Diversification for Keyword Search over Structured Databases

Outline

- Introduction
- Related Background
- *DivQ*
- Evaluation metrics
- Experiment
- Conclusion

Introduction

- Keywords queries over structured database
  - an organized collection of data
  - Data may be stored in different tables
  - Computationally expensive if too much data need to be retrieved cross multiple tables
  - Not attract much attention
- Compared with unstructured document
  - keyword queries need to be interpreted in terms of the underlying database
  - take advantage of the structure of the database

Introduction

	movie	actor
Item 1	War	Tom Hanks
Item 2	…	…

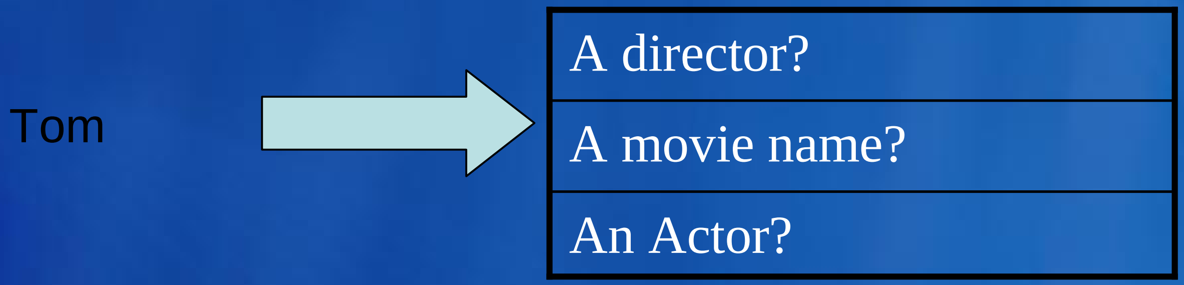
	movie	director
Item 1	…	…
Item 2	War	Tom Billy

	movie	year
Item 1	…	…
Item 2	War	2011

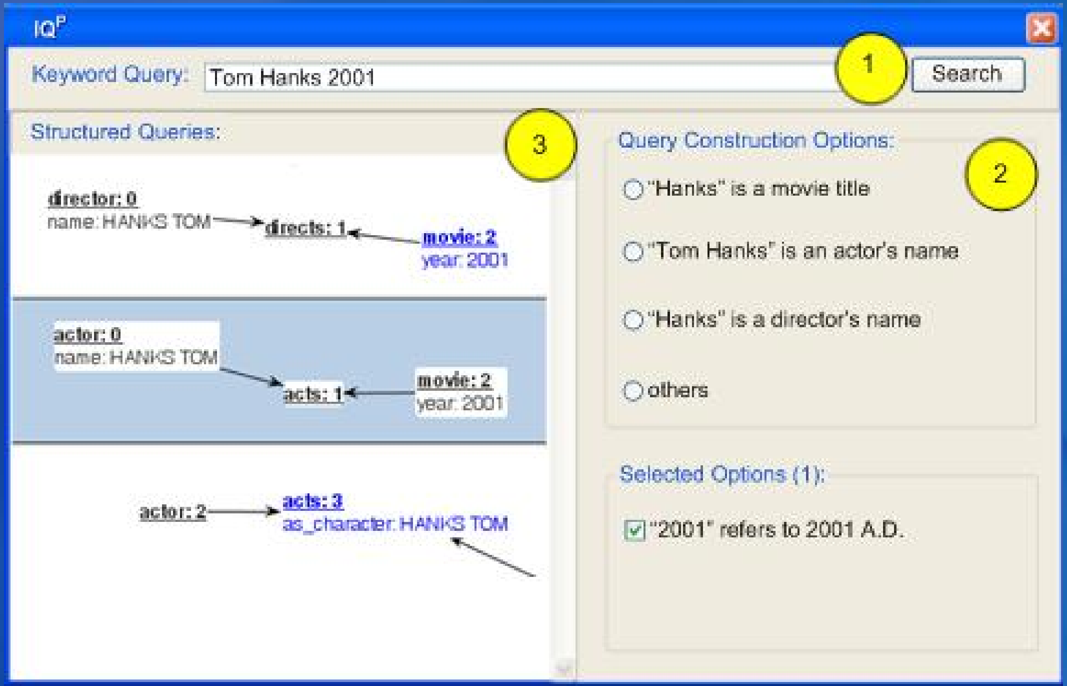
- Keyword queries over structured databases are notoriously ambiguous
  - Single interpretation of a keyword query is not enough
  - Multiple interpretation will yield to overlapping results
- Search "Tom 2011"
  - Tom : a director?
  - Tom: an actor?

Introduction

- Diversification aims at minimizing user’s dissatisfaction
  - Provide users a quick glance of the major plausible interpretations, so that the users can simply choose.



Introduction



Keyword Query: Tom Hanks 2001

Query Construction Options:

- ○ "Hanks" is a movie title
- ○ "Tom Hanks" is an actor's name
- ○ "Hanks" is a director's name
- ○ others

Selected Options (1):

- ☑ "2001" refers to 2001 A.D.

