

## Kapitel 2: Suchmaschinentechnologie für Intranets und das Web

- 2.1 Information-Retrieval-Systeme
- 2.2 Web-Crawling und Indexierung
- 2.3 Vektorraummodell für IR mit Ranking
- 2.4 Anfrageausführung mit Ranking
- 2.5 Grundlagen aus der Linearen Algebra
- 2.6 Latent Semantic Indexing

Informationssysteme SS 2004

2-1

## 2.1 Information-Retrieval-Systeme

Information Retrieval (IR) ist die Technologie zum Suchen in Kollektionen (Korpora, Intranets, Web) schwach strukturierter Dokumente: Text, HTML, XML, ...

Darunter fällt auch:

- Text- und Strukturanalyse
- Inhaltserschließung und -repräsentation
- Gruppierung und Klassifikation
- Zusammenfassung
- Filtern und Personalisieren (z.B. von Nachrichten-"Feeds")
- „Routing“ (Metasuche)

*Globales Ziel:  
Informationsbedürfnisse befriedigen - und dabei  
Besetzung des Engpasses (teurer) intellektueller Zeit !*

Informationssysteme SS 2004

2-2

## Schnittstellen von IR-Systemen

- **Ausgabe:**
  - Menge von Dokumenten, die Suchstring(s) enthalten: **Freitextsuche**
  - Menge inhaltlich relevanter Dokumente: **Inhaltssuche**
    - ungeordnete Menge: **Boolesches Retrieval**
  - nach Relevanz absteigend sortierte Rangliste: **Ranked Retrieval** (Ähnlichkeitssuche)
- **Eingabe:**
  - Keywords (positiv/negativ) (plus Phrasen, ganze Sätze)
  - (Boolesche) Ausdrücke über Keyword-Bedingungen
  - Strukturbedingungen (z.B. Tags, Links)
  - ontologisch basierte Bedingungen
  - Suchsprache (z.B. SQL mit interMedia)

Informationssysteme SS 2004

2-3

## Beispiel: SQL in Oracle/Text

Beispielanfrage: `Select URL, Content, Year From Docs  
Where Year > 1995 And Category Like ,%drama%'  
And Contains (Content, ,BT(king)', 1) > 10  
And Contains (Content, ,SYN(traitor)NT(traitor)', 2) > 0  
Order By Score(1)*Score(2)`

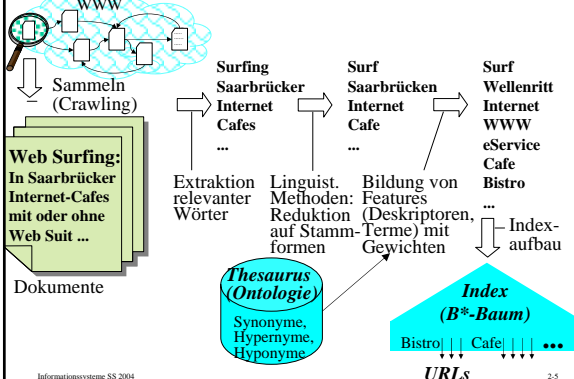
Weitere Operatoren (anhand von Beispielen):

~, &,	Not, And, Or
NEAR (king, David, 10)	höchstens 10 Wörter auseinander
king&David WITHIN Sentence	im selben Satz
!dog	ähnliche Aussprache (z.B. doc, dock)
\$sing	gleicher Wortstamm (z.B. singer, sings, sang)
?apple	ähnliche Schreibweise (z.B. applet, apply)
NTP(computer)	narrower term partative (z.B. hard drive)
NTG(rodent)	narrower term generic (z.B. rat)
NTI(fairytale)	narrower term instance (z.B. Cinderella)
ABOUT(miracles by Jesus)	thematische Suche (verwendet intern selbst andere Operatoren)

Informationssysteme SS 2004

2-4

## Sammeln, Analyse und Indexierung von Web-Dokumenten



Informationssysteme SS 2004

2-5

## Problem: Inhaltserschließung

Umgang mit „unscharfen“ Daten (und „unscharfen“ Anfragen)

→ Dokumente werden typischerweise durch

**Features** charakterisiert, z.B.:

- Wörter, Wortpaare oder Phrasen
- Worthäufigkeiten
- Anzahl eingehender Hyperlinks
- title, weitere Tags, Struktur von HTML- oder XML-Seiten
- Farbhäufigkeiten in Bildern (Bildmitte, oberer Rand, etc.)
- usw. usw.

