

Vector-Oriented Retrieval in XML Data Collections



Jaroslav Pokorny,
Faculty of Mathematics and Physics,
Charles University, Praha



Motivation

Observation: XPath, XQuery rather for data-centric than for document centric XML data

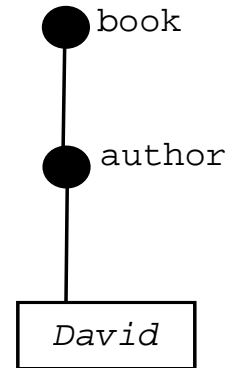
Goal: more from IR into querying

Strategy: combine weighting (vector space model) and XML structure.

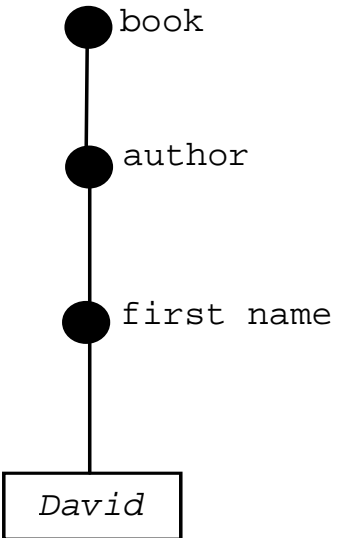
Consequently: ranking is possible

Example

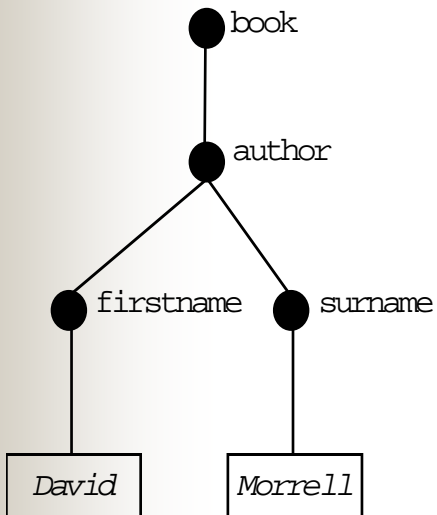
query Q_1 :



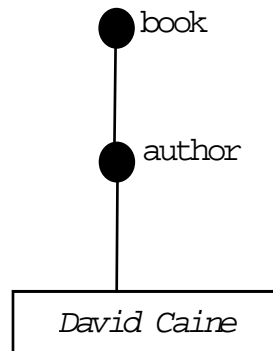
query Q_2 :



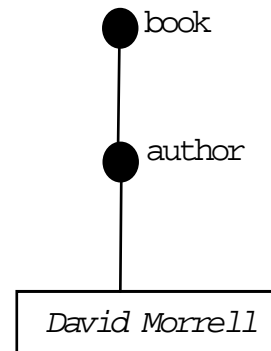
document D_1 :



document D_2 :



document D_3 :



Vector Space Model (VSM)

- collection C of documents

$$D_i = [w_{i,1}, w_{i,2}, \dots, w_{i,m}] \in \langle 0, 1 \rangle^m$$

- query

$$Q = [q_1, q_2, \dots, q_m] \in \langle 0, 1 \rangle^m$$

- e.g. cosine similarity

$$Sim(D_i, Q) = \frac{\sum_{j=1}^m w_{i,j} * q_j}{\sqrt{\sum_{j=1}^m (w_{i,j})^2 * \sum_{j=1}^m (q_j)^2}}$$

- Questions: how to use it for XML data and XML querying (index unit, index terms, retrievable unit, how do a weighting)



Outline

- Related works
- Matrix model
 - DataGuide for collection of XML documents
 - Weighting and ranking
 - Path transform matrix
 - Issues
- Extended Vector Space Model (EVSM)
- Renewing Matrix Model based on EVSM
- Experiments with INEX collection
- Conclusions



Related works

- First attempts: ApproXML (Schlieder, 2001), EVSM (Carmel et al, 2002)
- The best example: XIQRL (Fuhr, Grossjohann, 2001)
- XQuery – attempts to weighting and ranking, e.g. TeXQuery (Amer-Zahia et al, 2004)
- Recently: randomized indexes (Kakade, Raghavan, 2005)

Matrix Model*

- a document is modelled by a matrix

$$D_i = \begin{bmatrix} w_{i,1,1} & w_{i,1,2} & \dots & w_{i,1,k} \\ w_{i,2,1} & w_{i,2,2} & \dots & w_{i,2,k} \\ \dots & \dots & \dots & \dots \\ w_{i,m,1} & w_{i,m,2} & \dots & w_{i,m,k} \end{bmatrix} \in \langle 0,1 \rangle^{m,k}$$

k ... the number of paths in the XML structure of the entire document collection C ,

$w_{i,j,s} \in \langle 0,1 \rangle$ is the weight of the term t_j on the path s in the document D_i .

