Boolean Model

Hongning Wang
CS@UVa
Abstraction of search engine architecture

Indexed corpus
Crawler
Doc Analyzer
Doc Representation
Indexer

Indexed corpus

Crawler
Doc Analyzer
Doc Representation
Indexer

Ranking procedure

Feedback
Query Rep

Evaluation
User
(results)

Index

Ranker

Query Rep

Evaluation
User
(results)
Search with Boolean query

• Boolean query
  – E.g., “obama” AND “healthcare” NOT “news”

• Procedures
  – Lookup query term in the dictionary
  – Retrieve the posting lists
  – Operation
    • AND: intersect the posting lists
    • OR: union the posting list
    • NOT: diff the posting list
Search with Boolean query

• Example: AND operation

Time complexity: $O(|L_1| + |L_2|)$

*Trick for speed-up:* when performing multi-way join, starts from lowest frequency term to highest frequency ones
Deficiency of Boolean model

• The query is unlikely precise
  – “Over-constrained” query (terms are too specific): no relevant documents found
  – “Under-constrained” query (terms are too general): over delivery
  – It is hard to find the right position between these two extremes (hard for users to specify constraints)

• Even if it is accurate
  – Not all users would like to use such queries
  – All relevant documents are not equally relevant
    • No one would go through all the matched results

• Relevance is a matter of degree!
Document Selection vs. Ranking

True Rel(q)

Doc Selection
\[ f(d,q) = ? \]

Doc Ranking
\[ \text{rel}(d,q) = ? \]

\[
\begin{align*}
0.98 d_1 & + \\
0.95 d_2 & + \\
0.83 d_3 & - \\
0.80 d_4 & + \\
0.76 d_5 & - \\
0.56 d_6 & - \\
0.34 d_7 & - \\
0.21 d_8 & + \\
0.21 d_9 & - \\
\end{align*}
\]
Ranking is often preferred

- Relevance is a matter of degree
  - Easier for users to find appropriate queries
- A user can stop browsing anywhere, so the boundary is controlled by the user
  - Users prefer coverage would view more items
  - Users prefer precision would view only a few
- Theoretical justification: Probability Ranking Principle
Retrieval procedure in modern IR

• Boolean model provides all the ranking candidates
  – Locate documents satisfying Boolean condition
    • E.g., “obama healthcare” -> “obama” OR “healthcare”

• Rank candidates by relevance
  – Important: the notation of relevance

• Efficiency consideration
  – Top-k retrieval (Google)
Notion of relevance

Relevance

$\Delta(\text{Rep}(q), \text{Rep}(d))$

Similarity

$P(r=1|q,d)$  \( r \in \{0,1\} \)

Probability of Relevance

$P(d \rightarrow q) \text{ or } P(q \rightarrow d)$

Probabilistic inference

Different rep & similarity

Regression Model

(Fox 83)

Generative Model

Doc generation

Classical prob. Model

(Robertson & Sparck Jones, 76)

Query generation

LM approach

(Ponte & Croft, 98)

Prob. concept space model

(Wong & Yao, 95)

Different inference system

Inference network model

(Turtle & Croft, 91)

Vector space model

(Salton et al., 75)

Prob. distr. model

(Wong & Yao, 89)

Today’s lecture
Intuitive understanding of relevance

• Fill in magic numbers to describe the relation between documents and words

<table>
<thead>
<tr>
<th></th>
<th>information</th>
<th>retrieval</th>
<th>retrieved</th>
<th>is</th>
<th>helpful</th>
<th>for</th>
<th>you</th>
<th>everyone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Doc1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Doc2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

E.g., 0/1 for Boolean models, probabilities for probabilistic models
Some notations

