LC029 정보검색

Information Retrieval

- Information Retrieval (IR) is **finding material** (usually documents) of an **unstructured** nature (usually text) that satisfies an **information need** from within **large collections** (usually stored on computers).

Information Retrieval Model

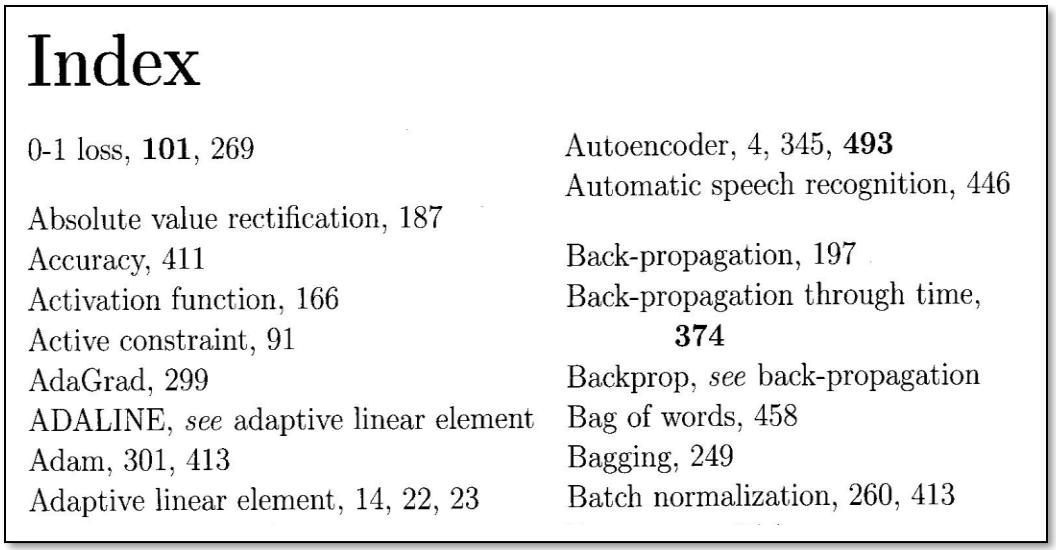
- Boolean Retrieval Model
- Vector Space Model

Boolean Retrieval Model

- Boolean Retrieval
- The Term Vocabulary and Postings Lists
- Dictionaries and Tolerant Retrieval
- Index Construction
- Index Compression

Boolean Retrieval Model

- Boolean Retrieval 주기적 업데이트 필요



Index

0-1 loss, **101**, 269

Absolute value rectification, 187
Accuracy, 411
Activation function, 166
Active constraint, 91
AdaGrad, 299
ADALINE, *see* adaptive linear element
Adam, 301, 413
Adaptive linear element, 14, 22, 23
Autoencoder, 4, 345, **493**
Automatic speech recognition, 446

Back-propagation, 197
Back-propagation through time, **374**
Backprop, *see* back-propagation
Bag of words, 458
Bagging, 249
Batch normalization, 260, 413

- The Term Vocabulary and Postings Lists
- Dictionaries and Tolerant Retrieval 키워드가 완벽하지 않아도 검색 가능
- Index Construction
- Index Compression 인덱스 압축

Boolean Retrieval Model

- Boolean Retrieval
- The Term Vocabulary and Postings Lists
  - Tokenization
  - Stop Words 불용어
  - Normalization
- Dictionaries and Tolerant Retrieval
- Index Construction
- Index Compression

Tokenization

Documents to be indexed

Friends, Romans, countrymen.