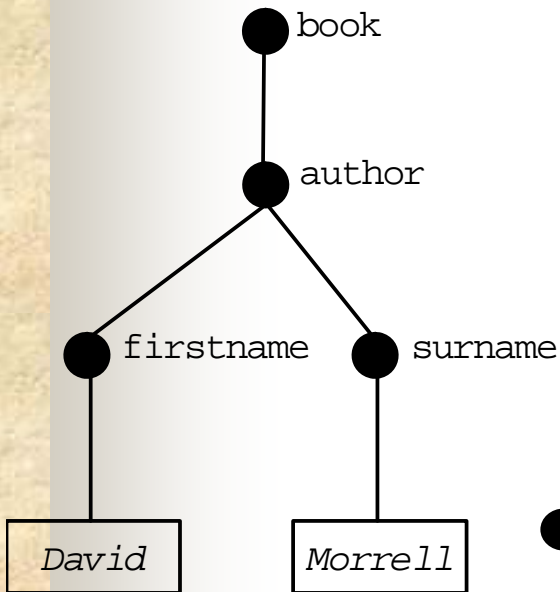
*Pokorný, J., Rejlek, V.: A Matrix Model for XML Data. Chap. in: Databases and Information Systems, Selected Papers from the 6th Int. Baltic Conf. DB&IS'2004, Vol. 118, Frontiers in Artificial Intelligence and Applications, IOS Press, 2005, pp. 53-64.



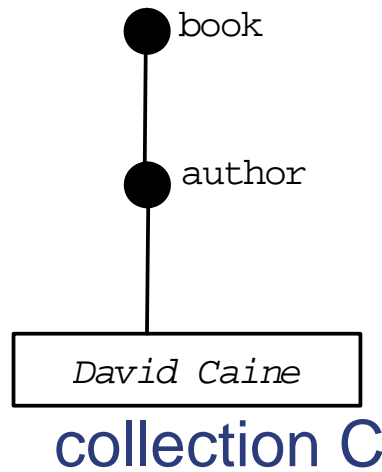
DataGuide for a collection

- How to model a collection C of XML data?
- A **DataGuide** for a collection C , DG_C , of XML documents is a tree T_C fulfilling the following conditions:
 - each path in C exists in T_C ,
 - each path in T_C exists in C ,
 - paths in T_C are unique (but not in C).
- There is a linear time construction of DG_C .

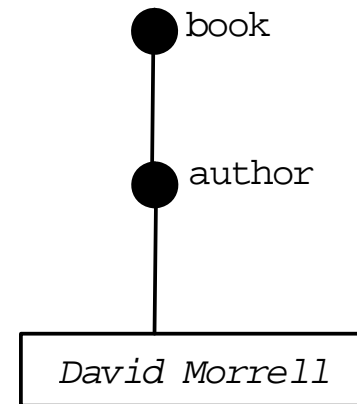
document D_1 :



document D_2 :

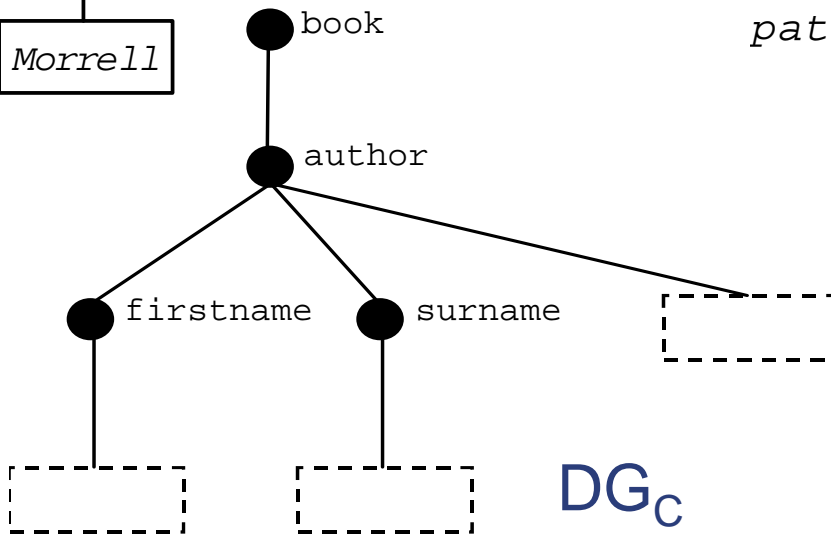


document D_3 :



paths:

- a) book->author
- b) book->author->firstname
- c) book->author->surname



Example 1



The approach

- weighting terms in paths
- similarity measure
 - naive
 - with the path transform matrix
 - normalized

Weighting

Given a term t , path p , and document D_i , the **term weight** $w_{i,t,p}$ is defined as

$$TF_{t,p}/m_p$$

where $TF_{t,p}$ is the number of t occurrences on p and m_p is the number of all terms associated with p .

