CS317

Information Retrieval

Week 05

Muhammad Rafi

March 11, 2021

Computing Scores in Complete Search Systems(VSM)- Optional

Chapter No. 7

Agenda

- Efficient scoring and ranking
- Inexact Top k documents
- Index Elimination
- Champion Lists
- Static Quality Scores
- Impact Ordering
- Cluster Pruning
- Conclusion

Efficient Scoring and Ranking

- Modified Cosine computation as discussed in the last lecture.
  - Only compute the partial scores for each q term and document d (only common dimensions will add values to score).
  - Should we go for all N document- obviously not.
  - We ae only interested in k top ranked documents
  - How we can make it efficient?
    - If we know in advance that which documents are high-scoring for a given query?
    - If we can arrange all the documents score in an efficient top retrieval data structures (Max. Heap)
    - Any rough estimates of ranking which can be computed quickely will be helpful.

Fast Cosine Scores

```
FASTCOSINESCORE($q$)
float $Scores[N] = 0$
for each $d$
do Initialize $Length[d]$ to the length of doc $d$
for each query term t
do calculate $w_{t,q}$ and fetch postings list for $t$
    for each pair($d$, $tf_{t,d}$) in postings list
    do add $wf_{t,d}$ to $Scores[d]$
Read the array $Length[d]$
for each $d$
do Divide $Scores[d]$ by $Length[d]$
return Top $K$ components of $Scores[]$
```

Index Elimination

- Only Consider high-Idf query terms
- Only Consider doc containing many query terms.

Champion List

- Precompute for each dictionary term t, k documents of highest weight in t's postings.
  - Call it champion list for term t
  - Only compute the scores for k documents.
  - This k may not be same for every term t.
  - Pick the top (10-20 documents based on scores)

Static Quality Scores

- Top ranking documents should be both "Relevant" and "Authoritative"
  - Relevant – Terms / cosine scores
  - Authoritative – document host (PageRank)
- Modeling Authority
  - Assign a query independent score to each document – quality score (0-1)
- NET Score
  - NET-score (q,d) = g(d) + cos(q,d)
  - May be weighted combination
  - Return top K documents on NET-Score

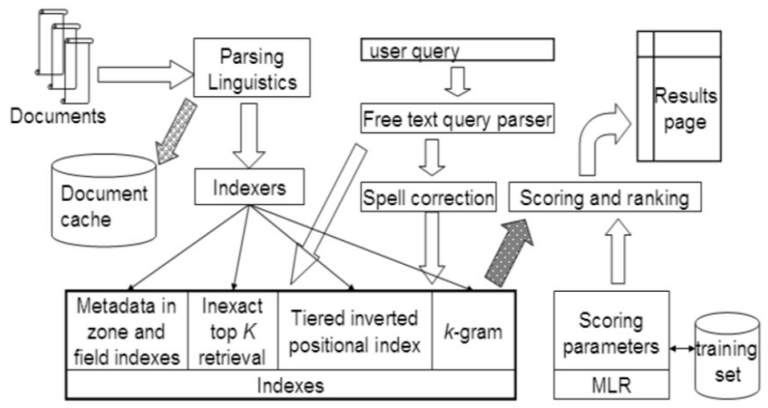
Impact Ordering

- Early Termination
  - Process t posting list with fixed k documents.
  - Stop for documents having wf(t,d) < alpha
  - For each term in query – terminate early and get a union for all selected documents.
- idf – ordering terms
  - High idf contribute more to score
  - Stop if the doc score is relatively unchanged for next documents.

Cluster Pruning

- Preprocessing
  - Pick some landmark documents for each term
  - From all other documents – apply knn to link these documents to a landmark document. (Leader <-> Followers)
- Query Processing
  - Given a q find the nearest leader (landmark document)
  - Get the k nearest document to this leader.
  - Apply Cosine similarity to these

Complete Search Systems



Conclusion

- Large Scale search systems – perform a lot of optimizations
- If we can make a user HAPPY we can present results in a Proxy form.
- She will never feel the difference