↳ 문장들임

Tokenizer

토큰화

Token stream

Friends | Romans | Countrymen

Linguistic modules

언어에 따라 다른 처리

예 : 한국어는 조사 떼기

Modified tokens

friend | roman | countryman

Indexer

Inverted index

friend	2 → 4 →
roman	1 → 2 →
countryman	13 → 16

Stop Words

- Most frequent words considered insignificant
- Significantly reduces the size of index

너무 자주 나와서 안 중요함

인덱스 사이즈 줄임

Normalization 정규화

query

USA

document

In **U.S.A.** bla bla …

사용자 입장에서 USA 와 U.S.A. 는 같다. 그래서 이 두 단어의 인덱스를 합치는 게 정규화

car

Best **SUV** in the world bla bla …

Boolean Retrieval Model

- Boolean Retrieval
- The Term Vocabulary and Postings Lists
- Dictionaries and Tolerant Retrieval
  - Dictionary Structures
  - Wild-card queries
  - Spelling correction
- Index Construction
- Index Compression

Dictionary Structures

사전 구조

빠른 검색을 위해 보통 메인 메모리에 있음

- How do we store a dictionary in memory efficiently, so that we could quickly look up elements at query processing time?
  - Hash table
  - Tree

Wild-card queries

와일드카드를 쓰는 상황들

- When you are uncertain of the spelling

  Sydney vs. Sidney ? → **S*dney**

- When you are aware that the term has variants of spelling

  color vs. colour → **colo*r**

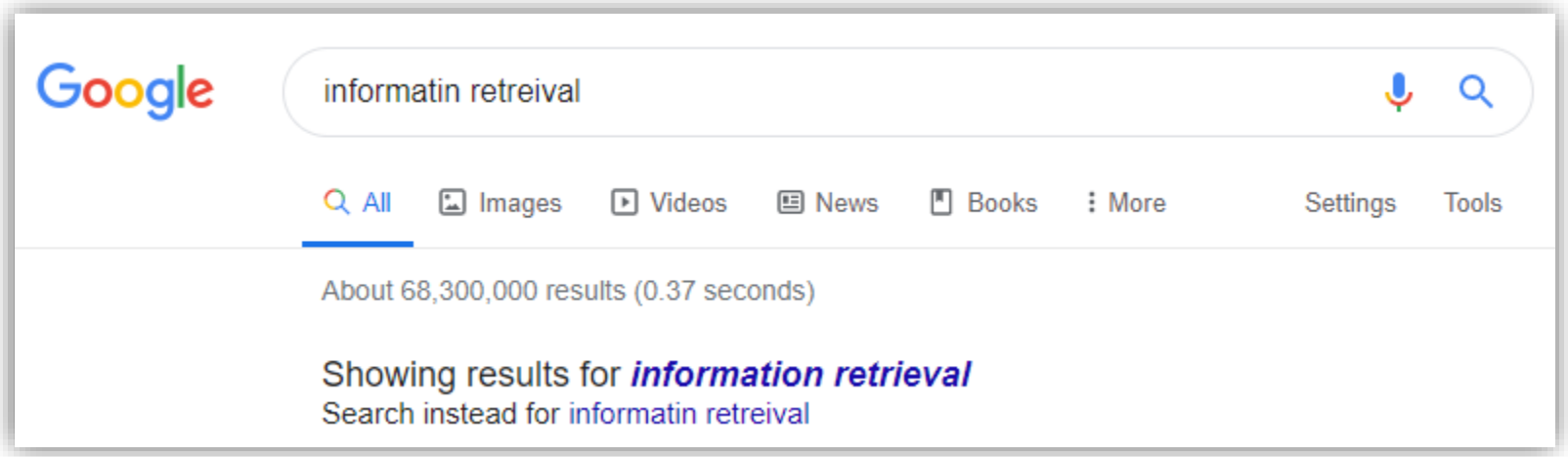
- When you want documents containing variants of a term

  computation, computing, computer, computational, …
  → **comput***

Spelling correction

오타 고쳐주기

- Correcting documents being indexed ← 문서 고치기
- Correcting user queries to retrieve “right” answers ← 사용자의 검색어 고쳐주기



Boolean Retrieval Model

- Boolean Retrieval
- The Term Vocabulary and Postings Lists
- Dictionaries and Tolerant Retrieval
- Index Construction 인덱스 만들기
  - Sort-based Index Construction 정렬 → 한계가 있음
  - Scalable Index Construction
    - BSBI: Blocked Sort-Based Indexing
    - SPIMI: Single-Pass In-Memory Indexing
    - Distributed Indexing
- Index Compression