Ranking

A **similarity** between a document and a query by a (normalized) similarity

$$Sim_2(D_1, Q) = \sum_{l=1}^m \frac{\sum_{j=1}^k w_{1,l,j} * q_{l,j}}{\sqrt{\sum_{j=1}^k (w_{1,l,j})^2 * \sum_{j=1}^k (q_{l,j})^2}}$$

	a	b	c	a	b	c	a	b	c
D_1 :	(0	1	0)	(0	0	1)	(0	0	0)]
D_2 :	(0,5	0	0)	(0	0	0)	(0,5	0	0)]
D_3 :	(0,5	0	0)	(0,5	0	0)	(0	0	0)]

"david"

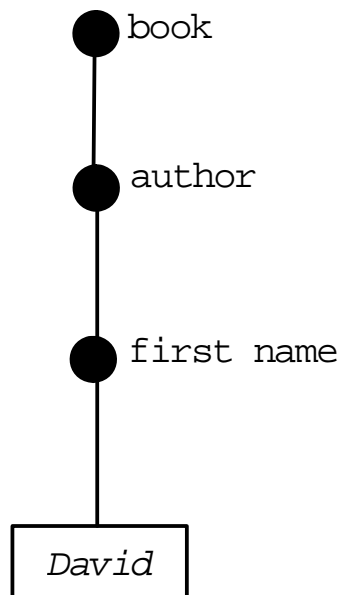
"morrell"

"caine"

query Q_1 :



query Q_2 :



queries

Example 1 cont.

	a	b	c	a	b	c	a	b	c
Q_1 :	[(1	0	0)	(0	0	0)	(0	0	0)]
Q_2 :	[(0	1	0)	(0	0	0)	(0	0	0)]
	"david"			"morrell"			"caine"		

Problems: some similarities are zeros; this is not in accordance with our intuition

Solution: to express similarities among paths in DG_C

Path Transform Matrix

path transform matrix ... a square matrix A of dimension $k \times k$, where k is the number of paths in DG_C , $a_{i,j} \in \langle 0,1 \rangle$, and $a_{i,i} = 1$.

A_1	a	b	c	
a	1	0,2	0,2	book→author
b	0,5	1	0	book→author→firstname
c	0,5	0	1	book→author→surname

e.g. book→author→firstname have to be $0.5 \times$ transformed on book→author

Transformation in detail

D ... a document matrix of dimension $m \times k$,

A ... a path transform matrix of dimension k ,

A **one step transformation** is a mapping $1st_tr(D, A)$ returning again a matrix of dimension $m \times k$, TD_1 with

$$w_{l,j} = \max(w_{l,j}, \max_{j=1 \dots k} (a_{j,i} * w_{l,j}))$$

A **transformation of the document matrix D by the path transform matrix A** is the transitive closure of $1st_tr(D, A)$.

Practice: (1) Transitive closure of A
(2) $1st_tr(D, A)$

Matrices TD a TQ

	a	b	c	a	b	c	a	b	c
TD_1 :	[(0,5	1	0,1)	(0,5	0,1	1)	(0	0	0)]
TD_2 :	[(0,5	0,1	0,1)	(0,0	0	0)	(0,5	0,1	0,1)]
TD_3 :	[(0,5	0,1	0,1)	(0,5	0,1	0,1)	(0	0	0)]
TQ_1 :	[(1	0,2	0,2)	(0	0	0)	(0	0	0)]
TQ_2 :	[(0,5	1	0,1)	(0	0	0)	(0	0	0)]

Normalization

$$Sim_2(D_1, Q_1) = \mathbf{0.62}$$

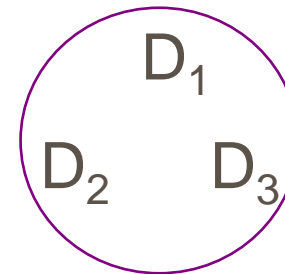
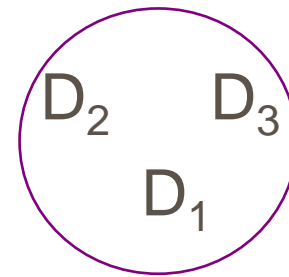
$$Sim_2(D_2, Q_1) = \mathbf{1}$$

$$Sim_2(D_3, Q_1) = \mathbf{1}$$

$$Sim_2(D_1, Q_2) = \mathbf{1}$$

$$Sim_2(D_2, Q_2) = \mathbf{0.62}$$

$$Sim_2(D_3, Q_2) = \mathbf{0.62}$$



satisfiable results



Experiences and critique

- First experiments (2004):
 - Shakespeare's plays,
 - synthetic data generated by XBench
 - comparisons with a vector model
 - More balanced results than VSM
- Issues:
 - construction of \mathbf{A} + its transitive closure $O(k^4)$ ✓
 - time complexity of \mathbf{A} transformation to \mathbf{D} $O(mk^2)$ ✓
 - static collections
 - norm choice

Extended Vector Space Model (EVSM)

Features:

- term weight is replaced by a term weight in context $w_D(t_i, c_j)$
- context resemblance $cr(c_j, c_k)$
 - cr values from [0,1] (when EVSM \approx VSM?)
 - based on algorithm Longest Common Subsequence
 - more general: approximate embedding of trees (see ApproXML)

$$Sim_3(D, Q) = \frac{\sum_{(t_i, c_j) \in Q} \sum_{(t_i, c_k) \in D} w_Q(t_i, c_j) * w_D(t_i, c_k) * cr(c_j, c_k)}{\|Q\| * \|D\|}$$



Matrix Model vs. EVSM

If cr is defined as $cr(x,y) = a_{x,y}^2$, then similarity Sim_2 in the matrix model is equal to similarity Sim_3 except for norms $||Q||$ and $||D||$

