Anfragen und einfache Datenstrukturen

Das **Boolesche Modell** oder **Boolesche Retrieval-Modell** ist ein Information-Retrieval-Modell der folgenden art:

* Die logische Repräsentation betrachtet die Dokumente als **Menge von Wörtern**
* Anfrangen werden aus Index-Termen zusammen mit den **Booleschen Operatoren** AND, OR und NOT gebildet

## Dokumente

* **Dokumente** sind die Einheiten des Datenbestandes bezeichnet, die durch das jeweilige Information Retrieval-System bearbeitet werden
* Dokumente können beispielsweise Bücher, die Kapitel eines Buchs, Notizen, etc. sein
* Die Grundmenge an Dokumenten, für die Information Retrieval durchgeführt wird, wird als **Dokumentkollektion** bezeichnet
* Üblicherweise wird das Symbol **D** verwendet

## Terme

* Als **Term** oder **Index**-**Term** bezeichnet man im Information Retrieval diejenigen Einheiten der Dokumente, die Gegenstand der logischen Repräsentation sind
* Die Terme bilden eine Menge **repräsentativer** **Stichwörter**. Terme sind meistens Wörter oder Wortkombinationen. Über die Terme kann man einen **Index** als Repräsentation eines oder mehrere Textdokumente aufbauen

## Informationsbedarf und Ad-hoc-Anfragen

* Standardaufgabe eines Information Retrieval
* Gesucht: Dokumente aus der Dokumentkollektion, die für eine Anfrage „relevant“ im Hinblick auf den jeweiligen Informationsbedarf sind. Relevanz durch denjenigen definiert, der Anfrage gestellt hat
* Algorithmen zur Anfragebeantwortung sollen effizient (d.h. schnell ihre Ergebnisse liefern) und effektiv (d.h. möglichst genau die Menge der „relevanten“ Dokumenten auffinden) sein
* Informationsbedarf
  + Sachverhalt, über den ein Nutzer etwas in Erfahrung bringen möchte
  + Nicht exakt definiert

## Term-Dokument Inzidenzmatrix

* Die **Termin-Dokument Inzidenzmatrix** *M* enthält eine Zeile für jeden betrachteten Term *t* und eine Spalte für jedes im Grundbestand vorkommende Dokument *d*
* Tritt *t* in dem Dokument *d* auf, so enthält das Matrixelement *(t,d)* eine 1, sonst eine 0
* Die Inzidenzmatrix erlaubt verschiedene Sichtweisen:
  + Jede Zeile *(t,•)* stellt einen Vektor dar, der angibt, in welchen Dokumenten der Term *t* vorkommt
  + Jede Spalte *(•,d)* bildet einen Vektor, der angibt, welche Terme in dem Dokument *d* auftreten

## Invertierter Index

Ein **invertierter Index** oder **invertierte Datei** für eine Dokumentenkollektion *D* besteht aus einem **Vokabular** (Dictionary) und den **Positionen** (Postings)

* Das **Vokabular** enthählt alle Index-Terme zu *D*
* Die **Position-Tabelle** enthält zu jedem Term aus dem Vokabular alle Dokument-IDs
* Die Positionsliste eines Terms heißt auch **invertierte Liste** des Terms

**Einfache konjunktive Boolesche Anfrage**

* Durchschnitt von p1 ∩ p2 repräsentiert die Treffermenge

**Disjunktive Boolesche Anfrage**

* Vereinigung von p1 ∪ p2 repräsentiert die Treffermenge

**Negierte Boolesche Anfrage**

* Entferne aus p1 alle Einträge, die auch in p2 enthalten sind

# Boolesches Retrieval – Vorverarbeitung von Dokumenten und Indexierung

## Indexierung

Die klassischen Dokumentmodelle abstrahieren ein Dokument auf eine Menge von sogenannten **Indextermen** oder **Deskriptoren**

* Idealerweise sollten Indexterme so gewählt sein, dass sie
  + Den **Inhalt** der einzelnen Dokumente adäquat repräsentieren
  + Eine möglichst klare **Abgrenzung** der einzelnen Dokumente gewährleisten
  + Die Verknüpfung von **thematisch ähnlichen** Dokumenten ermöglicht
* Der Prozess der Auswahl von Indextermen heißt **Indexierung**

## Konstruktion eines invertierten Indexes

1. Identifikation und Aufsammeln der zu indizierenden Dokumente
2. Repräsentation jedes Dokument als Listen von **Tokens** (Tokenizing)
3. Optionale Normalisierung der Tokenliste durch **linguistische Vorverarbeitung**; Resultat: **Index-Terme**
4. Aufbau des invertierten Index aus Vokabular und Positionslisten

## Token und Terme

* Ein **Token** ist die Instanz einer begrenzten Zeichenreihe, die in dem gegebenen Dokument auftritt und zu einer für die Weiterverarbeitung semantisch sinnvollen Einheit gruppiert ist
  + Ein Token kann in einem Dokument mehrfach auftreten
* Ein **Typ** ist die Klasse aller Token, die dieselbe Zeichenreihe enthalten
* Ein **Term** ist ein (ggf. „normalisierter“) Typ, der in das Vokabular aufgenommen werden kann.
* Die Normalisierung kann z.B. hinsichtlich Groß-/Kleinschreibung, Morphologie, Rechtschreibung erfolgen

## Tokenisierung und Normalisierung

* **Problembereiche**
  + Satzzeichen
    - .,;:?!‘“: üblicherweise ignoriert
  + Binde- bzw. Trennstriche
  + Trennung am Zeilenende
* Chinesischer Text
  + **Fehlende Leerzeichen**, damit Tokenisierung sehr schwierig
  + **Ambiguität** von Symbolen, Bedeutung und Segmentierung hängt vom Kontext ab
* Japanischer Text
  + Vier Arten von Schriftzeichen
  + Es werden keine Leerzeichen verwendet
* Arabischer Text
  + Hauptleserichtung von rechts nach links
  + Zahlen jedoch umgekehrt

## Vorverarbeitung

* Nicht alle Wörter, die in einem Dokument auftreten, haben die gleiche Signifikanz
* Meistens wird daher ein Dokument einer Vorverarbeitung unterzogen, um die tatsächlich zu verwendenden Index-Terme zu ermitteln.
  + Normalisierung
  + Reduktion auf Wortstämme und Lemmatisierung
  + Thesaurusbildung
  + Elimination von Stoppwörtern

### Normalisierung

* In der Regel möchte man auch bei gewissen **Abweichungen** zwischen den Dokumenttermen und den Anfragetermen gültige Anfrageergebnisse erzielen
* **Normalisierung** ist der Prozess **der** **Kanonisierung** **von** **Token**, damit irrelevante Abweichungen nicht ins Gewicht fallen.
* **Terme** sind also die „Normalformen“ von Token
* Gängiges Normalisierungsverfahren besteht in der Bildung von **Äquivalenzklassen von Token**
* **Ziffern, Zahlen, Daten** müssen ebenfalls segmentiert und in ein Standardformat gebracht werden
* Umlaute und Sonderzeichen
* Schreibfehler
* Case-Folding
  + Der Übergang zur Kleinschreibung
  + Eigennamen weiterhin in Großschrift
  + Anfragen oft ohne Differenzierung von Groß- und Kleinschreibung

