벡터 스페이스 모델을 위한 문서 정규화

고려대학교

미디어 공학과
전희원

개요

- 현 검색엔진의 검색 속도를 올릴 여러 방안이 필요
- 그 방안 중에서 미리 계산을 줄일 수 있는 방법을 많은 검색엔진에서 사용중임
- 그 방법 중에 문서 길이 **정규화 인자**를 미리 저장하는 방법을 개선하고자 함

그럼 정규화 인자 (normalization factor) 란 ?

- 문서의 길이가 긴 문서일수록 색인어의 빈도때문에 검색될 확률이 높아진다 .
- 길이가 짧은 문서가 정확한 문서일 가능성이 많다 .
- 이런 사실을 기반으로 길고 짧은 문서에 대한 보정을 하는것이 normalization factor 이다 .

기존의 검색엔진은 ?

- 대표적으로 Lucene 이라는 검색엔진

$$score(q,d) = \sum_{t \in q} (tf(t \in d) \times idf(t)^2 \times getBoost(t \in q) \times getBoost(t.field \in d) \times lengthNorm(t.field \in d)) \times coord(q,d) \times queryNorm(sumOfSquaredWeights)$$

문서 길이 정규화 펙터

$$lengthNorm = \frac{1.0}{\sqrt{numTerms}}$$

어떻게 이런 식이 도출되었나 ?

- From a common similarity (cosine measure)

$$sim(Q, D_i) = \frac{\sum_{j=1}^{V} w_{Q,j} \times w_{i,j}}{\sqrt{\sum_{j=1}^{V} w_{Q,j}^2 \times \sum_{j=1}^{V} w_{i,j}^2}}$$

Normalization factor

$$sim(Q, D_i) = \frac{\sum_{j=1}^{V} w_{Q,j} \times w_{i,j}}{\sqrt{\text{number of terms in } D_i}}$$

