Ghislain Fourny

Information Retrieval

8. Scoring in a Bigger Picture

Picture copyright: johan2011/123RF Stock Photo
What we have seen so far
Boolean retrieval

Input
Set of documents

Output
Subset of documents

query

lawyer AND Penang AND NOT silver
Today: Ranked retrieval

Input
Set of documents

Output
Ranked subset of documents

lawyer
Penang
silver

query

1
2
3
4
<table>
<thead>
<tr>
<th>Title</th>
<th>Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author</td>
<td></td>
</tr>
<tr>
<td>Publication Date</td>
<td></td>
</tr>
<tr>
<td>Language</td>
<td>▼</td>
</tr>
<tr>
<td>Country</td>
<td>▼</td>
</tr>
<tr>
<td>Cost</td>
<td>$ to $</td>
</tr>
</tbody>
</table>

Search
Parametric indices

Title

Author

Publication Date

Language

Country

Cost

Search structure

Posting lists
**Shared inverted index with zones**

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Count</th>
<th>1.title</th>
<th>2.abstract</th>
<th>3.body</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETH</td>
<td>3</td>
<td>1.title</td>
<td>2.abstract</td>
<td>3.body</td>
</tr>
<tr>
<td>Zürich</td>
<td>2</td>
<td>1.title</td>
<td>3.title, 3.abstract</td>
<td></td>
</tr>
<tr>
<td>computer</td>
<td>2</td>
<td>1.title, 1.abstract</td>
<td>4.body</td>
<td></td>
</tr>
<tr>
<td>data</td>
<td>3</td>
<td>2.abstract</td>
<td>5.abstract</td>
<td>6.body</td>
</tr>
<tr>
<td>CPU</td>
<td>3</td>
<td>1.body</td>
<td>2.abstract</td>
<td>5.body</td>
</tr>
<tr>
<td>information</td>
<td>3</td>
<td>1.body</td>
<td>4.title, 4.body, 4.abstract</td>
<td>5.body</td>
</tr>
<tr>
<td>retrieval</td>
<td>3</td>
<td>1.body</td>
<td>6.body</td>
<td>7.title, 7.body, 7.abstract</td>
</tr>
</tbody>
</table>
**Single-term query**

<table>
<thead>
<tr>
<th>Zone</th>
<th>weight</th>
<th>information</th>
<th>retrieval</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>0.3</td>
<td>0.3</td>
<td>0.3</td>
<td>0.6</td>
</tr>
<tr>
<td>Abstract</td>
<td>0.2</td>
<td></td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>Body</td>
<td>0.5</td>
<td>0.5</td>
<td></td>
<td>0.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Total</td>
<td></td>
<td>1.3</td>
</tr>
</tbody>
</table>
Learning weights

\[
\text{argmin}_g \sum_j \text{error}_j(g) = \text{argmin}_g \sum_j (r_j - \vec{g}_j \cdot \vec{s}_j)^2
\]
Term frequency, (Inverted) Document frequency,

<table>
<thead>
<tr>
<th>tf</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>foo</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>bar</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>foobar</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>tf-idf</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>foo</td>
<td>25</td>
<td>5</td>
</tr>
<tr>
<td>bar</td>
<td>0</td>
<td>40</td>
</tr>
<tr>
<td>foobar</td>
<td>6</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>idf</th>
</tr>
</thead>
<tbody>
<tr>
<td>foo</td>
</tr>
<tr>
<td>bar</td>
</tr>
<tr>
<td>foobar</td>
</tr>
</tbody>
</table>
Model and abstraction

Document as a list of words (with duplicates)

Simplification

Document as a set of words

(0 1 0 1 0 1 1 1 0 0 1 1 0 1 0 1 0 1 0 1 0 1 0 1 0)

Document as a vector of booleans
Model and abstraction

Document as a list of words (with duplicates)

Simplification

Document as a bag of words

Document as a vector of numbers

(0.1 2.0 1.5 0.1 0.3 4.2 2.4 23.5 4.3 24.5 0.13)

Linearization
Vector-Space Model

Documents = vectors in the first quadrant of $\mathbb{R}^M$
Queries as vectors

Queries = points in the first quadrant of $\mathbb{R}^M$

$d3$ is a good result of $q2$!
Standard inverted index adapted to VSM

\[
\frac{tf_{tq} \times idf_t \times tf_{td} \times idf_t}{\|q\| \times \|d\|}
\]
SMART notation

