Information Retrieval
Topic- The complete search system
Lecture-26

Prepared By

Dr. Rasmita Rautray & Dr. Rasmita Dash
Associate Professor
Dept. of CSE

Content

- Vector space model
- Similarity measure
- Cosine similarity

- Term frequency weight
- Inverse document frequency weight
- Tf-idf weight
- Cosine similarity

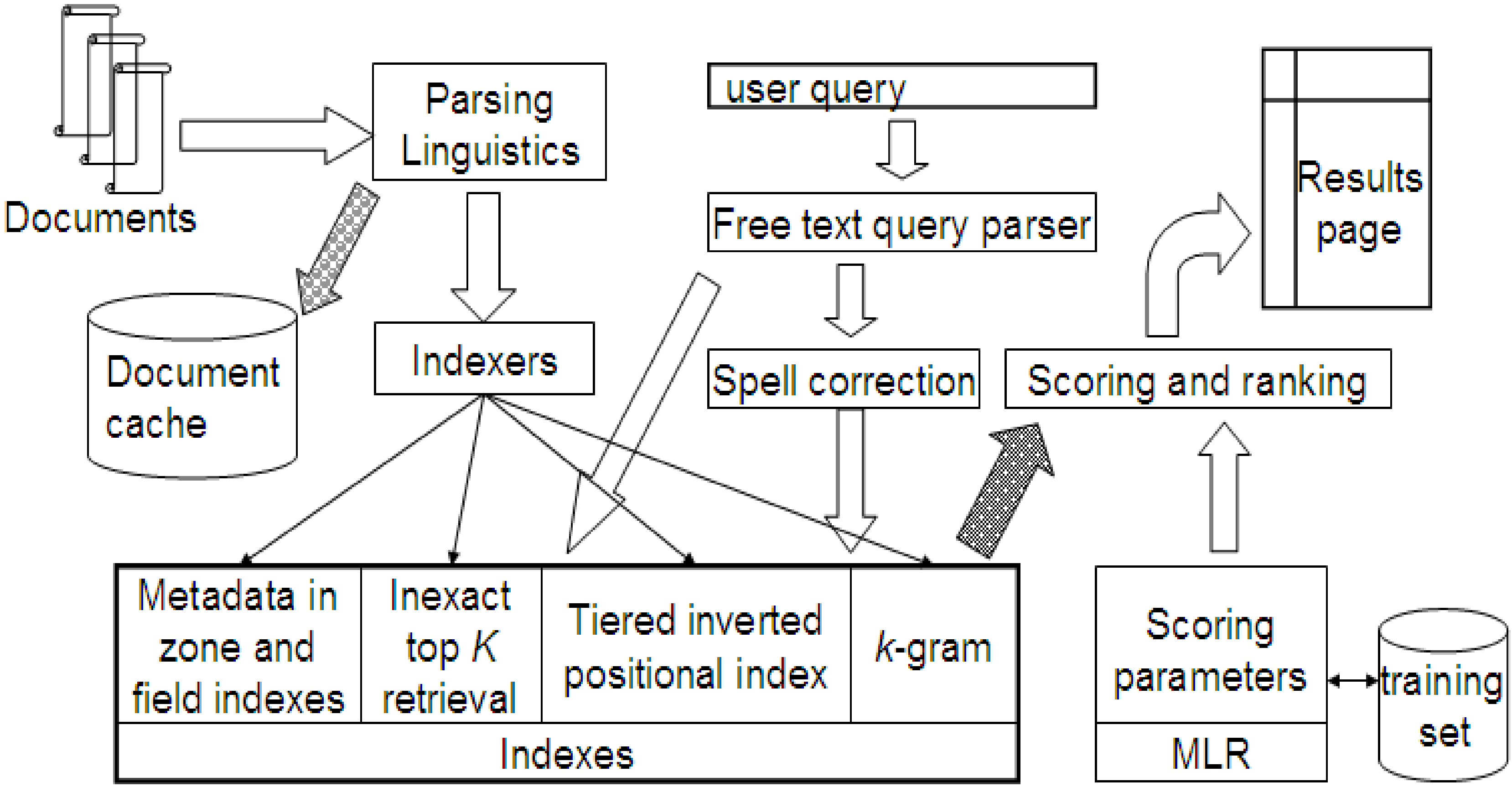
Why is ranking so important?

- Last lecture: Problems with unranked retrieval
  - Users want to look at a few results – not thousands.
  - It’s very hard to write queries that produce a few results.
  - Even for expert searchers
    - Ranking is important because it effectively reduces a large set of results to a very small one.
- Next: More data on “users only look at a few results”

Importance of ranking: Summary

- Viewing abstracts: Users are a lot more likely to read the abstracts of the top-ranked pages (1, 2, 3, 4) than the abstracts of the lower ranked pages (7, 8, 9, 10).
- Clicking: Distribution is even more skewed for clicking
- In 1 out of 2 cases, users click on the top-ranked page.
- Even if the top-ranked page is not relevant, 30% of users will click on it.
  - Getting the ranking right is very important.
  - Getting the top-ranked page right is most important.

