HUMBOLDT-UNIVERSITÄT ZU BERLIN

Information Retrieval

Modeling Information Retrieval 2

Ulf Leser

Content of this Lecture

- IR Models
- Boolean Model
- Vector Space Model
- Relevance Feedback in the VSM
- Probabilistic Model
- Latent Semantic Indexing
- Other IR Models
- Outlook: Word Semantics and Word Embeddings

A Probabilistic Interpretation of Relevance

- VSM is fairly heuristic – some kind of similarity with some kind of weighting in some vector space
- Probabilistic models build on well-established and mathematically consistent probability theory
  - Derive relevance formulas from a few basic and sound principles
- Probabilistic model
  - Words appearing in docs are seen as independent events
  - A doc (or query) is a conjunction of events
  - Compute the probability that a doc d is relevant to query q
    - Actually, we will compute a score (using probabilities)

Basic Model

- Given a corpus D and a vocabulary K
- Let R be a set of docs, d be a doc, k be a term, and n=|d|
- We model terms as events and documents as conjunction of events
  - $p(R) = |R| / |D|$
  - $p(k|R) = \{d \mid k \in d \wedge d \in R\} / |R|$
  - $p(d) = p(k_1,k_2,...k_n) = p(k_1)*p(k_2)*...p(k_n)$
    - Assuming statistical independence
  - $p(d|R) = p(k_1,k_2,...k_n|R) = p(k_1|R)*p(k_2|R)*...p(k_n|R)$
  - $p(k|d) = 1$ if $k \in d$ else 0
    - We actually won’t need this quantity

Process

- Initial queries too short for probabilistic reasoning
  - We need relevant docs to learn about “relevance”
  - Determined iteratively using feedback (automatic, explicit, implicit)
    - Similar to VSM with relevance feedback
- Process for answering q
  - Subset $R \subseteq D$ of only relevant documents
  - Subset $N \subseteq D$ of only irrelevant documents
  - Compute $p(R|d)$, the probability that document d belongs to R
  - Typically performed iteratively

Odds-Score

- We want to compute rel(d,q), the relevance of d for q
- Since words k_i of d appear both in relevant and in irrelevant docs, we look at the ratio p(R|d) / p(N|d)
  - Also called odds-score

$$rel(d,q) = \frac{p(R|d)}{p(N|d)} = \frac{p(R|k_1,...,k_n)}{p(N|k_1,...,k_n)}$$

- Assuming statistical independence of words, we get

$$rel(d,q) = \frac{p(R|k_1,...,k_n)}{p(N|k_1,...,k_n)} = \frac{p(R|k_1)*...*p(R|k_n)}{p(N|k_1)*...*p(N|k_n)}$$

Using Bayes

- Using Bayes Theorem

$$rel(d,q) = \frac{p(R|d)}{p(N|d)} = \frac{p(d|R)*p(R)*p(d)}{p(d|N)*p(N)*p(d)} \sim \frac{p(d|R)}{p(d|N)}$$

- p(R), p(N): relative frequency of (ir-)relevant docs in D
  - A-Priori probability of a doc to be (ir-)relevant
  - Constant for a given q and thus irrelevant for ranking docs
- p(d|R) is the probability of drawing the combination of words forming d when drawing words at random from R
  - We need the probability of drawing the words in d from R
  - And we need the probability of not drawing the other words from R

Binary Independence Model

- p(d|R) is the probability of drawing words from d from R and not drawing words not in d from R
- Binary Independence Model

$$rel(d,q) = \frac{p(d \mid R)}{p(d \mid N)} = \frac{\prod_{k \in d} p(k \mid R) \; * \; \prod_{k \notin d} p(\neg k \mid R)}{\prod_{k \in d} p(k \mid N) \; * \; \prod_{k \notin d} p(\neg k \mid N)}$$

  - Having words that are frequent in R raises the relevance of d
  - Not having words that are frequent in R lowers the relevance of d
  - Having words that are frequent in N lowers the relevance of d
  - Not having words that are frequent in N raises the relevance of d

Binary Independence Model

Good: Important terms for R present?

Bad: Important terms for R missing?

$$rel(d,q)=\frac{\prod_{k\in d} p(k\,|\,R) \;*\; \prod_{k\notin d} p(\neg k\,|\,R)}{\prod_{k\in d} p(k\,|\,N) \;*\; \prod_{k\notin d} p(\neg k\,|\,N)}=\frac{\prod_{k\in d} p(k\,|\,R) \;*\; \prod_{k\notin d}\left(1-p(k\,|\,R)\right)}{\prod_{k\in d} p(k\,|\,N) \;*\; \prod_{k\notin d}\left(1-p(k\,|\,N)\right)}$$

Bad: Important terms for N present?

Good: Important terms for N missing?

Continuation

- Rephrasing using q

$$rel(d,q)=\frac{\prod_{k\in d\cap q} p(k\,|\,R)}{\prod_{k\in d\cap q} p(k\,|\,N)} * \cancel{\frac{\prod_{k\in d\setminus q} p(k\,|\,R)}{\prod_{k\in d\setminus q} p(k\,|\,N)}} * \frac{\prod_{k\in q\setminus d} p(\neg k\,|\,R)}{\prod_{k\in q\setminus d} p(\neg k\,|\,N)} * \cancel{\frac{\prod_{k\notin d\cup q} p(\neg k\,|\,R)}{\prod_{k\notin d\cup q} p(\neg k\,|\,N)}}$$

- Since we are not sure about R and N: Focus on terms in q

$$\ldots \approx \prod_{k\in d\cap q}\frac{p(k\,|\,R)}{p(k\,|\,N)} * \prod_{k\in q\setminus d}\frac{p(\neg k\,|\,R)}{p(\neg k\,|\,N)} = \prod_{k\in d\cap q}\frac{p(k\,|\,R)}{p(k\,|\,N)} * \prod_{k\in q\setminus d}\frac{1-p(k\,|\,R)}{1-p(k\,|\,N)}$$

Last Step

$$\prod_{k\in d\cap q}\frac{p(k\,|\,R)}{p(k\,|\,N)}*\prod_{k\in q\setminus d}\frac{1-p(k\,|\,R)}{1-p(k\,|\,N)}$$

All matching terms All non-matching terms

- Some reformulating (duplicating the terms in q)

$$=\prod_{k\in d\cap q}\frac{p(k\,|\,R)*(1-p(k\,|\,N))*(1-p(k\,|\,R))}{p(k\,|\,N)*(1-p(k\,|\,R))*(1-p(k\,|\,N))}*\prod_{k\in q\setminus d}\frac{1-p(k\,|\,R)}{1-p(k\,|\,N)}$$

$$=\prod_{k\in d\cap q}\frac{p(k\,|\,R)*(1-p(k\,|\,N))}{p(k\,|\,N)*(1-p(k\,|\,R))}*\prod_{k\in q}\frac{1-p(k\,|\,R)}{1-p(k\,|\,N)}$$

All matching terms All query terms

Problem

- Last quotient is identical for all d and can be dropped

$$rel(d,q) \approx \prod_{k \in d \cap q} \frac{p(k|R)*(1-p(k|N))}{p(k|N)*(1-p(k|R))}$$

- But: Computing rel(d,q) requires knowledge of R and N
    - If R and N were known for sure, we could simply use p(k|R) / p(k|N) as relative frequencies of terms in R/N and use these weights for ranking
    - [Also called maximum likelihood estimation]
- In reality, we actually want to find R and N

