Tutorial: From Ctrl-F to Information Retrieval - An Excursion into the World of Searching in Text

Andreas Schmidt

Institute for Automation and Applied Informatics
Karlsruhe Institute of Technologie
Germany

Department of Informatics and Business Information Systems
University of Applied Sciences Karlsruhe
Germany
Latest Version of this Slides

https://www.smiffy.de/dbkda-2022/

- Slideset
- Exercise
- Command refcard
- Many examples
- Example datasets
- Further resources

1. all materials copyright 2017, 2018, 2019, 2020, 2021, 2022 by andreas schmidt
Tutorial: Around the Inverted Index

Andreas Schmidt

Institute for Automation and Applied Informatics
Karlsruhe Institute of Technologie
Germany

Department of Informatics and Business Information Systems
University of Applied Sciences Karlsruhe
Germany
Outlook

- Boolean Search
- Inverted Index & Extensions
- Term Relevance and Ranking
- Vector Space Model
- Summary
Boolean Retrieval

Queries of the Form:

• Q1: Wimbledon
• Q2: Agassi AND Federer
• Q3: Becker OR Boris
• Q4: Federer AND NOT Wimbledon
• Q5: (Federer OR Agassi) AND (Wimbledon OR Rothenbaum)
## Term-Document Matrix

<table>
<thead>
<tr>
<th></th>
<th>Doc 1</th>
<th>Doc 2</th>
<th>Doc 3</th>
<th>Doc 4</th>
<th>Doc 5</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agassi</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Australia</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Australian Open</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Britain</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Davenport</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Federer</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>France</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Hewitt</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Kuznetsova</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Melbourne</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
In which documents appear the words *Agassi* and *Federer*?

<table>
<thead>
<tr>
<th></th>
<th><strong>Agassi</strong></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Agassi</strong></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

&

<table>
<thead>
<tr>
<th></th>
<th><strong>Federer</strong></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Federer</strong></td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>...</td>
</tr>
</tbody>
</table>

= 

<table>
<thead>
<tr>
<th></th>
<th><strong>Agassi &amp; Federer</strong></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Agassi &amp; Federer</strong></td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
In which documents appear the words *Agassi* and *Federer*, but not *Melbourne*?

<table>
<thead>
<tr>
<th></th>
<th>Agassi</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Federer</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>&amp;</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>not(Melbourne)</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

=

| Agassi & Federer & not (Melbourne) | 0 | 0 | 0 | 1 | 0 | ... |
Quantitative Aspects

- Document Collection:
  - 1,000,000 documents
  - Average size of a document: ~1000 words
  - Average word length: ~6 characters
  - 50,000 - 500,000 distinct words in corpus

=> total volume: ~6 GB
Quantitative Aspects of the Term-Document Matrix

- Term-Document-Matrix:
  - 1.000.000 Columns (the documents)
  - 500.000 Rows (the different words)

=> total volume: 1.000.000 x 500.000 bits = 62.5 GB

- but:
  - about 99.8% of all cells in a row contain ’0’

=> we only store the information about the ’1’cells
Inverted Index: Quantitative Aspects

- 1,000,000 x 500,000 * 0.2% ~ 1,000,000 entries (Document IDs)
  => 4 MB (with 4 byte for each entry)

- 500,000 rows (vocabulary)
  - average word length: ~ 8 characters
    => memory requirement: 4 MB
      + additional 2 MB for the pointers to postlinglists

- total memory consumption for our index: ~10 MB
Inverted Index

<table>
<thead>
<tr>
<th>Vocabularity</th>
<th>Doc 1</th>
<th>Doc 2</th>
<th>Doc 4</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agassi</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Australia</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Australian Open</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Britain</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Davenport</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Federer</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>France</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hewitt</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kuznetsova</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Melbourne</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Inverted Index

Vocabularity

- Agassi
- Australia
- Australian Open
- Britain
- Davenport
- Federer
- France
- Hewitt
- Kuznetsova
- Melbourne

Term

Posting

Postinglist
Inverted Index: Characteristics

- Fast access to posting list of a term e.g. via hash function, complexity: O(1)
- Postings are sorted by Document ID
- AND, OR operations have linear complexity (number of documents)

Example Queries:

- All documents containing the term „Melbourne“
  - trivial, already available

- All documents containing the terms „Cacablanca“ and „Martini“:

  Casablanca: 1, 7, 23, 61, 109, 207
  Martini: 2, 23, 24, 51, 109, 211, 220
Inverted Index: Query Processing

Casablanca: 1, 7, 23, 61, 109, 207
Martini: 2, 23, 24, 51, 109, 211, 220

Casablanca: 1, 7, 23, 61, 109, 207
Martini: 2, 23, 24, 51, 109, 211, 220

Casablanca: 1, 7, 23, 61, 109, 207
Martini: 2, 23, 24, 51, 109, 211, 220

Casablanca: 1, 7, 23, 61, 109, 207
Martini: 2, 23, 24, 51, 109, 211, 220

Casablanca: 1, 7, 23, 61, 109, 207
Martini: 2, 23, 24, 51, 109, 211, 220

L = \{\}
Inverted Index: Query Processing

Casablanca: 1, 7, 23, **61**, 109, 207
Martini: 2, 23, **24**, 51, 109, 211, 220

L = {23}

Casablanca: 1, 7, 23, **61**, 109, 207
Martini: 2, 23, **24**, 51, 109, 211, 220

L = {23}

Casablanca: 1, 7, 23, **61**, 109, 207
Martini: 2, 23, 24, 51, **109**, 211, 220

L = {23}

Casablanca: 1, 7, 23, **61**, 109, 207
Martini: 2, 23, 24, 51, **109**, 211, 220

L = {23, 109}

Casablanca: 1, 7, 23, **61**, **109**, 207
Martini: 2, 23, 24, 51, **109**, 211, 220

L = {23, 109}
Inverted Index: Query Processing

• Analog strategy also for OR, NOT

Search query with n (n>2) terms:

• Posting lists: $L_1, L_2, ..., L_n$

  $L_1 \text{ AND } L_2 \rightarrow L_{12}$
  $L_{12} \text{ AND } L_3 \rightarrow L_{123}$
  ...
  $L_{12...(n-1)} \text{ AND } L_n \rightarrow L_{12...n}$

• Optimization (for AND): start with shortest list
  (general: sort the posting lists by their length in ascending order)
Inverted Index: Construction

- **Given::**
  - \(<n>\) Documents with Document IDs (Document collection, sorted by id)
  - Dictionary with text keys (for the terms) and list based values (posting lists)

- **Algorithmus:**

```python
index = {}
for doc in document_collection:
    for word in doc.words.as_set():
        if index[word]:
            index[word].append(doc.id)
        else:
            index[word] = list(doc.id)
return index
```
**bbcsport:** Example dataset [1]

- Consists of documents from the BBC Sport website corresponding to sports news articles in five topical areas from 2004-2005.
- Documents: 737, Terms: 4613
- Natural Classes: 5 (athletics, cricket, football, rugby, tennis)

```
$ ls bbcsport/
athletics  cricket  football  rugby  tennis  README.TXT

$ ls bbcsport/tennis/
001.txt  009.txt  017.txt  025.txt  033.txt  041.txt  049.txt
057.txt  065.txt  073.txt  081.txt  089.txt  097.txt
002.txt  010.txt  018.txt  026.txt ...
```

Tim Henman saved a match point before fighting back to defeat British rival Greg Rusedski 4–6 7–6 (8–6) 6–4 at the Dubai Tennis Championships on Tuesday.

World number 46 Rusedski broke in the ninth game to take a tight opening set. Rusedski had match point at 6–5 in the second set tie-break after Henman double-faulted, but missed his chance and Henman rallied to clinch the set. The British number one then showed his superior strength to take the decider and earn his sixth win over Rusedski. Serve was held by both players with few alarms until the seventh game of the final set, when Rusedski's wild volley gave Henman a vital break. A furious Rusedski slammed his racket onto the ground in disgust and was warned by the umpire.

Henman, seeded three, then held his serve comfortably thanks to four serve-and-volley winners to take a clear 5–3 lead. Rusedski won his service game but Henman took the first of his three match points with a service winner to secure his place in the second round at Dubai for the first time in three years. It was the first match between the pair for three years – Henman last ...
Construction of the Index using Shell Commands ...