- Vocabulary $V = \{w_1, w_2, \ldots, w_N\}$ of language
- Query $q = t_1, \ldots, t_m$, where $t_i \in V$
- Document $d_i = t_{i1}, \ldots, t_{in}$, where $t_{ij} \in V$
- Collection $C = \{d_1, \ldots, d_k\}$
- $\text{Rel}(q,d)$: relevance of doc $d$ to query $q$
- $\text{Rep}(d)$: representation of document $d$
- $\text{Rep}(q)$: representation of query $q$
Vector Space Model

Hongning Wang
CS@UVa
Relevance = Similarity

• Assumptions
  – Query and documents are represented in the same form
    • A query can be regarded as a “document”
  – Relevance(d,q) ∝ similarity(d,q)

• R(q) = {d ∈ C | rel(d,q) > θ}, rel(q,d) = Δ(Rep(q), Rep(d))

• Key issues
  – How to represent query/document?
  – How to define the similarity measure Δ(x,y)?
Vector space model

• Represent both doc and query by **concept vectors**
  – Each concept defines one dimension
  – \( K \) concepts define a high-dimensional space
  – Element of vector corresponds to concept weight
    • E.g., \( d=(x_1,\ldots,x_k) \), \( x_i \) is “importance” of concept \( i \)

• Measure relevance
  – Distance between the query vector and document vector in this concept space
VS Model: an illustration

• Which document is closer to the query?
What the VS model doesn’t say

- How to define/select the “basic concept”
  - Concepts are assumed to be orthogonal
- How to assign weights
  - Weight in query indicates importance of the concept
  - Weight in doc indicates how well the concept characterizes the doc
- How to define the similarity/distance measure
What is a good “basic concept”?

• Orthogonal
  – Linearly independent basis vectors
    • “Non-overlapping” in meaning
    • No ambiguity

• Weights can be assigned automatically and accurately

• Existing solutions
  – Terms or N-grams, i.e., bag-of-words
  – Topics, i.e., topic model

We will come back to this later
How to assign weights?

• **Important!**

• **Why?**
  – Query side: not all terms are equally important
  – Doc side: some terms carry more information about the content

• **How?**
  – Two basic **heuristics**
    • TF (Term Frequency) = Within-doc-frequency
    • IDF (Inverse Document Frequency)
TF weighting

• Idea: a term is more important if it occurs more frequently in a document

• TF Formulas
  – Let $f(t, d)$ be the frequency count of term $t$ in doc $d$
  – Raw TF: $tf(t, d) = f(t, d)$
TF normalization

• Two views of document length
  – A doc is long because it is verbose
  – A doc is long because it has more content
• Raw TF is inaccurate
  – Document length variation
  – “Repeated occurrences” are less informative than the “first occurrence”
  – Relevance does not increase proportionally with number of term occurrence
• Generally penalize long doc, but avoid over-penalizing
  – Pivoted length normalization
TF normalization

• Sublinear TF scaling

\[ tf(t, d) = \begin{cases} 
  1 + \log f(t, d), & \text{if } f(t, d) > 0 \\
  0, & \text{otherwise}
\end{cases} \]
TF normalization

- **Maximum TF scaling**

\[ tf(t, d) = \alpha + (1 - \alpha) \frac{f(t, d)}{\max_t f(t, d)} \]

- Normalize by the most frequent word in this doc
Document frequency

• Idea: a term is more discriminative if it occurs only in fewer documents
IDF weighting

• Solution
  – Assign higher weights to the rare terms
  – Formula
    • \( IDF(t) = 1 + \log\left(\frac{N}{df(t)}\right) \)
  – A corpus-specific property
    • Independent of a single document
Why document frequency

• How about total term frequency?
  \[ ttf(t) = \sum_d f(t, d) \]

Table 1. Example total term frequency v.s. document frequency in Reuters-RCV1 collection.