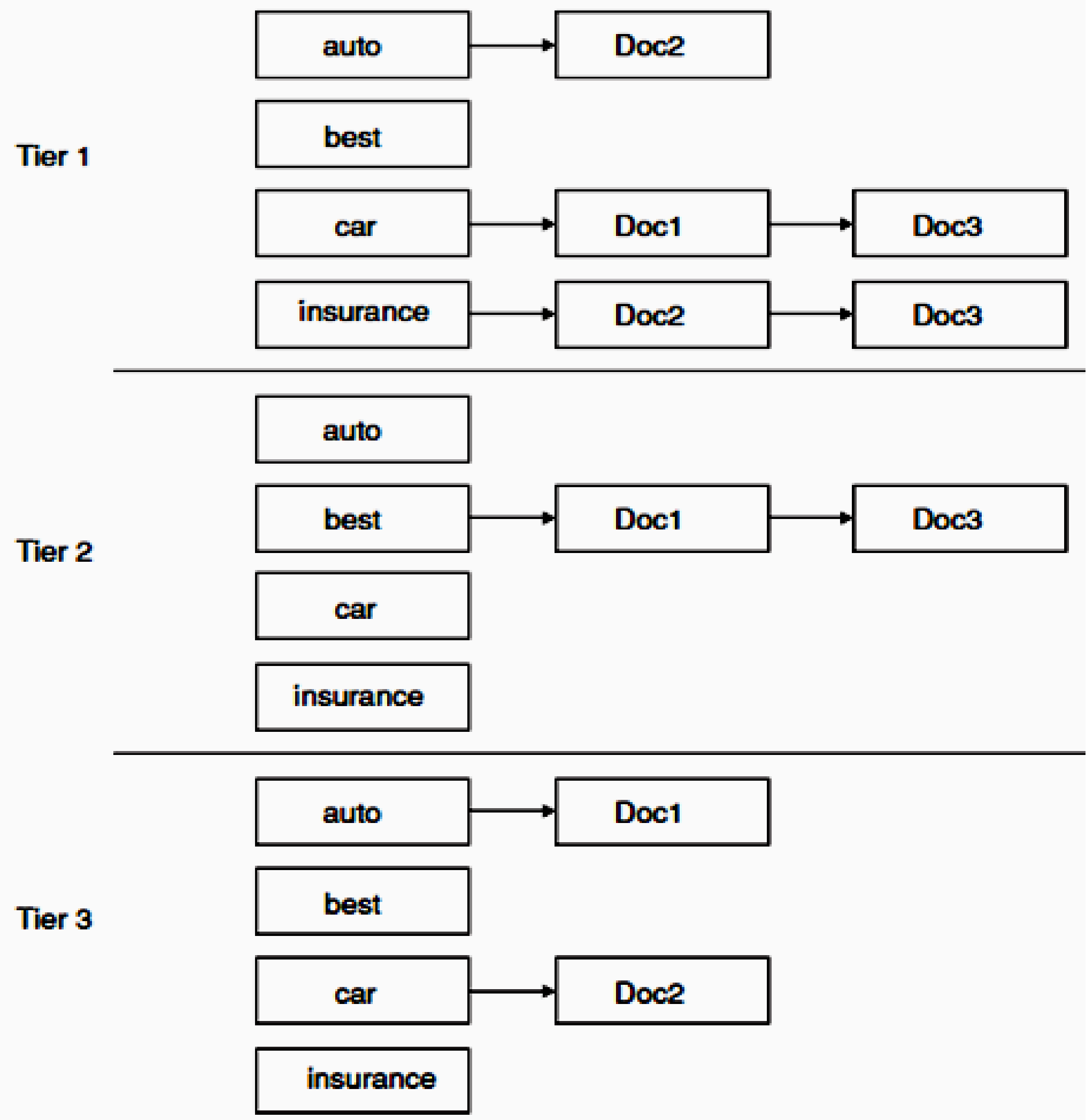
Complete search system



Tiered indexes

- Basic idea:
    - Create several tiers of indexes, corresponding to importance of indexing terms
    - During query processing, start with highest-tier index
    - If highest-tier index returns at least k (e.g., k = 100) results: stop and return results to user
    - If we’ve only found < k hits: repeat for next index in tier cascade
- Example: two-tier system
    - Tier 1: Index of all titles
    - Tier 2: Index of the rest of documents
    - Pages containing the search words in the title are better hits than pages containing the search words in the body of the text.

Tiered index



Tier 1

auto → Doc2

best

car → Doc1 → Doc3

insurance → Doc2 → Doc3

Tier 2

auto

best → Doc1 → Doc3

car

insurance

Tier 3

auto → Doc1

best

car → Doc2

insurance

Components we have introduced thus far

- Document preprocessing (linguistic and otherwise)
- Positional indexes
- Tiered indexes
- Spelling correction
- k-gram indexes for wildcard queries and spelling correction
- Query processing
- Document scoring

Components we haven’t covered yet

- Document cache: we need this for generating snippets (= dynamic summaries)
- Zone indexes: They separate the indexes for different zones: the body of the document, all highlighted text in the document, anchor text, text in metadata fields etc
- Machine-learned ranking functions
- Proximity ranking (e.g., rank documents in which the query terms occur in the same local window higher than documents in which the query terms occur far from each other)
- Query parser

Thank You