Back to Reality

- Idea: Approximation using an iterative process
  - Start with “educated guess” for R and set N=D\R
    - E.g. R ~ “all docs containing at least one word from q”
  - Compute relevance of all docs with respect to q
  - Chose relevant docs (by user feedback) or hopefully relevant docs (by selecting the top-r docs)
  - This gives new sets R and N
    - If top-r docs are chosen, we may decide to only change probabilities of terms in R (and disregard the questionable negative information)
  - Compute new conditional probabilities and new ranking
  - Iterate until satisfied
- [Variant of the Expectation Maximization Algorithm (EM)]

Initialization

$$rel(d,q) \approx \prod_{k \in d \cap q} \frac{p(k|R)*(1-p(k|N))}{p(k|N)*(1-p(k|R))}$$

- Typical simplifying assumptions for the start
  - Terms in non-relevant docs are equally distributed: p(k|N)~df_k/|D|
  - Terms in relevant doc get equal probability: p(k|R)=0.5
  - Much less computation, less weight to unstable first values
  - [But leaves axiomatic probability theory]
- Iterations: Assume we have a new R’ and N’. Then

$$p(k|R') = \frac{|\{d|\, k \in d \; and \; d \in R'\}|}{|R'|}$$

$$p(k|N') = \frac{df_k - |\{d|\, k \in d \; and \; d \in R'\}|}{|D| - |R'|}$$

Example

	Text	verkauf	haus	italien	gart	miet	blüh	woll
1	Wir verkaufen Häuser in Italien	1	1	1				
2	Häuser mit Gärten zu vermieten		1		1	1		
3	Häuser: In Italien, um Italien, um Italien herum		1	1				
4	Die italienschen Gärtner sind im Garten			1	1			
5	Um unser italiensches Haus blüht‘s		1	1			1	
6	Wir verkaufen Blühendes	1					1	
Q	Wir wollen ein Haus mit Garten in Italien mieten		1	1	1	1		1

Example: Initialization

$$rel(d,q) \approx \prod_{k \in d \cap q} \frac{p(k|R) * (1 - p(k|N))}{p(k|N) * (1 - p(k|R))}$$

	V	H	I	G	M	B	W
1	1	1	1				
2		1		1	1		
3		1	1				
4			1	1			
5		1	1			1	
6	1					1	
Q		1	1	1	1		1

- All docs with at least one word from q
  - R={1,2,3,4,5}, N={6}
- Initial estimations
  - p(k|R)=0.5, p(k|N)= df_k/|D| -> p(verkauf|N)=p(blüh|N)=2/6
  - Smoothing: If p(k|X)=0, set p(k|X)=0.01
- Initial ranking
  - rel(1,q)= p(haus|R)*(1-p(haus|N))*p(italien|R)*(1-p(italien|N)) /
    p(haus|N)*(1-p(haus|R))*p(italien|N)*(1-p(italien|R))
    = .5*(1-4/6)*.5*(1-4/6) / (4/6*(1-0.5)*4/6*(1-0.5))=
    = 0,66
  - rel(2,q)= p(haus|R)*(1-p(haus|N)*p(garten|R)*(1-p(garten|N)*p(mieten|R)*(1-p(mieten|N) / ...
  - rel(3,q)= ...

Adjustment

$$P(k|R) = \frac{|\{d \mid k \in d, d \in R\}|}{|R|}$$

$$P(k|N) = \frac{df_k - |\{d \mid k \in d, d \in R\}|}{|D| - |R|}$$

	V	**H**	**I**	**G**	**M**	**B**	**W**
1	1	1	1				
2		1		1	1		
3		1	1				
4			1	1			
5		1	1			1	
6	1					1	
Q		1	1	1	1		1

- Let’s use the top-2 docs as new R
  - Second chosen arbitrarily among 1,3,4,5
  - R={1,2}, N={3,4,5,6}
- Adjust scores
  - p(verkauf|R)=.5, p(verkauf|N)=(2-1)/(6-2)=1/4
  - p(haus|R)=1 (~.99), p(haus|N)=(4-2)/(6-2)=2/4
  - p(italien|R)=.5, p(italien|N)=(4-1)/(6-2)=3/4
  - p(gart|R)=.5, p(gart|N)=(2-1)/(6-2)=1/4
  - p(miet|R)=.5, p(miet|N)=(1-1)/(6-2)=0~0.01

Smoothing: Avoid 1-1=0

Re-Ranking

$$rel(d,q) \approx \prod_{k \in d \cap q} \frac{p(k|R)*(1-p(k|N))}{p(k|N)*(1-p(k|R))}$$

	V	H	I	G	M	B	W
1	1	1	1				
2		1		1	1		
3		1	1				
4			1	1			
5		1	1			1	
6	1					1	
Q		1	1	1	1		1

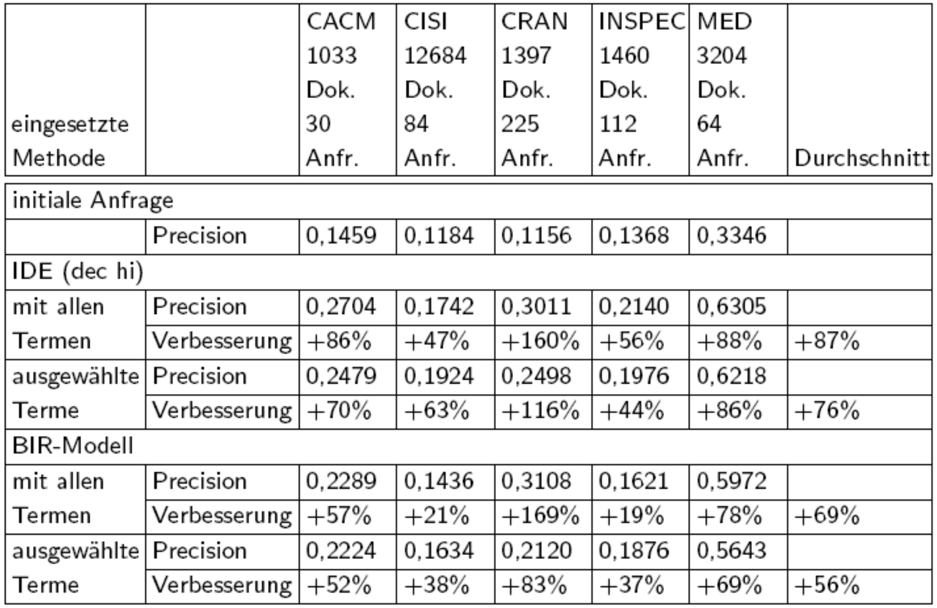
- New ranking
  - rel(1,q) = p(haus|R)*(1-p(haus|N))*p(italien|R)*(1-p(italien|N))
    p(haus|N)*(1-p(haus|R))*p(italien|N)*(1-p(italien|R))
    = ...
  - rel(2,q) = ...
  - ...