- Query weights
- Sublinear term frequency
- Natural document frequency
- Byte-size normalization
Efficient Scoring and Ranking
Standard inverted index adapted to VSM

\[
\frac{\text{tf-idf}_{tq} \times \text{tf-idf}_{td}}{||q|| \times ||d||}
\]
Standard inverted index adapted to VSM

\[ \frac{\text{tf-idf}_{tq} \times \text{tf-idf}_{td}}{\|q\| \times \|d\|} \]

Doubles are hard to compress
Standard inverted index adapted to VSM

\[
\frac{tf_{tq} \times idf_t \times tf_{td} \times idf_t}{||q|| \times ||d||}
\]
Standard inverted index adapted to VSM

The query length is useless

\[ \frac{tf_{tq} \times idf_t \times tf_{td} \times idf_t}{\|q\| \times \|d\|} \]
Standard inverted index adapted to VSM

\[ \frac{tf_{tq} \times idf_t \times tf_{td} \times idf_t}{||d||} \]
Standard inverted index adapted to VSM

Division by document length can be done at the end
Standard inverted index adapted to VSM

<table>
<thead>
<tr>
<th>ETH</th>
<th>6</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>tf_{tq}</td>
<td>idf_t</td>
<td>3</td>
<td>tf_{td}</td>
<td>7</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>computer</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>data</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ tf_{tq} \times idf_t \times tf_{td} \times idf_t \]
Standard inverted index adapted to VSM

We only need to maintain accumulators for documents in which the query terms appear.
Standard inverted index adapted to VSM

\[ \text{tf}_{tq} \times \text{idf}_t \times \text{tf}_{td} \times \text{idf}_t \]
Many queries have only 1 and 0 weights
Standard inverted index adapted to VSM

\[ \text{idf}_t \times \text{tf}_{td} \times \text{idf}_t \]
We can use a \( *n* \) SMART scheme for the query.

}\[ t f_{td} \times idf_t \]
Inexact top-K document retrieval
Posting lists can be very large!
Posting lists can be very large!

Many accumulators to maintain
Many computations
General optimization

All documents

Top 10
General optimization

All documents

Top 10 Preselected documents
General optimization

All documents

Preselected documents
General optimization

Only compute scores in this smaller set!

Preselected documents
General optimization

Only compute scores in this smaller set!

Inexact Top 10 Preselected documents
Index elimination

Query: "ETH Zürich and school"
Index elimination

Query: "ETH Zürich and school"
Index elimination

Query: "ETH Zürich and school"
Index elimination: first idea
Index elimination: first idea

ETH

Zürich

and

school
Index elimination: first idea
Index elimination: second idea
Index elimination: second idea

ETH

Zürich

and

school

Keep documents containing many (or all) terms
Index elimination: second idea

Drop documents containing only a few terms
Index elimination: second idea

Drop documents containing only a few terms

(Easy to figure out using intersection algorithm)
Champion lists
Champion lists

Sort by decreasing term frequency
Champion lists

ETH

Only keep the top $r$ documents
Champion lists

Intersect the top r from each term
Champion lists

Intersect the top $r$ from each term

What is the difficulty here?
Champion lists

Intersect the top $r$ from each term

What is the difficulty here?

The order of the documents is term-dependent!
Static Quality Scores

We assign each document $d$ a query score $g(d)$ in $[0,1]$.
We assign each document $d$ a query score $g(d)$ in $[0, 1]$

We sort \textit{statically} by $g(d)$
Static Quality Scores

We can build our champion lists sorting by $g(d).\text{tf-idf}_{t,d}$
Static Quality Scores
Static Quality Scores

We can also maintain two lists separately
Static Quality Scores

We can also maintain two lists separately

And only use the "low list" if we do not have enough results.
Impact ordering
Impact ordering

Sort by tf (or any other term-wise impact document score)
Impact ordering

This is where to start
Impact ordering

This is where to start

go down and right until score no longer improves
Clustering
Clustering

Randomly pick leaders

\[ \sqrt{N} \]
Clustering

Cluster document space (nearest leader)
Clustering

Pick leader closest to query
Clustering

Compute scores only for local cluster
Tiered indices

Inexact Top 10

Preselected documents
Tiered indices

What if this gets us less than 10 documents?