### Reduktion auf Grundformen

Durch die Reduktion von Wörtern auf eine **Grundform** oder auf einen **Wortstamm** können Äquivalenzklassen gebildet werden. Dadurch lässt sich die **Größe** **von** **Indexen** und die **Komplexität** **von** **Anfragen** stark reduzieren

**Lemmatisierung**

* Als Lemmatisierung wird die Reduktion von Wörtern auf ihre Grundform nach linguistisch gültigen Regeln bezeichnet
* Lemmatisierung beachtet die Regeln der **Flexion** (Beugung) und **Derivation** (Wortableitung) und berücksichtigen die dadurch hervorgerufenen Wortvarianten

**Stemming**

* Als **Stemming** wird eine heuristische Methode zur Reduktion von Wörtern auf einen **Wortstamm** bezeichnet.
* Durch Stemming werden Wortende abgeschnitten, um zu Äquivalenzklassen mit gleicher oder ähnlicher Bedeutung zu gelangen
* Im Gegensatz zu Lemmatisierung wird Stemming von Linguisten nicht als gültiges Verfahren akzeptiert
* N-gram-Stemmer
  + Bei Übereinstimmung **hinreichend vieler n-Gramme** gelten zwei Wörter als morphologisch ähnlich
  + Ähnlichkeitsmaß wird mit Hilfe der Bi-Gramme ermittelt
  + Aufbau einer Ähnlichkeitsmatrix
  + Clusterbildung
* Stemming mit Successor Variety
* Stemming mit „affix removal“
* Porter-Algorithmus
  + Reduktion erfolgt in fünf sequentiellen Phasen

**Bewertung von Lemmatisierung und Stemming**

* Lemmatisierung führt höchstens zu **sehr kleinen Vorteilen** beim Retrieval
* Stemming erhöht die Ausbeute, aber verschlechtert in der Regel die Präzision

### Thesauri

Ein **Thesaurus** (oder **Wortnetz**) beschreibt Äquivalenzklassen (**Synsets**) von Wörtern bzw. Phrasen (Sequenzen von Wörtern) gleicher Bedeutung, sogenannter **Synonyme**. Er verzeichnet in der Regel auch **Homonyme** und **Polyseme**, d.h. Wörter, die verschiedene Bedeutungen haben können (z.B. Bank)

### Stoppwörter

Ein **Stoppwort für eine Dokumentenmenge** *D* ist ein Wort, das als nicht signifikant für das Retrieval von Dokumenten aus *D* angesehen wird. **Stoppwortliste** ist abhängig von der Anfrage.

* Stoppwörter-Elimination wir heute nicht mehr von Web-Suchmaschinen verwendet

# Boolesches Retrieval – Weitere wichtige Retrievaloperatoren

## Phrasenanfragen

* Ein Auftreten einer Phrase in einem Dokument ist eine Sequenz von Einzelwörtern
* **Wortpaarindexe** indexieren jedes **aufeinanderfolgende Paar von Termen** in einem Dokument als Phrase
  + Warum werden Wortpaarindexe selten verwendet?
    - Falsch-positive Ergebnisse, die eine Filterung der Ergebnisse erforderlich machen
    - Index kann sehr groß werden, da das Vokabular sehr groß werden kann

## Positionsindexe

Ein **Positionsindex** besteht wie ein invertierter Index aus einem Vokabular und einer Positionsliste. Er speichert dabei zusätzlich für jeden Term **t** aus dem Vokabular seine **Position** für jedes Dokument, in dem er auftritt, in der Form

Dabei sind

* **DFreq**: Die **Dokumentenhäufigkeit,** d.h. die Anzahl der Dokumente, in denen **t** vorkommt
* **DocIDi**: Der Identifikatior des i-ten Dokuments, in dem **t** vorkommt
* **TFreqi**: Die Anzahl der Positionen, an denen **t** in Dokument DocId auftritt
* **Posi1, …, posini:** Die Position in aufsteigender Reihenfolge, an denen **t** in Dokument DocID auftritt

## Proximity-Queries (Nachbarschaftsanfragen)

* **Proximity-Queries** oder **Nachbarschaftsanfragen** stellen eine verallgemeinerte Form der Phrase Queries dar.
* Textstellen, in denen die angegebenen Einzelwörter einen bestimmten **Maximalabstand nicht überschreiten**.
* Reihenfolgen der Einzelwörter kann beachtet werden

## Rechtschreibkorrektur

* Es gibt zwei mögliche Ansätze für die Rechtschreibkorrektur
  + Korrektur von **Dokumenten** vor der Indizierung
  + Korrektur von **Anfragen**
* Methodische Ansätze der Rechtschreibkorrektur
  + **Korrektur isolierter Terme**
    - Jeder einzelne Anfrageterm wird separat behandelt
    - Diese Methode kann keine fehlerhafte Anfrage, die aus korrekten Termen besteht, erkennen
  + **Kontext-sensitive Korrektur**
    - Es wird die gesamte Anfrage behandelt
    - Es findet in der Regel keine grammatikalische Prüfung statt

### Korrektur isolierter Terme

* **Gewichtete Edit-Distanz**
  + Man kann die Edit-Distanz verfeinern, indem man das Gewicht einer Grundoperation in Abhängigkeit von den behandelten Zeichen definiert
* **Spelling Corrector von Peter Norvig**
  + Das Verfahren berechnet die Wahrscheinlichkeit P(c|w), das Term c gemeint ist, wenn Term w geschrieben wurde, mit dem Satz von Bayes

### Kontext-sensitive Rechtschreibkorrektur

* **Hit-basierte Rechtschreibkorrektur**
  + Einfaches und nicht sehr effizientes Verfahren
  + **Einzelnen Anfrageterme** werden durch Terme mit geringer Edit-Distanz ersetzt
* **Wahrscheinlichkeit einer Wortsequenz**
  + Sequenz werden in ihre Bigramme (auf Wortebene) zerlegt
* **Spelling Correction mit Wortsequenzen**
  + Hit-basierte Methode kombiniert mit der höchsten geschätzten Wahrscheinlichkeit
  + Annehmen, dass nur ein Fehler pro Anfrage auftritt
* **Phonetische Korrektur**
  + Neben der eigentlichen Rechtschreibkorrektur spielt auch die **phonetische Korrektur**, d.h. die Korrektur von Fehlern, die aufgrund des gleichen Klangs zweier Schreibweisen entstehen, eine Rolle
  + Korrekturalgortihmen: **SOUNDEX-Algorithmen**
    - Erster Buchstabe bleibt unverändert und wird als Großbuchstabe übernommen
    - Ersetze alle Vorkommen der folgenden Buchstaben mit 0 (Null) A,E,I,O,U,H,W,Y
    - Ersetze die übrigen Buchstaben nachfolgendem Schema (aus bestimmten Buchstaben werden bestimmte Zahlen)
    - Ersetze alle Paare von gleichen aufeinerfolgenden Ziffern
    - Lösche alle Nullen aus dem Ergebnis
    - Gib die ersten vier Stellen des Ergebnisses zurück, ggf. nach Auffüllen mit Nullen; das Ergebnis hat dann die Form **Großbuchstabe Ziffer Ziffer Ziffer**

## Schwachpunkte des Booleschen IR-Modells

Wesentliche Schwachpunkte des elementaren Booleschen IR Modells sind:

* Boolesche Anfragen werden schnell recht **komplex**
* Die Retrieval-Strategie basiert auf einer binären Entscheidung, lässt also **kein Ranking** zu

# Retrievalmodelle – Das Vektorraum-Modell

## Sichten auf ein Dokument

Die Automatisierung von Retrieval-Aufgaben erfordert die **Modellierung** und **Repräsentation** von Dokumenten auf einem Rechner. Dabei lassen sich drei orthogonale Sichten auf den Inhalt unterscheiden

1. **Layout-Schicht**: Darstellung eines Dokuments auf einem zweidimensionalen Medium
2. **Strukturelle bzw. logische Sicht**: Definiert den Aufbau bzw. die logische Struktur eines Dokuments
3. **Semantische Sicht:** Betrifft die Aussage eines Dokuments und ermöglicht dessen Interpretation

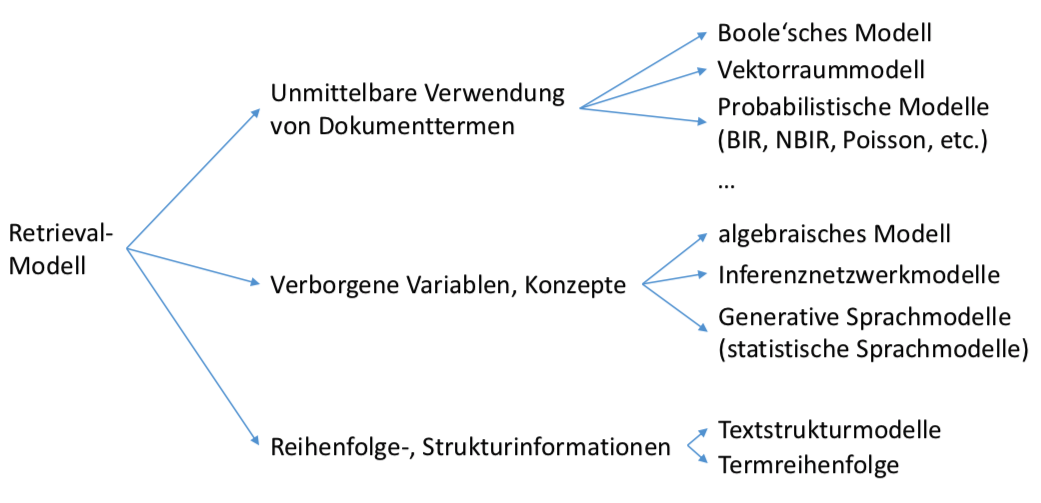
## Modelle

**Definition: (Dokumentmodell, Retrieval-Modell, Retrieval-Funktion)**

Sei *D* eine Menge von Dokumenten und *Q* eine Menge von Anfragen. Ein **Dokument-Modell** für *D,Q* ist ein Tupel **(D,Q,pR),** dessen Elemente wie folgt definiert sind:

* **D** ist die Menge der Repräsentationen der Dokumente d ∈ D. In d ∈ D können Layout-logische und semantische Sicht codiert sein.
* **Q** ist die Menge der **formalisierten Anfragen**
* **R** ist ein Retrieval-Modell und formalisiert ein Prinzip, ein Paradigma oder eine linguistische Theorie.
* Auf der Grundlage von **R** ist die **Retrieval-Funktion pR(q,d)** definiert. Sie **quantifiziert die Systemrelevanz** zwischen einer formalisierten Anfrage q ∈ Q und einer Dokumentrepräsentation d∈D: **pR : Q** x **D 🡪 R**
* Die von **pR** berechneten Werte heißen **Retrieval-Werte** (Retrieval Status Value, RSV) oder auch **Scores**

## Taxonomie von Retrieval-Modellen



## Klassisches Retrieval-Modell

Die **klassischen Retrieval-Modelle** abstrahieren ein Dokument d∈D zu einer unstrukturierten Menge von Indextermen, die sich quasi unmittelbar und automatisch aus *d* gewinnen lassen

Die Dokumentenrepräsentation **d** eines Dokumentes *d* besteht aus gewichteten Indextermen, die aus d stammen

Unterscheidung der klassischen Retrieval-Modelle:

1. Art und Weise, wie sich Gewichte wi für die Indexterme ti berechnen
2. Art und Weise, wie formalisierte Anfragen **q** konstruierbar sind
3. Art und Weise, wie sich die Retrieval-Funktion **pR(q,d)** berechnet
4. Art und Weise, wie die Menge relevanter Dokumente R(q) konstruiert wird

## Boolesches Modell

**Dokumentenrepräsentationen D:**

* Typischerweise bilden normalisierte Terme eines Korpus die Menge der IOndexterme T = {t1,…tm}.
* Die Repräsentation **d** eines Dokumentes *d* ist eine Abbildung von T nach {0,1}, wobei **d**(w) = 1 bzw. **d**(w)= 0 als „Term in d vorhanden“ bzw. „nicht vorhanden“ interpretiert wird

**Formalisierte Anfragemenge Q:**

* Eine formalisierte Anfrage **q€Q** entspricht einer logischen Formel über dem Alphabet Σ=T, in der die Junktoren ⋀, ⋁, ¬ und Klammern verwendet werden können

**Retrieval-Funktion pR:**

* Die Dokumentenrepräsentation **d** eines Dokumentes d induziert eine Interpretation Id für **q**; man setzt **pR(q,d) =** Id**(q)**
* Gilt **pR(q,d) = 1,** wird das Dokument **d** Element der Antwortmenge R(q)

**Vorteile:**

* Mächtigkeit: Prinzipiell kann mit einer Booleschen Anfrage jede beliebige Teilmenge von Dokumenten aus einer Kollektion selektiert werden
* Einfache und genaue Implementierbarkeit

**Nachteile:**

* die Größe der Antwortmenge ist schwierig zu kontrollieren
* keine Möglichkeit zur Gewichtung von Fragetermen
* schlechte Retrieval-Qualität

## Vektorraummodell

**Dokumentenrepräsentationen D:**

* Typischerweise bilden die normalisierten Terme, ggf. nach Entfernung der Stoppwörter, eines Korpus die Menge der **Indexterme** T = {t1,…,tm}.
* Der Wertebereich der Termgewichte ist R (reele Zahlen)
* für die Gewichtsberechnung existieren verschiedene Konzepte

**Formalisierte Anfragenmenge Q:**

* Eine formale Anfrage **q€Q** hat den gleichen Aufbau wie eine Dokumentenrepräsentation **d€D**

**Retrieval-Funktion pR:**

* Dokumentrepräsentationen und formalisierte Fragen werden als Punkte eines **orthonormalen Vektorraums** interpretiert, der durch die Terme aufgespannt wird
* Wichtige Ansätze zur Berechnung von **pR**sind das **Cosinus-Ähnlichkeitsmaß** und die euklidische Distanz.