→ Abbildung von natürlichsprachlichem Text auf Features:

- Behandlung von morphologischer Variation
- Behandlung von Synonymen, Hypernymen/Hyponymen und Polysemen (u.a. mittels Thesaurus)

Informationssysteme SS 2004

2-6

**Problem: Effektivität (Suchresultatsgüte)**  
*query = „Chernoff theorem“*

**AltaVista:** Fermat's last theorem. Previous topic. Next topic. ...  
 URL: [www.groups.dcs.st-and.ac.uk/~history/His...st\\_theorem.html](http://www.groups.dcs.st-and.ac.uk/~history/His...st_theorem.html)

**Northernlight:** J. D. Biggins- Publications. Articles on the Branching Random Walk  
<http://www.shef.ac.uk/~st1jdb/bibliog.html>

**Excite:** The Official Web Site of Playboy Lingerie Model Mikki Chernoff <http://www.mikkichernoff.com/>

**Google:** ...strong convergence \cite{Chernoff}. \begin{theorem} \label{T1} Let...  
[http://mpej.unige.ch/mp\\_arc/p/00-277](http://mpej.unige.ch/mp_arc/p/00-277)

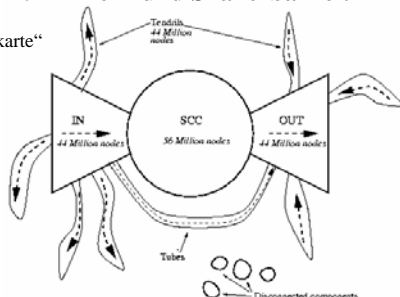
**Yahoo:** Moment-generating Functions; Chernoff's Theorem;  
<http://www.siam.org/catalog/mcc10/bahadur.htm>

**Mathsearch:** No matches found.

Informationssysteme SS 2004 2-7

**Problem: Effizienz und Skalierbarkeit**

Aktuelle „Landkarte“ des Webs:



!!! Aber:

Suchmaschinen überdecken das „Surface Web“:  
 4 Mrd. Dokumente, 20 TBytes  
 Die meisten Daten sind im „Deep Web“ hinter Portalen:  
 500 Mrd. Dokumente, 8 PBytes

Informationssysteme SS 2004 2-8

**Bewertung der Retrieval-Güte (Effektivität)**

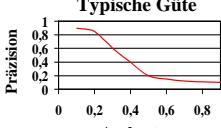
Fähigkeit, zu einer Anfrage **nur** relevante Dokumente zu liefern:

$$\text{Präzision (precision)} = \frac{\text{Anzahl relevanter Dokumente unter Top } r}{r}$$

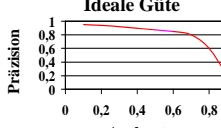
Fähigkeit, zu einer Anfrage **alle** relevanten Dokumente zu liefern:

$$\text{Ausbeute (recall)} = \frac{\text{Anzahl relevanter Dokumente}}{\text{Anzahl aller relevanten Dokumente}}$$

**Typische Güte**



**Ideale Güte**



Informationssysteme SS 2004 2-9

**2.2 Web-Crawling und Indexierung**

**Komponenten einer Web-Suchmaschine:**

- **URL Queue Server:** verwaltet Priority-Queue (noch) zu traversierender Links
- **Crawler (Robot, Spider):** holt Dokumente unter Beachtung von Nebenbedingungen (Filetyp, Robot Exclusion Protocol, usw.)
- **Repository Server:** verwaltet DocumentRepository
- **Indexer (inkl. Parser, Stemmer):** analysiert Dokumente und erzeugt Einträge in Lexicon, Anchors und DocumentIndex
- **URL Resolver:** übersetzt URLs in DocIds
- **Link Analyzer:** berechnet Autoritäts-Ranking aufgrund von Links
- **Query Processor:** wertet Anfragen durch Index-Lookups aus und berechnet Resultats-Ranking

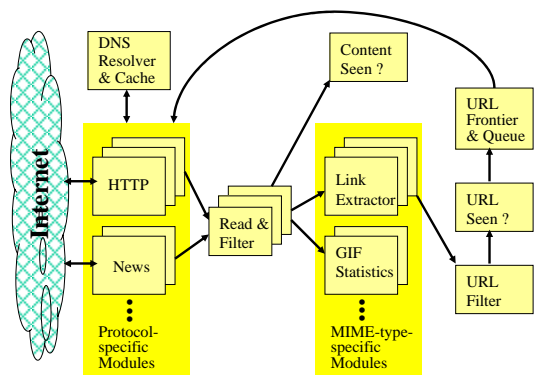
Informationssysteme SS 2004 2-10

**Datenstrukturen einer Web-Suchmaschine**

- **DocumentRepository (DocId, DocContent):**  
 alle (HTML-) Dokumente in komprimierter Form
- **Lexicon (TermId, Term):**  
 alle vorkommenden Stammformen jeweils mit TermId
- **DocumentIndex (DocId, TermId, Weight, ...):**  
 alle Vorkommen von Termen in Dokumenten, optimiert für Zugriff nach DocId
- **TermIndex (TermId, DocId, Weight, ...):**  
 alle Dokumente zu allen Termen, optimiert für Zugriff nach TermId
- **Anchors (SourceDocId, TargetDocId, AnchorText):**  
 alle Hyperlinks
- **URLIndex (URL, DocId):**  
 Umsetzung von URLs auf interne Ids

Informationssysteme SS 2004 2-11

**Architektur eines skalierbaren Crawlers**



Informationssysteme SS 2004 2-12

## Dimensionen sehr großer Web-Suchmaschinen

- > 4 Mrd. Web-Dokumente + 1 Mrd. News-Dokumente
- > 20 Terabytes Rohdaten
- > 10 Mio. Terme
- > 4 Terabytes Index
- > 150 Mio. Anfragen pro Werktag
- < 1 Sek. mittlere Antwortzeit
- < 30 Tage Indexaktualität
- > 1000 Webseiten pro Sek. Crawling

High-End-Server-Farm:

- > 10 000 Intel-Server mit jeweils
- > 1 GB Hauptspeicher, 2 Platten und
- partitionierten, gespiegelten Daten, die über alle Server verteilt sind,
- sowie Lastbalancierung der Queries, Remote-Administration, usw.