Pros and Cons

- Advantages
  - Sound (probabilistic) framework
    - Many researchers feel more comfortable – explanations for all steps
    - But: Several steps are very heuristic
  - Results converge to most relevant docs (empirically shown)
    - Under the assumption that relevant docs are similar by sharing term distributions that are different from distributions in irrelevant docs
- Disadvantages
  - First guesses often are pretty bad – slow convergence
  - Assumes statistical independence of terms (as many methods)
  - “Has never worked convincingly better in practice” [MS07]

Probabilistic Model versus VSM with Rel. Feedback

- Published 1990 by Salton & Buckley
- Comparison based on various corpora
- Improvement after 1 feedback iteration



eingesetzte Methode		CACM 1033 Dok. 30 Anfr.	CISI 12684 Dok. 84 Anfr.	CRAN 1397 Dok. 225 Anfr.	INSPEC 1460 Dok. 112 Anfr.	MED 3204 Dok. 64 Anfr.	Durchschnitt
initiale Anfrage							
	Precision	0,1459	0,1184	0,1156	0,1368	0,3346	
IDE (dec hi)							
mit allen Termen	Precision	0,2704	0,1742	0,3011	0,2140	0,6305	
	Verbesserung	+86%	+47%	+160%	+56%	+88%	+87%
ausgewählte Terme	Precision	0,2479	0,1924	0,2498	0,1976	0,6218	
	Verbesserung	+70%	+63%	+116%	+44%	+86%	+76%
BIR-Modell							
mit allen Termen	Precision	0,2289	0,1436	0,3108	0,1621	0,5972	
	Verbesserung	+57%	+21%	+169%	+19%	+78%	+69%
ausgewählte Terme	Precision	0,2224	0,1634	0,2120	0,1876	0,5643	
	Verbesserung	+52%	+38%	+83%	+37%	+69%	+56%

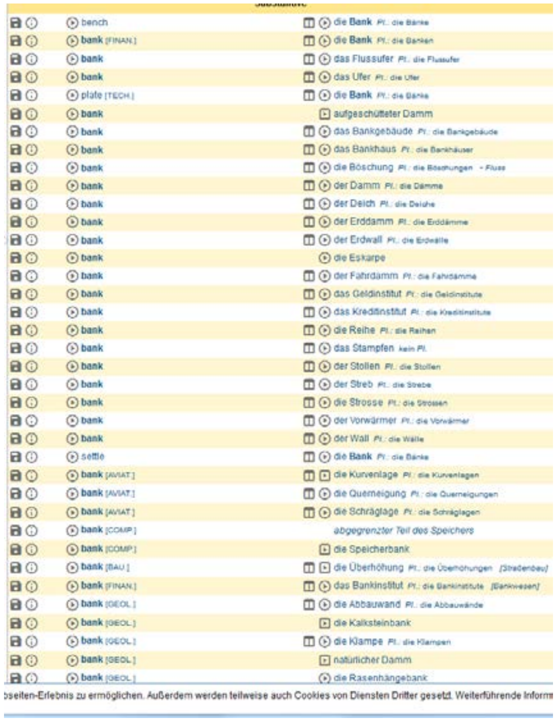
- Probabilistic model (BIR) in general worse than VSM+rel feedback (IDE)
  - Probabilistic model does not weight terms in documents
  - Probabilistic model does not allow to weight terms in queries

Content of this Lecture

- IR Models
- Boolean Model
- Vector Space Model
- Relevance Feedback in the VSM
- Probabilistic Model
- Latent Semantic Indexing
- Other IR Models
- Outlook: Word Semantics and Word Embeddings

Latent Semantic Indexing

- We so-far ignored semantic relationships between terms
  - Homonyms: bank (money, river, place)
  - Synonyms: House, building, hut, villa, ...
  - Hyperonyms: officer – lieutenant

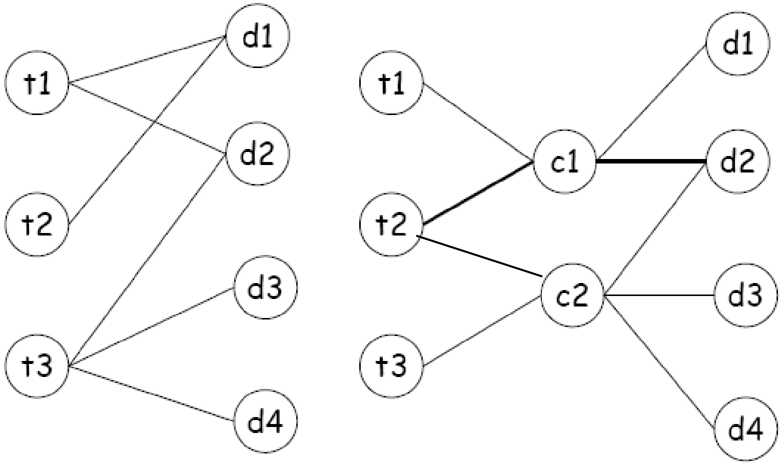


bench	die Bank Pl.: die Bänke
bank [FINAN.]	die Bank Pl.: die Banken
bank	das Flussufer Pl.: die Flussufer
bank	das Ufer Pl.: die Ufer
plate [TECH.]	die Bank Pl.: die Bänke
bank	aufgeschütteter Damm
bank	das Bankgebäude Pl.: die Bankgebäude
bank	das Bankhaus Pl.: die Bankhäuser
bank	die Böschung Pl.: die Böschungen - Fluss
bank	der Damm Pl.: die Dämme
bank	der Deich Pl.: die Deiche
bank	der Erddamm Pl.: die Erddämme
bank	der Erdwall Pl.: die Erdwälle
bank	die Eskarpe
bank	der Fahrdamm Pl.: die Fahrdämme
bank	das Geldinstitut Pl.: die Geldinstitute
bank	das Kreditinstitut Pl.: die Kreditinstitute
bank	die Reihe Pl.: die Reihen
bank	das Stampfen kein Pl.
bank	der Stollen Pl.: die Stollen
bank	der Streb Pl.: die Strebe
bank	die Strosse Pl.: die Strossen
bank	der Vorwärmer Pl.: die Vorwärmer
bank	der Wall Pl.: die Wälle
settle	die Bank Pl.: die Bänke
bank [AVIAT.]	die Kurvenlage Pl.: die Kurvenlagen
bank [AVIAT.]	die Querneigung Pl.: die Querneigungen
bank [AVIAT.]	die Schräglage Pl.: die Schräglagen
bank [COMP.]	*abgegrenzter Teil des Speichers*
bank [COMP.]	die Speicherbank
bank [BAU.]	die Überhöhung Pl.: die Überhöhungen *[Straßenbau]*
bank [FINAN.]	das Bankinstitut Pl.: die Bankinstitute *[Bankwesen]*
bank [GEOL.]	die Abbauwand Pl.: die Abbauwände
bank [GEOL.]	die Kalksteinbank
bank [GEOL.]	die Klampe Pl.: die Klampen
bank [GEOL.]	natürlicher Damm
bank [GEOL.]	die Rasenhängebank

seiten-Erlebnis zu ermöglichen. Außerdem werden teilweise auch Cookies von Diensten Dritter gesetzt. Weiterführende Inform

- Latent Semantic Indexing (LSI)
  - Deerwester et al. (1990). "Indexing by latent semantic analysis." JASIS 41(6): 391-407.
    - 2011: ~7500 cit.; 2014: ~9400, 2018: ~13500
  - Map (many) terms into (fewer) semantic concepts
    - Discover the concepts hidden (“latent”) in the docs
  - Compare docs and query in concept space instead of term space
- May find docs that don’t contain a single query term