Notice: construction of cr acts also transitively in some extent: if $cr(a,b) \neq 0$ and $cr(b,c) \neq 0$ then a and b have at least one common element as well as b and c . Hence a and c have a common element and $cr(a,c) \neq 0$.

$$Sim_4(D_i, Q) = \sum_{t=1}^m \frac{\sum_{x=1}^k \sum_{y=1}^k w_{t,l,x} * q_{t,y} * cr(x, y)}{\sqrt{\sum_{x=1}^k (w_{i,t,x})^2 * \sum_{y=1}^k (q_{t,y})^2}}$$

Combination of Matrix Model and EVSM

$$Sim_4(D_i, Q) = \sum_{t=1}^m \frac{\sum_{x=1}^k \sum_{y=1}^k w_{t,l,x} * q_{t,y} * cr(x, y)}{\sqrt{\sum_{x=1}^k (w_{i,t,x})^2 * \sum_{y=1}^k (q_{t,y})^2}}$$

Statement 1: There is an algorithm for computing Sim_4 with time complexity $O(m * k^2)$ (compare with calculation of TD)

In fact, it is much more effective, since the number of terms in Q and at once in D is of order less than in C . Similarly, the #paths in D is much less than the #paths in DG_C .

Intuitively: the algorithm does not calculate all products in sums.



Experiments with the INEX Collection

Up to 2004: the full-texts, marked up in XML, of 12107 articles of the IEEE Journals (494 MB, 8 million elements).

Topics (queries) are of types CAS and CO (30 CAS topics and 36 CO topics which are distinguished by attribute `/inex_topic/@query_type` in INEX 2003).

Measures: *exhaustivity (e-value)*, *specificity (s-value)*

Each dimension has 4 grades of relevance:

0: Irrelevant,

1: Marginally relevant,

2: Fairly relevant,

3: Highly relevant.

We did a transformation $\{0,1,2,3\}^2 \rightarrow \langle 0,1 \rangle$ by $f(e,s) = \frac{e * s}{10}$

Experiments with the INEX Collection

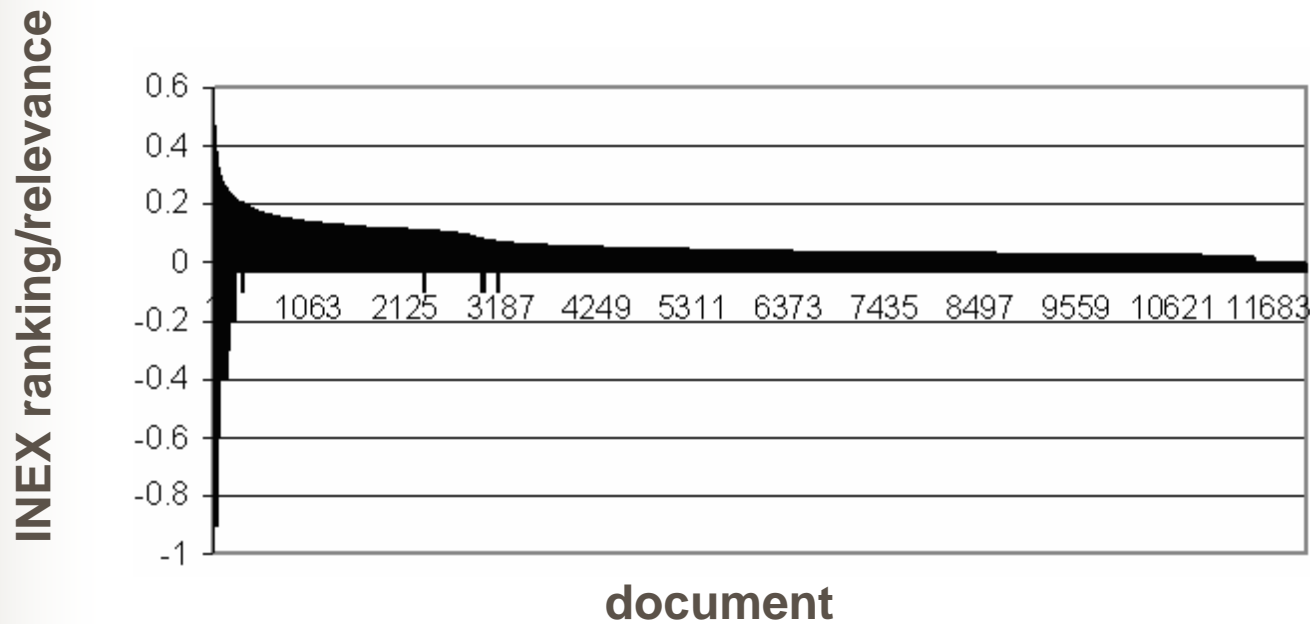
topic 62

Topics were transformed into (XML) queries, e. g.

```
<inex_topic topic_id="62" query_type="CAS" ct_no="5">
  <title> //article[about(.,'security +biometrics')AND
    about(./sec,'"facial recognition"')]</title>
  <description> Find articles about biometrical security systems
    with a section about facial recognition. </description>
  <narrative> To be relevant the document has to discuss the
    usage of facial recognition in biometrical security
    systems.</narrative>
  <keywords>facial recognition, face recognition, security,
    biometrics </keywords>
</inex_topic>
<article>security biometrics biometrical systems
  <sec>facial recognition face recognition</sec>
</article>
```