Informationssysteme SS 2004

2-13

## 2.3 Vektorraummodell für IR mit Ranking

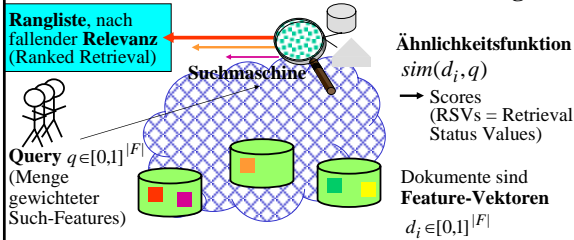
### Grundprinzipien:

- **Featureraum:** Wörter in Dokumenten werden auf Terme reduziert.
- **Dokumentenmodell:** Jedes Dokument  $d_i$  wird als Vektor  $\in [0,1]^{|F|}$  repräsentiert, wobei  $d_{ij}$  das Gewicht des  $j$ -ten Terms in  $d_i$  angibt.
- **Anfragemodell:** Anfragen sind Vektoren  $q \in [0,1]^{|F|}$
- **Relevanz:** Suchresultatsranking basiert auf einer Ähnlichkeitsfunktion im Vektorraum  $[0,1]^{|F|}$
- **Crawling:** Das Web wird entlang von Hyperlinks traversiert, um Dokumente zu analysieren und zu indexieren.
- **Indexierung:** Zu jedem Term wird eine Liste von Dokumenten-Ids (z.B. URLs) mit dem jeweiligen Gewicht in einem „invertierten File“ (Suchbaum oder Hash-File) angelegt.
- **Anfrageverarbeitung:** Anfragen werden zerlegt in Index-Lookups für Einzeltermine, um Trefferkandidatenlisten zu bestimmen.

Informationssysteme SS 2004

2-14

## Vektorraummodell für Relevanz-Ranking



Verwendete Ähnlichkeitsfunktionen sind z.B.:

$$sim(d_i, q) := \sum_{j=1}^{|F|} d_{ij} q_j \quad \text{(Skalarprodukt)}$$

$$sim(d_i, q) := \frac{\sum_{j=1}^{|F|} d_{ij} q_j}{\sqrt{\sum_{j=1}^{|F|} d_{ij}^2 \sum_{j=1}^{|F|} q_j^2}} \quad \text{(Cosinus-Maß)}$$

Informationssysteme SS 2004

2-15

## Termgewichtung in Dokumenten

Betrachtet werden die folgenden Werte

(für N Dokumente und M Terme):

- $tf_{ij}$ : Häufigkeit (term frequency) des Terms  $t_i$  in Dokument  $d_j$
- $df_i$ : Anzahl der Dokumente mit Term  $t_i$  (doc. frequency)
- $idf_i$ :  $N / df_i$  (inverse document frequency)
- $cf_i$ : Häufigkeit von  $t_i$  in allen Dokumenten (corpus frequency) (ggf. mit separater Berücksichtigung von Termen in title u.ä.)

### Grundprinzip:

Das Gewicht  $w_{ij}$  von Term  $t_i$  in Dokument  $d_j$  sollte mit  $tf_{ij}$  und mit  $idf_i$  monoton wachsen.

→ erster Ansatz:  $w_{ij} = tf_{ij} * idf_i$  (**tf-idf-Formel**)

Ggf. sollten die Gewichte  $w_{ij}$  wie folgt zu  $\omega_{ij}$  normiert werden:

$$\omega_{ij} := w_{ij} / \sqrt{\sum_k w_{kj}^2}$$

Informationssysteme SS 2004

2-16

## Variationen der Termgewichtung mit tf und idf

Empirische Resultate zeigen, daß in der Regel die tf- und idf-Werte normalisiert und/oder gedämpft sein sollten.

Normalisierung tf-Werte:  $tf_{ij} := \frac{tf_{ij}}{\max_k tf_{kj}}$

Gedämpfte tf-Werte:  $tf_{ij} := (1 + \log tf_{ij})$

Gedämpfte idf-Werte:  $idf_i := \log \frac{N}{df_i}$

Häufige Variante:

$$w_{ij} := \frac{tf_{ij}}{\max_k tf_{kj}} \log \frac{N}{df_i} \quad \text{tf*idf-Formel}$$

$$\omega_{ij} := w_{ij} / \sqrt{\sum_k w_{kj}^2}$$

Informationssysteme SS 2004

2-17

## Query-Verfeinerung mit Relevanz-Feedback nach Rocchio

Für Resultat D der Query q bestehe das Relevanz-Feedback des Benutzers aus einer Partitionierung von D in

- $D^+$ : die Menge der relevanten Dokumente in D und
- $D^-$ : die Menge der nicht relevanten Dokumente in D.