Terms and Concepts



Quelle: K. Aberer, IR

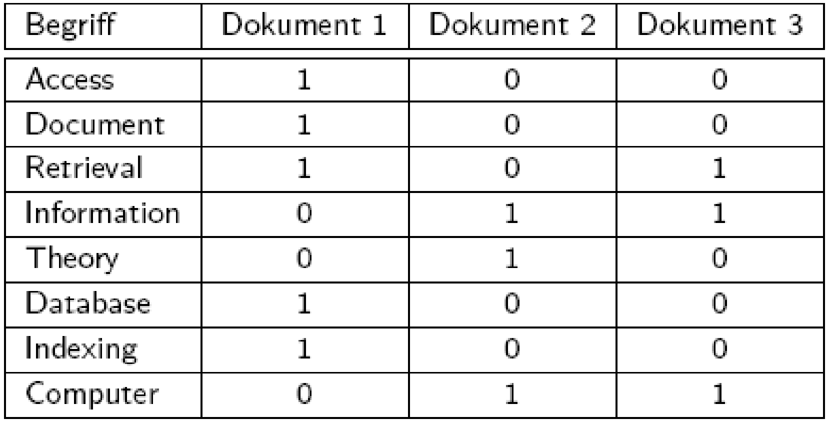
- Concepts are more abstract than terms
- Concepts are related to terms and to docs
- LSI models concepts as sets of strongly co-occurring terms
  - Can be computed using matrix manipulations
  - Concepts from LSI cannot be “spelled out”, but are matrix columns

Term-Document Matrix

- Definition

  The term-document matrix M for docs D and terms K has $n=|D|$ *columns and* $m=|K|$ *rows.* $M[i,j]=1$ *iff document* d_j *contains term* k_i*.*

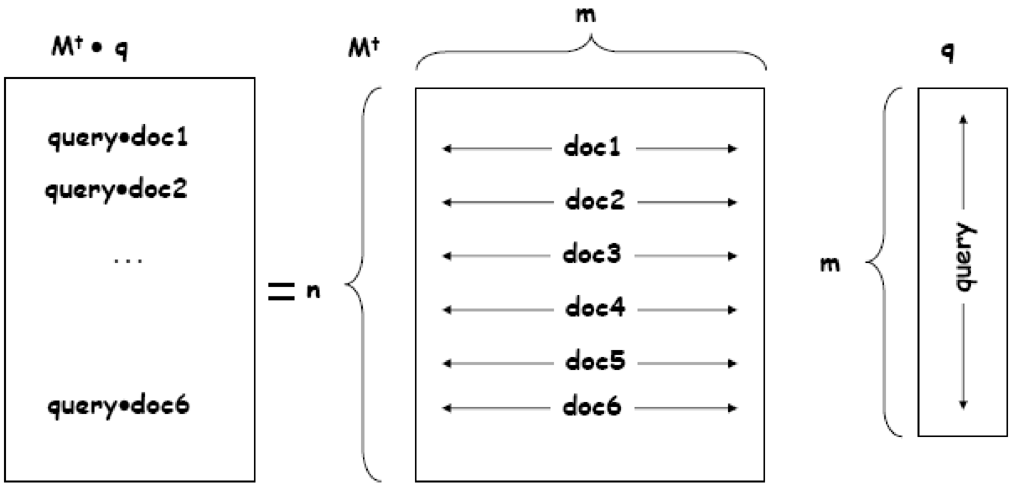
  - Works equally well for TF or TF*IDF values



Begriff	Dokument 1	Dokument 2	Dokument 3
Access	1	0	0
Document	1	0	0
Retrieval	1	0	1
Information	0	1	1
Theory	0	1	0
Database	1	0	0
Indexing	1	0	0
Computer	0	1	1

Term-Document Matrix and VSM

- VSM uses the transposed document-term matrix (=M^t)
- Having M, we can in principle compute the vector v of the VSM-scores for q of all docs as $v=M^t \bullet q$
  - Only the dot product, normalization missing



What to do with a Term-Document Matrix

- M is not just a comfortable way of representing the term vectors of all documents
- In the following, we approximate M by a particular M’
  - M’ should be smaller than M
    - Less dimensions; faster computations; abstraction
  - M’ should abstract from terms to concepts
    - The fewer dimensions capture the most frequent co-occurrences
  - Approach: Find an M’ such that $M'^t*q' \approx M^t*q$
    - Produce the least error among all M’ of the same dimension

	D1	**D2**	**D3**	**D4**
and	1	1		
cat	1	1	1	
eat	1	1	1	
…				1
zoo		1		1

	D1	**D2**	**D3**	**D4**
C1	0,3	0,2	0	0,4
C2	0,7	0	0,1	0,9
C3	0,1	0	0,5	0,3

Term and Document Correlation

- $M \bullet M^t$ is called the term correlation matrix
  - Has |K| columns and |K| rows
  - “Similarity” of terms: how often do they co-occur in a doc?
- $M^t \bullet M$ is called the document correlation matrix
  - Has |D| columns and |D| rows
  - “Similarity” of docs: how many terms do they share?
- Example

	1	**2**	**3**	**4**	**5**
A	1	1	1		
B	1	1	1		1
C		1	1		
D				1	1

M (A..: terms; 1...: docs)

•

	A	**B**	**C**	**D**
1	1	1		
2	1	1	1	
3	1	1	1	
4				1
5		1		1

M^t

=

	A	**B**	**C**	**D**
A	3	3	2	0
B	3	4	2	1
C	2	2	2	0
D	0	1	0	2

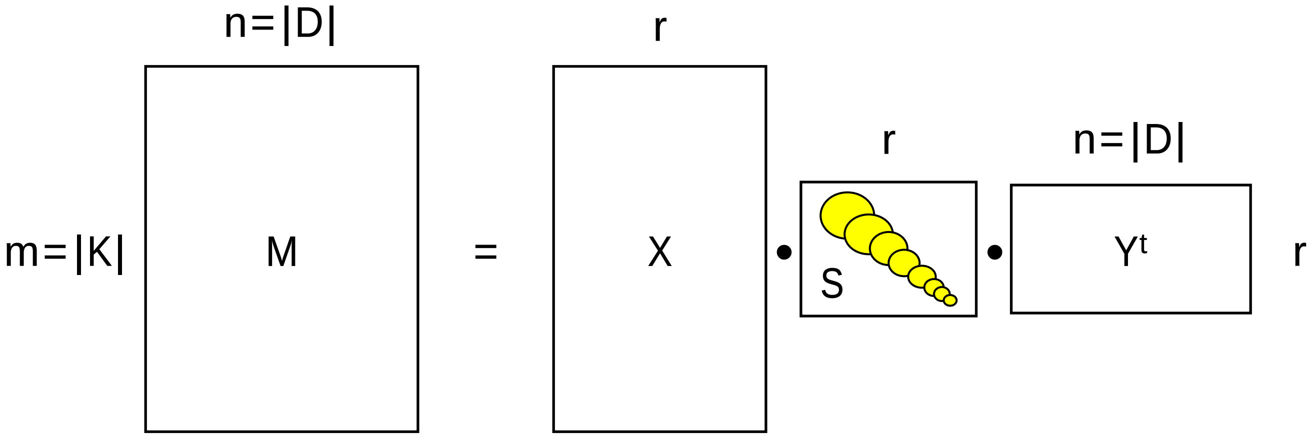
Term correlation matrix

Some Linear Algebra

- The rank r of a matrix M is the maximal number of linearly independent rows of M
- If $Mx-\lambda x=0$ for a vector $x\neq 0$, then λ is called an Eigenvalue of M and x is his associated Eigenvector
  - Eigenvectors/-werte are useful for many things
  - In particular, a matrix M can be transformed into a diagonal matrix L with $L=U^{-1}*M*U$ with U formed from the Eigenvectors of M iff M has “enough” Eigenvectors
    - L represents M in another vector space, based on another basis
    - L can be used in many cases instead of M and is easier to handle
  - However, our M usually will not have “enough” Eigenvectors
  - We use another factorization of M

