Search Model

- Vector Space Model

Contents

Definitions

- is called "term vector model".
- is an algebraic model for representing text documents as vectors of identifiers.
- Similarity of document is a distance between each terms.
- Each dimension corresponds to a separate term.
- If a term occurs in the document, its value in the vector is non-zero.
- used TF*IDF weighting often.
- is used information retrieval, information filtering, indexing …

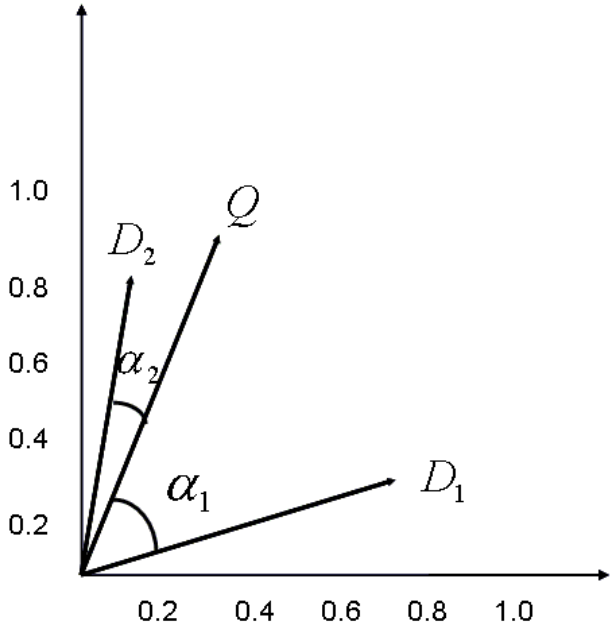
Applications

- In practice, it is easier to calculate the *cosine* of the angle between the vectors instead of the angle.

- A *cosine* value of zero means that the query and document vector were orthogonal and had no match
  → the query term did not exist in the document.

- Represents

$$\cos\theta = \frac{\mathbf{v_1} \cdot \mathbf{v_2}}{\|\mathbf{v_1}\| \, \|\mathbf{v_2}\|}$$



TF*IDF

TF (Term Frequency) : number of occurred term in a document
DF(Document Frequency) : number of documents where term occurs
IDF(Inverse Document Frequency) : Inverse DF

Weight = TF * IDF

→ TF가 크고, DF가 작을수록 가중치는 커진다.
→ 전체 문서에서 공통적으로 등장하는 단어들은 걸러지게 된다.
많은 문서에서 출현하는 term은 의미가 없다는 의미

문서 d가 있다면, Vector d는

$$\mathbf{V}_d = [w_{1,d}, w_{2,d}, \ldots, w_{N,d}]^T$$

에서

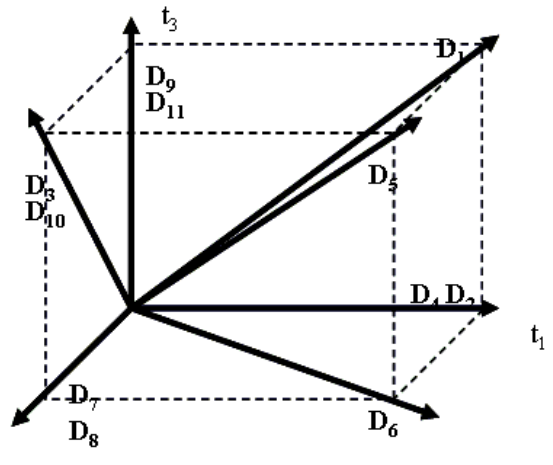
$$w_{t,d} = \mathrm{tf}_t \cdot log\frac{|D|}{|\{t \in d\}|}$$

TF

IDF

문서의 총 개수

Term을 포함한 문서의 개수



Similarity of documents

Assume Term Vector exist. (using TF)
- 주어진 term에 대해서 두 문서간의 거리를 구할 수 있다.
- 거리가 가까우면 가까울 수록 더 유사하다고 판단 가능

documents	"정보"의 TF	"검색"의 TF	비고
A	10	5	정보검색 관련 문서
B	8	9	정보검색 관련 문서
C	0	0	촛불집회 관련 문서