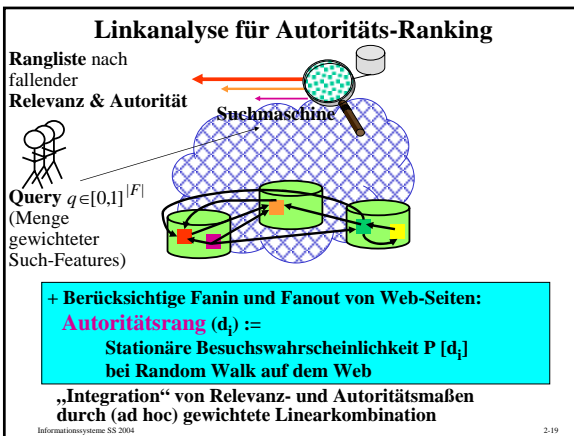
Generiere verfeinerte Query  $q'$ :

$$\vec{q}' := \alpha \vec{q} + \frac{\beta}{|D^+|} \sum_{d_j \in D^+} \vec{d}_j - \frac{\gamma}{|D^-|} \sum_{d_j \in D^-} \vec{d}_j$$

mit geeigneten Gewichten  $\alpha, \beta, \gamma \in [0,1]$  (typischerweise  $\alpha > \beta > \gamma$ )

Informationssysteme SS 2004

2-18



### Weitere IR-Modelle

- **Probabilistisches Retrieval & Statistische Sprachmodelle:**  
Ranking aufgrund von Relevanzwahrscheinlichkeiten, die aus - geschätzten - Basisparametern abgeleitet werden.
- **Fuzzy-Set-Modell:**  
Queries (inkl. einzelner Terme) beschreiben Fuzzy-Mengen mit Dokumenten als Elementen vom Grad  $\mu \in [0,1]$ . Mengenoperationen verwenden Funktionen max, min, 1- $\mu$ .
- **Latent Semantic Indexing:**  
Berücksichtigung von Termkorrelationen durch Transformation des Term-Vektorraums in einen Themen-Vektorraum niedrigerer Dimensionalität
- **Neuronale Netze** und andere Inferenznetze zum Lernen von Termgewichten

Informationssysteme SS 2004 2-20

### 2.4 Anfrageausführung mit Ranking

Suchbaum

algorithm	...	performance	...	z-transform
17: 0.3		12: 0.5		11: 0.6
44: 0.4		14: 0.4		17: 0.1
52: 0.1		28: 0.1		28: 0.7
53: 0.8		44: 0.2		...
55: 0.6		51: 0.6		
...		52: 0.3		

Indexliste mit (DocId, tf\*idf) sortiert nach DocId

Gegeben: Query  $q = t_1 t_2 \dots t_z$  mit  $z$  Keywords  
 Ähnlichkeitsfunktion  $\text{score}(q,d)$  für Dok.  $d \in D$ , z.B.:  $\bar{q} \cdot \bar{d}$

Finde: Top-k-Resultate bzgl.  $\text{score}(q,d)$  (z.B.:  $\sum_{i \in q} s_i(d)$ )

**Naiver QP Algorithmus:**  
 candidate-docs :=  $\emptyset$ ;  
 for  $i=1$  to  $z$  do {  
   candidate-docs := candidate-docs  $\cup$  index-lookup( $t_i$ ) };  
 for each  $d_j \in$  candidate-docs do {compute  $\text{score}(q,d_j)$  };  
 sort candidate-docs by  $\text{score}(q,d_j)$  descending;

Informationssysteme SS 2004 2-21

### Fagin's TA (Threshold Algorithm)

scan all lists  $L_i$  ( $i=1..m$ ) in parallel:  
 consider  $d_j$  at position  $\text{pos}_i$  in  $L_i$ ;  
 $\text{high}_i := s_i(d_j)$ ;  
 if  $d_j \notin \text{top-k}$  then {  
   look up  $s_v(d_j)$  in all lists  $L_v$  with  $v \neq i$ ; // random access  
   compute  $s(d_j) := \text{aggr} \{s_v(d_j) \mid v=1..m\}$ ;  
   if  $s(d_j) > \text{min score among top-k}$  then  
     add  $d_j$  to top-k and remove min-score  $d$  from top-k; }  
 if  $\text{min score among top-k} \geq \text{aggr} \{ \text{high}_v \mid v=1..m \}$  then exit;

	L1	L2	L3	top-k:
m=3	f: 0.5	a: 0.55	h: 0.35	
aggr: sum	b: 0.4	b: 0.2	d: 0.35	-f: 0.75-
k=2	c: 0.35	f: 0.2	b: 0.2	a: 0.95
	a: 0.3	g: 0.2	a: 0.1	b: 0.8
	h: 0.1	c: 0.1	c: 0.05	
	d: 0.1		f: 0.05	

