Result summaries, Relevance Feedback, Query Expansion

Pavel Pecina
pecina@ufal.mff.cuni.cz

Institute of Formal and Applied Linguistics
Faculty of Mathematics and Physics
Charles University

Original slides are courtesy of Hinrich Schütze, University of Stuttgart.
Contents

Result summaries
  Static summaries
  Dynamic summaries

Relevance feedback
  Rocchio algorithm
  Pseudo-relevance feedback

Query expansion
  Thesauri
Result summaries
How do we present results to the user?

- Most often: as a list of hits – aka “10 blue links” – with description

- The hit description is crucial:
  - The user often can identify good hits based on the description.
  - No need to “click” on all documents sequentially.

- The description usually contains:
  - document title, url, some metadata
  - summary

- How do we “compute” the summary?
Summaries

Two basic kinds: (i) static (ii) dynamic:

(i) A static summary of a document is always the same, regardless of the query that was issued by the user.

(ii) Dynamic summaries are query-dependent. They attempt to explain why the document was retrieved for the query at hand.
Static summaries

- In typical systems, the static summary is a subset of the document.
- Simplest heuristic: the first 50 or so words of the document
- More sophisticated: an extract consisting of a set of “key” sentences
  - Simple NLP heuristics to score each sentence
  - Summary is made up of top-scoring sentences.
  - Machine learning approach
- Most sophisticated: complex NLP to synthesize/generate a summary
  - For most IR applications: not quite ready for prime time yet
Dynamic summaries

- Present one or more “windows” or snippets within the document that contain several of the query terms.
- Prefer snippets where query terms occurred as a phrase or jointly in a small window (e.g., paragraph).
- The summary that is computed this way gives the entire content of the window – all terms, not just the query terms.
Query: “new guinea economic development”

Snippets (in bold) that were extracted from a document:

... In recent years, Papua New Guinea has faced severe economic difficulties and economic growth has slowed, partly as a result of weak governance and civil war, and partly as a result of external factors such as the Bougainville civil war which led to the closure in 1989 of the Panguna mine (at that time the most important foreign exchange earner and contributor to Government finances), the Asian financial crisis, a decline in the prices of gold and copper, and a fall in the production of oil. PNG’s economic development record over the past few years is evidence that governance issues underly many of the country’s problems. Good governance, which may be defined as the transparent and accountable management of human, natural, economic and financial resources for the purposes of equitable and sustainable development, flows from proper public sector management, efficient fiscal and accounting mechanisms, and a willingness to make service delivery a priority in practice. ...
Generating dynamic summaries

- Where do we get these other terms in the snippet from?

- We cannot construct a dynamic summary from the positional inverted index – at least not efficiently.

- We need to cache documents.

- The positional index tells us: query term occurs at position 4378 in the document.

- Byte offset or word offset?

- Note that the cached copy can be outdated

- Don’t cache very long documents – just cache a short prefix
Dynamic summaries

- Space on the search result page is limited.
- The snippets must be short but also long enough to be meaningful.
- Snippets should communicate whether and how the document answers the query.
- Ideally:
  - linguistically well-formed snippets
  - should answer the query, so we don’t have to look at the document.
- Dynamic summaries are a big part of user happiness because ...
  - we can quickly scan them to find the relevant document to click on.
  - in many cases, we don’t have to click at all and save time.
Relevance feedback
How can we improve recall in search?

- Two ways of improving recall: relevance feedback, query expansion

- Example:
  - query $q$: [aircraft]
  - document $d$: containing “plane”, but not containing “aircraft”

- A simple IR system will not return $d$ for $q$ even if $d$ is the most relevant document for $q$!

- We want to return relevant documents even if there is no term match with the (original) query.
Goal: increasing the number of relevant documents returned to user

- This may actually decrease recall on some measures e.g., when expanding “jaguar” with “panthera”
- which eliminates some relevant documents, but increases relevant documents returned on top pages.

- Options for improving recall:
  1. Local: on-demand analysis for a user query – relevance feedback
  2. Global: on-time analysis to produce thesaurus – query expansion
1. The user issues a (short, simple) query.

2. The search engine returns a set of documents.

3. User marks some docs as relevant, some as nonrelevant.

4. Search engine computes a new representation of the information need. Hope: better than the initial query.

5. Search engine runs new query and returns new results.

6. New results have (hopefully) better recall.
We can iterate this: several rounds of relevance feedback.

We will use the term **ad-hoc retrieval** to refer to regular retrieval without relevance feedback.