<table>
<thead>
<tr>
<th>Word</th>
<th>ttf</th>
<th>df</th>
</tr>
</thead>
<tbody>
<tr>
<td>try</td>
<td>10422</td>
<td>8760</td>
</tr>
<tr>
<td>insurance</td>
<td>10440</td>
<td>3997</td>
</tr>
</tbody>
</table>

– Cannot recognize words frequently occurring in a subset of documents
TF-IDF weighting

• Combining TF and IDF
  – Common in doc $\rightarrow$ high tf $\rightarrow$ high weight
  – Rare in collection $\rightarrow$ high idf $\rightarrow$ high weight
  – $w(t, d) = TF(t, d) \times IDF(t)$

• Most well-known document representation schema in IR! (G Salton et al. 1983)

“Salton was perhaps the leading computer scientist working in the field of information retrieval during his time.” - wikipedia

Gerard Salton Award – highest achievement award in IR
How to define a good similarity measure?

• Euclidean distance?
How to define a good similarity measure?

• Euclidean distance

\[ dist(q, d) = \sqrt{\sum_{t \in V} [tf(t, q)idf(t) - tf(t, d)idf(t)]^2} \]

– Longer documents will be penalized by the extra words
– We care more about how these two vectors are overlapped
From distance to angle

• Angle: how vectors are overlapped
  – Cosine similarity – projection of one vector onto another

The choice of angle
The choice of Euclidean distance

TF-IDF space

Finance
Sports
Query

D₂
D₁
Cosine similarity

- Angle between two vectors
  \[ \text{cosine}(V_q, V_d) = \frac{V_q \times V_d}{|V_q|_2 \times |V_d|_2} = \frac{V_q}{|V_q|_2} \times \frac{V_d}{|V_d|_2} \]
  - Document length normalized

TF-IDF vector

Unit vector

TF-IDF space

Finance

D_2

D_1

Query

Sports
Fast computation of cosine in retrieval

- \( \text{cosine}(V_q, V_d) = V_q \times \frac{V_d}{|V_d|_2} \)
  
  - \( |V_q|_2 \) would be the same for all candidate docs
  
  - Normalization of \( V_d \) can be done in index time
  
  - Only count \( t \in q \cap d \)
  
  - Score accumulator for each query term when intersecting postings from inverted index
Fast computation of cosine in retrieval

• Maintain a score accumulator for each doc when scanning the postings

Query = “info security”

\[ S(d,q) = g(t_1) + \ldots + g(t_n) \] [sum of TF of matched terms]

Info: (d1, 3), (d2, 4), (d3, 1), (d4, 5)

Security: (d2, 3), (d4, 1), (d5, 3)

Accumulators:

<table>
<thead>
<tr>
<th>Accumulators:</th>
<th>d1</th>
<th>d2</th>
<th>d3</th>
<th>d4</th>
<th>d5</th>
</tr>
</thead>
<tbody>
<tr>
<td>(d1,3) =&gt;</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(d2,4) =&gt;</td>
<td>3</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(d3,1) =&gt;</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(d4,5) =&gt;</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>(d2,3) =&gt;</td>
<td>3</td>
<td>7</td>
<td>1</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>(d4,1) =&gt;</td>
<td>3</td>
<td>7</td>
<td>1</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>(d5,3) =&gt;</td>
<td>3</td>
<td>7</td>
<td>1</td>
<td>6</td>
<td>3</td>
</tr>
</tbody>
</table>

Can be easily applied to TF-IDF weighting!

Keep only the most promising accumulators for top K retrieval
Advantages of VS Model

• Empirically effective! (Top TREC performance)
• Intuitive
• Easy to implement
• Well-studied/ Mostly evaluated
• The Smart system
  – Developed at Cornell: 1960-1999
  – Still widely used
• **Warning: Many variants of TF-IDF!**
Disadvantages of VS Model

• Assume term independence
• Assume query and document to be the same
• Lack of “predictive adequacy”
  – Arbitrary term weighting
  – Arbitrary similarity measure
• Lots of parameter tuning!
What you should know

• Document ranking v.s. selection
• Basic idea of vector space model
• Two important heuristics in VS model
  – TF
  – IDF
• Similarity measure for VS model