Singular Value Decomposition (SVD)

- SVD decomposes any matrix into M = X • S • Y^t
  - S is the diagonal matrix of the singular values of M in descending order and has size rxr (with r=rank(M))
  - X is the matrix of Eigenvectors of M • M^t
  - Y is the matrix of Eigenvectors of M^t • M
  - This decomposition is unique and can be computed in O(r^3)
    - Use approximation in practice



Example

- Assume for now M is quadratic and has full rank
  - Example for r=|K|=|D|=3

$$\begin{array}{|c|c|c|}\hline M_{11} & M_{12} & M_{13} \\ \hline M_{21} & \ldots & \ldots \\ \hline M_{31} & \ldots & M_{33} \\ \hline \end{array} = \begin{array}{|c|c|c|}\hline x_{11} & \ldots & \ldots \\ \hline x_{21} & \ldots & \ldots \\ \hline \ldots & \ldots & x_{33} \\ \hline \end{array} \bullet \begin{array}{|c|c|c|}\hline s_{11} & 0 & 0 \\ \hline 0 & s_{22} & 0 \\ \hline 0 & 0 & s_{33} \\ \hline \end{array} \bullet \begin{array}{|c|c|c|}\hline y_{11} & \ldots & \ldots \\ \hline y_{21} & \ldots & \ldots \\ \hline \ldots & \ldots & y_{33} \\ \hline \end{array}$$

- $$\begin{aligned} M_{11} = {} & (x_{11}*s_{11}+x_{12}*s_{12}+x_{13}*s_{13})*y_{11} + \\ & (x_{11}*s_{21}+x_{12}*s_{22}+x_{13}*s_{23})*y_{21} + \\ & (x_{11}*s_{31}+x_{12}*s_{32}+x_{13}*s_{33})*y_{31} \\ = {} & x_{11}*s_{11}*y_{11} + x_{12}*s_{22}*y_{21} + x_{13}*s_{33}*y_{31} \end{aligned}$$
- $M_{12} = \ldots$

Approximating M

- LSI idea: What if we stop the sums earlier?
  - s_{ii} are sorted by descending value
  - Aggregating only over the first s_{ii}–values captures “most” of M

$$\begin{array}{|c|c|c|}\hline M_{11} & M_{12} & M_{13} \\ \hline M_{21} & \ldots & \ldots \\ \hline M_{31} & \ldots & M_{33} \\ \hline \end{array} = \begin{array}{|c|c|c|}\hline x_{11} & \ldots & \ldots \\ \hline x_{21} & \ldots & \ldots \\ \hline \ldots & \ldots & x_{33} \\ \hline \end{array} \bullet \begin{array}{|c|c|c|}\hline s_{11} & 0 & 0 \\ \hline 0 & s_{22} & 0 \\ \hline 0 & 0 & s_{33} \\ \hline \end{array} \bullet \begin{array}{|c|c|c|}\hline y_{11} & \ldots & \ldots \\ \hline y_{21} & \ldots & \ldots \\ \hline \ldots & \ldots & y_{33} \\ \hline \end{array}$$

- $M_{11} = x_{11}*s_{11}*y_{11} + x_{12}*s_{22}*y_{21} + x_{13}*s_{33}*y_{31}$

largest s_{ij}

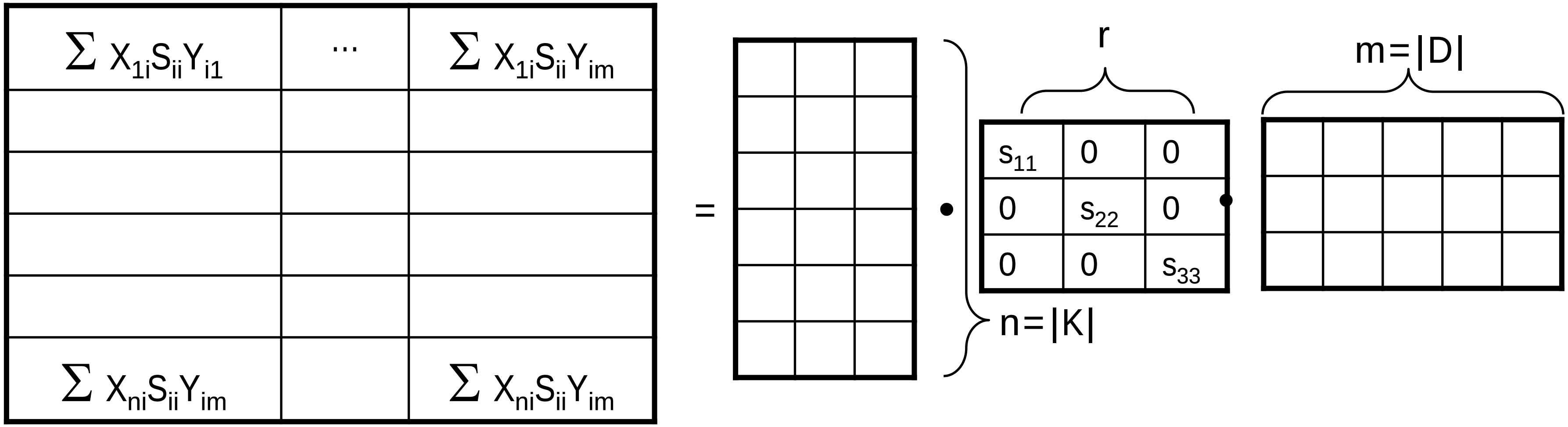
2nd largest s_{ij}

3rd largest s_{ij}

- What if $M_{11}' = x_{11}*s_{11}*y_{11} + x_{12}*s_{22}*y_{21}$

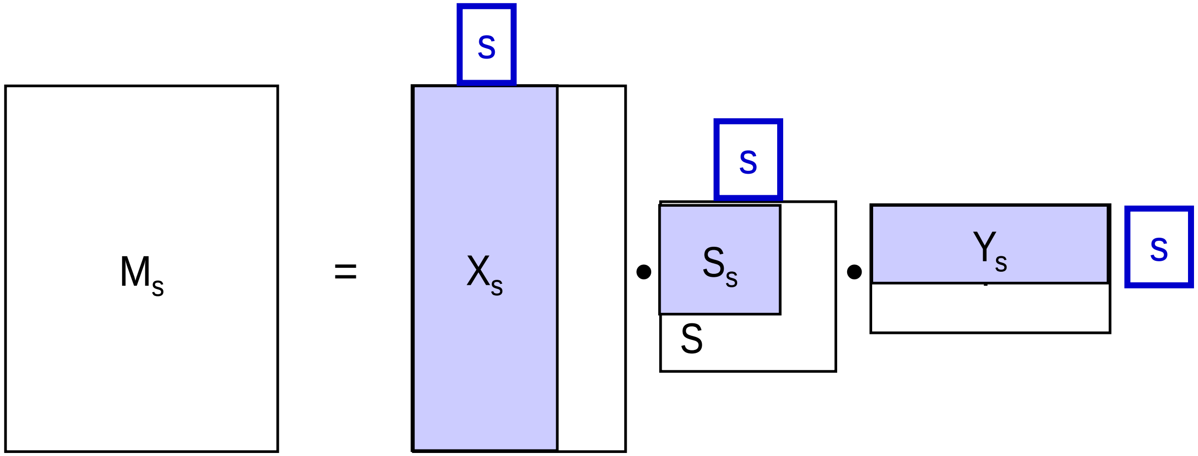
General Case

- In general, M is not quadratic and r < min(|K|,|D|)
  - All sums range from 1 to r