We will now look at three different examples of relevance feedback that highlight different aspects of the process.
Relevance Feedback: Example 1

Shopping related 607,000 images are indexed and classified in the database
Only One keyword is allowed!!!

bike

Designed by Baris Sumengen and Shawn Newsam

Powered by JLAMP2000 (Java, Linux, Apache, Mysql, Perl, Windows2000)
Results for initial query
User feedback: Select what is relevant
Results after relevance feedback
Vector space example: query “canine” (1)

source: Fernando Díaz
Similarity of docs to query “canine”

source: Fernando Díaz
User feedback: Select relevant documents

source: Fernando Díaz
Results after relevance feedback

source: Fernando Díaz
Example 3: A real (non-image) example

Initial query: [new space satellite applications]

Results for initial query: $(r = \text{rank}, s = \text{score})$

<table>
<thead>
<tr>
<th>$r$</th>
<th>$s$</th>
<th>title</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>1</td>
<td>0.539</td>
</tr>
<tr>
<td>+</td>
<td>2</td>
<td>0.533</td>
</tr>
<tr>
<td>3</td>
<td>0.528</td>
<td>Science Panel Backs NASA Satellite Plan, But Urges Launches of Smaller Probes</td>
</tr>
<tr>
<td>4</td>
<td>0.526</td>
<td>A NASA Satellite Project Accomplishes Incredible Feat: Staying Within Budget</td>
</tr>
<tr>
<td>5</td>
<td>0.525</td>
<td>Scientist Who Exposed Global Warming Proposes Satellites for Climate Research</td>
</tr>
<tr>
<td>6</td>
<td>0.524</td>
<td>Report Provides Support for the Critics Of Using Big Satellites to Study Climate</td>
</tr>
<tr>
<td>7</td>
<td>0.516</td>
<td>Arianespace Receives Satellite Launch Pact From Telesat Canada</td>
</tr>
<tr>
<td>+</td>
<td>8</td>
<td>0.509</td>
</tr>
</tbody>
</table>

User then marks relevant documents with “+”.

Expanded query after relevance feedback

<table>
<thead>
<tr>
<th>Term</th>
<th>Score</th>
<th>Term</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>new</td>
<td>2.074</td>
<td>space</td>
<td>15.106</td>
</tr>
<tr>
<td>satellite</td>
<td>30.816</td>
<td>application</td>
<td>5.660</td>
</tr>
<tr>
<td>nasa</td>
<td>5.991</td>
<td>eos</td>
<td>5.196</td>
</tr>
<tr>
<td>launch</td>
<td>4.196</td>
<td>aster</td>
<td>3.972</td>
</tr>
<tr>
<td>instrument</td>
<td>3.516</td>
<td>arianespace</td>
<td>3.446</td>
</tr>
<tr>
<td>bundespost</td>
<td>3.004</td>
<td>ss</td>
<td>2.806</td>
</tr>
<tr>
<td>rocket</td>
<td>2.790</td>
<td>scientist</td>
<td>2.053</td>
</tr>
<tr>
<td>broadcast</td>
<td>2.003</td>
<td>earth</td>
<td>1.172</td>
</tr>
<tr>
<td>oil</td>
<td>0.836</td>
<td>measure</td>
<td>0.646</td>
</tr>
</tbody>
</table>

Compare to original query: [new space satellite applications]
## Results for expanded query

<table>
<thead>
<tr>
<th></th>
<th>r</th>
<th>s</th>
<th>title</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>1</td>
<td>0.513</td>
<td>NASA Scratches Environment Gear From Satellite Plan</td>
</tr>
<tr>
<td>*</td>
<td>2</td>
<td>0.500</td>
<td>NASA Hasn’t Scrapped Imaging Spectrometer</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.493</td>
<td>When the Pentagon Launches a Secret Satellite, Space Sleuths Do Some Spy Work of Their Own</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.493</td>
<td>NASA Uses ‘Warm’ Superconductors For Fast Circuit</td>
</tr>
<tr>
<td>*</td>
<td>5</td>
<td>0.492</td>
<td>Telecommunications Tale of Two Companies</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0.491</td>
<td>Soviets May Adapt Parts of SS-20 Missile For Commercial Use</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.490</td>
<td>Gaping Gap: Pentagon Lags in Race To Match the Soviets In Rocket Launchers</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>0.490</td>
<td>Rescue of Satellite By Space Agency To Cost $90 Million</td>
</tr>
</tbody>
</table>
Key concept for relevance feedback: Centroid