Informationssysteme SS 2004 2-22

### TA-Sorted (Fagin, Güntzer et al., ...)

scan index lists in parallel:  
 consider  $d_j$  at position  $\text{pos}_i$  in  $L_i$ ;  
 $E(d_j) := E(d_j) \cup \{i\}$ ;  $\text{high}_i := s_i(q,d_j)$ ;  
 $\text{bestscore}(d_j) := \text{aggr}\{x_1, \dots, x_m\}$   
 with  $x_i := s_i(q,d_j)$  for  $i \in E(d_j)$ ,  $\text{high}_i$  for  $i \notin E(d_j)$ ;  
 $\text{worstscore}(d_j) := \text{aggr}\{x_1, \dots, x_m\}$   
 with  $x_i := s_i(q,d_j)$  for  $i \in E(d_j)$ , 0 for  $i \notin E(d_j)$ ;  
 top-k := k docs with largest worstscore;  
 if  $\text{min worstscore among top-k} \geq \text{bestscore}\{d \mid d \text{ not in top-k}\}$   
 then exit;

	L1	L2	L3	top-k:
m=3	f: 0.5	a: 0.55	h: 0.35	a: 0.95
aggr: sum	b: 0.4	b: 0.2	d: 0.35	b: 0.8
k=2	c: 0.35	f: 0.2	b: 0.2	
	a: 0.3	g: 0.2	a: 0.1	
	h: 0.1	c: 0.1	c: 0.05	
	d: 0.1		f: 0.05	

Informationssysteme SS 2004 2-24

### 2.5 Grundlagen aus der Linearen Algebra

Eine Menge  $S$  von Vektoren heißt **linear unabhängig**, wenn sich kein  $x \in S$  als Linearkombination der anderen Vektoren aus  $S$  schreiben lässt. Der **Rang** einer Matrix  $A$  ist die maximale Anzahl linear unabhängiger Zeilen- oder Spaltenvektoren.

Eine **Basis** einer  $n \times n$ -Matrix  $A$  ist eine Menge  $S$  von Zeilen- bzw. Spaltenvektoren, so dass alle Zeilen bzw. Spalten Linearkombinationen der Vektoren aus  $S$  ist.

Eine Menge  $S$  von  $n \times 1$ -Vektoren heißt **Orthonormalbasis**, wenn für alle  $x, y \in S$  gilt:

$$\|x\|_2 := \sqrt{\sum_{i=1}^n x_i^2} = 1 = \|y\|_2 \quad \text{und} \quad x \cdot y = 0$$

Informationssysteme SS 2004 2-24

## Eigenwerte und Eigenvektoren

Seien  $A$  eine reellwertige  $n \times n$ -Matrix,  $x$  ein reellwertiger  $n \times 1$ -Vektor und  $\lambda$  ein reeller Skalarwert. Die Lösungen  $x$  und  $\lambda$  der Gleichung  $A \times x = \lambda x$  heißen **Eigenvektor** bzw. **Eigenwert** von  $A$ . Die Eigenvektoren von  $A$  sind Vektoren, deren Richtungen bei der durch  $A$  beschriebenen Linearabbildung erhalten bleiben.

Die Eigenwerte von  $A$  sind die Nullstellen des charakteristischen Polynoms  $f(\lambda)$  von  $A$ :  $f(\lambda) = |A - \lambda I| = 0$

mit der Determinantenentwicklung nach der  $i$ -ten Zeile:

$$|A| = \sum_{j=1}^n (-1)^{i+j} a_{ij} |A^{(ij)}| \quad \text{wobei man die Matrix } A^{(ij)} \text{ aus } A \text{ durch Streichung der } i. \text{ Zeile und der } j. \text{ Spalte erhält}$$

Die reellwertige  $n \times n$ -Matrix  $A$  heißt **symmetrisch**, wenn  $a_{ij} = a_{ji}$  für alle  $i, j$ .  $A$  heißt **positiv definit**, wenn für alle  $n \times 1$ -Vektoren  $x \neq 0$  gilt:  $x^T \times A \times x > 0$ .

Wenn  $A$  symmetrisch ist, sind alle Eigenwerte von  $A$  reell.

Wenn  $A$  symmetrisch und positiv definit ist, sind alle Eigenwerte positiv.

Informationssysteme SS 2004

2-25

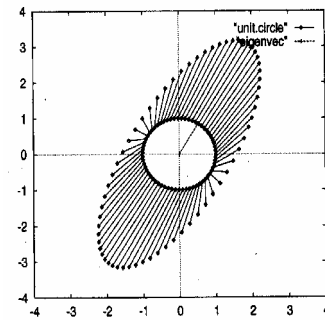
## Illustration von Eigenvektoren

$$\text{Matrix } A = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}$$

beschreibt  
affine Abb.  $x \mapsto Ax$

Eigenvektor  
 $x_1 = (0.52 \ 0.85)^T$   
zum Eigenwert  $\lambda_1 = 3.62$

Eigenvektor  
 $x_2 = (0.85 \ -0.52)^T$   
zum Eigenwert  $\lambda_2 = 1.38$



Informationssysteme SS 2004

2-26

## Spektralsatz der Linearen Algebra

### Spektralsatz

(*Hauptachsentransformation, Principal Component Analysis, PCA*):

Sei  $A$  eine symmetrische  $n \times n$ -Matrix mit Eigenwerten  $\lambda_1, \dots, \lambda_n$  und Eigenvektoren  $x_1, \dots, x_n$ , so dass  $\|x_i\|_2 = 1$  für alle  $i$ .