Approximating M

- LSI: Use S to approximate M
- Fix some s<r; Compute $M_s = X_s \bullet S_s \bullet Y_s^t$
  - X_s : First s columns in X
  - S_s : First s columns and first s rows in S
  - Y_s : First s rows in Y
- M_s has the same size as M, but different values
  - In fact, we don't need to compute M_s, but only need X_s, S_s and Y_s



s-Approximations

- Formal: M_s is the matrix where $||M-M_s||_2$ is the smallest

- Since the s_{ii} are sorted in decreasing order
  - The approximation is the better, the larger s
  - The computation is the faster, the smaller s
- LSI: Only consider the top-s singular values
  - s must be small enough to filter out noise (spurious co-occurrences) and to provide “semantic reduction”
  - s must be large enough to represent the diversity in the documents
  - Typical value: 200-500
    - While r is typically >100.000
- Universal idea: LSI has ample applications outside IR

LSI for Information Retrieval

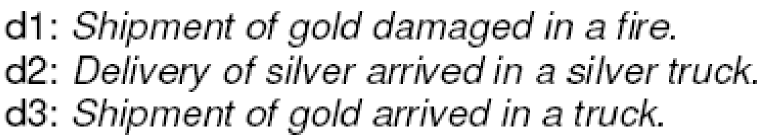
- We map document vectors from a m-dimensional space into a s-dimensional space
  - Approximated docs (still) are represented by columns in Y_s^t
- SVD as much as possible preserves distances between docs (depending on number of shared co-occurring terms)
- To this end, SVD (in a way) maps frequently co-occurring terms onto the same dimensions
  - Because these terms have little impact on distance
- Linear terms-combinations can be seen as concepts
  - But they cannot easily be "named"
  - We cannot easily abstract the terms that are mapped into a new dimension – it is always a bit of everything (a linear combination)

Query Evaluation

- After LSI, docs are represented by columns in Y_s^t
- How can we compute the distance between a query and a doc in concept space?
  - Transform q into concept space
  - Assume q as a new column in M
    - Of course, we can transform M offline, but need to transform q online
  - This would generate a new column in Y_s^t
  - To only compute this column, we apply the same transformations to q as we did to all other columns of M
  - With a little algebra, we get: $q' = q^t \bullet X_s \bullet S_s^{-1}$
  - This vector is compared to the transformed doc vectors as usual

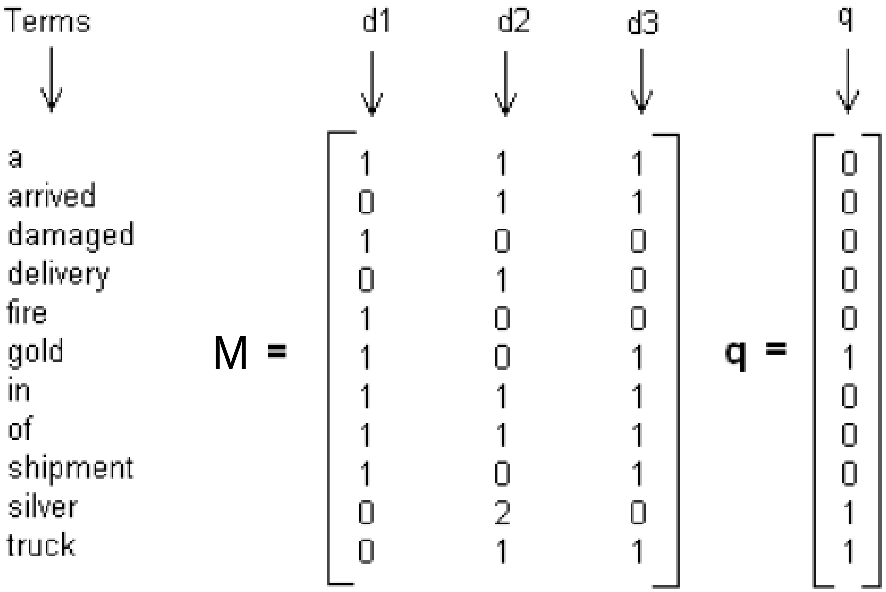
Example: Term-Document Matrix

- Taken from Mi Islita: “Tutorials on SVD & LSI”
  - http://www.miislita.com/information-retrieval-tutorial/svd-lsi-tutorial-1-understanding.html
    - Who took if from the Grossman and Frieder book



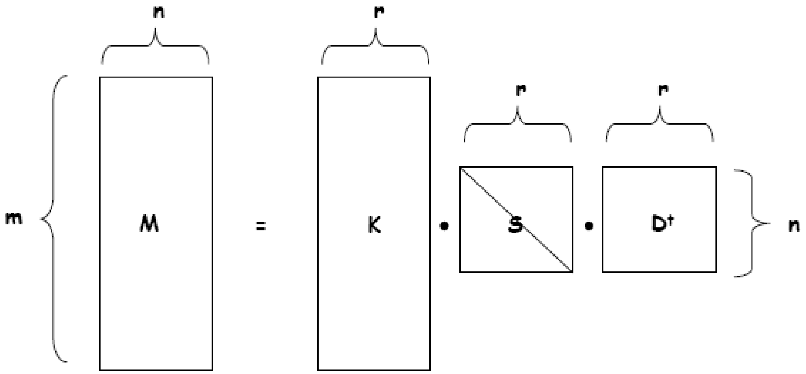
d1: *Shipment of gold damaged in a fire.*
d2: *Delivery of silver arrived in a silver truck.*
d3: *Shipment of gold arrived in a truck.*

Query: „gold silver truck“



$$\begin{array}{l} \text{Terms} \\ \downarrow \\ \text{a} \\ \text{arrived} \\ \text{damaged} \\ \text{delivery} \\ \text{fire} \\ \text{gold} \\ \text{in} \\ \text{of} \\ \text{shipment} \\ \text{silver} \\ \text{truck} \end{array} \quad M = \begin{array}{c} \begin{array}{ccc} d1 & d2 & d3 \\ \downarrow & \downarrow & \downarrow \end{array} \\ \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 2 & 0 \\ 0 & 1 & 1 \end{bmatrix} \end{array} \quad \mathbf{q} = \begin{array}{c} q \\ \downarrow \\ \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} \end{array}$$