- diversification should take advantage of the structure of the database
  - Query disambiguation before actual execution
  - Avoid computational overhead for retrieving and filtering actual result
  - executes only the top-ranked query interpretations at last

Introduction

- This scheme balances the relevance and novelty of keyword search results
  - A probabilistic model helps to rank the possible interpretations, to create semantic interpretations
  - a scheme to diversify the search results by re-ranking query interpretations, generating the top-k most relevant and diverse query interpretations

Related Background

- In document retrieval
  - Diversification performs as a post-processing or re-ranking step
  - first retrieve relevant results and then filter or re-order the result list to achieve diversification
- However in structured database
  - computationally expensive
  - obtained by joining multiple tables
- So in *DivQ*, take the advantage of rich structure
  - clear semantics in database before retrieving any results
  - Only the results of the top ranked interpretations are retrieved from the database

Related Background

- Diversification by classifying search results
  - based on similarity
  - understandable for end user
  - classes are usually pre-defined
- In *DivQ* -- a special kind of classes
  - Well-defined semantics
  - Query interpretations are generated based on users' keyword
  - consider the similarity between query interpretations to avoid redundant search results

Related Background

- Some ideas from traditional IR - variance
  - to select top-n documents first
  - order them by balancing the overall relevance of the list against its variance
- Other complementation in *DivQ*
  - categorization, which takes into account user preferences

Major part -- DivQ

DivQ

- *DivQ* translates a keyword query to a set of structured queries, taking not only relevance but also diversification into consideration

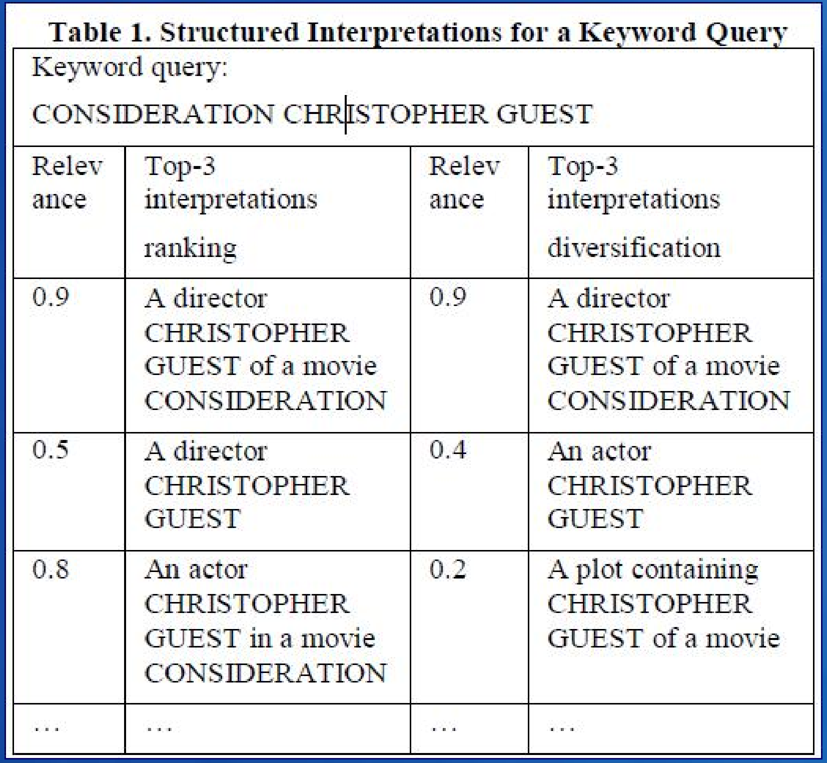


Table 1. Structured Interpretations for a Keyword Query

Keyword query:

CONSIDERATION CHRISTOPHER GUEST

Relevance	Top-3 interpretations ranking	Relevance	Top-3 interpretations diversification
0.9	A director CHRISTOPHER GUEST of a movie CONSIDERATION	0.9	A director CHRISTOPHER GUEST of a movie CONSIDERATION
0.5	A director CHRISTOPHER GUEST	0.4	An actor CHRISTOPHER GUEST
0.8	An actor CHRISTOPHER GUEST in a movie CONSIDERATION	0.2	A plot containing CHRISTOPHER GUEST of a movie
…	…	…	…

DivQ - Bringing Keywords into Structure

- translate a keyword query *K* to a structured query *Q*
  - a set of ***keyword interpretations*** *Ai*:*ki*, map each *ki* to *Ai*
  - Then joins the keyword interpretations using a predefined ***query template*** *T*
- For example
  - Search "CONSIDERATION CHRISTOPHER GUEST"
  - "director:CHRISTOPHER", "director:GUEST", "movie: CONSIDERATION"
  - query interpretation: "A director *X* of a movie *Y*"
- "complete query interpretation", "partial query interpretation"