Index Construction

symbol	statistic	value
N	documents	800,000
$\mathbf{L_{ave}}$	average number of tokens / doc	200
M	number of terms	400,000
	average number of bytes / token (including spaces/punctuation)	6
	average number of bytes / token (without spaces/punctuation)	4.5
	average number of bytes / term	7.5
	number of non-positional postings	100,000,000

Boolean Retrieval Model

- Boolean Retrieval
- The Term Vocabulary and Postings Lists
- Dictionaries and Tolerant Retrieval
- Index Construction
- Index Compression
  - Dictionary Compression
  - Postings Compression
  - Huffman Code

Vector Space Model

- Why Vector Space Model?
  - Problems with Boolean Search
    - Documents either match or don't ← 주어진 키워드가 포함된 모든 문서 반환.
    - Ranking the results is impossible ← 사용자가 필요할 것 같은 문서 순서대로 랭크를 매겨야 한다.
- How can we rank the documents in the collection with respect to a query?
  - Assign a *score*, say in [0, 1], to each document
  - This score measures how well document and query match

Vector Space Model

- Documents as Vectors
  - Very high-dimensional

< 5.25 , 1.21 , 8.59 , 0 , 2.85 , 1.51 , 1.37 >

Antony	Brutus	Caeser	Carpurnia	Cleopatra	Mercy	Worser

- Query as a Vector

< 1 , 0 , 1 , 0 , 0 , 0 , 0 >

- Compute similarity to the query and then rank documents according to their **similarity** score

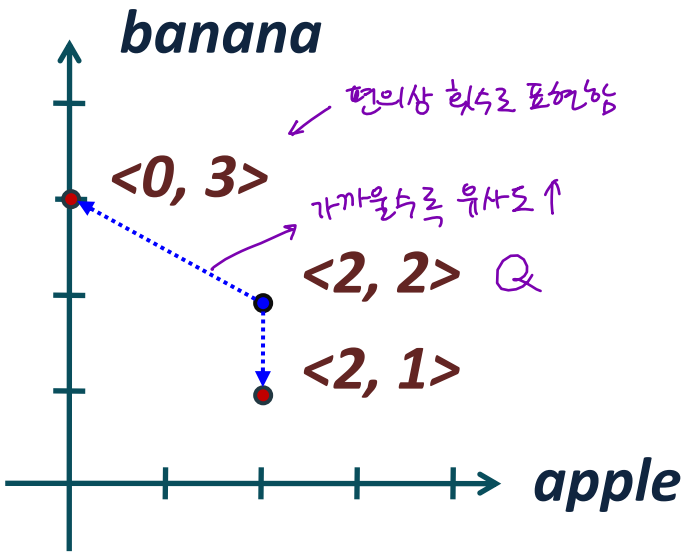
Vector Space Model

- How to get the *weight* of each term in a document?
  - Term Frequency
    - Term frequency is used to compute query-document match score
  - Document Frequency
    - Rare terms are more informative than frequent terms
  - *tf-idf* Weight
    - Best known weight scheme in information retrieval
    - Increases with the number of occurrences within a document
    - Increases with the rarity of the term in the collection

