Related Content Finder: A Search Engine that works!

IEEE Computer Society Presentation
Friday October 28, 2005

By Douglas J. Matzke, Ph.D.
matzke@IEEE.org
Abstract

Search has become indispensable in our electronic and networked virtual communities. This has led to a large compounded growth in the search product markets, where Google is very visible to the general market. The question being asked by many, “Are these search engines finding what people want?”. This presentation discusses this topic in the context of a relatively new search technology called the Relational Content Finder or RCF developed by my company Lawrence Technologies, LLC.

RCF is integrated into the Synthetix® products marketed by Syngence. Synthetix is fast becoming the dominate search product in their particular market segment of litigation support, since it has been integrated into most of the litigation document tool vendors. The Synthetix customers are dominantly “tech-agnostic” lawyers and paralegals who demand easy to use yet reliable search technology, using “search by example”.
Outline

- Approaches to Search
- Full-Text Boolean Search
  - Optional, required, excluded terms
  - Divergence, convergence
  - Recall versus Precision
  - Boolean Search Problems
- Related Content Finder
  - Description of RCF approach
  - RCF scores and ranking
  - High recall and ranked precision
  - RCF advantages and disadvantages
  - RCF Application Scenarios
- Summary
Approaches to Search

- Attribute search (table of contents)
  - Format, keywords, metadata, status, etc
- Category search (indexes)
  - Use fields such as title, author, dates, etc
- Full-text Search (reading)
  - Boolean combinations of terms
- Concept Search (meaning)
  - Clustering, synonyms, natural language
- Search by Example (similarity)
  - Find similar documents
- Combinations of above
Full-Text Boolean Search

- **Optional** terms means logical OR
  - Example: termA termB termC
  - Means: OR(termA, termB, termC)
  - Produces: growing set size or *divergent*

- **Required** terms (“+”) means logical AND
  - Example: +termA +termB +termC
  - Means: AND(termA, termB, termC)
  - Produces: shrinking set size or *convergent*

- **Excluded** term (“−”) means logical NOT
  - Example: −termA +termB +termC
  - Means: AND(NOT(termA), termB, termC)
  - Produces: restricts to exclude terms
**Divergent and Convergent**

<table>
<thead>
<tr>
<th>Logic</th>
<th>Recall</th>
<th>Precision</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>OR Logic</td>
<td>High</td>
<td>Low</td>
<td>Too many</td>
</tr>
<tr>
<td>AND Logic</td>
<td>Low</td>
<td>High</td>
<td>May miss</td>
</tr>
</tbody>
</table>

- **Recall** is the percentage of *relevant* records that are *located*.
- **Precision** is the percentage of *retrieved* records that are *relevant*.
Recall versus Precision

- **Recall** is the percentage of relevant records that are located.
- **Precision** is the percentage of retrieved records that are relevant.
Recall versus Precision (cont)

- **Recall** is the percentage of *relevant* records that are *located*.
- **Precision** is the percentage of *retrieved* records that are *relevant*.
Boolean Search Problems

Blair & Maron: *Com. of the ACM*, Mar, ‘85

- “An Evaluation of Retrieval Effectiveness for a Full-Text Document-Retrieval System”
- Six-month study of full-text retrieval using a 350,000 page full text database
- *Users found less than 20% of relevant records*, even though believed results were good.
- User manually trades off recall versus precision
- User can't retrieve/find a known document
Related Content Finder

Approach:
- “Search by example” reinvents full-text
- Finds records “like” some example page
- Word count features act as fingerprint
- Scoring using information theory
- Ranking based on sorting record scores

Goals:
- High recall (all pages essentially have score)
- High precision (ranking of all records)
Search as Sparse Matrix

Entries $c_{ji}$ are either a bit or count.