DivQ - Estimating Query Relevance

- estimate relevance as the conditional probability, P(*Q*|*K*)
  - keyword query *K*
  - *Q* is the user's intended interpretation of *K*
- probability P(*Q*|*K*) can be expressed as $P(Q \mid K) = P(I,T \mid K)$.
  - query interpretation *Q* is composed of a query template *T*
  - I: a set of keyword interpretations

$$I = \{A_j : \{k_{j1}, k_{jn}\} \mid A_j \in T, \{k_{ji}, k_{jn}\} \subset K, \{k_{i1}, k_{im}\} \cap \{k_{j1}, k_{jn}\} = \{\}\ \mathit{for}\ i \neq j\}$$

DivQ - Estimating Query Relevance

- Two assumptions for simplifying the computation
  - each keyword has one particular interpretation intended by the user
  - The probability of a keyword interpretation is independent

- Based on these assumptions and Bayes' rule

$$P(Q \mid K) \propto \left(\prod_{A_j \in T} P\left(A_j : \{k_{j1}, k_{jn}\} \mid A_j\right) \right) \times \left(\prod_{k_u \in K \cap k_u \notin Q} P_u \right) \times P(T)$$

DivQ - Estimating Query Relevance

$$P(Q \mid K) \propto \left(\prod_{A_j \in T} P\left(A_j : \{k_{j1}, k_{jn}\} \mid A_j\right) \right) \times \left(\prod_{k_u \in K \cap k_u \notin Q} P_u \right) \times P(T)$$

- P(*Aj*:{*kj1*,*kjn*}|*Aj*) represents the probability that *Aj*:{*kj1*,*kjn*} are a part of the query interpretation, estimated using attribute specific term frequency
- Constant smoothing factor *Pu,* the probability that keyword *Ku* does not match any available attribute in the database, smaller than the minimum probability of any existing keyword interpretation
- P(*T*) is the prior probability that the template *T* is used to form a query interpretation, a frequency of the template's occurrence in the database query log if available

DivQ - Estimating Query Similarity

- The resulting query interpretations should be not only relevant but also as dissimilar to each other

$$Sim(Q_1, Q_2) = \frac{|I_1 \cap I_2|}{|I_1 \cup I_2|}$$

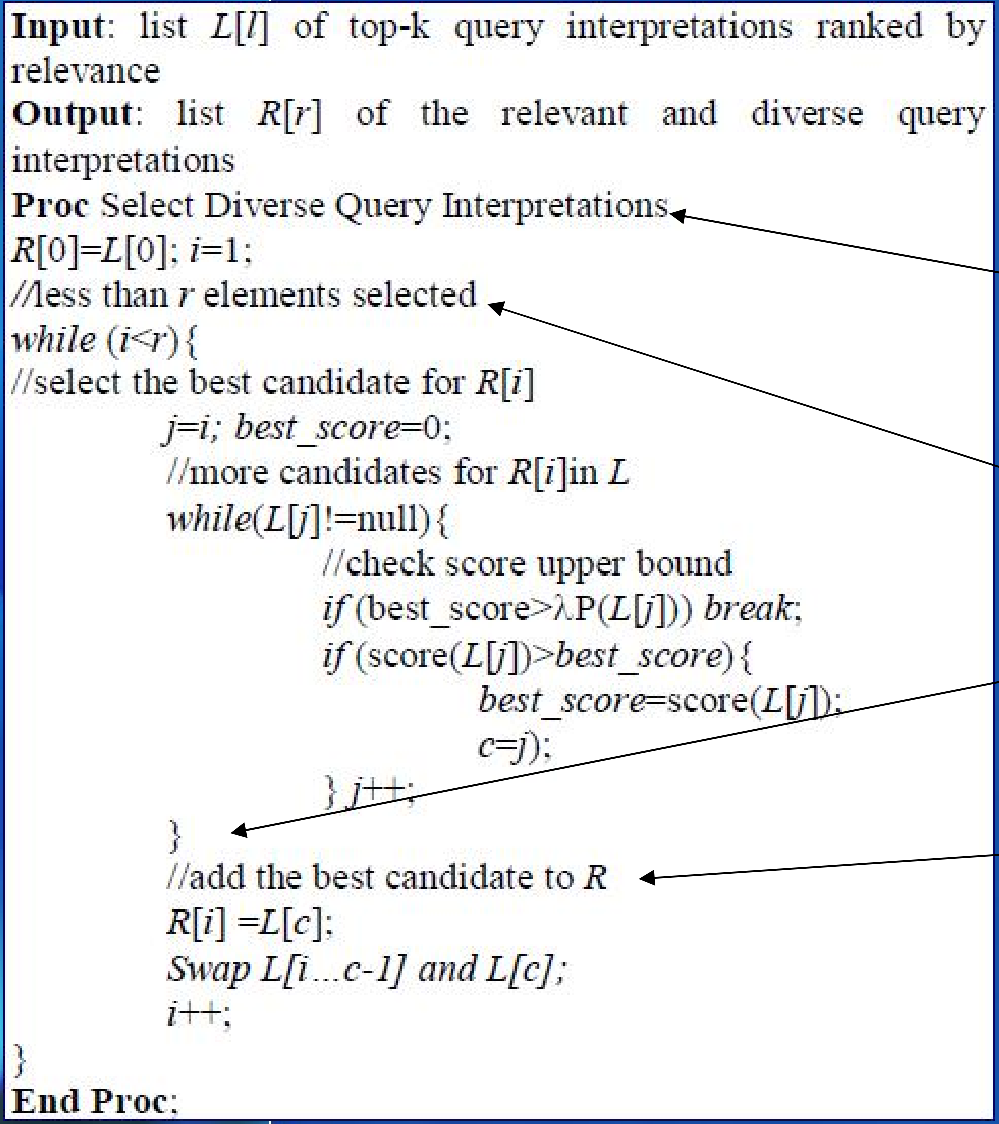
  – *Q1* and *Q2* be two query interpretations of a keyword query *K*
  – *I1* and *I2* be the sets of keyword interpretations contained by *Q1* and *Q2*
  – resulting similarity value should always fall in [0, 1], 1 means the highest possible similarity