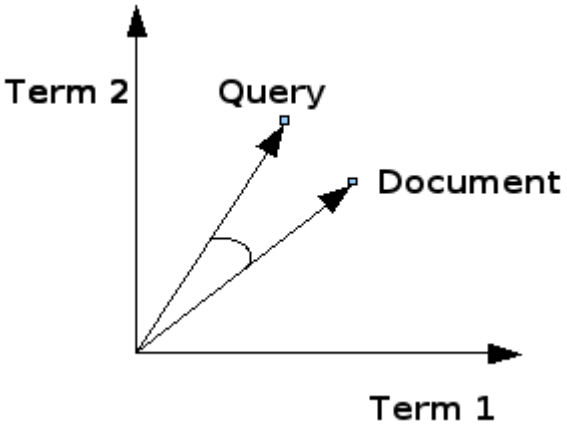
- d(A, B) = $2^{\wedge 2}$ + $4^{\wedge 2}$ = 20
- d(A, C) = $10^{\wedge 2}$ + $5^{\wedge 2}$ = 125
- d(B, C) = $8^{\wedge 2}$ + $9^{\wedge 2}$ = 145

이 결과를 보자면, A와 B는 유사하지만, A와 C, B와 C는 유사하지 않다고 볼 수 있음.

그러나, 유사하다 유사하지 않다라는 잣대는 없다. 단지 상대적으로 가까운 거에 비해서 유사하지 않다고 판단할 뿐이다.

Similarity of documents for Query

Assume Term Vector exist. (using TF)



Query와 Document간의 거리 계산한 후 가까울수록 더 연관성이 있는 문서라고 판단

Limitations

- 긴 문서들은 유사도 값이 작기 때문에, 제대로 표현되지 않는다.
  * term의 개수가 많아질수록 similarity가 더 낮은 값은 가지게 된다.
  * 예를 들면, 3번 출현한 문서와 100번 출현한 문서의 거리는 | 100 -3 | ^2 가 되어 값이 커지는 문제가 생겨 큰 문서일수록 저평가될 수 있다.

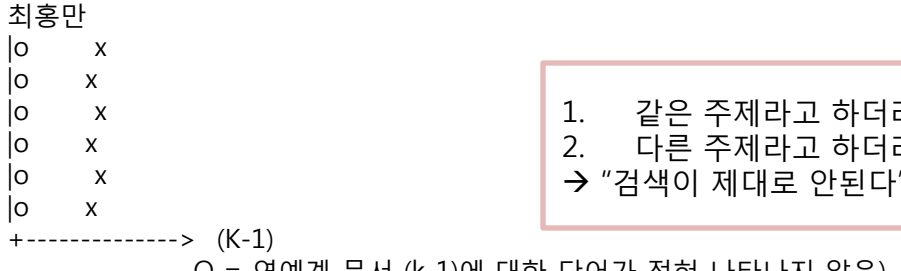
- 검색 키워드는 문서 내의 단어와 정확하게 일치해야 한다.

-- 벡터 공간 표현에서는 단어들이 나타나는 순서가 무시된다.

- 비슷한 의미를 갖고 있는 문서들일지라도 사용된 단어가 다르면 연관성을 갖지 않게 된다.

최홍만은 k-1에서 활약하는 격투선수이면서 특유의 쇼맨쉽으로 이런 저런 프로에 게스트 혹은 고정출현을 하고 있다. 그러므로 연예계 문서 중 몇 개에서는 '*최홍만*이 어느 정도의 빈도수를 가지게 될 것이다. k-1 으로 눈을 돌려서 생각해보면, 최홍만이 출전한 경기와 관련된 문서의 경우 빈도수가 높지만, 그렇지 않은 경기와 관련된 문서인 경우 낮은 빈도수를 가지게 될 것이다.