Die Eigenvektoren bilden eine Orthonormalbasis von  $A$ .

Dann gilt:

$$D = Q^T \times A \times Q,$$

wobei  $D$  eine Diagonalmatrix ist mit den Diagonalelementen  $\lambda_1, \dots, \lambda_n$  und  $Q$  aus den Spaltenvektoren  $x_1, \dots, x_n$  besteht.

Informationssysteme SS 2004

2-27

## 2.6 Latent Semantic Indexing (LSI): Grundidee

### Ziel:

Transformation der Dokumentvektoren vom hochdimensionalen Termvektorraum in einen **Themenvektorraum** niedrigerer Dimensionalität unter

- Ausnutzung von Korrelationen zwischen Termen (z.B. „Web“ und „Internet“ häufig zusammen)
- implizite Differenzierung von Polysemen, die sich in ihren Korrelationen mit anderen Termen unterscheiden (z.B. „Java“ mit „Library“ vs. „Java“ mit „Kona Blend“ vs. „Java“ mit „Borneo“)

### mathematisch:

gegeben:  $m$  Terme,  $n$  Dokumente (i.d.R.  $n > m$ ) und eine  $m \times n$ -Term-Dokument-Ähnlichkeitsmatrix  $A$ ,  
gesucht: möglichst gute – ähnlichkeitsbewahrende – Abbildung der Spaltenvektoren von  $A$  in einen  $k$ -dimensionalen Vektorraum ( $k \ll m$ ) für gegebenes  $k$

Informationssysteme SS 2004

2-28

## Exkurs: Singulärwertdekomposition (SVD)

### Satz:

Jede reellwertige  $m \times n$ -Matrix  $A$  mit Rang  $r$  kann zerlegt werden in die Form  $A = U \times \Delta \times V^T$  mit einer  $m \times r$ -Matrix  $U$  mit orthonormalen Spaltenvektoren, einer  $r \times r$ -Diagonalmatrix  $\Delta$  und einer  $n \times r$ -Matrix  $V$  mit orthonormalen Spaltenvektoren. Diese Zerlegung heißt Singulärwertdekomposition und ist eindeutig, wenn die Elemente von  $\Delta$  der Größe nach geordnet werden.

### Satz:

In der Singulärwertdekomposition  $A = U \times \Delta \times V^T$  der Matrix  $A$  sind  $U$ ,  $\Delta$  und  $V$  wie folgt bestimmt:

- $\Delta$  besteht aus den Singulärwerten von  $A$ , d.h. den positiven Wurzeln der Eigenwerte von  $A^T \times A$ ,
- die Spaltenvektoren von  $U$  sind die Eigenvektoren von  $A \times A^T$ ,
- die Spaltenvektoren von  $V$  sind die Eigenvektoren von  $A^T \times A$ .

Informationssysteme SS 2004

2-29

## Schematische Darstellung der SVD

$$\begin{pmatrix} \text{Term-Dokument-Ähnlichkeitsmatrix} \end{pmatrix}_{m \times n} = \begin{pmatrix} \text{Term-Themen-Ähnlichkeit} \end{pmatrix}_{m \times r} \begin{pmatrix} \sigma_1 & & 0 \\ & \ddots & \\ 0 & & \sigma_r \end{pmatrix}_{r \times r} \begin{pmatrix} \text{Themen-Dokument-Ähnlichkeit} \end{pmatrix}_{r \times n}$$

$U \qquad \qquad \Delta \qquad \qquad V^T$

Informationssysteme SS 2004

2-30

## Exkurs: SVD als Regressionsverfahren

### Satz:

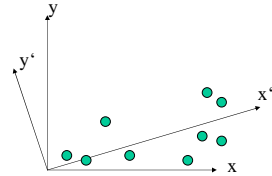
Sei  $A$  eine  $m \times n$ -Matrix mit Rang  $r$  und sei  $A_k = U_k \times \Delta_k \times V_k^T$ , wobei die  $k \times k$ -Diagonalmatrix  $\Delta_k$  die  $k$  größten Singulärwerte von  $A$  enthält und die  $m \times k$ -Matrix  $U_k$  sowie die  $n \times k$ -Matrix  $V_k$  aus den zugehörigen Eigenvektoren der Singulärwertdekomposition von  $A$  bestehen.

Unter allen  $m \times n$ -Matrizen  $C$  mit einem Rang, der nicht größer als  $k$  ist, ist  $A_k$  diejenige Matrix, die den Wert

$$\|A - C\|_F^2 = \sum_{i=1}^m \sum_{j=1}^n (A_{ij} - C_{ij})^2$$

minimiert (die Frobenius-Norm).

