Information Retrieval and Web Search

Introduction to IR models and methods

Rada Mihalcea

(Some of the slides in this slide set come from IR courses taught at UT Austin and Stanford)
Information Retrieval

• The indexing and retrieval of textual documents.
• Searching for pages on the World Wide Web is the most recent and perhaps most widely used IR application.
• Concerned firstly with retrieving relevant documents to a query.
• Concerned secondly with retrieving from large sets of documents efficiently.
Typical IR Task

• Given:
  – A corpus of textual natural-language documents.
  – A user query in the form of a textual string.

• Find:
  – A ranked set of documents that are relevant to the query.
Typical IR System Architecture
Web Search System

Web

Spider

Document corpus

Query String

IR System

Ranked Documents

1. Page1
2. Page2
3. Page3
...
Relevance

• Relevance is a subjective judgment and may include:
  – Being on the proper subject.
  – Being timely (recent information).
  – Being authoritative (from a trusted source).
  – Satisfying the goals of the user and his/her intended use of the information (*information need*).

• Main **relevance criterion**: an IR system should fulfill the user’s information need.
Basic IR Approach: Keyword Search

- Simplest notion of relevance is that the query string appears verbatim in the document.
- Slightly less strict notion is that the words in the query appear frequently in the document, in any order (*bag of words*).
Problems with Keywords

• May not retrieve relevant documents that include synonymous terms.
  – “restaurant” vs. “café”
  – “PRC” vs. “China”

• May retrieve irrelevant documents that include ambiguous terms.
  – “bat” (baseball vs. mammal)
  – “Apple” (company vs. fruit)
  – “bit” (unit of data vs. act of eating)

• In this course:
  – We will cover the basics of keyword-based IR
  – Also address more complex techniques for “intelligent” IR
Techniques for Intelligent IR

- Take into account the *meaning* of the words used
- Take into account the *order* of words in the query
- Adapt to the user based on automatic or semi-automatic *feedback*
- *Extend* search with related terms
- Perform automatic *spell checking / diacritics restoration*
- Take into account the *authority* of the source.
IR System Architecture

User Interface

User Need

User Interface

Text Operations

query Operation

Indexing

Index

Retrieved Docs

Retrieved Docs

Inverted file

Text Database

Database Manager

Logical View

User Feedback

User Interface

Ranked Docs

Searching

Ranking

Text

Query

User Interface

User Interface

User Interface
IR System Components

- **Text Operations** forms index words (tokens).
  - Tokenization
  - Stopword removal
  - Stemming

- **Indexing** constructs an *inverted index* of word to document pointers.
  - Mapping from keywords to document ids

```
Doc 1
I did enact Julius Caesar I was killed i' the Capitol;
Brutus killed me.

Doc 2
So let it be with Caesar. The noble Brutus hath told you
Caesar was ambitious
```

<table>
<thead>
<tr>
<th>Term</th>
<th>Doc #</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>did</td>
<td>1</td>
</tr>
<tr>
<td>enact</td>
<td>1</td>
</tr>
<tr>
<td>julius</td>
<td>1</td>
</tr>
<tr>
<td>caesar</td>
<td>1</td>
</tr>
<tr>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>was</td>
<td>1</td>
</tr>
<tr>
<td>killed</td>
<td>1</td>
</tr>
<tr>
<td>i'</td>
<td>1</td>
</tr>
<tr>
<td>the</td>
<td>1</td>
</tr>
<tr>
<td>capitol</td>
<td>1</td>
</tr>
<tr>
<td>brutus</td>
<td>1</td>
</tr>
<tr>
<td>killed</td>
<td>1</td>
</tr>
<tr>
<td>me</td>
<td>1</td>
</tr>
<tr>
<td>so</td>
<td>2</td>
</tr>
<tr>
<td>let</td>
<td>2</td>
</tr>
<tr>
<td>it</td>
<td>2</td>
</tr>
<tr>
<td>be</td>
<td>2</td>
</tr>
<tr>
<td>with</td>
<td>2</td>
</tr>
<tr>
<td>caesar</td>
<td>2</td>
</tr>
<tr>
<td>the</td>
<td>2</td>
</tr>
<tr>
<td>noble</td>
<td>2</td>
</tr>
<tr>
<td>brutus</td>
<td>2</td>
</tr>
<tr>
<td>hath</td>
<td>2</td>
</tr>
<tr>
<td>told</td>
<td>2</td>
</tr>
<tr>
<td>you</td>
<td>2</td>
</tr>
<tr>
<td>caesar</td>
<td>2</td>
</tr>
<tr>
<td>was</td>
<td>2</td>
</tr>
<tr>
<td>ambitious</td>
<td>2</td>
</tr>
</tbody>
</table>
IR System Components

• Searching retrieves documents that contain a given query token from the inverted index.

• Ranking scores all retrieved documents according to a relevance metric.

