Chapter 5: Information Retrieval and Web Search

An introduction

Most slides courtesy Bing Liu
Introduction

- Text mining refers to data mining using text documents as data.
- Most text mining tasks use Information Retrieval (IR) methods to pre-process text documents.
- These methods are quite different from traditional data pre-processing methods used for relational tables.
- Web search also has its root in IR.
Information Retrieval (IR)

- Conceptually, IR is the study of finding needed information. I.e., IR helps users find information that matches their information needs.
  - Expressed as queries

- Historically, IR is about document retrieval, emphasizing document as the basic unit.
  - Finding documents relevant to user queries

- Technically, IR studies the acquisition, organization, storage, retrieval, and distribution of information.
IR architecture
IR queries

- Keyword queries
- Boolean queries (using AND, OR, NOT)
- Phrase queries
- Proximity queries
- Full document queries
- Natural language questions
Information retrieval models

- An IR model governs how a document and a query are represented and how the relevance of a document to a user query is defined.

- Main models:
  - Boolean model
  - Vector space model
  - Statistical language model
  - etc
Boolean model

- Each document or query is treated as a “bag” of words or terms. Word sequence is not considered.
- Given a collection of documents $D$, let $V = \{t_1, t_2, \ldots, t_{|V|}\}$ be the set of distinctive words/terms in the collection. $V$ is called the vocabulary.
- A weight $w_{ij} > 0$ is associated with each term $t_i$ of a document $d_j \in D$. For a term that does not appear in document $d_j$, $w_{ij} = 0$.

$$d_j = (w_{1j}, w_{2j}, \ldots, w_{|V|j}),$$
Boolean model (contd)

- Query terms are combined logically using the Boolean operators **AND**, **OR**, and **NOT**.
  - E.g., \((data \text{ AND} mining) \text{ AND } (NOT\text{ text})\)

- Retrieval
  - Given a Boolean query, the system retrieves every document that makes the query logically true.
  - Called **exact match**.

- The retrieval results are usually quite poor because term frequency is not considered.
Vector space model

- Documents are also treated as a “bag” of words or terms.
- Each document is represented as a vector.
- However, the term weights are no longer 0 or 1. Each term weight is computed based on some variations of TF or TF-IDF scheme.

**Term Frequency (TF) Scheme:** The weight of a term $t_i$ in document $d_j$ is the number of times that $t_i$ appears in $d_j$, denoted by $f_{ij}$. Normalization may also be applied.
The most well known weighting scheme
- TF: still term frequency
- IDF: inverse document frequency.

$N$: total number of docs
$df_i$: the number of docs that $t_i$ appears.

The final TF-IDF term weight is:

$$tf_{ij} = \frac{f_{ij}}{\max\{f_{1j}, f_{2j}, \ldots, f_{|V|j}\}}$$

$$idf_i = \log \frac{N}{df_i}$$

$$w_{ij} = tf_{ij} \times idf_i.$$
Retrieval in vector space model

- Query $q$ is represented in the same way or slightly differently.
- **Relevance of $d_i$ to $q$:** Compare the similarity of query $q$ and document $d_i$.
- Cosine similarity (the cosine of the angle between the two vectors)

$$
\text{cosine}(d_j, q) = \frac{\langle d_j \cdot q \rangle}{\|d_j\| \times \|q\|} = \frac{\sum_{i=1}^{V} w_{ij} \times w_{iq}}{\sqrt{\sum_{i=1}^{V} w_{ij}^2} \times \sqrt{\sum_{i=1}^{V} w_{iq}^2}}
$$

- Cosine is also commonly used in text clustering
An Example

- A document space is defined by three terms:
  - hardware, software, users
  - the vocabulary

- A set of documents are defined as:
  - $A_1=(1, 0, 0)$, $A_2=(0, 1, 0)$, $A_3=(0, 0, 1)$
  - $A_4=(1, 1, 0)$, $A_5=(1, 0, 1)$, $A_6=(0, 1, 1)$
  - $A_7=(1, 1, 1)$, $A_8=(1, 0, 1)$, $A_9=(0, 1, 1)$

- If the Query is “hardware and software”
- what documents should be retrieved?
An Example (cont.)