Beispiel:  
 $m=2, n=8, k=1$   
 Projektion auf  $x'$ -Achse  
 minimiert „Fehler“ bzw.  
 maximiert Varianz  
 im  $k$ -dimensionalen Raum



Informationssysteme SS 2004

2-31

$r \times q$

## Anwendung der SVD auf das Vektorraummodell

$A$  ist die  $m \times n$ -Term-Dokument-Ähnlichkeitsmatrix. Dann sind:

- $U$  bzw.  $U_k$  die  $m \times r$ - bzw.  $m \times k$ -Term-Themen-Ähnlichkeitsmatrix,
- $V$  bzw.  $V_k$  die  $n \times r$ - bzw.  $n \times k$ -Dokument-Themen-Ähnlichkeitsmatrix
- $A \times A^T$  bzw.  $A_k \times A_k^T$  die  $m \times m$ -Term-Term-Ähnlichkeitsmatrix,
- $A^T \times A$  bzw.  $A_k^T \times A_k$  die  $n \times n$ -Dokument-Dokument-Ähnlichkeitsmatrix

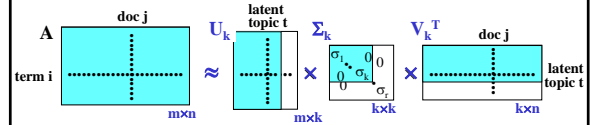


Abbildung von  $m \times 1$ -Vektoren in Themenraum:  $d_j \mapsto U_k^T \times d_j =: d_j'$   
 $q \mapsto U_k^T \times q =: q'$

Skalarprodukt-Ähnlichkeit im Themenraum:  $d_j'^T \times q' = ((\Delta_k V_k^T)_{kj})^T \times q'$

Informationssysteme SS 2004

2-32

## Indexierung und Anfrageauswertung

- Die Matrix  $\Delta_k V_k^T$  entspricht einem „Themen-Index“ und ist in einer geeigneten Datenstruktur zu verwalten.  
 Statt  $\Delta_k V_k^T$  kann man auch vereinfachend  $V_k^T$  als Index verwenden.
- Zusätzlich muß die Term-Themen-Abbildung  $U_k$  gespeichert werden.
- Eine Anfrage  $q$  (ein  $m \times 1$ -Spaltenvektor) im Termvektorraum wird in die Anfrage  $q' = U_k^T \times q$  (ein  $k \times 1$ -Spaltenvektor) transformiert und dann im Themenvektorraum (also  $V_k$ ) ausgewertet (z.B. mittels Skalarproduktmaß  $V_k^T \times q'$  oder Cosinusmaß)
- Ein neues Dokument  $d$  (ein  $m \times 1$ -Spaltenvektor) wird in  $d' = U_k^T \times d$  (ein  $k \times 1$ -Spaltenvektor) transformiert und als neue Spalte an den „Index“  $V_k^T$  angefügt („folding-in“)

Informationssysteme SS 2004

2-33

## Beispiel 1 für Latent Semantic Indexing

$m=5$  (Bush, Schröder, Korea, Klose, Völler),  $n=7$

$$A = \begin{pmatrix} 1 & 2 & 1 & 5 & 0 & 0 & 0 \\ 1 & 2 & 1 & 5 & 0 & 0 & 0 \\ 1 & 2 & 1 & 5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 3 & 1 \\ 0 & 0 & 0 & 0 & 2 & 3 & 1 \end{pmatrix} = \begin{pmatrix} 0.58 & 0.00 \\ 0.58 & 0.00 \\ 0.58 & 0.00 \\ 0.00 & 0.71 \\ 0.00 & 0.71 \end{pmatrix} \times \begin{pmatrix} 9.64 & 0.00 \\ 0.00 & 5.29 \end{pmatrix} \times \begin{pmatrix} 0.18 & 0.36 & 0.18 & 0.90 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.53 & 0.80 & 0.27 \end{pmatrix}$$

$U \quad \Delta \quad V^T$

Anfrage  $q = (0 \ 0 \ 1 \ 0 \ 0)^T$  wird in

$q' = U^T \times q = (0.58 \ 0.00)^T$  transformiert und gegen  $V^T$  evaluiert.

Neues Dokument  $d8 = (1 \ 1 \ 0 \ 0 \ 0)^T$  wird in

$d8' = U^T \times d8 = (1.16 \ 0.00)^T$  transformiert und an  $V^T$  angefügt.

Informationssysteme SS 2004

2-34

## Beispiel 2 für Latent Semantic Indexing

$m=6$  terms

t1: bak(e,ing)  
 t2: recipe(s)  
 t3: bread  
 t4: cake  
 t5: pastr(y,ies)  
 t6: pie

$n=5$  documents

d1: How to bake bread without recipes  
 d2: The classic art of Viennese Pastry  
 d3: Numerical recipes: the art of scientific computing  
 d4: Breads, pastries, pies and cakes: quantity baking recipes  
 d5: Pastry: a book of best French recipes

$$A = \begin{pmatrix} 0.5774 & 0.0000 & 0.0000 & 0.4082 & 0.0000 \\ 0.5774 & 0.0000 & 1.0000 & 0.4082 & 0.7071 \\ 0.5774 & 0.0000 & 0.0000 & 0.4082 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.4082 & 0.0000 \\ 0.0000 & 1.0000 & 0.0000 & 0.4082 & 0.7071 \\ 0.0000 & 0.0000 & 0.0000 & 0.4082 & 0.0000 \end{pmatrix}$$