### Termhäufigkeit

* Die Anzahl des Auftretens eines Terms in einem Dokument

### Termgewichte

* **Erinnerung:** Die Term-Dokumenten-Inzidensmatrix des Booleschen Retrieval-Modells enthält für jeden Term t und jedes Dokument d einen Eintrag **m(t,d)**. Sein Wert ist 1, falls t in d auftritt und 0 sonst
* **Gewicht w(t,d)**, dass im Zusammenhang mit der Anzahl der Auftreten des Terms in dem jeweiligen Dokument steht
* **Term Frequency**  oder **Term-Häufigkeit** bezeichnet das Gewichtungsschema, in dem direkt die Anzahl tft,d der Auftreten des Terms t in Dokument d als Gewicht verwendet wird
* **Tft,d = Term-Häufigkeit**

### Term-Dokument Häufigkeitsmatrix

Eine **Term-Dokument-Häufigkeitsmatrix M** enthält eine Zeile für jeden Term t€V aus dem Vokabular V und eine Spalte für jedes in der Dokumentkollektion D vorkommende Dokument d€D. Tritt t in dem Dokument d an k Stellen auf, so enthält das Matrixelement m(t,d) den Wert k, sonst eine 0:

In diesem Modell wird also jedes Dokument d durch einen Vektor (w1,d,…,W|V|,d) mit **Termgewichten** repräsentiert, dessen i-te Komponente die Häufigkeit tft,d des Terms ti in dem Dokument *d* angibt

### Bag-of-Words-Modell

Wortordnung innerhalb eines Dokumentes wird nicht berücksichtigt

### Absolute Termhäufigkeit

Die Absolute Anzahl von Auftreten eines Terms t in dem Dokument d ist als Maß nicht geeignet

* Logarithmische Häufigkeitsmaße
* Dämpfungsfunktion (Ergebnis zwischen 0 und 1)

### Dokumentenhäufigkeit

Sei D eine Dokumentenkollektion. Sei t ein Term des Vokabulars. Dann bezeichnet die **Dokumenthäufigkeit** **dft** die Anzahl der Dokumente d€D, in denen t auftritt

* **Hohe Dokumenthäufigkeit** dft bedeutet **geringe Signifikanz** oder Trennschärfe von t
* **Geringe Dokumenthäufigkeit** dft bedeutet **hohe Signifikanz** (Trennschärfe) von t

### Inverse Dokumenthäufigkeit

Sei D eine Dokumentkollektion, die N = |D| Dokumente enthält. Für einen Term t des Vokabulars ist die **inverse Dokumenthäufigkeit** **idft von t in der Kollektion D** definiert durch

**Bemerkung:**

* Hohe inverse Dokumenthäufigkeit idft bedeutet hohe Trennschärfe von t
* Geringe inverse Dokumenthäufigkeit idft bedeutet geringe Trennschärfe von t
* Der Einfluss der Dokumenthäufigkeit wird durch das logarithmische Maß gedämpft

### Kollektionshäufigkeit

**Kollektionshäufigkeit** eines Terms t: Anzahl des Auftretens von t in der gesamten Dokumentkollektion

🡪 **Dokumenthäufigkeit** erweist sich jedoch als das besser geeignete Maß

### tf•idf-Gewichtung

Falls t in d vorkommt, ist das Termgewicht wt,d also

* **am höchsten**, wenn t häufig in d, aber insgesamt in einer geringen Zahl von Dokumenten der Kollektion auftritt
* **geringer,** wenn t seltener in d oder insgesamt in einer größeren Zahl von Dokumenten der Kollektion auftritt
* **am geringsten,** wenn t in praktisch allen Dokumenten auftritt

### Dokumentvektoren

* Wenn man die tf•idf-Gewichtung anwendet, wird jedes Dokument d durch ein Vektor repräsentiert.
* Für Terme t€V des Vokabulars V, die nicht in d vorkommen, hat das entsprechende Gewicht den Wert 0
* Zusammen bilden die Dokumente eine |V|-dimensionalen reelen Vektorraum
* Die Terme t€V des Vokabulars bilden die Dimensionen des Vektorraums
* Der Vektorraum der Dokumente ist im Allgemeinen von sehr hoher Dimension

### Anfragen als Vektoren

* Anfragen werden im Vektorraummodell als **Freiformanfragen** aufgefasst, die nur durch eine Menge von Termen des Vokabulars spezifiziert werden
* Gewichte orientieren sich dabei nur an **der Häufigkeit der Terme in der Anfrage** (ohne idf-Komponente)
* Der Score eines Dokuments d für die Anfrage q wird als **Cosinus-Ähnlichkeit** sim(d,q) der entsprechenden Vektoren berechnet
* Unterstützt Freitextanfragen
  + Anfrage als Menge von Termen ohne Verknüpfung
* Bei Web-Suchmaschinen werden Anfrageterme oft als konjunktive Anfrage aufgefasst

# Retrievalmodelle – Probabilistische Modelle

* Die grundlegende Idee ist es, Dokumente nach absteigender Relevanzwahrscheinlichkeit zu ordnen

## Probability Ranking Principle

* Sei d ein Dokument aus der Kollektion. Die binäre Zufallsvariable R beschreibe die Relevanz eines Dokuments: R = 1 bedeutet also relevant, R = 0 bedeutet nicht relevant
* Dokumente werden in **absteigender Relevanzwahrscheinlichkeit** zurückgegeben
* In der Praxis wird nach dem Verhältnis zwischen den Wahrscheinlichkeiten für Relevanz und Nichtrelevanz sortiert
* Mit korrekt geschätzten Wahrscheinlichkeiten ist diese Ergebnisreihenfolge **optimal bezüglich der Ergebnisgüte**

## Binary Independece-Model

## Okapi BM25

# Retrievalmodelle – Generative Sprachmodelle

## Statistische Sprachmodelle

* Modelliert die in einer Sprache auftretenden Sätze **statistisch**
* Es erlaubt die **Wahrscheinlichkeit** zu bestimmen, mit der einen vorgegebene Wortfolge vorkommt
* Wird mit vielen Beispielsätzen gelernt
* Verwendet keine Grammatikregeln
* Konzept aus der **kontext-sensitiven Rechtschreibkorrektur**
* Solche komplexen Abhängigkeiten kann man praktisch nicht bestimmen, wir verwenden daher vereinfachte Approximationen
* Die am häufigsten eingesetzten Sprachmodelle verwenden **Unigramme** und **Bigramme**
  + Man schätzt die jeweiligen Wahrscheinlichkeiten auf Basis der **Vorkommen der Wörter** (bzw. der Wortpaare bei Bigramm-Modellen) **in einer großen Menge von Dokumenten**, die charakteristisch für die jeweilige Sprache sind
  + **Unigramm-Sprachmodell**
    - Wird die Wahrscheinlichkeit einer Wortsequenz auf die Wahrscheinlichkeit der einzelnen Wörter zurückgeführt
    - Dieses Modell berücksichtigt die Reihenfolge der Wörter nicht, es betrachtet als nur die Wortsequenzen als Bag-of-Words
  + **Bigramm-Sprachmodell**
    - Konditioniert dagegen die Teilwahrscheinlichkeiten mit dem vorhergehenden Wort
    - Wird für die Fehlerkorrektur verwendet

## Sprachmodelle im Information Retrieval (Query-Likelihood-Modell)

* Im Information Retrieval müssen **Relevanzwahrscheinlichkeiten P(R = 1 | q,d)** schätzen
* Anfrage q die „Eingabe“ der Schätzung => **P(d|q)**
* P(q) ist für eine feste Anfrage konstant, spielt also keine Rolle für das Ranking
* P(d) könnte verwendet werden, um „guten“ Dokumenten eine höhere Wahrscheinlichkeit zu geben
* **Glättung von Sprachmodellen**
  + Bisher konjunktive Semantik, jedoch zu strikt
  + **Glättungs-** oder **­Smoothing-Methoden,** um auch im Fall fehlender Terme eine gewisse Wahrscheinlichkeit berücksichtigen zu können
  + Verwendung eines **Hintergrund-Sprachmodells**