?
Reminder: Quality Scores

We can also maintain two lists separately.

And only use the "low list" if we do not have enough results.
Tiered indices
Tiered indices
Tiered indices

ETH

Zürich

Computer
Tiered indices

Tier I
- ETH
- Zürich
- Computer

Tier II
- ETH
- Zürich
- Computer

Tier III
Tiered indices

ETH

Zürich

Computer

Tier I

Do we have enough results?
Tiered indices

Tier I

ETH

Zürich

Computer

Tier II

ETH

Zürich

Computer

Do we have enough results?
Tiered indices

ETH

Zürich

Computer

Tier I

Tier II

Tier III

ETH

Zürich

Computer
Query term proximity
Query term proximity

ETH Zurich

One of the two Swiss ETH is in Zurich.
Query term proximity

One of the two Swiss ETH is in Zurich.

\[ \omega = 4 \]
Query term proximity

There is an active ETH cryptocurrency ecosystem in Zug, near Zurich.
Query term proximity

There is an active ETH cryptocurrency ecosystem in Zug, near Zurich.

$\omega = 7$
Query term proximity

ETH Zurich

ETH Zurich has a department of Computer Science.
Query term proximity

\[ \omega = 2 \]

ETH Zurich has a department of Computer Science.
Query term proximity

ETH Zurich

One of the two Swiss ETH is in Zurich.

There is an active ETH cryptocurrency ecosystem in Zug, near Zurich.

ETH Zurich has a department of Computer Science.
Query term proximity

ETH Zurich

One of the two Swiss ETH is in Zurich.

ETH Zurich has a department of Computer Science.

There is an active ETH cryptocurrency ecosystem in Zug, near Zurich.
Query term proximity

ETH Zurich

ETH Zurich has a department of Computer Science.

One of the two Swiss ETH is in Zurich.

Top 2
Query interfaces
Query interfaces

ETH AND Zurich
OR (Computer
AND NOT Science)

Boolean queries
Query interfaces

ETH AND Zurich
OR (Computer
AND NOT Science)

Boolean queries

"ETH Zurich
Computer Science"

Phrase queries
Query interfaces

ETH AND Zurich OR (Computer AND NOT Science)

Boolean queries

"ETH Zurich Computer Science"

Phrase queries

Title: Pascal
Author: Niklaus Wirth

Zone queries
Query interfaces

Boolean queries

ETH AND Zurich
OR (Computer
AND NOT Science)

Phrase queries

"ETH Zurich
Computer Science"

Set of words

ETH Zurich
Pascal Algorithm

Zone queries

Title: Pascal
Author: Niklaus Wirth
Query interfaces

**Boolean queries**

- ETH AND Zurich
- OR (Computer AND NOT Science)

**Phrase queries**

- "ETH Zurich Computer Science"

**Wildcard queries**

- ETH Z*rich

**Set of words**

- ETH Zurich
- Pascal Algorithm

**Zone queries**

- Title: Pascal
- Author: Niklaus Wirth
Query interfaces

- **Boolean queries**
  - ETH AND Zurich
  - OR (Computer AND NOT Science)

- **Phrase queries**
  - "ETH Zurich Computer Science"

- **Wildcard queries**
  - ETH Z*rich

- **Set of words**
  - Pascal Algorithm

- **Inverted index**
  - Vector-space model
  - k-gram index
  - permuterm index

- **Biword indices**
  - Zone indices

- **Field/zone queries**
  - Title: Pascal
  - Author: Niklaus Wirth
Input collection

Parsing Linguistics

Document cache
Indexing

 Parsing Linguistics

 Indexers

 Zone indices  Inexact top K retrieval  Tiered inverted positional index  biword index  k-gram index

 Indices
Querying

User query
(Free text) Query parser
Spell correction

Zone indices
Inexact top K retrieval
Tiered inverted positional index
biword index
k-gram index

Indices
Returning results

Document cache

Evidence accumulation

Zone indices
Inexact top K retrieval
Tiered inverted positional index
Biword index
K-gram index

Indices

Scoring Ranking

Results

Weights
Fine tuning and training
Overall architecture

Document cache

Indexers

Zone indices
Inexact top K retrieval
Tiered inverted positional index
biword index
k-gram index

Indices

Parsing Linguistics

User query

(Free text) Query parser

Spell correction

Results

Scoring Ranking

Weights
Machine Learning

Training set