• Tokenization
  
grep -E -o '[A-Za-z]+' bbcsport/tennis/001.txt

• Lowercase text:
  
tr < bbcsport/tennis/001.txt 'A-Z' 'a-z'

• Construct Inverted Index (Uppercase words only)
  
mkdir -p InvIndex
grep -o -E '[A-Z][a-z]+' bbcsport/*//*.txt | sort | uniq |
  awk -F: '{print $1 >> "InvIndex/"$2}'

• Query: All documents containing the words Alex and Rusedski
  
comm InvIndex/Alex InvIndex/Rusedski -1 -2

• Query: All documents containing the word Rusedski, but not the word Alex
  
comm InvIndex/Alex InvIndex/Rusedski -1 -3
Boolean Retrieval: Summary & Extensions

- Based on Set-of-Word model
- Advantage: mathematically based (set theory)
- Queries are based on boolean expressions with AND, OR, NOT, (, )
- Disadvantage: no order (ranking) of the returned documents.
- Possible Extensions:
  - Partition document into different fields (title, abstract, ..., literature) and query specific fields, e.g.:
    
    \[
    \text{TITLE: (trump AND impeachment) AND AUTHOR: sanders}
    \]
  - Consider frequency of search terms (switch to the bag-of-words model) for ranking the results.
  - Phrase-Search, NEAR Operator, z.B.:
    \[
    \text{“The world is my oyster“}
    \text{trump NEAR corona}
    \]
Boolean Retrieval: Summary & Extensions

- Based on Set-of-Word model
- Advantage: mathematically based (set theory)
- Queries are based on boolean expressions with AND, OR, NOT, (, )
- Disadvantage: no order (ranking) of the returned documents.
- Possible Extensions:
  - Partition document into different fields (title, abstract, ..., literature) and query specific fields, e.g.:
    \[ \text{TITLE:} (\text{trump AND impeachment}) \text{ AND AUTHOR: sanders} \]
  - Consider frequency of search terms (switch to the bag-of-words model) for ranking the results.
  - Phrase-Search, NEAR Operator, z.B.:
    \[ "\text{The world is my oyster"} \text{ trump NEAR corona} \]

more fine-grained subdivision of documents
requires extension of the inverted index
Inverted Index: Extension 1

- Additionally consider the number of times a term occurs in a document.

- Handling of multiple search terms?
  - Example: *mike stonebraker*

  ![Diagram](image)

  - The term "country" appears 2-times in document D3.

  ![Search Results](image)

  - 2.410.000.000 matches
  - 6.780.000 matches
Inverted Index: Extension 1b

- Additionally consider the number of times a term occurs in a document.

  - Example:
    - `mike stonebraker`
    - `australien (2)`: D1(2), D2(1)
    - `country (4)`: D1(1), D2(3), D3(2), D4(1)

  - The term "country" appears 2-times in document D3.

- Handling of multiple search terms?

  - The weighting of "stonebraker" should be higher, like that of "mike".
Term Weighting

• How often a term occurs in a document (Term Frequency - TF)
• In how many documents of the collection does a term occur (Document Frequency - DF)

• Intuitive solution:
  • A document should get a high weight if a search term occurs very often in the document
  • Search terms themselves have different weights: A search term that occurs in many documents in the document collection is not as important as a term that occurs in relatively few documents.

=> see Vector-Model later for concrete calculation
Construction of Inverted Index with TF

• Inverted Index with additional Term Frequency (in 2nd column):
  $ grep -o -E '[A-Z][a-z]+' bbcsport/tennis/*.txt |sort| \ 
  uniq -c | tr -s ' ' : | \ 
  awk -F: '{print $3"\t"$2 >> "InvIndexTf/"$4}''

• Calculate Document Frequency
  wc -l InvIndex/*|head -n -1| sed 's#InvIndex/##'| \ 
  awk '{print $2"\t"$1}''

• Create Stopword-list (most common 50 words in corpus)
  cat bbcsport/*/*.txt | tr 'A-Z' 'a-z' | grep -o -E '[a-z]+' |\ 
  sort | uniq -c | sort -nr| head -n50
Inverted Index: Extension 2

- Additionally consider the position(s) of a term in a document

```
australien (2)  D1(2)_{12,21}  D2(1)_{14}
```

```
country (4)  D1(1)_{19}  D2(3)_{11,21,101}  D3(2)_{2,37}  D4(1)_{51}
```

The term „country“ appears 3-times in document D3 at position 11, 21, and 101.