- **In Boolean query matching:**
  - document $A_4$, $A_7$ will be retrieved ("AND")
  - retrieved: $A_1$, $A_2$, $A_4$, $A_5$, $A_6$, $A_7$, $A_8$, $A_9$ ("OR")

- **In similarity matching (cosine):**
  - $q = (1, 1, 0)$
  - $S(q, A_1) = 0.71$, $S(q, A_2) = 0.71$, $S(q, A_3) = 0$
  - $S(q, A_4) = 1$, $S(q, A_5) = 0.5$, $S(q, A_6) = 0.5$
  - $S(q, A_7) = 0.82$, $S(q, A_8) = 0.5$, $S(q, A_9) = 0.5$
  - Document retrieved set (with ranking) =
    - $\{A_4, A_7, A_1, A_2, A_5, A_6, A_8, A_9\}$
Okapi relevance method

- Another way to assess the degree of relevance is to directly compute a relevance score for each document to the query.
- The **Okapi** method and its variations are popular techniques in this setting.

The Okapi relevance score of a document $d_j$ for a query $q$ is:

$$
onapi(d_j, q) = \sum_{t_i \in q, d_j} \ln \frac{N - df_i + 0.5}{df_i + 0.5} \times \frac{(k_1 + 1)f_{ij}}{k_1 (1 - b + b \frac{dl_j}{avdl}) + f_{ij}} \times \frac{(k_2 + 1)f_{iq}}{k_2 + f_{iq}},$$

where $k_1$ (between 1.0-2.0), $b$ (usually 0.75) and $k_2$ (between 1-1000)
Relevance feedback

Relevance feedback is one of the techniques for improving retrieval effectiveness. The steps:

- the user first identifies some relevant \((D_r)\) and irrelevant documents \((D_{ir})\) in the initial list of retrieved documents
- the system expands the query \(q\) by extracting some additional terms from the sample relevant and irrelevant documents to produce \(q_e\)
- Perform a second round of retrieval.

Rocchio method \((\alpha, \beta \text{ and } \gamma \text{ are parameters})\)

\[
q_e = \alpha q + \frac{\beta}{|D_r|} \sum_{d_r \in D_r} d_r - \frac{\gamma}{|D_{ir}|} \sum_{d_{ir} \in D_{ir}} d_{ir}
\]
Text pre-processing

- Word (term) extraction: easy
- Stopwords removal
- Stemming
- Frequency counts and computing TF-IDF term weights.
Stopwords removal

- Many of the most frequently used words in English are useless in IR and text mining – these words are called *stop words*.
  - the, of, and, to, ....
  - Typically about 400 to 500 such words
  - For an application, an additional domain-specific stopwords list may be constructed

- Why do we need to remove stopwords?
  - Reduce indexing (or data) file size
    - stopwords accounts 20-30% of total word counts.
  - Improve efficiency and effectiveness
    - stopwords are not useful for searching or text mining
    - they may also confuse the retrieval system.
Stemming

- Techniques used to find out the root/stem of a word. E.g.,
  - user
  - users
  - used
  - using
  - engineering
  - engineered
  - engineer

- stem: use engineer

**Usefulness:**
- improving effectiveness of IR and text mining
  - matching similar words
  - Mainly improve recall
- reducing indexing size
  - combing words with same roots may reduce indexing size as much as 40-50%.
Basic stemming methods

Using a set of rules. E.g.,

- **remove ending**
  - if a word ends with a consonant other than s, followed by an s, then delete s.
  - if a word ends in es, drop the s.
  - if a word ends in ing, delete the ing unless the remaining word consists only of one letter or of th.
  - If a word ends with ed, preceded by a consonant, delete the ed unless this leaves only a single letter.
  - ......

- **transform words**
  - if a word ends with “ies” but not “eies” or “aies” then “ies --> y.”
Frequency counts + TF-IDF

- Counts the number of times a word occurred in a document.
  - Using occurrence frequencies to indicate relative importance of a word in a document.
    - if a word appears often in a document, the document likely “deals with” subjects related to the word.

- Counts the number of documents in the collection that contains each word

- TF-IDF can be computed.
Evaluation: Precision and Recall

- Given a query:
  - Are all retrieved documents relevant?
  - Have all the relevant documents been retrieved?

- Measures for system performance:
  - The first question is about the precision of the search
  - The second is about the completeness (recall) of the search.
**Example 2:** Following Example 1, we obtain the interpolated precisions at all 11 recall levels in the table of Fig. 6.4. The precision-recall curve is shown on the right.