Singular Value Decomposition



$$M = X \bullet S \bullet Y^t$$

$$X = \begin{bmatrix} -0.4201 & 0.0748 & -0.0460 \\ -0.2995 & -0.2001 & 0.4078 \\ -0.1206 & 0.2749 & -0.4538 \\ -0.1576 & -0.3046 & -0.2006 \\ -0.1206 & 0.2749 & -0.4538 \\ -0.2626 & 0.3794 & 0.1547 \\ -0.4201 & 0.0748 & -0.0460 \\ -0.4201 & 0.0748 & -0.0460 \\ -0.2626 & 0.3794 & 0.1547 \\ -0.3151 & -0.6093 & -0.4013 \\ -0.2995 & -0.2001 & 0.4078 \end{bmatrix} \quad S = \begin{bmatrix} 4.0989 & 0.0000 & 0.0000 \\ 0.0000 & 2.3616 & 0.0000 \\ 0.0000 & 0.0000 & 1.2737 \end{bmatrix}$$

$$Y = \begin{bmatrix} -0.4945 & 0.6492 & -0.5780 \\ -0.6458 & -0.7194 & -0.2556 \\ -0.5817 & 0.2469 & 0.7750 \end{bmatrix} \quad Y^t = \begin{bmatrix} -0.4945 & -0.6458 & -0.5817 \\ 0.6492 & -0.7194 & 0.2469 \\ -0.5780 & -0.2556 & 0.7750 \end{bmatrix}$$

A Two-Approximation (s=2)

$$X_2 = \begin{bmatrix} -0.4201 & 0.0748 \\ -0.2995 & -0.2001 \\ -0.1206 & 0.2749 \\ -0.1576 & -0.3046 \\ -0.1206 & 0.2749 \\ -0.2626 & 0.3794 \\ -0.4201 & 0.0748 \\ -0.4201 & 0.0748 \\ -0.2626 & 0.3794 \\ -0.3151 & -0.6093 \\ -0.2995 & -0.2001 \end{bmatrix} \qquad S_2 = \begin{bmatrix} 4.0989 & 0.0000 \\ 0.0000 & 2.3616 \end{bmatrix}$$

$$Y_2 = \begin{bmatrix} -0.4945 & 0.6492 \\ -0.6458 & -0.7194 \\ -0.5817 & 0.2469 \end{bmatrix} \qquad Y_2^t = \begin{array}{c} \begin{bmatrix} -0.4945 & -0.6458 & -0.5817 \\ 0.6492 & -0.7194 & 0.2469 \end{bmatrix} \\ \begin{array}{ccc} \uparrow & \uparrow & \uparrow \\ d_1 & d_2 & d_3 \end{array} \end{array}$$

Transforming the Query

$$q' = q^t \bullet X_2 \bullet S_2^{-1}$$

$$q' = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} -0.4201 & 0.0748 \\ -0.2995 & -0.2001 \\ -0.1206 & 0.2749 \\ -0.1576 & -0.3046 \\ -0.1206 & 0.2749 \\ -0.2626 & 0.3794 \\ -0.4201 & 0.0748 \\ -0.4201 & 0.0748 \\ -0.2626 & 0.3794 \\ -0.3151 & -0.6093 \\ -0.2995 & -0.2001 \end{bmatrix} \begin{bmatrix} \frac{1}{4.0989} & 0.0000 \\ 0.0000 & \frac{1}{2.3616} \end{bmatrix}$$

$$= \begin{bmatrix} -0.2140 & -0.1821 \end{bmatrix}$$

Computing the Cosine of the Angle

$$\text{sim}(q, d) = \frac{q \bullet d}{|q|\,|d|}$$

$$\text{sim}(q, d_1) = \frac{(-0.2140)(-0.4945) + (-0.1821)(0.6492)}{\sqrt{(-0.2140)^2 + (-0.1821)^2}\sqrt{(-0.4945)^2 + (0.6492)^2}} = -0.0541$$

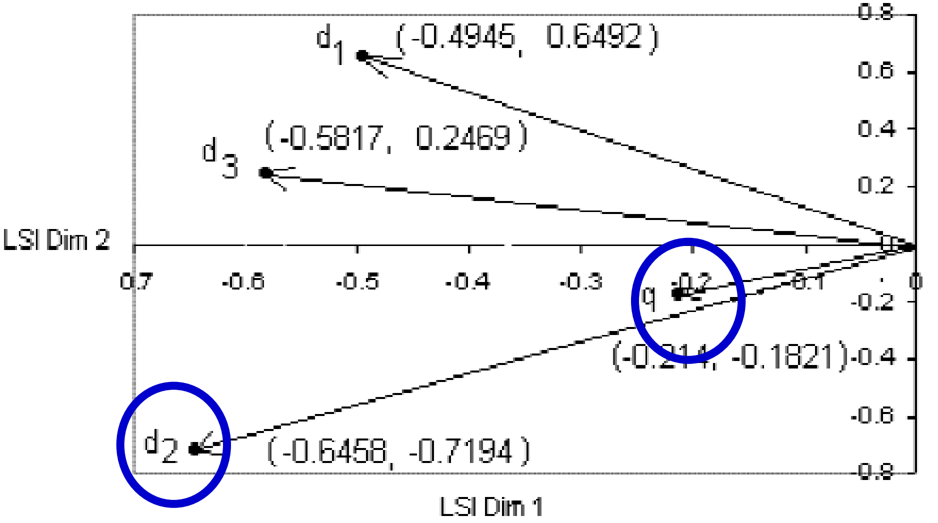
$$\text{sim}(q, d_2) = \frac{(-0.2140)(-0.6458) + (-0.1821)(-0.7194)}{\sqrt{(-0.2140)^2 + (-0.1821)^2}\sqrt{(-0.6458)^2 + (-0.7194)^2}} = 0.9910$$

$$\text{sim}(q, d_3) = \frac{(-0.2140)(-0.5817) + (-0.1821)(0.2469)}{\sqrt{(-0.2140)^2 + (-0.1821)^2}\sqrt{(-0.5817)^2 + (0.2469)^2}} = 0.4478$$

Visualization of Results in 2D

$$
M = \begin{array}{l|ccc}
\text{Terms} & d1 & d2 & d3 \\
\downarrow & \downarrow & \downarrow & \downarrow \\
\hline
\text{a} & 1 & 1 & 1 \\
\text{arrived} & 0 & 1 & 1 \\
\text{damaged} & 1 & 0 & 0 \\
\text{delivery} & 0 & 1 & 0 \\
\text{fire} & 1 & 0 & 0 \\
\text{gold} & 1 & 0 & 1 \\
\text{in} & 1 & 1 & 1 \\
\text{of} & 1 & 1 & 1 \\
\text{shipment} & 1 & 0 & 1 \\
\text{silver} & 0 & 2 & 0 \\
\text{truck} & 0 & 1 & 1
\end{array}
\qquad
q = \begin{array}{c}
q \\
\downarrow \\
\hline
0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1
\end{array}
$$

Very large distance in original space



Pros and Cons

- Pro
  - Practical implementations exist, but not if corpus is very large
    - [MPS08] says: “no more than 1M docs”
  - Increases recall (and usually decreases precision)
- Contra
  - Computing SVD is expensive
    - Fast approximations exist, especially for extremely sparse matrices
    - Use stemming, stop-word removal etc. to shrink the original matrix
  - Ranking requires less dimensions than |D|, but more than |q|
    - Mapping the query turns a few keywords into an s-dimensional vector
    - We cannot simply index the “concepts” of M_s using inverted files etc.
    - Thus, LSI needs other techniques than inverted files
      - Means: lots of memory
    - Query speed is not reduced (despite less dimensions)

Content of this Lecture

- IR Models
- Boolean Model
- Vector Space Model
- Relevance Feedback in the VSM
- Probabilistic Model
- Latent Semantic Indexing
- Other Classical IR Models
- Outlook: Word Semantics and Word Embeddings