DivQ - Combining Relevance and Similarity

- For generating the top-k query interpretations that are both relevant and diverse
  - First, select the most relevant interpretation as the top-1 interpretation
  - Then select the interpretation based on both its relevance and novelty

$$Score(Q) = \lambda \cdot P(Q \mid K) - (1-\lambda) \cdot \sum_{q \in QI} \frac{Sim(Q,q)}{|QI|}$$

  - a query interpretation *Q*
  - a set of query interpretations *QI* that are already presented to the user
  - λ is a parameter to trade-off query interpretation relevance against novelty, λ =1 only care about relevance, 0 otherwise
  - The interpretation with highest score will be next interpretation



Input: list $L[l]$ of top-k query interpretations ranked by relevance
Output: list $R[r]$ of the relevant and diverse query interpretations

```
Proc Select Diverse Query Interpretations
R[0]=L[0]; i=1;
//less than r elements selected
while (i<r){
//select the best candidate for R[i]
        j=i; best_score=0;
        //more candidates for R[i]in L
        while(L[j]!=null){
                //check score upper bound
                if (best_score>λP(L[j])) break;
                if (score(L[j])>best_score){
                        best_score=score(L[j]);
                        c=j);
                } j++;
        }
        //add the best candidate to R
        R[i] =L[c];
        Swap L[i…c-1] and L[c];
        i++;
}
End Proc;
```

- For creating a set *R* of the most relevant and diverse query interpretations
  - starts with the most relevant query interpretation at the top of *L*
  - scan the remaining candidate elements in *L*, compare their scores according to the formula
  - Add item

DivQ - The Diversification Algorithm

- Worst case
  - Worst complexity is $O(l*r)$
  - maximal number of similarity computations is $(l^2-l)/2$
  - l is the number of query interpretations and r is the number of interpretations in the result list R

Evaluation

Evaluation metrics

- In document retrieval
  - α -NDCG (normalized Discounted Cumulative Gain)
  - S-recall
- In structured data
  - α -NDCG-W
  - Weighted S-Recall
- Differences
  - primary key -- notion of information nugget -- subtopic
  - α -NDCG and S-recall assume equal relevance of information nuggets and subtopics in a document. However relevance of primary keys in a query result may vary a lot

Evaluation metrics - CG

- What is CG?
  - Cumulative Gain (CG) is the predecessor of DCG
  - The value: The gain *G*[*k*] at rank *k*
  - does not care the position of a result in result set.
  - The CG at a particular rank position *p* is defined as:

$$\mathrm{CG_p} = \sum_{i=1}^{p} rel_i$$

CG = 3+2+3+0+1+2

D1	3
D2	2
D3	3
D4	0
D5	1
D6	2

Evaluation metrics - DCG

- What is DCG?
  - Discounted Cumulative Gain
  - Take position into consideration
  - Change the position, the value changes

$$\mathrm{DCG_p} = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2 i}$$

D1	3
D2	2
D3	3
D4	0
D5	1
D6	2

$$\mathrm{DCG_6} = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2 i} = 3 + (2 + 1.887 + 0 + 0.431 + 0.772) = 8.09$$

Evaluation metrics - nDCG

- What is nDCG
  - Normalized DCG Discounted Cumulative Gain
  - Sort the order first, then calculate

$$\mathrm{nDCG}_6 = \frac{DCG_6}{IDCG6} = \frac{8.09}{8.693} = 0.9306$$

  - IDCG: ideal DCG

D1	3
D3	3
D2	2
D6	2
D5	1
D4	0

Evaluation metrics - α -NDCG

- α -nDCG
  - *G*[*k*] is extended with a parameter α ,
  - a trade-off between relevance and novelty
  - α -nDCG views a document as the set of information nuggets
  - α is in the interval [0, 1]; 0 just care about the relevance, increasing α , novelty is rewarded with more credit