Vector Space Model

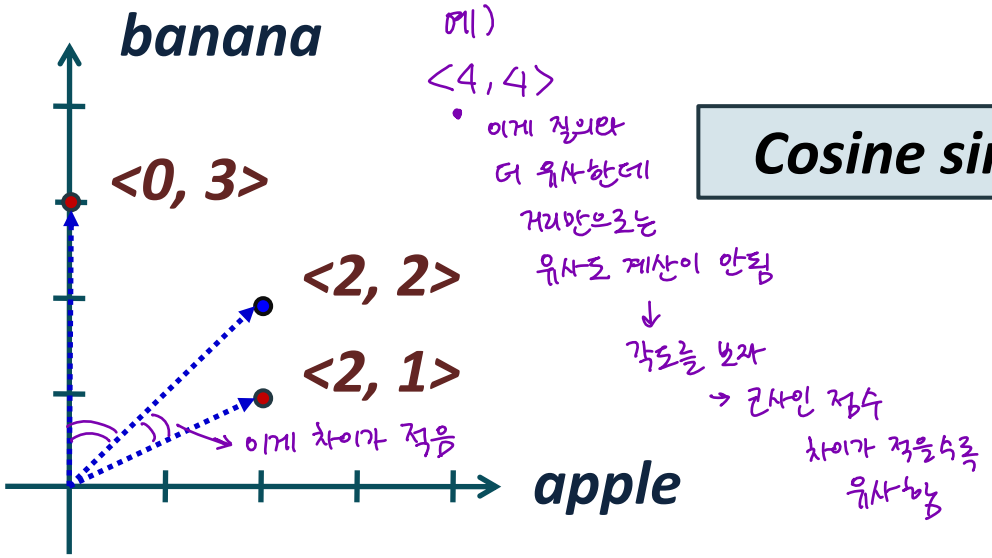
- How to get the **similarity** score

각각의 벡터는 그 길이만큼의 차원에 하나의 점으로 표현됨



Vector Space Model

- How to get the **similarity** score



Cosine similarity

벡터 유사도 계산하는 식

$$\cos(\vec{q},\vec{d}) = \frac{\vec{q}\cdot\vec{d}}{|\vec{q}||\vec{d}|} = \frac{\vec{q}}{|\vec{q}|}\cdot\frac{\vec{d}}{|\vec{d}|} = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2}\sqrt{\sum_{i=1}^{|V|} d_i^2}}$$

- The following two notions are equivalent
  - Rank documents in **increasing** order of the angle between query and document
  - Rank documents in **decreasing** order of ***cos*(*q*, *d*)**

Vector Space Model

- Scoring, Term Weighting and the Vector Space Model
- Computing Scores in a Complete Search System
  - Efficient Cosine Ranking
  - Inexact top *K* document retrieval

↳ 정확하게 검색하면 너무 오래걸리니까 대충 찾기

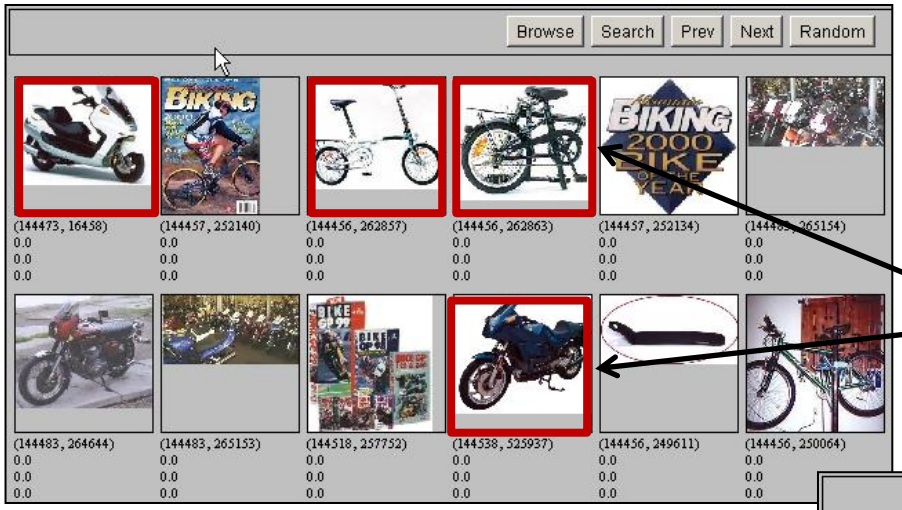
More on Information Retrieval

- Evaluation in Information Retrieval 평가
- Relevance Feedback and Query Expansion 피드백