이것을 가정해서 **최홍만**의 빈도수를 y축으로 **k-1**의 빈도수를 x축으로 하고 연예방송과 k-1 두 종류의 문서를 그래프로 그려보면 아래와 같이 각각의 주제가 **y**축으로 길게 늘어선 형태를 가지게 된다.

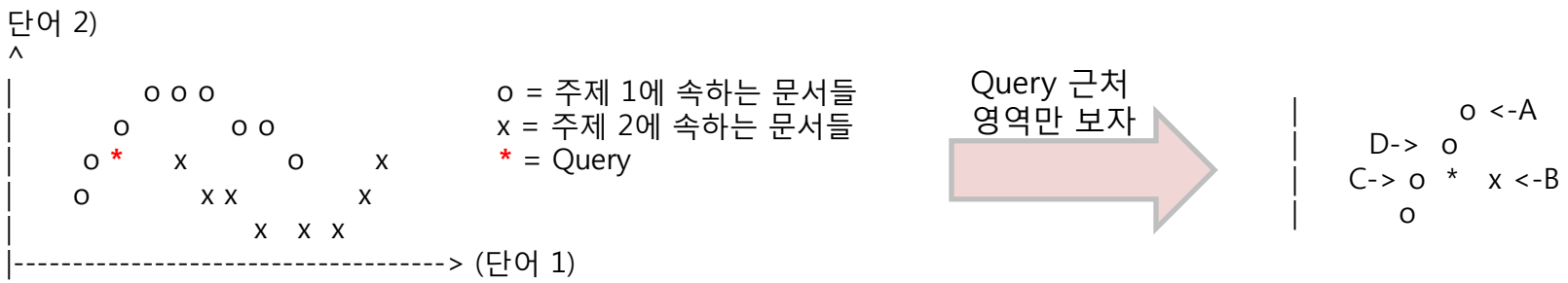


1. 같은 주제라고 하더라도 y축의 이쪽 끝과 저쪽 끝의 문서는 거리가 멀어진다.
2. 다른 주제라고 하더라고 y축상에 있는 문서끼리는 거리가 가깝다.

→ "검색이 제대로 안된다" 주제간의 차이를 반영하지 못하기 때문이다.

Markov random walk

- 주제의 성격이 비슷해서 원하는 문서를 가져오기가 애매모호해지는 경우가 문제임.



사용자가 *와 같은 Query를 줬을 때, 올바른 결과는 주제 1에 속한 문서들 중 쿼리와 가까이에 있는 문서가 상위에 올라오는 경우다. 그런데 그림을 보면 알겠지만 주제 2에 속하는 문서들 중 몇 개가 주제 1의 문서들보다 Query에 더 가까이 있기 때문에, 이들이 상위에 올라오는 문제가 발생한다. 이 경우 주제 2에 속하는 문서를 어떻게 제거할 것인가가 중요한 이슈.

문제 : distance(*, A) > distance(*, B) → VSM적용 → B가 A보다 높은 유사성
어떻게 해결??

1. distance(*, C) < distance(*, B)
2. distance(*, D) < distance(*, B)
3. distance(D, C) < distance(*, B)
4. distance(D, A) < distance(*, B)

각각의 Pointer로 이동하게 될 확률을 곱한다.

거리가 가까우면 움직일 확률이 ↑, 멀면 ↓
→ Markov random walk로 처리 ←

Query의 위치에서 시작을 해서 A 혹은 B로 도착할 때까지 매 턴마다 랜덤하게 임의의 Pointer로 이동을 하는 방식

예) 가. Query에서 A로 갈 수 있는 유망한 경로와 이에 따른 확률값
a) * → C → D → A : 확률값 P1 = P(*→C) x P(C→D) x P(D→A)
b) * → D → A : 확률값 P2 = P(*→D) x P(D→A)
c) 기타 : * → C → * → D → A and so on. : 확률값 P3
→ *에서 A까지 도달할 확률 P = P1 + P2 + P3
나. Query에서 B로 갈 수 있는 유망한 경로
a) * → B : 확률값 = P(*→B)

distance(C, B), distance(D, B), distance(A, B)가 너무 멀기 때문에 확률 자체가 ↓ → 다른 경로를 통한 확률은 * → B보다 낮아질 수 밖에 없게 된다.