Evaluation metrics - α -NDCG-W

- α - NDCG-W
  - For reflecting the graded relevance assessment on the PK
  - Take overlapping and diversification into consideration

$$G[k] = relevance(Q_k) \cdot (1-\alpha)^r$$

  - *r* expresses overlap in result list at ranks 1…*k*-1.
  - Each *pk* is distinct with others in other interpretations
  - for each primary key pk_i in the result of Q_k, count how many query interpretations with pk_i were seen before

$$r = \sum_{pk_i \in Q_k} \sum_{j \in [1,k-1]} \left| pk_i \in Q_j \right|$$

Evaluation metrics – weighted S-Recall

- S-recall is the number of unique subtopics covered by the first *k* results, divided by the total number of subtopics
- In database keyword search
  - single primary key corresponds to a subtopic in S-recall
  - take the graded relevance of subtopics into account
  - WS-Recall is computed as the aggregated relevance of the subtopics divided by the maximum possible relevance

$$WS-recall@k = \frac{\sum_{pk \in Q_{1..k}} relevance(pk)}{\sum_{pk \in U} relevance(pk)}$$

  - *U* is the set of relevant subtopics (primary keys)
  - Same to S-recall, if only binary relevance assessments are available

Experiments – Dataset and Queries

- two real-world datasets
  - a crawl of the Internet Movie Database (IMDB)
    - seven tables
    - more than 10,000,000 records
  - a crawl of a lyrics database from the web
    - five tables
    - around 400,000 records
- No associated query log
  - extracted the keyword queries from logs of MSN and AOL
  - obtained thousands of queries for the IMDB and lyrics domains

Experiments – Dataset and Queries

- most popular keyword queries
  - first sorted the queries based on frequency in the log
    - each domain, select 200 most frequent queries with non-empty results exist in the database
    - *single concept queries* -- mostly either single keyword or single concept queries
  - manually selected for more complex queries
    - 100 queries for each dataset from the query log
    - *multi-concept queries*
- for each keyword query
  - ranked interpretations
  - entropy in the top-10 ranks of the resulting list -- ambiguity
  - selected 25 single concept and 25 multi-concept queries with the highest entropy for each dataset

Experiments – User Study

- at most the top-25 interpretations

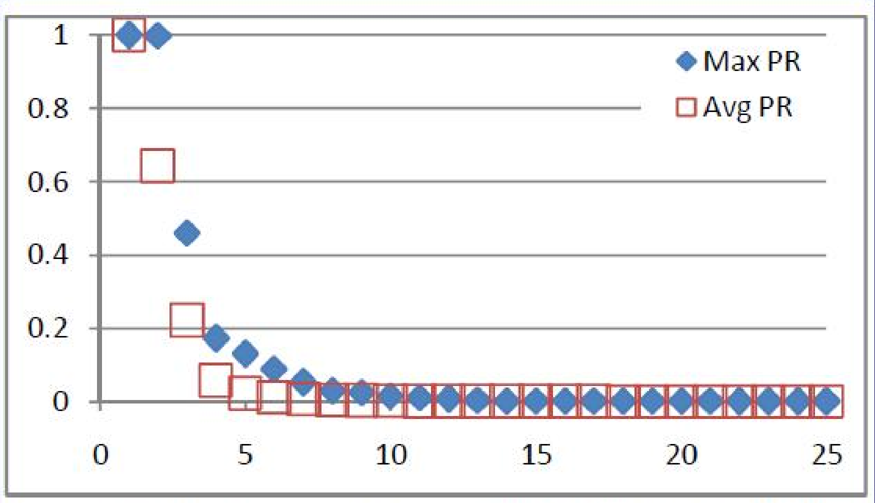


Figure 1a. Maximum and Average Probability Ratio, IMDB.

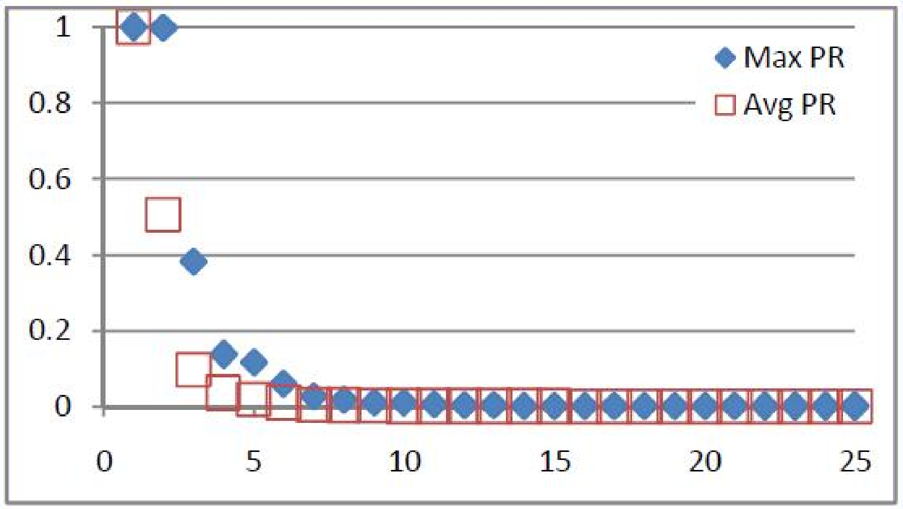


Figure 1b. Maximum and Average Probability Ratio, Lyrics.