$$lengthNorm = \frac{1.0}{\sqrt{numTerms}}$$

Approximate normalization

Approximate normalization 의 근거

- Full normalization 에서 계산의 부하 때문임
- 실험 결과의 근거에 따름

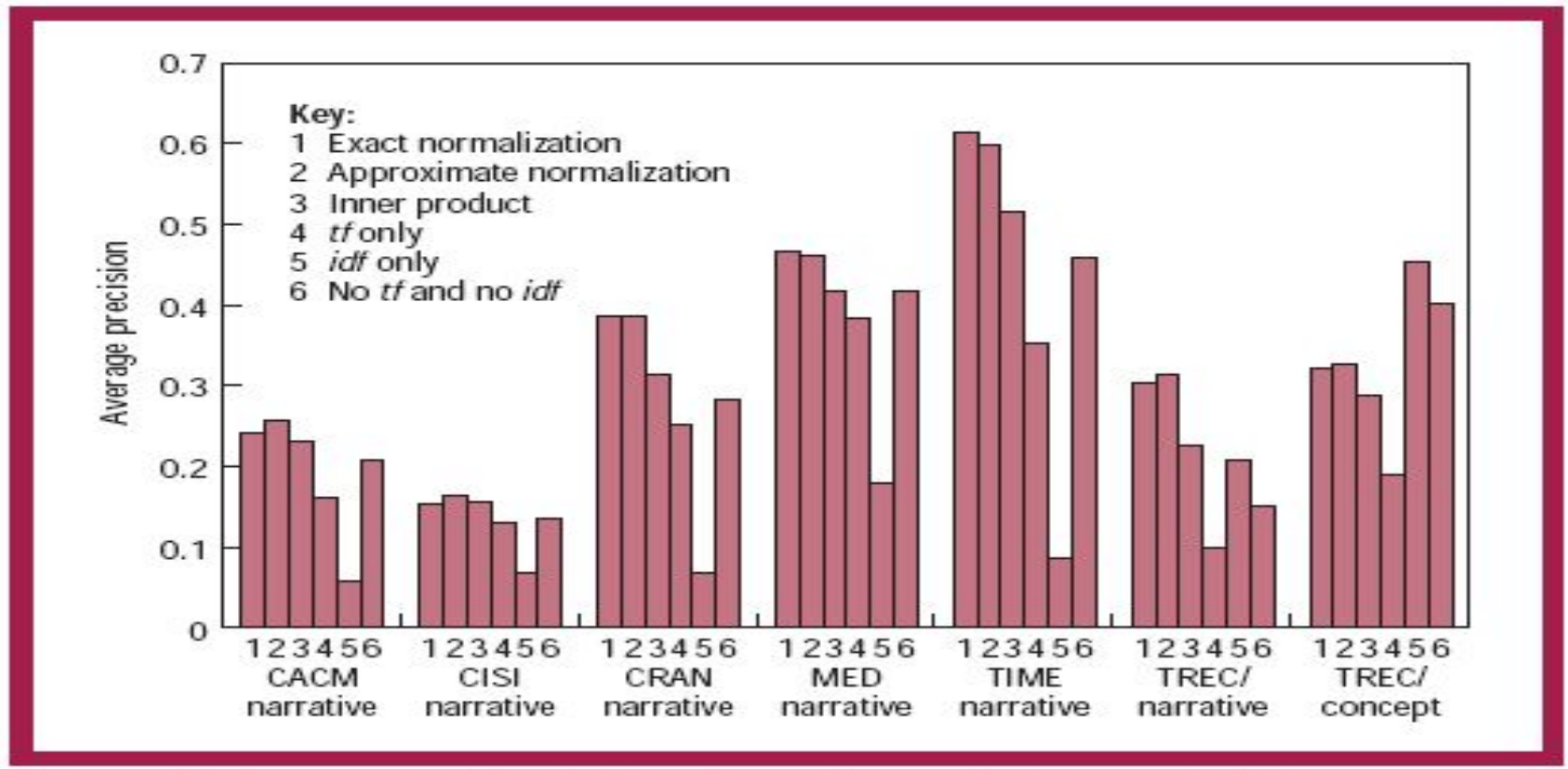
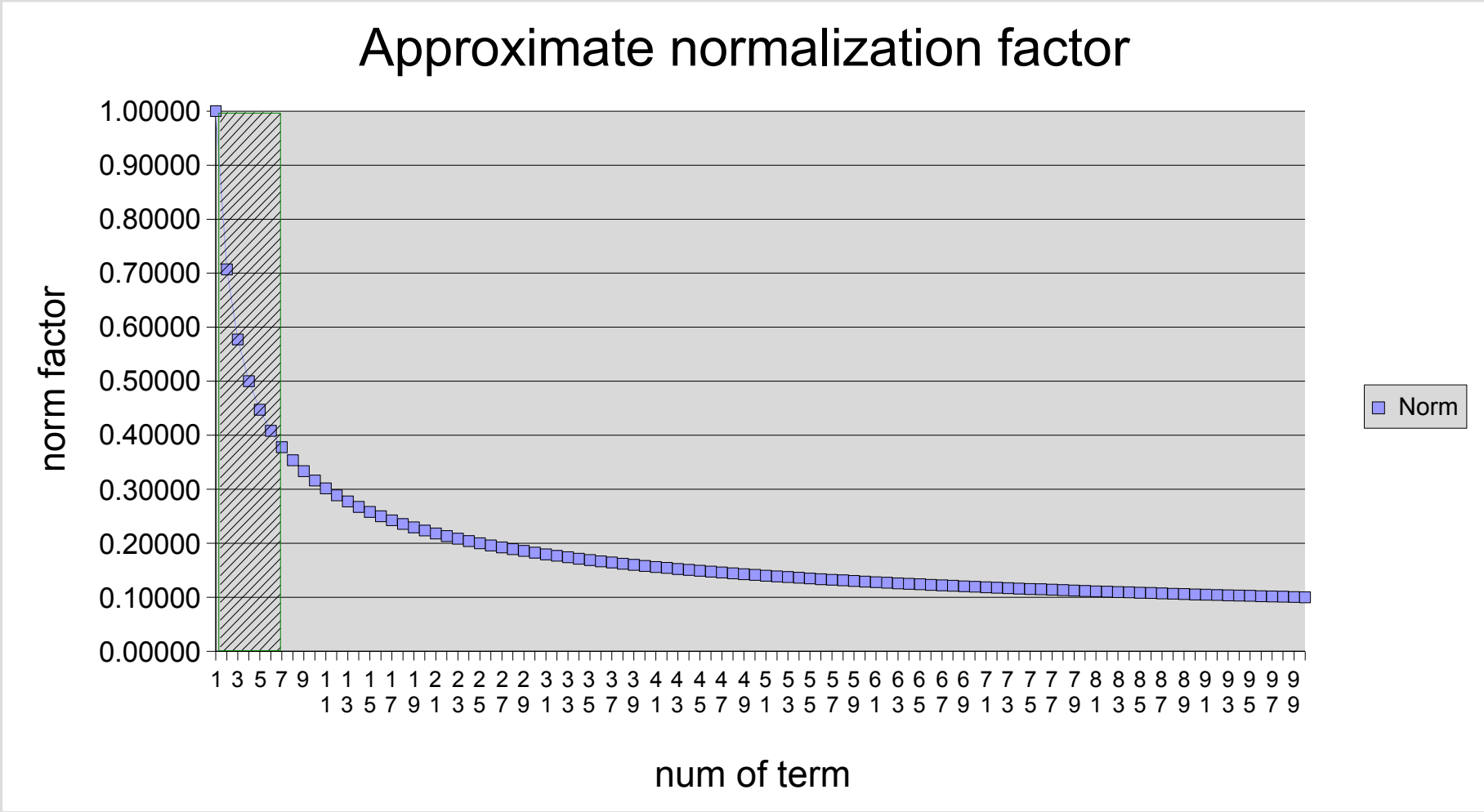


Figure 4. *Average precision for the six ranking methods on each of the seven sample document collections.*

이 함수의 장점과 단점

- 장점
  - Norm factor 를 미리 계산해서 저장할 수 있다 .
  - 저장용량 또한 문서의 갯수 만큼만 저장하므로 간편하고 용량이 적다 .
  - 실험 결과 처럼 정확도가 full norm 만큼 좋다 .
- 단점
  - 굉장히 짧은 문서에 대해서는 오류가 발생할 여지가 높다 .(예 . Rss feed 문서)

Approximate normalization 의 문제점



해결방안들

- 1. Nom of term 을 중복 텀을 제거하는 방향으로 보완 시킨다 . (ex {x,x,y,y,z,z} 의 num of term 은 3 이다 . tf 를 이용하면 쉽게 제거가 가능하리라 생각된다 .)
- 2. 문서의 평균 길이를 도출해 정규분포식을 이용 문서의 norm 을 책정한다 . 하지만 평균 이하의 길이의 문서에 대한 대책이 필요하다 . 왜냐면 norm 을 하는 목적이 짧은 문서의 score 를 보강하기 위함이기 때문이다 . (도리어 더 반감시키는 위험이 있다 .)
- 3. 그 밖의 추가적인 고민이 필요하다 .

참고 논문

- Document Ranking and the Vector-Space Model - *DIK L. LEE, Hong Kong University of Science and Technology HUEI CHUANG, Information Dimensions KENT SEAMONS, Transarc*