∴ 문서간의 유사성을 판단할 때 **문서간의 직선거리**를 생각하는 대신에 **문서에 도착할 확률**로 본다.

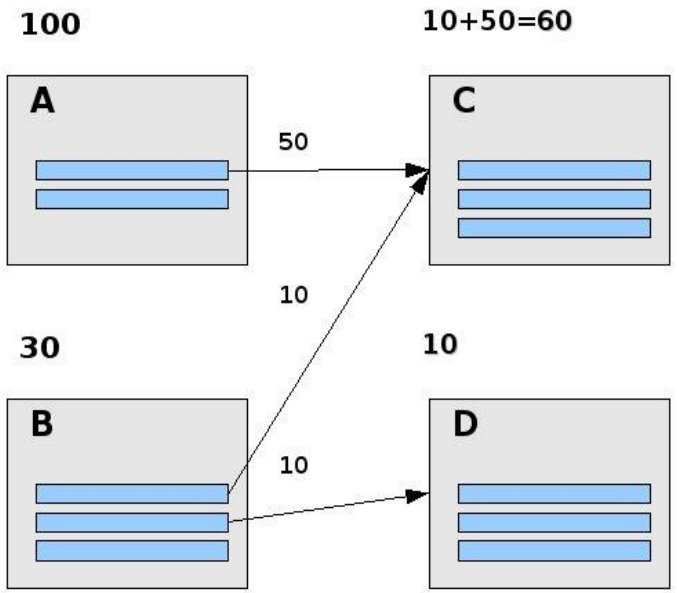
Markov random walk와 PageRank

- Page Rank is a algorithm of google for important of document
  Ex) A, B, C, D 4개의 웹문서가 있고, 이 중 B와 D가 A를 link하고 있을 때, A의 PageRank는
  Q(A) = Q(B) / N(B) + Q(D)/N(D) 가 된다.
  N은 해당 페이지가 가지고 있는 link의 개수를 의미
  → 이 계산을 모든 웹페이지에 대해서 반복해서 수행

PageRank는 웹페이지 혹은 링크와 같은 것들을 따로 Vector화 하지 않음
→ Markov random walk를 사용할 수 없음.
➔ 여러 개의 링크 중 어떤 링크로 이동할 확률이 가장 큰가를 어떻게 계산???

∴ Google은 **random click**이라는
아이디어를 사용한다.
(A라는 웹페이지에 N개의 링크가 있다고 가정했을 때, random click은 그 중 하나를 무작위로 click할 확률을 의미하며, 계산은 단순하게 1/N이다.)

➔ Q(A) = Q(B) * (1/N(B)) + Q(C)*0 + Q(D)*(1/N(D))



Markov chain

주사위를 던졌을 때, 1이 나올 확률은 1/6 ← 다른 조건의 영향을 받지 않는다.
서울에서 로또를 하든 부산에서 로또를 하든 확률은 동일

But….

오늘의 비가 오지 않을 확률은 어제의 날씨 상태가 어떠했는지에 따라서 달라질 수 있다.
어제 해가 쨍쨍했는지, 달무리가 생겼는지 기타 등등의 조건에 따라서 오늘 비가 올 확률이 변한다.
← 주어진 조건에 따라서 확률값이 어떻게 변하는지를 분석하는 게 **Markov chain 모형**
← 주어진 조건의 변화에 따라 변하는 확률값을 **천이확률(Transition Probability)**

Models based on and extending the vector space model

- Generalized vector space model

-(enhanced) Topic-based Vector space model

- Latent semantic analysis

- DSIR model

- Term Discrimination