query 62

Experiments

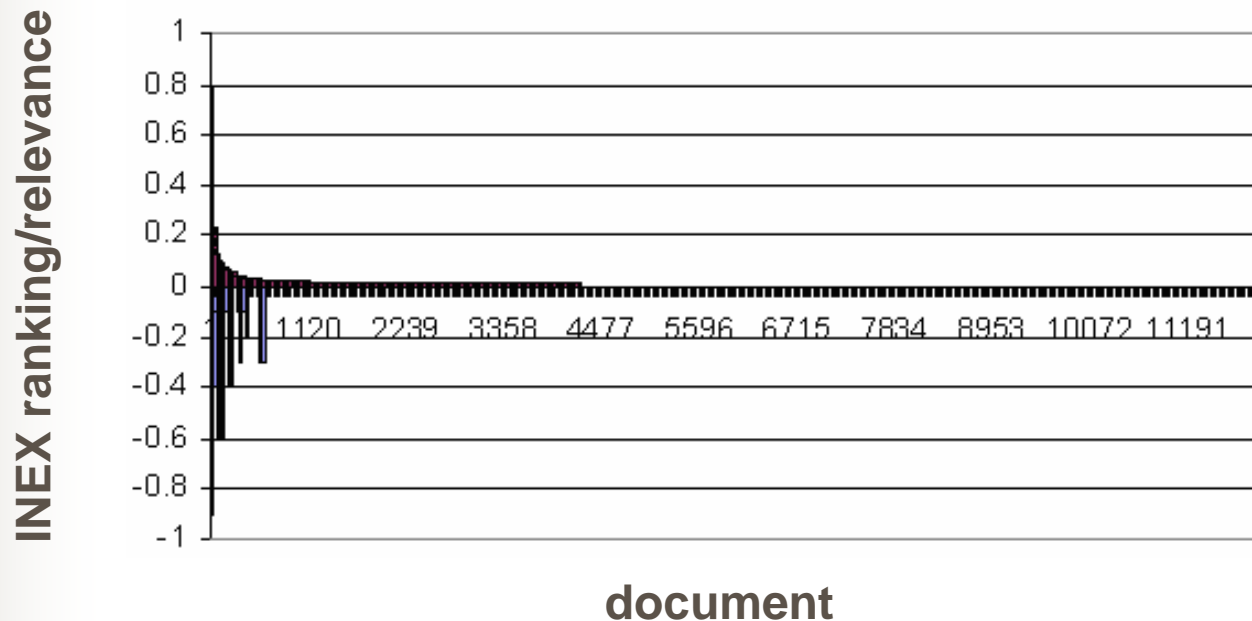


Comparison of the matrix model and INEX ranking for query q62

Experiments

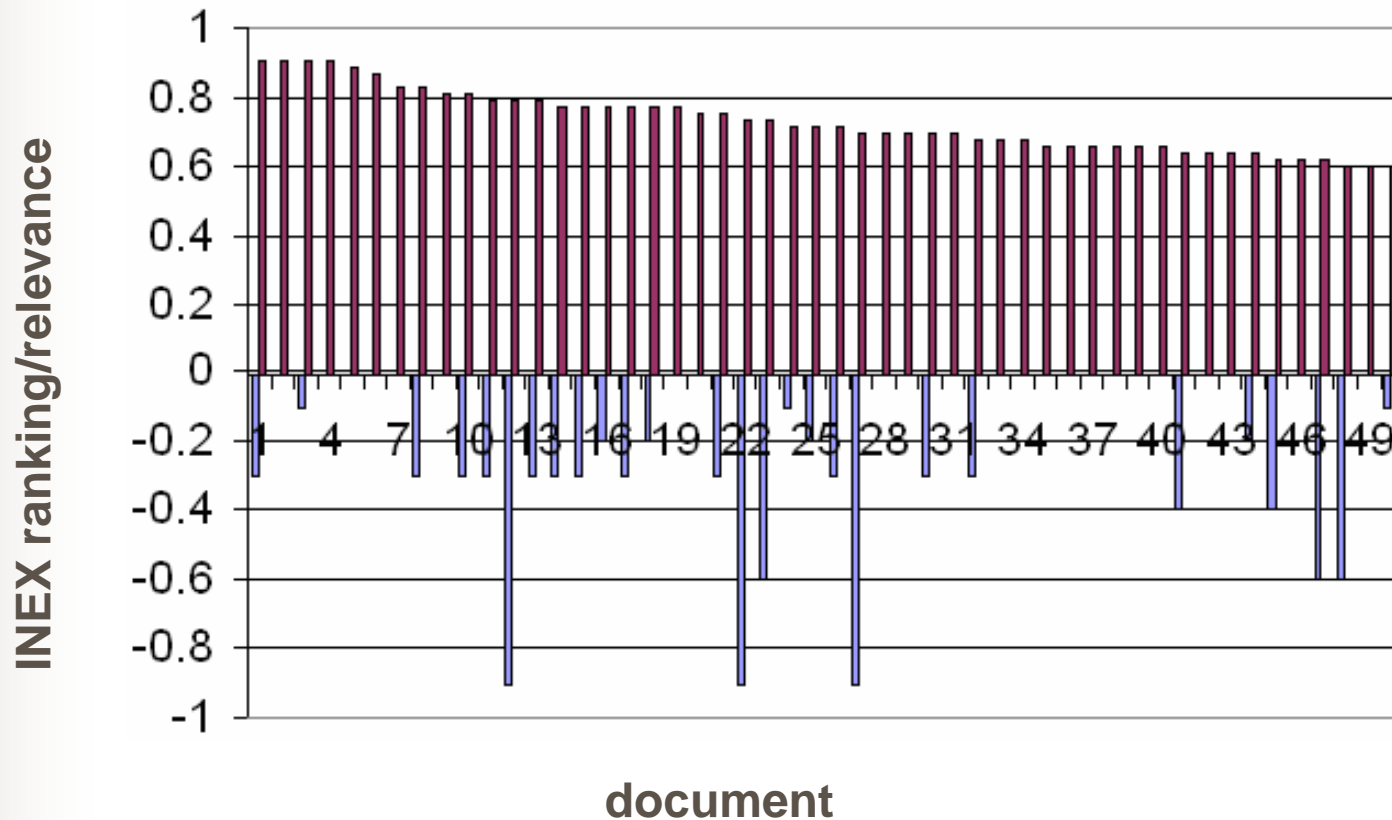
positive values – matrix model ranking