$w_i$ for each token column

$s_j$ for each record row

Record and token dictionaries map names to indexes.
Search as fingerprint match

<table>
<thead>
<tr>
<th>Search Record Fingerprint</th>
<th>Corpus Record fingerprints</th>
</tr>
</thead>
</table>

\[
\text{total count} = \sum \sum cols = \text{Master Fingerprint}
\]

Produces weights \( w_i \)
Huffman Weights for Tokens

\[ w_i = -\log \left( \frac{\text{Count}_{\text{token } i}}{\text{Total}_{\text{tokens}}} \right) = \log(\text{Total}) - \log(\text{Count}_{i}) \]

<table>
<thead>
<tr>
<th>For Count ( t_i )</th>
<th>( w_i ) with ( \log_2 )</th>
<th>( w_i ) with ( \log_{10} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 = \log(10^6)</td>
<td>19.93 bits</td>
<td>6.00</td>
</tr>
<tr>
<td>10</td>
<td>16.60 bits</td>
<td>5.00</td>
</tr>
<tr>
<td>100</td>
<td>13.28 bits</td>
<td>4.00</td>
</tr>
<tr>
<td>1000</td>
<td>9.96 bits</td>
<td>3.00</td>
</tr>
<tr>
<td>10000</td>
<td>6.64 bits</td>
<td>2.00</td>
</tr>
<tr>
<td>100000</td>
<td>3.32 bits</td>
<td>1.00</td>
</tr>
<tr>
<td>5000000</td>
<td>1.00 bit</td>
<td>0.30</td>
</tr>
</tbody>
</table>

Computed for 1,000,000 total tokens
RCF Scoring and Ranking

- Compute score for search records based on counts and weights
- Compute scores for each record by computing distance to search record
- Normalize results so exact match (or perfect subset) scores 100%
- Sort records by score and display

*USPTO has allowed RCF scoring formulas
RCF Recall and Precision

- All records in Corpus
- Zero Scores
- Exact Matches
- All Records scored
- Ranked Precision for Relevant Records

High Recall and Ranked Precision!!
Mimic Ranking with Boolean

\[
\begin{align*}
\text{or} & \quad \text{and (termA, termB, termC),} \\
& \quad \text{and (termA, termB, \neg termC),} \\
& \quad \text{and (termA, \neg termB, termC),} \\
& \quad \text{and (termA, \neg termB, \neg termC),} \\
& \quad \text{and (\neg termA, termB, termC),} \\
& \quad \text{and (\neg termA, termB, \neg termC),} \\
& \quad \text{and (\neg termA, \neg termB, termC)}
\end{align*}
\]

\{ 1 \text { at a time means lower ranking}\}
\{ 2 \text { at a time means medium ranking}\}
\{ 3 \text { at a time means highest ranking}\}

Number of sub-expressions explodes with lots of terms!!
RCF Advantages/Disadvantages

- **Advantages**
  - Search engine adapts to user
  - Ease of use with minimal training (copy & paste)
  - Eliminates query restructuring to converge
  - Perfect matches/subsets rank 100% score
  - Not brittle due to versioning or noise
  - “Think it Find it” is Synthetix’s marketing slogan

- **Disadvantages**
  - Paradigm shift for user trained in Boolean search
  - Token counts rather than Boolean matrix
  - All records are scored (actually or conceptually)
  - More effort to score and rank
  - No numerical range searches
RCF Application Scenarios

- Litigation Support (Syngence.com)
  - “Find Similar” that actually works
  - Synthetic search (write the smoking gun)
  - Redaction detection (both sides)
  - Integrated with Concordance, IPRO, iCONECT, etc

- Search by example for online newspapers
- Plagiarize detection at universities
- Tokenized search in other markets
- Leverage professionals (with little training)
  - Lawyers
  - Doctors
  - Professors
  - Business executives
  - Geophysicists
Search by Example Interfaces

Click and Drag, Right-Click in Concordance

Synthetix Icon w/drop-down menu in IPRO
RCF Summary

- RCF is novel “search by example”
- Linguistic feature based fingerprints
- Information theory based scoring
- Patented scoring ranking formula
- Finds perfect/near matches
- High Recall AND Ranked Precision
- Proven with 450 customers over 4 yrs.
- “Think it Find it”