## Abschließende Bemerkungen zu Sprachmodellen

* Während bei tf•idf und BM25 ein in der Kollektion **seltener Term** wichtiger als ein häufiger, ist es in Sprachmodellen umgekehrt: Im Hintergrundmodell haben in der Kollektion **häufige Terme** eine höhere Wahrscheinlichkeit als seltene Terme
* Sprachmodelle erzielen in den meisten Benchmarks **bessere Ergebnisse** als andere Modelle, z.B. BM 25

# Retrievalmodelle – Algebraische Modelle

## Idee

Transformation der **hochdimensionalen Dokumentvektoren** in einen **niedrigdimensionalen Raum** bei möglichst genauer Erhaltung der Information

## Vorteile

* Automatische Entdeckung verborgener **Konzepte**
* Syntaktische Erkennung von **Synonymen**
* **Semantische Erweiterung** von Anfragen aufgrund syntaktischer Analyse – und nicht durch Relevanz-Feedback oder die Bemühung von Thesauri

## Nachteile

* Die Wirkungsweise von LSI ist nicht vollständig verstanden; eine theoretische fundierte Brücke zur Linguistik ist nur ansatzweise vorhanden
* LSI entfaltet die volle Wirkung nur in einer **geschlossenen Retrieval-Situation**: die Kollektion ist bekannt, gegeben und ändert sich nur wenig
* Die Singulärwertzerlegung ist **rechenaufwendig**, O(n­^3)

# Retrievalmodelle – Kombination mehrere Modelle

* Kombination von Relevanzsignalen verschiedener Art zu einem **Gesamtscore**
* Übliche Klassen von **Relevanzsignalen** (oder **Features**) sind
  + **Dynamische Signale**, die von der Anfrage und vom Dokument abhängen
  + **Statische Signale**, die nur vom Dokument abhängen
  + **Anfrageeigenschaften**,
* Orthogonal und teilweise ergänzend kann man die Features auch nach ihrer Quelle gruppieren
  + **Inhaltssignale**, die den Inhalt eines einzelnen Dokuments betrachten
  + **Struktursignale**, die die Verlinkung von Seiten im Web ausnutzen
  + **Verhaltens- und Benutzersignale**, die das Clickverhalten von Benutzern berücksichtigen
* Die einzelnen Signale aus einer Menge F werden zu einem **gewichteten Gesamtscore** kombiniert

# Evaluation von IR-Systemen

Evaluation ist der Schlüssel, um

* **Effektive** (Finden wir die richtigen Dokumente?)
* **Effiziente** (Machen wir es schnell / mit hohem Durchsatz)

**Evaluations-Korpora**

* Testkollektionen, die aus **Dokumenten, Anfragen** und **Relevanzbewertungen** bestehen
  + CACM
  + AP
  + GOV2

## Poolbildung

* Erzeugt eine große Anzahl von Relevanzbewertungen für jede Anfrage, jedoch **immer noch unvollständig**

## Anfragelogs

* Werden für das **Tunen und Evaluieren** von Suchmaschinen eingesetzt
* Inhalte der Anfragelogs
  + Benutzeridentifikator
  + Anfrageterm
  + Liste der Ergebnis-URLs, und ob sie angeklcikt wurden
  + Zeitstempel
* Klicks sind **keine Relevanzbewertungen**
  + **Verfälscht** durch Faktoren wie den Rang in der Ergebnisliste
* Man kann Klickdaten verwenden, um **Präferenzen zwischen Paaren von Dokumenten** vorherzusagen
* Klickdaten können auch **aggregiert** werden, um „Noise“ (Klicks, die nicht zu relevanten Dokumenten führen) zu entfernen

## False Negatives und False Positives

|  |  |  |
| --- | --- | --- |
|  | Relevant | Nicht Relevant |
| Gefunden | True Positives | False Positives |
| Nicht gefunden | False Negatives | True Negatives |

* **True Positives (tp)** = Gefundene relevante Dokumente
* **False Positives (fp)** = Gefundene irrelevante Dokumente
* **True Negatives (tn)** = Nicht gefundene irrelevante Dokumente
* **False Nefatives (fn)** = Nicht gefundene relevante Dokumente

## Percision und Recall

Die **Präzision** oder **Percision P** gibt an, wie groß der Anteil der korrekten Treffer an der gesamten Menge der gefundenen Dokumente ist.

Die **Ausbeute** oder der **Recall R** gibt an, wie groß der Anteil der korrekten Treffer an der Menge der relevanten Dokumente ist

* **Recall-Orientierung**
  + Wenn es wichtig ist, in jedem Fall alle relevanten Dokumente zu finden
  + Beispiel: Patent-Recherche
* **Precision-Orientierung**
  + Wenn die Wahrscheinlichkeit, dass ein positives Ergebnis auch korrekt ist, wichtig ist
  + Beispiel: Alert
* Fast immer stehen die Ziele Recall und Precision im Konflikt!

### F-Maß

* **Harmonisches Mittel** von Recall und Precision
  + hebt die Bedeutung kleinerer Werte hervor
  + während **arithmetische Mittel** mehr von Außreißern, die gewöhnlich **groß** sind, beeinflusst werden

### Optionen zum Zusammenfassen eines Rankings

1. Berechnung von Recall und Precision an festgelegten Rang-Positionen
2. Precision wird an **Standard-Recall-Punkten** von 0,0 bis 1,0 im Abnstand von 0,1 berechnet
3. Bilden von **Durchschnittswerten über die Percision-Werte** der Rangpositionen, an denen ein relevantes Dokument abgerufen wurde

### Benutzermodell für AP

* AP evaluiert gleichzeitig verschiedene Retrievaltasks (recall-oriented und precision-oriented Tasks) und ist daher nicht ideal

### Durchschnittsbildung

* **Percision:** für **eine Anfrage** an einem Recall-Punkt
* **Average Percision (AP):** Mittelwertbildung **über die Recall Punkte einer Anfrage**
* **Mean Average Precision (MAP)**
  + Mittelwertbildung über **mehrere Anfragen**
  + Fasst **Rankings für mehrere Anfragen** zusammen, indem ein Durchschnitt über die mittleren AP-Werte gebildet wird
  + Sehr **verbreitetes Maß** in Forschungsliteratur
  + Benötigt viele Relevanzbewertungen in der Textkollektion
* **GMAP**
  + Höheres Gewicht auf Anfragen mit geringer AP

### Interpolation

Wichtig?