negative values – INEX ranking



Comparison of the vector model and INEX ranking for query q62

Experiments

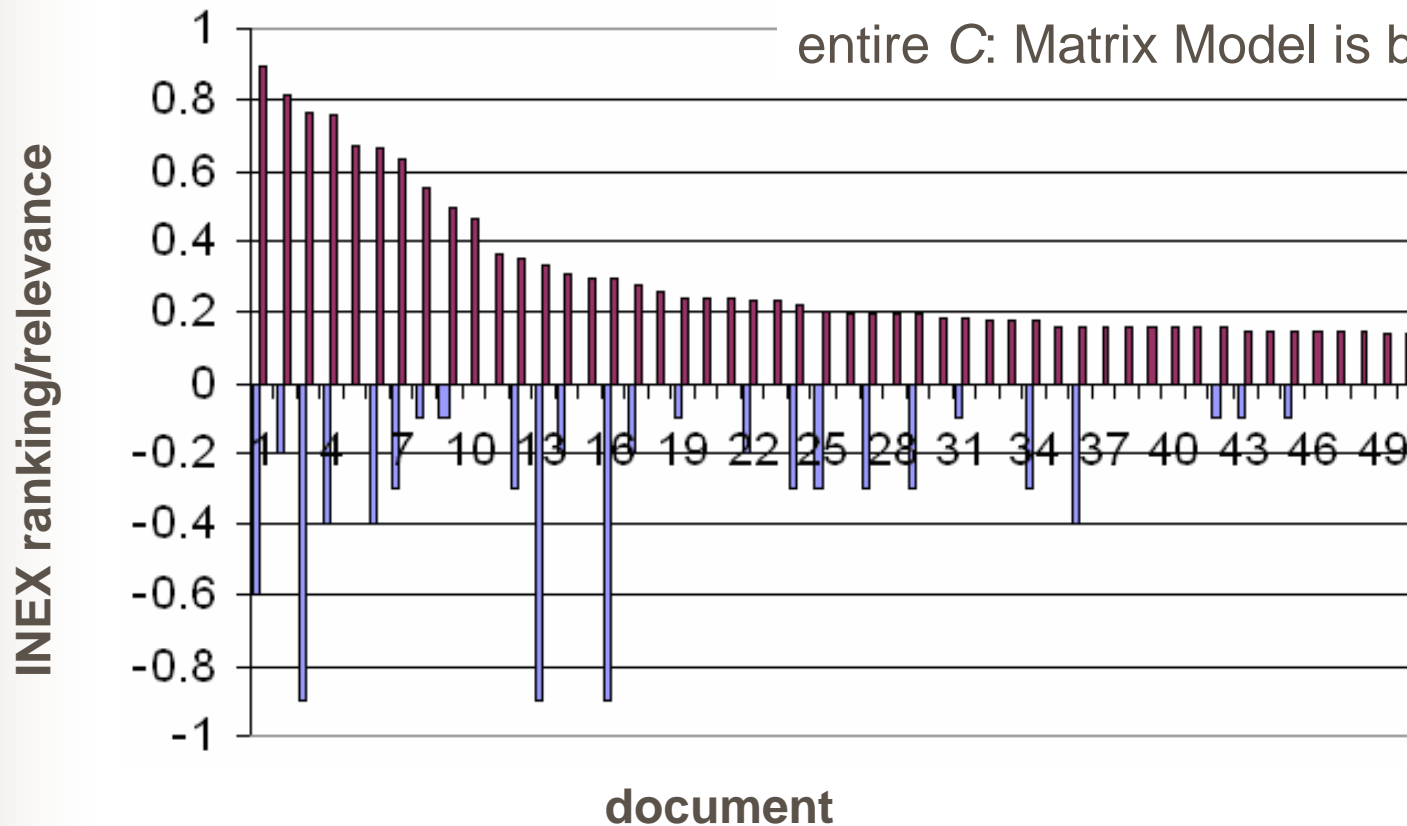


Comparison of matrix model and INEX ranking for the first 50 documents and query q62