• User Interface manages interaction with the user:
  – Query input and document output
  – Relevance feedback
  – Visualization of results

• Query Operations transform the query to improve retrieval:
  – Query expansion using a thesaurus
  – Query transformation using relevance feedback
IR Models

User Task
- Retrieval: Adhoc Filtering
- Browsing

Structured Models
- Non-Overlapping Lists
- Proximal Nodes

Classic Models
- boolean vector probabilistic

Set Theoretic
- Fuzzy Extended Boolean

Algebraic
- Generalized Vector Lat. Semantic Index
- Neural Networks

Probabilistic
- Inference Network
- Belief Network

Browsing
- Flat Structure
- Guided Hypertext
Classic IR Models - Basic Concepts

• Each document is represented by a set of representative keywords or index terms

• An index term is a document word that may be searched for

• Index terms may be selected to be only nouns, since nouns have meaning by themselves
  – Should reduce the size of the index
  – ... But it requires the identification of nouns \(\Rightarrow\) Part of Speech tagger

• However, search engines assume that all words are index terms (full text representation)
Classic IR Models - Basic Concepts

• Not all terms are equally useful for representing the document contents: less frequent terms allow for the identification of a narrower set of documents

• The importance of the index terms is represented by weights associated to them

• Let
  – \( k_i \) be an index term
  – \( d_j \) be a document
  – \( w_{ij} \) is a weight associated with \((k_i,d_j)\)

• The weight \( w_{ij} \) quantifies the importance of the index term to describe the document contents
Boolean Model

• Simple model based on set theory

• Queries specified as boolean expressions
  – precise semantics
  – neat formalism
  – \( q = k_a \wedge (k_b \lor \neg k_c) \)

• Terms are either present or absent. Thus, \( w_{ij} \in \{0,1\} \)

• Can always be transformed into DNF
  – \( q_{dnf} = (1,1,1) \lor (1,1,0) \lor (1,0,0) \)
Boolean Model

\[ q = k_a \land (k_b \lor \neg k_c) \]
Drawbacks of the Boolean Model

- Retrieval based on binary decision criteria with no notion of partial matching
- No ranking of the documents is provided (absence of a grading scale)
- Information need has to be translated into a Boolean expression which most users find awkward
- The Boolean queries formulated by the users are most often too simplistic
- As a consequence, the Boolean model frequently returns either too few or too many documents in response to a user query
Vector-based Model

- Use of binary weights is too limiting
- Non-binary weights provide consideration for partial matches
- These term weights are used to compute a degree of similarity between a query and each document
- Ranked set of documents provides for better matching
Vector-based Model

• Define:
  – $w_{ij} > 0$ whenever $k_i \in d_j$
  – $w_{iq} \geq 0$ associated with the pair $(k_i, q)$
  – $\text{vec}(d_j) = (w_{1j}, w_{2j}, ..., w_{tj})$
  – $\text{vec}(q) = (w_{1q}, w_{2q}, ..., w_{tq})$
  – Assign to each term $k_i$ a vector $\text{vec}(i)$
  – The vectors $\text{vec}(i)$ and $\text{vec}(j)$ are assumed to be orthonormal (i.e., index terms are assumed to occur independently within the documents)

• The $t$ vectors $\text{vec}(i)$ form an orthonormal basis for a $t$-dimensional space

• In this space, queries and documents are represented as weighted vectors
Vector-based Model

- $\text{Sim}(q,d_j) = \cos(\Theta)$
  $= [\text{vec}(d_j) \cdot \text{vec}(q)] / |d_j| \times |q|$
  $= [\sum w_{ij} * w_{iq}] / |d_j| \times |q|$

- Since $w_{ij} > 0$ and $w_{iq} > 0$, $0 \leq \text{sim}(q,d_j) \leq 1$

- A document is retrieved even if it matches the query terms only partially
Vector-based Model

- $\text{Sim}(q,d_j) = \left[ \sum w_{ij} * w_{iq} \right] / |d_j| * |q|$
- How to compute the weights $w_{ij}$ and $w_{iq}$?
- A good weight must take into account two effects:
  - quantification of intra-document contents
    - $tf$ factor, the term frequency within a document
  - quantification of inter-documents separation
    - $idf$ factor, the inverse document frequency
  - $w_{ij} = tf(i,j) * idf(i)$
Probabilistic Model

• Objective: to capture the IR problem using a probabilistic framework

• Given a user query, there is an *ideal* answer set

• Guess at the beginning what that could be (i.e., guess initial description of ideal answer set)

• Improve by iteration
Probabilistic Model

- An initial set of documents is retrieved somehow
  - Can be done using vector-space model, boolean model
- User inspects these docs looking for the relevant ones (in truth, only top 10-20 need to be inspected)
- IR system uses this information to refine description of ideal answer set
- By repeating this process, it is expected that the description of the ideal answer set will improve
- Have always in mind the need to guess at the very beginning the description of the ideal answer set
- Description of ideal answer set is modeled in probabilistic terms
Probabilistic Ranking Principle

- Given a user query $q$ and a document $d_j$, the probabilistic model tries to estimate the probability that the user will find the document $d_j$ interesting (i.e., relevant).

- The model assumes that this probability of relevance depends on the query and the document representations only.
Key Terms Used in IR

- QUERY: a representation of what the user is looking for - can be a list of words or a phrase.
- DOCUMENT: an information entity that the user wants to retrieve
- COLLECTION: a set of documents
- INDEX: a representation of information that makes querying easier
- TERM: word or concept that appears in a document or a query
<table>
<thead>
<tr>
<th>Classification</th>
<th>Inverted File</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster</td>
<td>Query Expansion</td>
</tr>
<tr>
<td>Similarity</td>
<td>Relevance</td>
</tr>
<tr>
<td>Information Extraction</td>
<td>Relevance Feedback</td>
</tr>
<tr>
<td>Term Frequency</td>
<td>Stemming</td>
</tr>
<tr>
<td>Inverse Document Frequency</td>
<td>Stopword</td>
</tr>
<tr>
<td>Precision</td>
<td>Vector Space Model</td>
</tr>
<tr>
<td>Recall</td>
<td>Weighting</td>
</tr>
<tr>
<td></td>
<td>TREC/TIPSTER/MUC</td>
</tr>
</tbody>
</table>