Extended Boolean Model

- Critique to Boolean Model: If 1 conjunctive term out of 10 is missing, we get same result as if 10 were missing
- Idea: Measure "distance" for each conjunctive / disjunctive subterm of the query expression to the document
  - Example: X-ary AND: use a projection into x-dim space
  - Query expression is (1,1,1,...,1)
  - Doc is $(a_1,a_2,\ldots,a_x)=(0/1?,0/1?,\ldots)$
  - Similarity is distance between these two points
  - Other formulas for OR and NOT
- This model mimics the VSM
  - But no terms weights

Generalized Vector Space Model

- One critique to the VSM: Terms are not independent
- Thus, term vectors cannot be assumed to be orthogonal
- Generalized Vector Space Model
  - Build a much larger vector space with $2^{|K|}$ dimensions
    - Each dimension ("minterm") stands for all docs containing a particular set of terms
    - Minterms are not orthogonal but correlated by term co-occurrences
  - Convert query and docs into minterm space
  - Finally, rel(q, d) is the cosine of the angel in minterm space
- Nice theory, considers term co-occurrence, much more complex than ordinary VSM, no proven advantage

Content of this Lecture

- IR Models
- Boolean Model
- Vector Space Model
- Relevance Feedback in the VSM
- Probabilistic Model
- Latent Semantic Indexing
- Other Classical IR Models
- Outlook: Word Semantics and Word Embeddings

Word Semantics

- VSM considers two tokens as different when they have different spelling
  - No shades: Equal or not, dimensions in VSM are orthogonal
  - King, princess, earl, milk, butter, cow, white, crown, emperor, ...
- This makes models very specific – bad generalization
- Humans do compare words in a multi-facetted way
  - King is similar to princess to earl to queen, but not to cow
    - But all are mammals
  - Kings use crowns much more often than cows
- How can we capture word semantics to derive meaningful similarity scores?

Knowledge-based: WordNet, Wikipedia, ...

- Let‘s dream: A comprehensive resource of all words and their relationships
  - Specialization, synonymy, partonomy, relatedness, is_required_for, develops_into, is_possible_with, ...
  - Example: WordNet
    - Roughly 150K concepts, 200K senses, 117K synsets
    - Specialization, partonomy, antonomy
  - Can be turned into a semantic similarity measure, e.g., length of shortest path between two concepts
- Problem: Incomplete, costly, outdated
  - Especially in specific domains like Biomedicine
- Much research to automatically expand WordNet, but no real breakthrough

Distributional Semantics

- „You shall know a word by the company it keeps" [Firth, 1957]
  - The distribution of words co-occurring (context) with a given word X is characteristic for X
  - To learn about X, look at its context
  - If X and Y are semantically similar, also their contexts are similar
  - If X and Y are a bit different, also their contexts will be a bit different
  - Holds in all domains and all corpora of sufficient size
- Central idea: Represent a word by its context
- For similarity: Compare contexts, not strings
- How can we do this efficiently and effectively?

Naive Approach

- Given a large corpus D and a vocabulary K
- Define a context window (typically sentence)
- Represent every $k \in K$ as a $|K|$-dimensional vector v_k
  - Find set W of all context windows in D containing k
  - For every $k' \neq k$, count frequency of k' in W: $v_k[k'] = freq(k', W)$
  - May be normalized, e.g. tf*idf
- Similarity: Compute cosine similarity between word-vectors
- Problem: Our model for each $d \in D$ grew from $|K|$ to $|K|^2$
  - Infeasible
  - We need an efficient and conservative dimensionality reduction
    - Efficient: Fast to compute; conservative: Distances are preserved
  - LSD too expensive

Word Embeddings

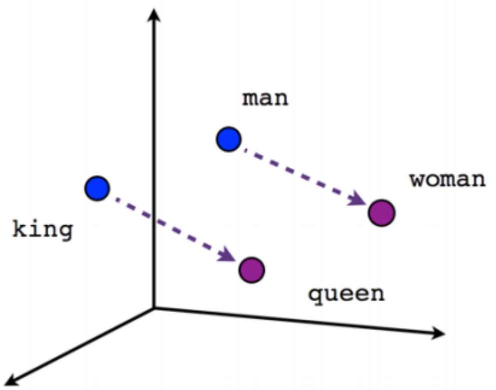
- Very popular technique since app. 2015
- Goal: Learning word vectors (“word embeddings”)
  - Low dimensional – typically 100-500 (a hyper parameter)
  - Unsupervised learning – may use extremely large corpora
  - Specific techniques to scale-up training (e.g. GPUs)
  - Can be precomputed and used without re-training in apps
- Approach: Use Machine Learning, not algebra
  - Though the border is not clear at all

Word2Vec [Mikolov et al. 2013]

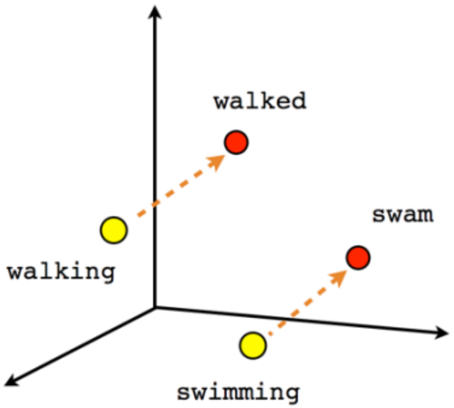
- Recall language models
  - Goal: Given a prefix of a sentence, predict next word
  - Can be understood as multi-class classification problem
    - As many classes as words
  - We computed word probabilities using a simple N-gram model
- Idea of Word2Vec
  - Cast the problem as classification
  - Given the context of a word – predict the word
    - Obviously related to language modelling
    - Note the “context” – we are close to distributional semantics

K2 is the second ? mountain in the world.

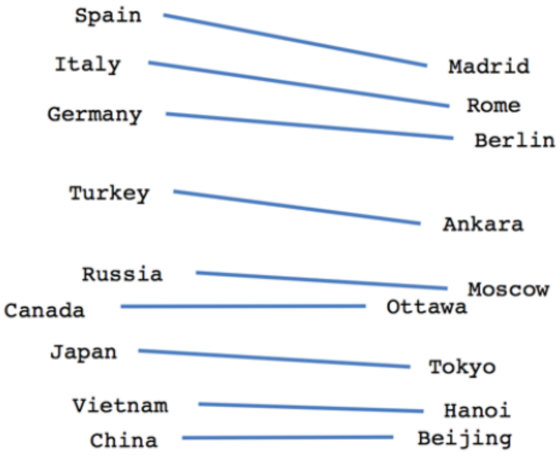
Does it Work?



Male-Female



Verb tense



Country-Capital

king – man ~ queen – woman

walking – walked ~ swimming – swam

Russia – Moscow ~ Vietnam – Hanoi

man - computer programmer ~ woman – homemaker

father - doctor ~ mother - nurse

Usage in Information Retrieval?

- Problem: We want to compare a query to a doc, not a word to a word
- Simple
  - Represent a doc by the average of all its word vectors
  - Same for query
  - Compute cosine of vectors
- More advanced
  - Compute sentence embeddings as average over words in sentence
  - Cluster sentence embeddings to find document segments
  - Match doc segments to query vector
- Fancy: Compute document embeddings

Self Assessment

- Explain the general approach of the probabilistic relevance model in IR
- How does one typically bootstrap this model?
- Which relevance model we discussed does consider the non-existent of terms in docs not existing in the query?
- Discuss the performance (speed) of the LSI approach to IR
- What is the difference between concept space and term space in LSI?
- Explain the Extended Boolean Model. Which of the shortcomings of the Boolean Model does it address?