– average ratio of the probability of a query at rank *i* and the aggregated probabilities of queries at rank *j<i*

$$\boxed{PR_i = P(Q_i \mid K) / \sum_{j<i} P(Q_j \mid K)}$$

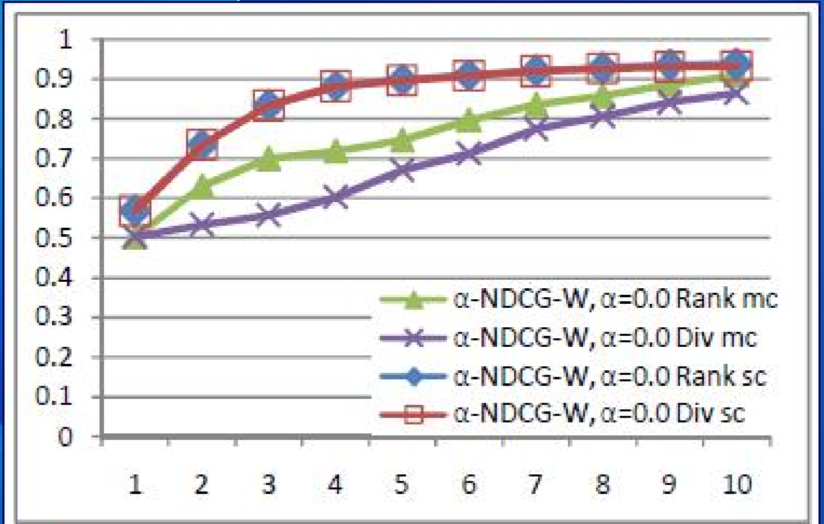
Experiments – User Study

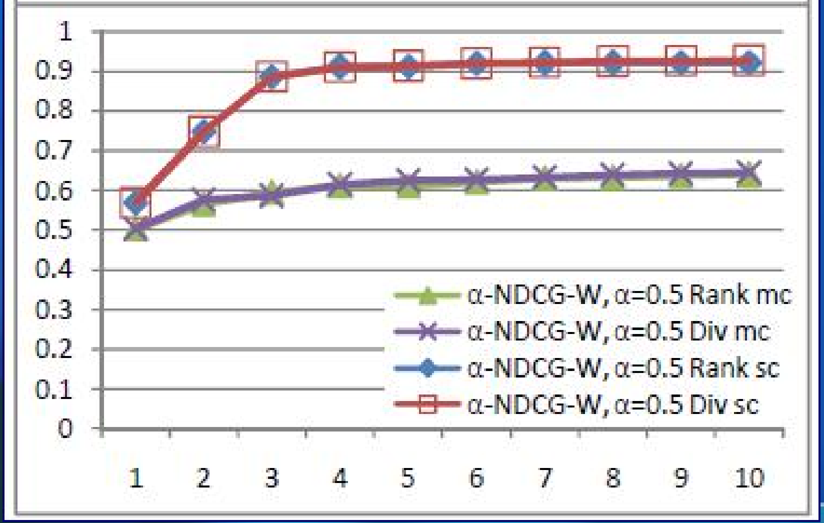
- For each query
  - pruned all query interpretations *Qi* whose probability constituted less than 0.1% of the aggregated probability
  - included at most five more interpretations with probability below the threshold
  - randomized the order when presented for user assessment
- In total
  - Each user -- 630 interpretations for IMDB, 517 for Lyrics
  - 10 persons - all tasks, 6 persons - 30% IMDB, 9% Lyrics
  - two-point Likert scale for each interpretation
  - Agreement in kappa statistics: 0.33 in IMDB; 0.28 in Lyrics
    - Such low agreement, additional indication of ambiguity of queries

Experiments – α-nDCG-W

- For assessing quality of ranking and diversification
  - measure α - NDCG-W by varying α parameter
  - α =0, novelty of results is completely ignored = NDCG
  - α =0.5, novelty is given a certain credit
  - α =0.99 results without novelty are regarded as redundant

Experiments – α-nDCG-W





- Y-axis: α -NDCG-W value
- Rank: without diversification
- Div: with diversification
- When α =0.99 and k>3
  - diversification on mc queries outperforms by about 7%