## Konzentration auf die Top-Dokumente

* **Benutzer** tendieren dazu, nur den **obersten Teil der Ergebnisliste** anzusehen, um relevante Dokumente zu finden
* -> Recall kein angemessenes Maß
* Stattdessen sollte gemessen werden, wie gut die Suchmaschine relevante Dokumente in Top-Rängen einstuft
* **Percision in Rang** R(P@R)
  + Einfach zu berechnen, einfache Mittelwertbildung, einfach zu verstehen
* **Reziproker Rang** (für Anfragen, bei denen es um ein relevantes Dokument geht)

## Discounted Cumulative Gain (DCG)

Verbreitetes Maß, um **Websuche** und verwandte Aufgaben zu evaluieren

* **Hochrelevante Dokumente** sind nützlicher als nur marginal relevante Dokumente
* Je höher der Rang eines relevanten Dokuments, desto weniger nützlich ist es für den Nutzer, da es mit geringerer Wahrscheinlichkeit betrachtet wird
* Verwendet **gestufte Relevanz** als Maß für die Nützlichkeit oder den Gewinn, der durch Betrachtung eines Dokuments erreicht wird
* Der **Gewinn** wird beginnend mit den bestplatzierten Ergebnissen akkumuliert und kann bei **höheren Rängen reduziert** werden

## Normalisierter DCG

* Der **Mittelwert** über DCG-Werte wird über eine Menge von Anfragen in spezifischen Rängen gebildet
* DCG-Werte werden oft normalisiert, indem die DCG-Werte in jedem Rang **mit den DCG-Werten für perfektes Ranking verglichen** werden

## BPREF

* Besonders wichtig für Anfragen mit unvollständigen Relevanzbewertungen
* Für eine Anfrage mit **R relevanten Dokumenten** werden nur die ersten **R als nicht relevant erkannten Dokumente** betrachtet
* *N* ist die Zahl der als nicht relevant erkannten Dokumente
* *dr* ist ein relevantes Dokument
* *Ndr*ist die Anzahl der ersten R nichtrelevanten Dokumente, die vom System höher eingestuft wurden als *dr*

## Effizienzmaße

|  |  |
| --- | --- |
| Maß | Beschreibung |
| Verstrichene Indexierungszeit (elapsed index time) | Misst den Zeitverbrauch für die Erstellung eines Index auf einem bestimmten System |
| Prozessorzeit für Indexierung (indexing processor time) | Misst die vom Prozessor für die Indexierung benötigte Zeit in Sekunden. Diese Zeit entspricht der verstrichenen Indexierungszeit, jedoch werden I/O- Wartezeiten und Zeitgewinne durch parallel Verarbeitung nicht beachtet. |
| Anfragendurchsatz (query throughput) | Anzahl der pro Sekunde verarbeiteten Anfragen |
| Anfragelatenzzeit (query latency) | Die Zeit in Millisekunden, die der Nutzer nach Abschicken der Anfrage auf eine Antwort durchschnittlich wartet (arithmetisches Mittel, besser Median) |
| Temporärer Speicherplatz für Indexierung (indexing temporary space) | Speicherplatz, der während der Indexerstellung benötigt wird |
| Indexgröße (index size) | Speicherplatz, den der fertige Index benötigt |

# Evaluierung von IR-Systemen – Tuning von Parametern

* Optimieren der Parameterwerte
  + Beste Leistung für verschiedenen Datentypen und Anfragetypen zu erzielen
* Finden der Parameterwerte
  + Viele Techniken für optimale Parameterwerte eingesetzt

## Online-Tests

* **Vorteil**
  + Echte Nutzer,
  + weniger voreingenommen,
  + großen Menge an Testdaten
* **Nachteil** 
  + Daten mit Noise behaftet
  + Kann die User Experience verschlechtern

## Zusammenfassung

* Es gibt **kein Maß**, das für **jede beliebige Applikation** korrektist
  + Das gewählte Maß muss **angemessen für die Aufgabe** sein
  + Verwendung von **Kombinationen**
* Analyse der Ergebnisse individueller Anfragen
* Wichtig: **Nutzerstandpunkt**
* **Analyse einzelner Anfragen** oft wichtiger als Durchschnittsbetrachtung
  + Effektivität bei **leichten/schweren Anfragen**
* Kleine Unterschiede in Kennzahlen haben oft keinen Zusammenhang zum **Nutzerempfinden**

# Websuchmaschinen

## Ansätze für die Informationsfindung

* **Verlinkung** thematisch ähnlicher Seiten
  + Vorteil: Verlinkte Seiten sind oft sehr hilfreich
  + Nachteil: Einschränkung auf verlinkte Seiten
  + Beispiel: Wikipedia
  + **In digitalen Bibliotheken**: Zitate in Veröffentlichungen
* Bildung **thematischer Indexe**
  + Vorteil: Großes Verzeichnis von ähnlichen Seiten
  + Nachteil: Für die thematische Einstufung und Ordnung von Webseiten sind Experten erforderlich
  + Beispiel: Yahoo, DMOZ
  + **In digitalen Bibliotheken**: Metadatenkataloge, Konferenzverzeichnisse
* **Suchmaschinen**
  + Vorteil: Automatisierte Erfassung sehr vieler Webseiten
  + Nachteil: Herausforderungen des Information Retrieval
  + Beispiel: Google, Bing
  + **In digitalen Bibliotheken:** Volltextsuche

**Lokale Marktführer**

* China: Baidu
  + Rückzug von Google
  + Chinesische Sprache
* Russland: Yandex
  + Erkennung von Flexionen (Wortbeugungen) in der Russischen Sprache

**Archie – die erste „Suchmaschine“**

* Werkzeug zur **Indizierung von FTP Archiven**

## Herausforderung an Websuchmaschinen

Größte Herausforderung für Information Retrieval im World Wide Web:

* Sehr **großer Datenbestand**
* Sehr **dynamischer Datenbestand**
* Unterscheidung wichtiger und belangloser Webseiten
* Aussortierung **bösartiger** Webseiten
* Verarbeitung unterschiedlichster Themengebiete (z.B. Schlagworte, Kartendienste, Aktienkurse)
* Kontextualisierung (z.B. Geolokalisierung)
* Personalisierung

## Crawler

Ein **Crawler** scannt das WWW nach Veränderungen ab. Seine Aufgabe ist es, **neue**, **modifizierte** und **gelöschte Webseiten** zu identifizieren. Seine Informationen gibt er an den Indexer weiter

### Anforderungen an Crawler

* **Robustheit** gegen **Spidertraps**
  + Syntaxfehler in Webseiten
  + Fehlerhafter Aufbau von Webseiten
  + Dynamische Applikationen
* **Höflichkeit** gegenüber **Webservern**
  + Implizite serverseitige Regeln für Crawler, z.B. Anfragehäufigkeit („politeness“)
  + Explizite serverseitige Regeln für Craweler, z.B. „robots.txt“, rel=“nofollow“ für HTML-Links und das Metatag robots
  + „Web Etiquette“

### Empfehlungen für Crawler

* Verteiltheit
* Skalierbarkeit, Effizienz
* Webseiten mit guter Qualität bevorzugen
* Aktualität der indizierten Webseiten gewährleisten
* Erweiterbarkeit

### Aktualisieren von Webseiten

Es ist also besser, nur die Seite zu aktualisieren, die sich selten ändert, da wir mit einer Aktualisierung pro Tag die Dynamik von p1 nicht nachvollziehen können

* **Wichtigkeit** für Optimierung mit einbeziehen

## Indexer

**Indexer-Modul**

* Nimmt die Informationen von Crawlern entgegen
* Baut daraus Indexe

**Anforderungen an den Indexer:**

* Verschiedene Dokumentformate
* Sprachliche Vorverarbeitung von Tokens
* Kompressionsmechanismen
* Verschiedene Indextypen ( z.B. BM25-Indexe, Positionsindexe..)