Informationssysteme SS 2004

2-35

## Beispiel 2 für Latent Semantic Indexing (2)

$$A = \begin{pmatrix} 0.2670 & -0.2567 & 0.5308 & -0.2847 \\ 0.7479 & -0.3981 & -0.5249 & 0.0816 \\ 0.2670 & -0.2567 & 0.5308 & -0.2847 \\ 0.1182 & -0.0127 & 0.2774 & 0.6394 \\ 0.5198 & 0.8423 & 0.0838 & -0.1158 \\ 0.1182 & -0.0127 & 0.2774 & 0.6394 \end{pmatrix} \times \begin{pmatrix} 1.6950 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 1.1158 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.8403 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.4195 \end{pmatrix} \times \begin{pmatrix} 0.4366 & 0.3067 & 0.4412 & 0.4909 & 0.5288 \\ -0.4717 & 0.7549 & -0.3568 & -0.0346 & 0.2815 \\ 0.3688 & 0.0998 & -0.6247 & 0.5711 & -0.3712 \\ -0.6715 & -0.2760 & 0.1945 & 0.6571 & -0.0577 \end{pmatrix}$$

$U \quad \Delta \quad V^T$

Informationssysteme SS 2004

2-36

### Beispiel 2 für Latent Semantic Indexing (3)

$$A_3 = \begin{pmatrix} 0.4971 & -0.0330 & 0.0232 & 0.4867 & -0.0069 \\ 0.6003 & 0.0094 & 0.9933 & 0.3858 & 0.7091 \\ 0.4971 & -0.0330 & 0.0232 & 0.4867 & -0.0069 \\ 0.1801 & 0.0740 & -0.0522 & 0.2320 & 0.0155 \\ -0.0326 & 0.9866 & 0.0094 & 0.4402 & 0.7043 \\ 0.1801 & 0.0740 & -0.0522 & 0.2320 & 0.0155 \end{pmatrix} = U_3 \times \Delta_3 \times V_3^T$$

Informationssysteme SS 2004

2-37

### Beispiel 2 für Latent Semantic Indexing (4)

Anfrage q: baking bread

$$q = (1 \ 0 \ 1 \ 0 \ 0 \ 0)^T$$

Transformation in den Themenraum mit k=3

$$q' = U_k^T \times q = (0.5340 \ -0.5134 \ 1.0616)^T$$

Skalarprodukt-Ähnlichkeit im Themenraum mit k=3:

$$\text{sim}(q, d1) = V_{k_1}^T \times q' \approx 0.86 \quad \text{sim}(q, d2) = V_{k_2}^T \times q' \approx -0.12$$

$$\text{sim}(q, d3) = V_{k_3}^T \times q' \approx -0.24 \quad \text{usw.}$$

Folding-in eines neuen Dokuments d6:

algorithmic recipes for the computation of pie

$$d6 = (0 \ 0.7071 \ 0 \ 0 \ 0 \ 0.7071)^T$$

Transformation in den Themenraum mit k=3

$$d6' = U_k^T \times d6 \approx (0.5 \ -0.28 \ -0.15)$$

d6' als neue Spalte an  $V_k^T$  anhängen

Informationssysteme SS 2004

2-38

### Mehrsprachiges Retrieval mit LSI

- Konstruiere LSI-Modell ( $U_k, \Delta_k, V_k^T$ ) anhand von Trainingsdokumenten, die mehrsprachig vorliegen:
  - Betrachte alle Sprachversionen eines Dokuments als ein einziges Dokument und
  - extrahiere zur Indexierung alle Terme oder Wörter unabhängig von der Sprache.
- Indexiere weitere Dokumente durch „folding-in“, also Abbildung in den Themen-Vektorraum und Anhängen an  $V_k^T$ .
- Anfragen können dann in beliebiger Sprache gestellt werden und liefern Antworten in allen Sprachen.

Beispiel:

d1: How to bake bread without recipes.

Wie man ohne Rezept Brot backen kann.

d2: Pastry: a book of best French recipes.

Gebäck: eine Sammlung der besten französischen Rezepte.

Terme sind dann z.B. bake, bread, recipe, backen, Brot, Rezept, usw. Dokumente und Terme werden auf einen kompakten Themenraum abgebildet.

Informationssysteme SS 2004

2-39

### Zusammenfassung zu LSI

- + Elegantes, mathematisch wohlfundiertes Modell
- + „Automatisches Lernen“ von Termkorrelationen (inkl. morphologischer Wortvarianten, Mehrsprachigkeit)
- + Impliziter Thesaurus (durch Korrelation von Synonymen)
- + Implizite Diskriminierung der verschiedenen Bedeutungen von Polysemen (durch verschiedene Korrelationen)
- + Verbesserte Retrievalgüte auf „geschlossenen“ Korpora (z.B. TREC-Benchmark, Finanznachrichten, Patentkollektionen u.ä.) mit empirisch günstigstem k in der Größenordnung 100-200
- Schwierige Wahl von günstigem k
- Rechen- und Speicheraufwand für sehr große (z.T. aber dünn besetzte) Matrizen
- Keine überzeugenden Resultate für Web-Suchmaschinen

Informationssysteme SS 2004

2-40