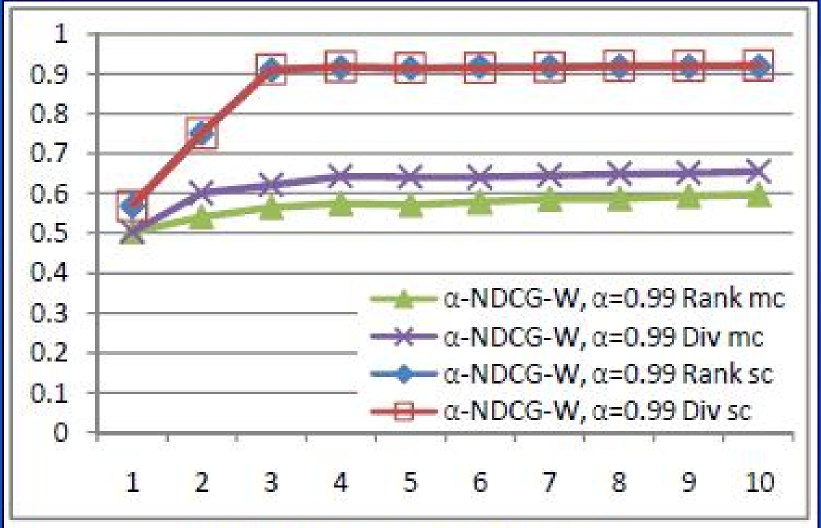
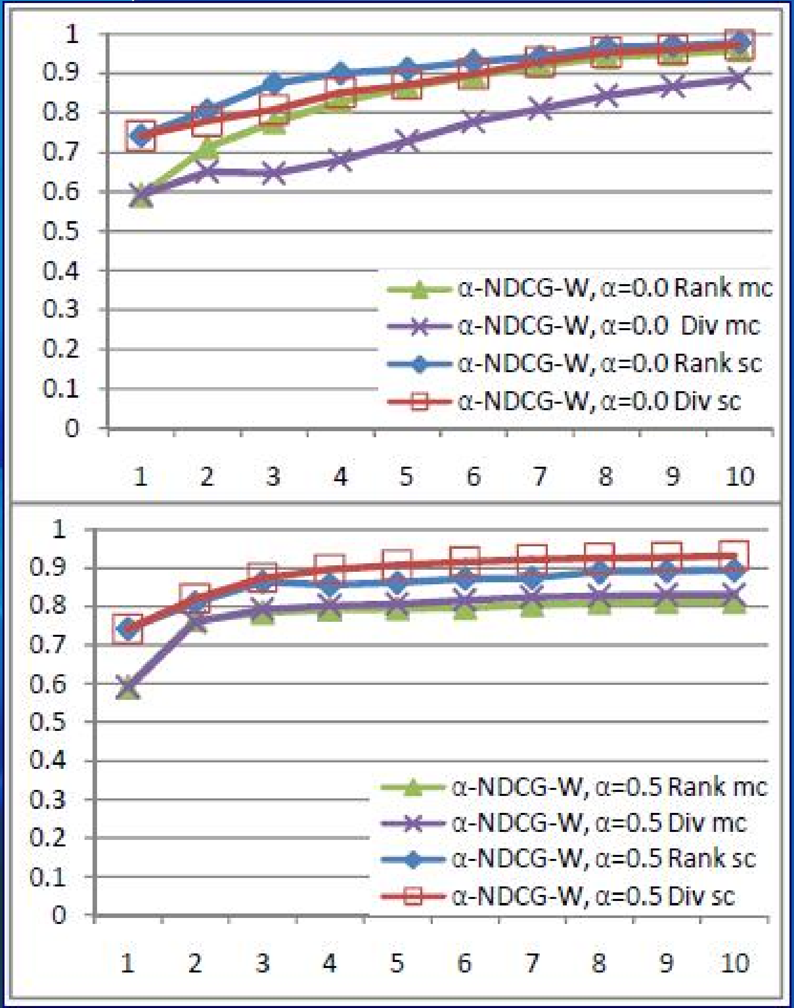


Figure 2a. α-NDCG-W, IMDB.

Experiments – α-nDCG-W



- Y-axis: α -NDCG-W value
- Rank: without diversification
- Div: with diversification
- When α =0.99 and k>3
  - diversification on mc queries outperforms by about 7%

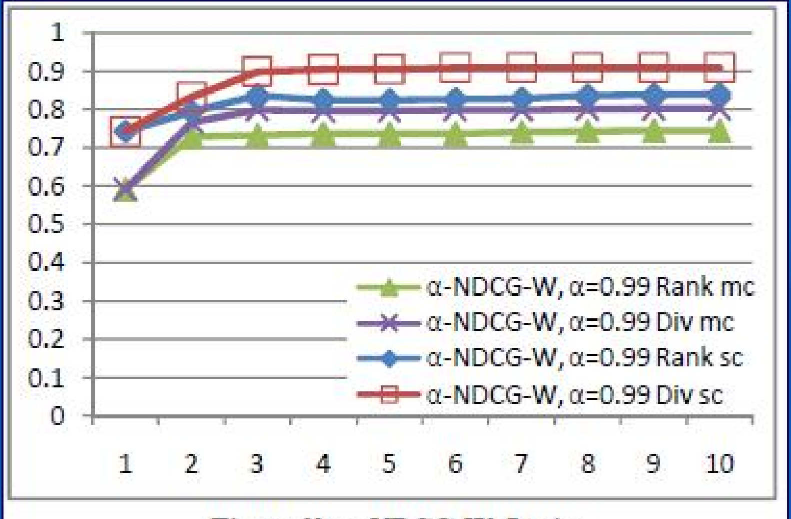


Figure 2b. α-NDCG-W, Lyrics.