- Allows phrase queries, i.e.
- „mike stonebraker“
  
Vectorspace Model

• General idea:
  
  • A document represents a vector in a high-dimensional vector space (dimension: |vocabulary|)
  • The query is also represented by a vector in the high-dimensional vector space
  
  • One can define various similarity measures on vectors, which can then be used for ranking documents.
  
  • Typically cosine measure is used to determine the similarity of the documents to the request
  
  • Can also be used to find similar documents to a given document (show similar documents)
### Again: Term-Document Matrix

<table>
<thead>
<tr>
<th>Term</th>
<th>tf(D1)</th>
<th>tf(D2)</th>
<th>tf(D3)</th>
<th>tf(D4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>australia</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>canada</td>
<td></td>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>continent</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>country</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>player</td>
<td></td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>participant</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>world</td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
... and now?

- Other view on Term-Document Matrix:

<table>
<thead>
<tr>
<th></th>
<th>tf(D1)</th>
<th>tf(D2)</th>
<th>tf(D3)</th>
<th>tf(D4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>australia</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>canada</td>
<td></td>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>continent</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>country</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>player</td>
<td></td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>participant</td>
<td>2</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>world</td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
**tf*idf**

- Answer to the question: how important is a word to a document in a collection

<table>
<thead>
<tr>
<th></th>
<th>df(t)</th>
<th>idf(t)</th>
<th>tf(D1)</th>
<th>tf(D2)</th>
<th>tf(D3)</th>
<th>tf(D4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>australia</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>canada</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>continent</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>country</td>
<td>4</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>player</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>participant</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>world</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

with $idf(t) = \log_2\left( \frac{|N|}{df(t)} \right)$

$|N|$: Number of documents in corpus
with $d(t) = tf(t,d) \times idf(t)$
Example Query:

- Query: country australia
  - Vector of query: \( q = (1 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0) \)
  - Vector document \( d_1 = (2 \ 0 \ 2 \ 0 \ 0 \ 0 \ 0) \)
  - Cosine similarity measure

\[
\text{sim}(q, d) = \frac{\sum_{i=1}^{n} q_i \cdot d_i}{\sqrt{\sum_{i=1}^{n} q_i^2 \cdot \sum_{i=1}^{n} d_i^2}}
\]

- \( \text{sim}(q, D1) = \frac{(1 \cdot 2 + 0 \cdot 0 + 0 \cdot 2 + 1 \cdot 0 + 0 \cdot 0 + 0 \cdot 0 + 0 \cdot 0)}{(\sqrt{1^2 + 1^2}) \cdot \sqrt{2^2 + 2^2})} = \frac{2}{4} = 0.5 \)
- \( \text{sim}(q, D2) = 0.123091 \)
- \( \text{sim}(q, D3) = \text{sim}(q, D4) = 0 \)
Vectorspace Model: Summary

- Document is represented as a vector in high-dimensional vector space
- The query will also be represented as a vector
- Cosine measure as similarity measure between vectors
- tf.idf: The number of occurrences of a term in a document and the number of documents in which the term occurs determine the relevance of a term
Preprocessing of Documents

- What happens before the Inverted Index is build?
  - Adaptation Character Sets
  - Removal of formatting
  - Tokenization (Breakdown into individual terms)
  - Normalisation (lowercase, handling dates, abbreviations, reduction, lemmatisation, stemming, stopwords)
- Further optional steps:
  - synonyms
  - integration of taxonomies, class-instance relationships
  - ...
Summary

- Inverted Index
  - Mapping from term to documents
  - Allows fast answer which document contain a single/multiple words
  - Extensions for ranking, phrase matches

- Vector Space Model
  - Every document is a vector
  - Query is a vector
  - tf*idf to handle number of occurrences/relevance of terms
  - Cosine measure as similarity measure

- The Shell
  - More powerful than we thought
  - Filter and Pipe Architecture
  - Typically incremental approach to program development