Experiments

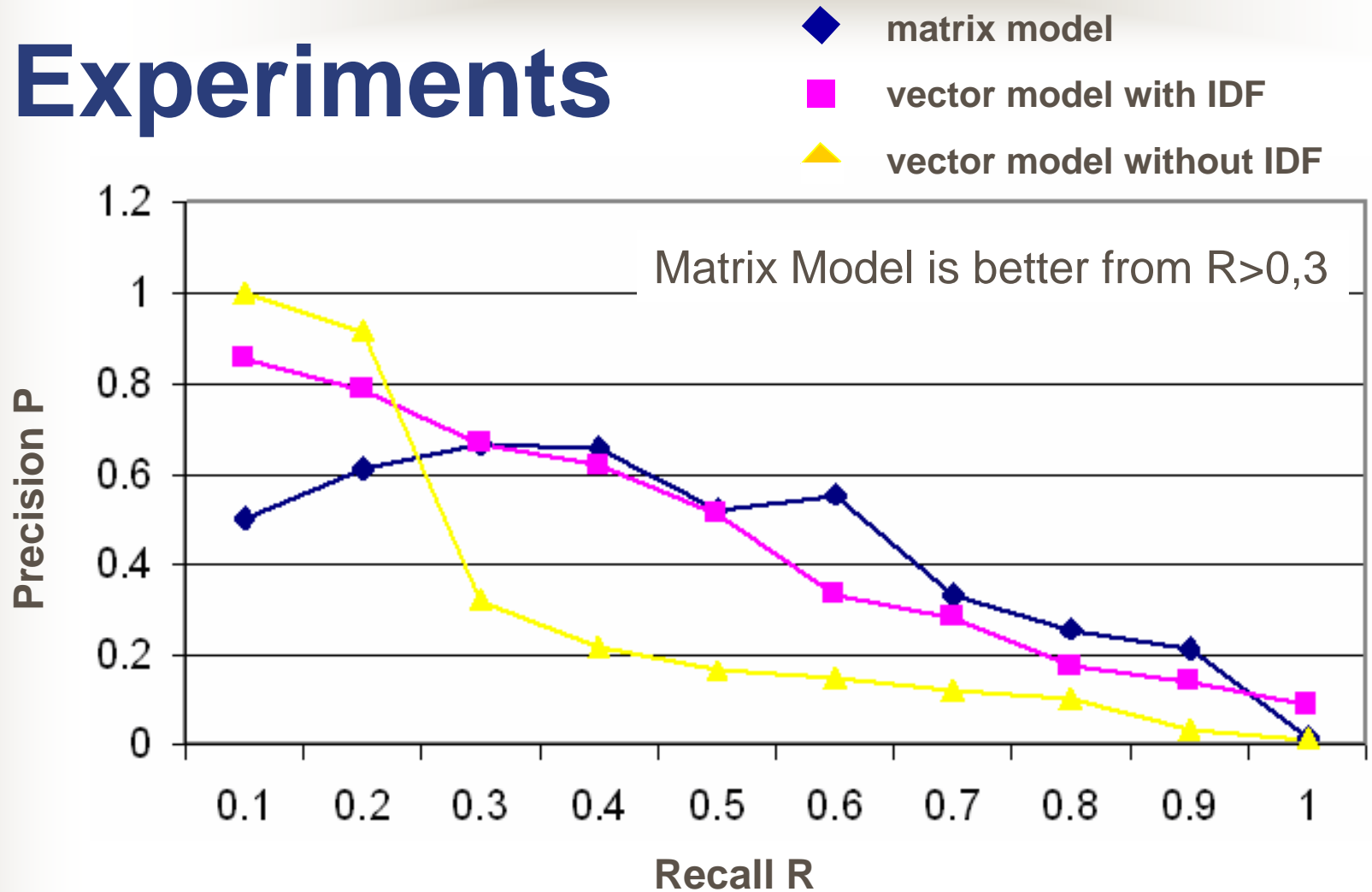
50-top: VSM is better for q62

entire C: Matrix Model is better



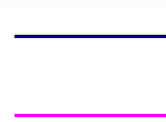
Comparison of vector model and INEX ranking for the first 50 documents and query q62