## Searcher

* Nimmt Suchanfragen von Benutzern entgegen
* Wertet diese anhand der vorhandenen Indexstrukturen aus
* Führt die Relevanzschätzung zur Bewertung der Ergebnisse bzgl. Der Suchanfrage durch

**Anforderungen an den Searcher:**

* Kommunikation mit Anwender
* Unterstützung des Anwenders bei Anfrageformulierungen
* Enge Zusammenarbeit mit dem Indexer: Verwendung der erstellten Indexe
* Relevanzschätzung, d.h. Ranking, von Dokumenten bzgl. Anfrage

## Google-Crawler

* Durchsucht täglich Milliarden Webseiten
* Die Updatehäufigkeit einer Webseite hängt von ihrer Relevanz (d.h. PageRank) ab
* Kann mittlerweile Flash-Animationen crawlen
* Laut Google keine kommerzielle Beeinflussung

## Google-Indexer

* **Linkindex**
  + Webgraph aus Knoten und Kanten
  + Speichert insbesondere Nachbarschaftsinformationen
* **Textindex**
  + Invertierter Index
  + Lexikon
  + Identifizierung gesuchter Webseiten
* **Relevanzindexe**

Die Indexierung unterstützt eine Vielzahl von Dateitypen

# Websuchmaschinen – Ranking mit Pagerank

**Ranking – Problem:**

* Sehr unterschiedliche Dokumente und Inhalte
* Verlinkung von Webseiten bringt Zusatzinformation

**Verlinkung im Web**

* Früher manuell
* Heute oftmals automatisch
* Viele Verlinkungen, hohe Qualität -> Erfolg von Google damals

## Webmodell

* Der **Webspace W** ist ein gerichteter Graph
* Die Knoten V repräsentieren Webseiten im Webspace
* Die Kanten E entsprechen der Verlinkung zwischen den Webseiten
* **Out-Nachbarn**
  + Menge aller Knoten (Webseiten), die durch eine Kante erreichbar sind
* **In-Nachbarn**
  + Menge aller Knoten (Webseiten, die eine Kante zu v besitzen

**Prinzip des Rankings: Seiten, auf denen der Random Surfer häufig ist, sind „wichtig“**

## Übergangsmatrix A

Die **Übergangsmatrix A** ist eine quadratische Matrix zur Darstellung der Verlinkung des Webspaces

* Zufallsgetriebener Surfer
  + Der u-te Spalentverktor enthält die **diskrete Wahrscheinlichkeitsverteilung** über alle ausgehenden Kanten des Knotens u
  + Der v-te Zeilenvektor benennt die möglichen **Quellen für Aufrufe von v** (Einträge > 0)

## Vereinfachter PageRank

* Der Surfer s wählt **zufällig und gleichverteilt** eine der vorhandenen Webseiten als **Startknoten**
* Anschließend wählt s in jedem Schritt **zufällig** eine der **aktuell erreichbaren Webseiten** aus.
* Die Auswahl einer Webseite wird **gleichverteilt** über alle aktuell erreichbaren Webseiten getroffen
* Sei s ein Surfer, der sich im Webspace W entlang der Links bewegt
* Der **PageRank PR(v)** einer Webseite v entspricht dem **Grenzwert der Auftrittswahrscheinlichkeit** von v nach unendlich vielen Bewegungen von s

## Rangsenken

* Stellen einen **Verbund von Webseiten** dar, die nur aufeinander verlinken, aber **keine Links nach außerhalb** besitzen
* Zur Modellierung erlaubt man dem Surfer in jedem Schritt mit Wahrscheinlichkeit *a* eine **Teleport-Operation** auszuführen, die ihn zu jeder beliebigen Webseite bringen kann

## Teleport-Operation

* Erlaubt es dem Surfer s, jede beliebige Webseite direkt anzuspringen (Wahrscheinlichkeit *a*)
* Wahrscheinlichkeit 1-*a* folgt der Surfer weiter einer Zufallsbewegung
* *a* = **Dämpfungsfaktor**

## Normaler PageRank

* Sei W ein Webspace und G = (V,E) der zugeordnete gerichtete Graph
* Dann kann der **PageRank-Vektor PR** über alle Webseiten approximiert werden mit einem **Grundrang**

## Suche mit PageRank

* Ansatz 1: **PageRank für Sortierung**
  + Suche alle zur Suchanfrage passenden Webseiten anhand eines IR-Verfahrens
  + Ordne die Webseiten in absteigender Reihenfolge entsprechend ihres PageRanks
  + Einfach zu implementieren
  + Die Sortierung der Webseiten erfolgt aufgrund ihrer Wichtigkeit im Webspace **unabhängig von deren geschätzter Relevanz** für die Suchanfrage
* Ansatz 2: **PageRank für Relevanzbestimmung**
  + Kombiniere PageRank mit einem IR-Verfahren zur Relevanzschätzung von Webseiten
  + Ordne die Webseiten in absteigender Reihenfolge entsprechend ihrer Relevanz
  + Erfordert eine **komplexe Kombination von PageRank und IR-Scoring**
  + Die Sortierung im Webspace und unter Berücksichtigung von deren Relevanz für die Suchanfrage

In der Praxis verwenden Suchmaschinen eine Variante des 2. Ansatzes

# Websuchmaschinen – Ranking mit HITS

## Adjazenzmatrix

Die **Adjazenzmatrix A** für einen gerichteten Graph G = (V,E) ist eine (VxV)-Matrix

## Authorities und Hubs

* Die Webseiten V können zwei verschiedenen Typen angehören
  + **Hubs (H)** und **Authorities (A)**
* Die Zugehörigkeit ist graduell und nicht zwangsweise exklusiv
* Eine **Authority** ist eine Webseite, auf die viele Hubs verlinken
  + A(v) = **Authority-Score**
* Ein **Hub** ist eine Webseite, die Links auf viele Authorities enthält
  + H(u) = **Hub-Score**
* Hubs sind gewissermaßen populäre Linksammlungen
* Jede Webseite ist zu einem gewissen Grad **gleichzeitig Hub und Authority**
  + Normalisierung der Zugehörigkeitsgrade

## HITS (Hyperlink-Induced Topic Search)

* Approximieren des Authority-Scores a und Hub-Scores h

### Suche mit HITS

Vorgehen bei der Beantwortung von Suchanfragen mit HITS-Ranking

1. Bestimme anhand der Suchanfrage die **Wurzelmenge** aller Webseiten (Alle Webseiten, die den Suchbegriff enthalten)
2. Bestimme anhand der Wurzelmenge die **Basismenge** aller Webseiten. Diese enthält die Wurzelmenge und sämtliche Webseiten, welche zu dieser verlinkt sind
3. Berechne Hub- und Authority-Scores der Webseiten in der Basismenge
4. Gib die Webseiten in **absteigender Reihenfolge** entsprechend ihren **Authority-Scores** aus

* Eine Webseite mit **gutem Authority-Score** könnte unter Umständen den Text der Suchanfrage gar nicht enthalten
* Falls eine Webseite mit **guten Hub-Score** den Suchtext enthält, sind häufig auch die **Authorities** gut, zu welchen die Webseiten einen Link besitzt
* Falls eine Webseite einen **guten Authority-Score** hat, sind häufig auch die **Hubs** gut, welche einen Link auf diese Webseite besitzen