<table>
<thead>
<tr>
<th>$i$</th>
<th>$p(r_i)$</th>
<th>$r_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>100%</td>
<td>0%</td>
</tr>
<tr>
<td>1</td>
<td>100%</td>
<td>10%</td>
</tr>
<tr>
<td>2</td>
<td>100%</td>
<td>20%</td>
</tr>
<tr>
<td>3</td>
<td>100%</td>
<td>30%</td>
</tr>
<tr>
<td>4</td>
<td>80%</td>
<td>40%</td>
</tr>
<tr>
<td>5</td>
<td>80%</td>
<td>50%</td>
</tr>
<tr>
<td>6</td>
<td>71%</td>
<td>60%</td>
</tr>
<tr>
<td>7</td>
<td>70%</td>
<td>70%</td>
</tr>
<tr>
<td>8</td>
<td>70%</td>
<td>80%</td>
</tr>
<tr>
<td>9</td>
<td>62%</td>
<td>90%</td>
</tr>
<tr>
<td>10</td>
<td>62%</td>
<td>100%</td>
</tr>
</tbody>
</table>

**Fig. 6.4.** The precision-recall curve
Compare different retrieval algorithms

Fig. 6.5. Comparison of two retrieval algorithms based on their precision-recall curves
Compare with multiple queries

- Compute the average precision at each recall level.
  \[
  \bar{p}(r_i) = \frac{1}{|Q|} \sum_{j=1}^{Q} p_j(r_i),
  \]
  where \( Q \) is the set of all queries and \( p_j(r_i) \) is the precision of query \( j \) at the recall level \( r_i \). Using the average precision at each recall level, we can also draw a precision-recall curve.

- Draw precision recall curves
  - Can also use the F-score evaluation measure.
Rank precision

- Compute the precision values at some selected rank positions.
- Mainly used in Web search evaluation.
- For a Web search engine, we can compute precisions for the top 5, 10, 15, 20, 25 and 30 returned pages
  - as the user seldom looks at more than 30 pages.
- Recall is not very meaningful in Web search.
  - Why?
Web Search as a huge IR system

- A Web crawler (robot) crawls the Web to collect all the pages.
- Servers establish a huge inverted indexing database and other indexing databases.
- At query (search) time, search engines conduct different types of vector query matching.
Inverted index

- The inverted index of a document collection is basically a data structure that
  - attaches each distinctive term with a list of all documents that contains the term.
- Thus, in retrieval, it takes constant time to
  - find the documents that contains a query term.
  - multiple query terms are also easy handle as we will see soon.
An example

Example 3: We have three documents of $id_1$, $id_2$, and $id_3$:

$id_1$: Web mining is useful.  
1 2 3 4

$id_2$: Usage mining applications.  
1 2 3

$id_3$: Web structure mining studies the Web hyperlink structure.  
1 2 3 4 5 6 7 8

Applications: $id_2$  
Hyperlink: $id_3$  
Mining: $id_1$, $id_2$, $id_3$  
Structure: $id_3$  
Studies: $id_3$  
Usage: $id_2$  
Useful: $id_1$  
Web: $id_1$, $id_3$

Applications: <$id_2$, 1, [3]>  
Hyperlink: <$id_3$, 1, [7]>  
Structure: <$id_3$, 2, [2, 8]>  
Studies: <$id_3$, 1, [4]>  
Usage: <$id_2$, 1, [1]>  
Useful: <$id_1$, 1, [4]>  
Web: <$id_1$, 1, [1]>, <$id_3$, 2, [1, 6]>

(A)  
(B)

Fig. 6.7. Two inverted indices: a simple version and a more complex version
Index construction

- Easy! See the example,

Fig. 6.8. The vocabulary trie and the inverted lists
Search using inverted index

Given a query $q$, search has the following steps:

- **Step 1 (vocabulary search):** find each term/word in $q$ in the inverted index.
- **Step 2 (results merging):** Merge results to find documents that contain all or some of the words/terms in $q$.
- **Step 3 (Rank score computation):** To rank the resulting documents/pages, using
  - content-based ranking
  - link-based ranking
Different search engines

The real differences among different search engines are:

- their index weighting schemes
  - Including location of terms, e.g., title, body, emphasized words, etc.
- their query processing methods (e.g., query classification, expansion, etc)
- their ranking algorithms
- Few of these are published by any of the search engine companies. They are tightly guarded secrets.
Summary

- We only give a **VERY** brief introduction to IR. There are a large number of other topics, e.g.,
  - Statistical language model
  - Latent semantic indexing (LSI and SVD).
  - (read an IR book or take an IR course)
- Many other interesting topics are not covered, e.g.,
  - Web search
    - Index compression
    - Ranking: combining contents and hyperlinks
  - Web page pre-processing
  - Combining multiple rankings and meta search
  - Web spamming
- **Want to know more? Read the textbook**