Experiments



Dependence of P on R for matrix and vector model with/without IDF for q62

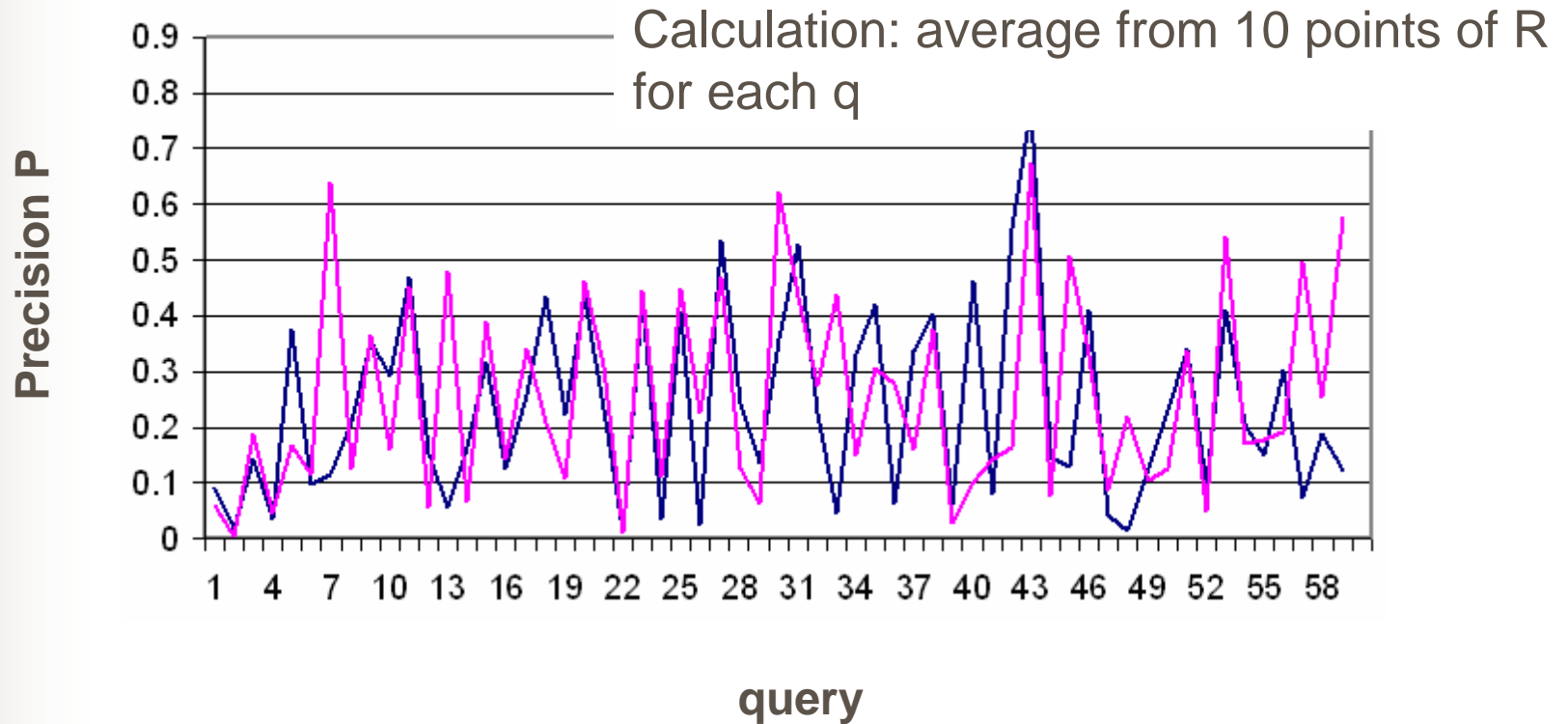
Experiments



matrix model

vector model with IDF

There is no unique winner.



Average precision for matrix model and vector with IDF



Comparison

- The behaviour on INEX collection is similar for matrix and vector models
- Hypothesis: matrix model is the same or better than the vector model, if q contains above-average number of terms
- Index sizes: 32% and 44% of the INEX size for VSM and matrix model, resp.

Q	type	#t	is s	is w	is b	Q	type	#t	is s	is w	is b
61	CAS	10	x			92*	CO	18			x
62	CAS	8	x			93	CO	2	x		
63	CAS	6		x		94	CO	5			x
64	CAS	7	x			95	CO	9	x		
65	CAS	5		x		96	CO	3	x		
66*	CAS	19	x			97	CO	6		x	
67	CAS	4			x	98	CO	12		x	
68	CAS	7			x	99	CO	5		x	
69	CAS	2		x		100	CO	4			x
70	CAS	3			x	101	CO	5			x
71*	CAS	18			x	102	CO	6		x	
72	CAS	9		x		103*#	CO	13		x	
73	CAS	8	x			104	CO	9			x
74*	CAS	22	x			107*	CO	20	x		
75*#	CAS	16		x		108	CO	7		x	
76*	CAS	13			x	109	CO	9			x
77*	CAS	17	x			110*	CO	21	x		
78	CAS	10	x			111*	CO	24			x
79	CO	6			x	112*	CO	14	x		
80	CAS	9		x		113*	CO	13			x
81*	CAS	18			x	116*#	CO	18		x	
83	CO	8	x			117*	CO	13			x
84	CO	5			x	119*	CO	17	x		
85*	CAS	22		x		121*	CO	13	x		
86	CO	11	x			122	CO	10	x		
87	CAS	9	x			123*	CO	13			x
88	CAS	9		x		124*	CO	30			x
89*	CAS	23	x			125*	CO	13	x		
90*	CAS	16			x	126*	CO	20	x		
91	CO	7	x			Total			24	15	20



Conclusions

- Advantages:
 - matrix model considers XML structure + content
 - A perspective for rich structured large documents
- Possibilities:
 - distinguishing national languages via the matrix A
 - Indexing (what to index)
- Extensions: including links