## Vergleich PageRank – HITS

### PageRank

* Sehr große Matrix
* Hoher initialer Berechnungsaufwand
* Beantwortet Anfragen online sehr schnell
* Konvergenzgeschwindigkeit justierbar
* Weniger anfällig für Link-Spamming
* Berechnet nur Authorities

### Hits

* Kleine Matrizen
* Kann Semantik von Anfragen berücksichtigen
* **Schwierig in Echtzeit**
* Anfällig für Link-Spamming
* Mindestens gleiche Ergebnisqualität wie PageRank
* **Tightly-Knit-Community-Effekt** (TKC-Effekt)
  + **HITS** bevorzugen dicht vernetzte Gruppen
  + Bewerten kleine vollständige bipartite Graphen sehr hoch
  + Problematische Folgen:
    - Kleine Gruppen können TKC-Effekt für Manipulationen nutzen
    - Topic-Drift: Anfrageergebnisse verschieben sich zu themenfremde dichtere Communities
    - Polarisierte Communities verlinken sich nicht

# Personalisierung

## Ziel: Auflösen der inhärenten Ambiguität von Suche

* Passendes Suchziel finden (Java)
* Suchergebnisse können vom aktuellen Kontext abhängen (der nicht konstant ist und sich mit der Zeit verändert)

## Dimensionen von Personalisierter Suche

* **Verschiedene Arten des Benutzerkontextes:**
  + **Global:** Hintergrund des Benutzers, Langzeitprofil
  + **Sitzung:** Menge der Anfragen mit ähnlichen Bedürfnissen
  + **Anfrage:** verwende letzte Anfrage und folgende Aktionen/Clicks
* Jeweils
  + Nur für Suchen
  + Für alle Browseraktionen
  + Für (andere/alle) Aktionen
* Kontext kann an verschiedenen Stellen gesammelt und genutzt werden
  + Dienstanbieter vs. Webserver vs. Lokaler Rechner
* Kontext kann auf verschiedene Weise genutzt werden:
  + Modifiziere Anfrage
  + Verändere Rankig der Ergebnisse („reranking“)

## Einfache Personalisierung: Relevance Feedback

* Sammle **Feedback** des Benutzer für Anfrageergebnisse
  + Explizites Feedback (Knopf im Interface)
  + Implizites Feedback (Klicks des Benutzers)
* Generiere **verbesserte Anfrage**
  + Füge neue Terme hinzu
  + Lösche existierende Terme
  + Ändere Gewicht von Termen

### Implizites Feedback durch Clicks

* **Geclickte Ergebnisse** sind **relevant** für die Anfrage
  + Außer der Benutzer hat die Seite sofort wieder verlassen
* **Nicht geclickte** Ergebnisse erlauben **keine Aussage**
  + Benutzer könnte sie sofort als nichtrelevant erkannt haben
  + Benutzer könnte das Ergebnis bereits kennen
  + Benutzer könnte das Ergebnis überhaupt nicht angesehen haben
* **Verbessertes impliziertes Feedback**
  + Wie lange bleibt ein Benutzer auf einer Seite
  + Wohin scrollt er, welche Bereiche sieht er wie lange an
  + Mausbewegungen, z.B. über Textstellen
  + Mausklicks
  + Geklickte Links
  + => Erlaubt eine bessere Schätzung der Relevanz

## Einfacher Einsatz von Feedback: Promoting

Idee: Verschiebe Ergebnisse mit **positivem Feedback** nach oben

* **Lokal** für jeden Benutzer einzeln:
  + Speichere Feedback für jeden Benutzer (z.B. Clicks auf Ergebnisse)
  + Verschiebe Ergebnisse mit Feedback nach oben, wenn Anfrage wiederkommt
  + Nutz Gewohnheit der Benutzer
* **Global** für alle Benutzer:
  + Sammle Feedback für häufige Anfragen
  + Verschiebe Ergebnisse mit Feedback der „meisten“ Benutzer nach oben
  + Funktioniert nicht gut für Anfragen mit unklarer Bedeutung
* => Ansatz basiert ausschließlich auf Reranking, Anfrage bleibt unverändert

## Benutzerprofile

Ziel: Konstruiere eine **Zusammenfassung der Interessen** eines Benutzers

* Aus den bisherigen Anfragen
* Aus den bisher zugegriffenen Seiten
* Aus den Dokumenten, den Mails, etc.

## Persistente vs. Sitzungsprofile

* Langzweitinteressen des Benutzers können sich von seinen Interessen in der aktuellen Sitzung unterscheiden
* => Verwaltet zwei Profile
  + **Sitzungsprofil**
    - Betrachte nur die Seiten, auf die in der aktuellen Sitzung zugegriffen wurden
    - Sitzungsgrenze durch Zeit oder inhaltliche Kohärenz
  + **Persistentes Langzeit-Profil**
    - Betrachte alle Seiten, auf die der Benutzer jemals zugegriffen hat
    - Geringeres Gewicht für alte Seiten
  + Profil ist Mischung von Sitzungsprofil und Langzeitprofil

## Personalisierung mit Benutzerprofilen

**Reranking** der Suchergebnisse basierend auf Übereinstimmung mit dem Profil

* Berechne vollständige Ergebnismenge R für die Anfrage
* Berechne für jedes Ergebnis p seine Ähnlichkeit mit dem Profilvektor (z.B. Cosinus-Ähnlichkeit)
* Sortiere Ergebnisse nach absteigender Ähnlichkeit

## Probleme beim Reranking: Ähnliche Ergebnisse

Reranking kann nicht funktionieren, wenn alle Ergebnisse ähnlich sind (und nicht relevant für die Anfrage)

## Diversifizierungsansatz

# Personalisierung – Empfehlungen

**Input – Output**

## Drei orthogonale Ansätze

* Coolaboratives Filtering („nächste Nachbarn“)
  + Benutzer A mag Item X .. Benutzer B ähnlich => Benutzer B könnte X mögen
* Content-based Filtering
  + Benutzer A mag Item X => Item X ähnlich zu Item Y
* Statischer Ansatz
  + Viele Leute kaufen x

### Kollaboratives Filtern

* Aktionen der Benutzer sind hochgradig dynamisch
  + Schwierig, Ähnlichkeiten vorauszuberechnen und zu aktualisieren
* Eine Empfehlung benötigt O(n+m) Zeit
* Empfehlungen müssen in Echtzeit berechnet werden

## Content-Based Filtering

* Beziehungen zwischen Items ist viel weniger dynamisch als die Beziehung zwischen Benutzern
* Ähnlichkeit ähnlich wie Benutzerähnlichkeit

## Offline-Evaluation vs. Benutzerexpermimente

* **Offline-Evaluation: Vergleiche** die vorhergesagte Bewertung mit der tatsächlichen Bewertung durch den Benutzer
* **Live-Experiment mit Benutzer: Frage** den Benutzer nach der Meinung oder **beobachte** Verhalten

## Probleme der Personalisierung

* Ansatz fokussiert auf **Maximierung der kurzfristigen Benutzerzufriedenheit**
* Teil der möglichen Ergebnisse wird ausgeblendet, weil das System sie für **nicht relevant für die Benutzer** hält
* Problematik der „**Filter-Bubble“**