Experiments – WS-recall

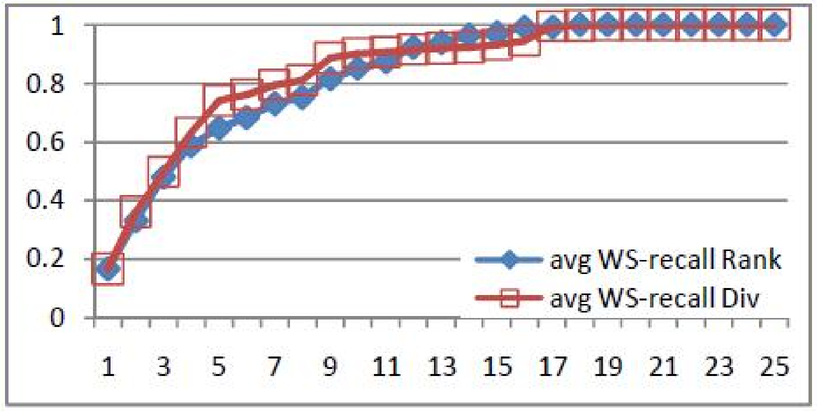


Figure 3a. WS-recall for Ranking and Diversification, IMDB.

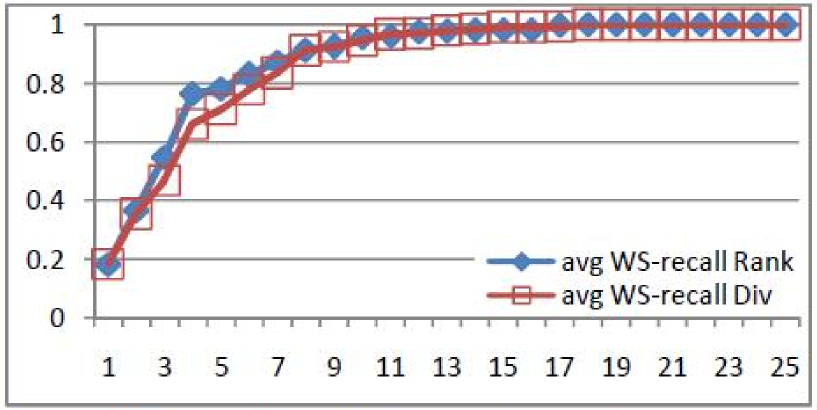


Figure 3b. WS-recall for Ranking and Diversification, Lyrics.

- Y-axis: WS-recall value
- Normalizing result sizes for WS-recall is future work
- No significant effect of diversification

Experiments – Balancing Relevance and Novelty

$$Score(Q) = \lambda \cdot P(Q \mid K) - (1-\lambda) \cdot \sum_{q \in QI} \frac{Sim(Q,q)}{|QI|}$$

- λ is parameter to balance relevance against novelty

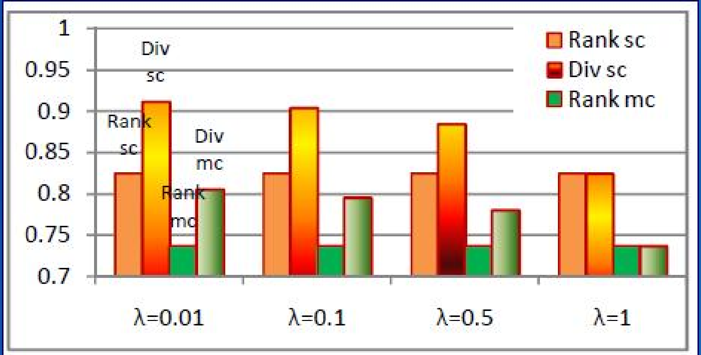


Figure 4. *α*-NDCG-W, *k*=5, *α*=0.99, Lyrics.

- α -NDCG-W values decrease with increasing λ , until λ = 1
- The smaller λ is, the more visible is the impact of diversification

Conclusion

- Advantages
  - Take diversification into consideration
  - A good attempt for queries under structured database
  - Evaluation results demonstrate that the novelty of keyword search results improved
  - Better characterized than initial relevance ranking
- Drawbacks
  - No significant improvement according to the evaluation
  - Still need improvement

Thank you for your attention

Thank you for your attention