검색어와 결과가 얼마나 비슷한지
사용자가 피드백 제공

↓
쿼리가 수정됨

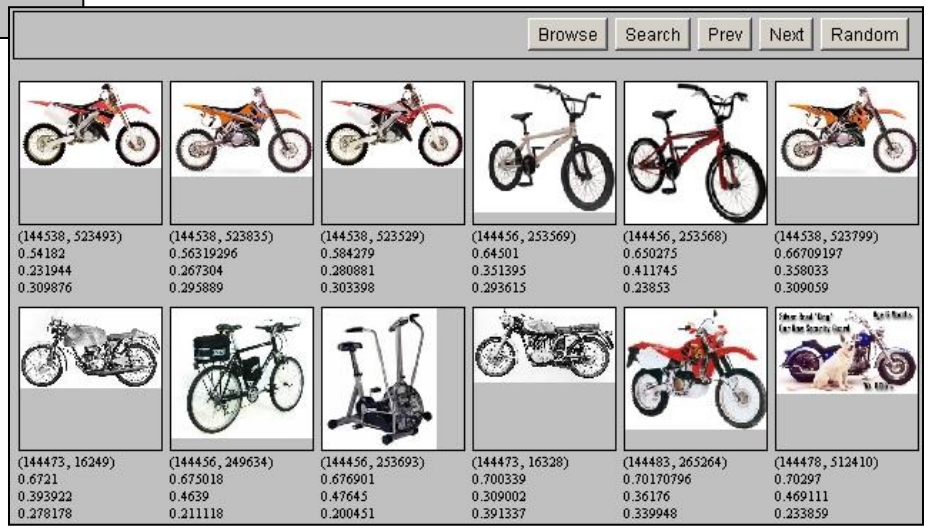
Relevance Feedback : Example 1



← initial search results

user feedback

improved search results →



Relevance Feedback : Example 2

Query: New space satellite applications

retrieved documents

relevance feedback

1. 0.539, 08/13/91, NASA Hasn't Scrapped Imaging Spectrometer
2. 0.533, 07/09/91, NASA Scratches Environment Gear From Satellite Plan
3. 0.528, 04/04/90, Science Panel Backs NASA Satellite Plan, But Urges Launches of Smaller Probes
4. 0.526, 09/09/91, A NASA Satellite Project Accomplishes Incredible Feat: Staying Within Budget

Improved results

1. 0.513, 07/09/91, NASA Scratches Environment Gear From Satellite Plan
2. 0.500, 08/13/91, NASA Hasn't Scrapped Imaging Spectrometer
3. 0.493, 08/07/89, When the Pentagon Launches a Secret Satellite, Space Sleuths Do Some Spy Work of Their Own
4. 0.493, 07/31/89, NASA Uses 'Warm' Superconductors For Fast Circuit

Relevance Feedback : Example 2

- Expanded query after relevance feedback

2.074	**new**	**15.12**	**space**
30.82	**satellite**	**5.660**	**application**
5.991	nasa	**5.196**	eos
4.196	launch	**3.972**	aster
3.516	instrument	**3.446**	arianespace
3.004	bundespost	**2.806**	ss
2.790	rocket	**2.053**	scientist
2.003	broadcast	**1.172**	earth
0.836	oil	**0.646**	measure

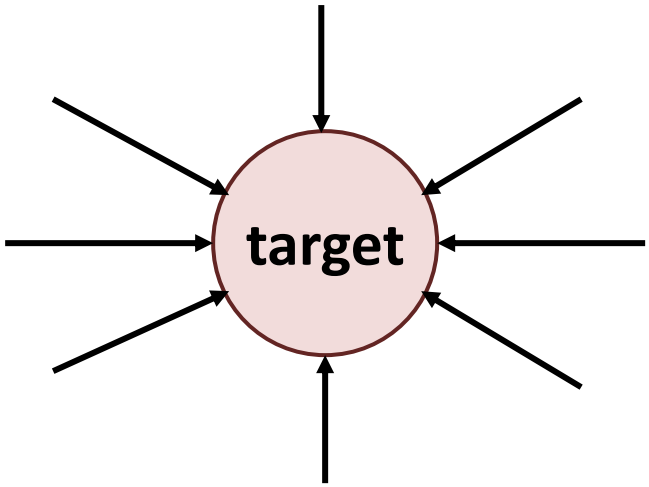
Query: New space satellite applications

More on Information Retrieval

- Web Search Basics
- Web Crawling and Indexes 검색 대상 확보
- Link Analysis

Link Analysis

- Query Processing
  - Retrieve all pages satisfying the query
  - Order retrieved documents by their *link popularity*



몇마나 많은 문서가 이 문서를 링크하는가
→ 이 문서가 중요하다
↓
악용 가능

- Problem : Link Spam
  - Set up multiple web pages pointing to a target web page