- The centroid is the center of mass of a set of points.
- We represent documents as points in a high-dimensional space.
- Thus: we can compute centroids of documents.
- Definition:

\[
\bar{\mu}(D) = \frac{1}{|D|} \sum_{d \in D} \bar{v}(d)
\]

- where:
  - \( D \) is a set of documents;
  - \( \bar{v}(d) = \vec{d} \) is the vector representing a document \( d \).
Centroid: Examples
Rocchio algorithm

- Rocchio implements relevance feedback in the vector space model.

- Rocchio chooses the query vector $\tilde{q}_{opt}$ that maximizes

$$\tilde{q}_{opt} = \arg \max_{\tilde{q}} [\text{sim}(\tilde{q}, \mu(D_r)) - \text{sim}(\tilde{q}, \mu(D_{nr}))]$$

- where:
  - $D_r$: set of relevant docs;
  - $D_{nr}$: set of nonrelevant docs

- $\tilde{q}_{opt}$ is the vector separating relevant and nonrelevant docs maximally

$$\tilde{q}_{opt} = \mu(D_r) + [\mu(D_r) - \mu(D_{nr})]$$
The optimal query vector is:

\[ \tilde{q}_{opt} = \mu(D_r) + [\mu(D_r) - \mu(D_{nr})] = \]

\[= \frac{1}{|D_r|} \sum_{\tilde{d}_j \in D_r} \tilde{d}_j + \left[ \frac{1}{|D_r|} \sum_{\tilde{d}_j \in D_r} \tilde{d}_j - \frac{1}{|D_{nr}|} \sum_{\tilde{d}_j \in D_{nr}} \tilde{d}_j \right] \]

The centroid of the relevant documents is moved by the difference between the two centroids.
Exercise: Compute Rocchio vector

circles: relevant documents, Xs: nonrelevant documents
circles: relevant documents, Xs: nonrelevant documents
\( \vec{\mu}_R \): centroid of relevant documents; does not separate relevant/nonrelevant.
\( \vec{\mu}_{NR} \): centroid of nonrelevant documents \( \vec{\mu}_R - \vec{\mu}_{NR} \): difference vector
Add difference vector to \( \vec{\mu}_R \) ... to get \( \vec{q}_{opt} \)
\( \vec{q}_{opt} \) separates relevant/nonrelevant perfectly.
Terminology

- So far, we have used the name Rocchio for the theoretically better motivated original version of Rocchio.

- The implementation that is actually used in most cases is the SMART implementation – this SMART version of Rocchio is what we will refer to from now on.
Rocchio 1971 algorithm (SMART)

\[ \vec{q}_m = \alpha \vec{q}_0 + \beta \mu(D_r) - \gamma \mu(D_{nr}) = \alpha \vec{q}_0 + \beta \frac{1}{|D_r|} \sum_{\vec{d}_j \in D_r} \vec{d}_j - \gamma \frac{1}{|D_{nr}|} \sum_{\vec{d}_j \in D_{nr}} \vec{d}_j \]

where:

- \( q_m \): modified query vector;
- \( q_0 \): original query vector;
- \( D_r, D_{nr} \): sets of known relevant and nonrelevant documents resp.;
- \( \alpha, \beta, \) and \( \gamma \): weights

- \( q_m \) moves towards relevant and away from nonrelevant documents.
- Tradeoff \( \alpha \) vs. \( \beta/\gamma \): in case of many judged documents → higher \( \beta/\gamma \).

- Set negative term weights to 0.
- Negative term weight doesn’t make sense in vector space model.
Rocchio relevance feedback illustrated

Initial query

Revised query

X known non-relevant documents
O known relevant documents
Positive vs. negative relevance feedback

- Positive feedback is more valuable than negative feedback.
- E.g., set $\beta = 0.75, \gamma = 0.25$ to give higher weight to positive feedback.
- Many systems only allow positive feedback.
Relevance feedback: Assumptions

- When can relevance feedback enhance recall?

- Assumption A1: The user knows the terms in the collection well enough for an initial query.

- Assumption A2: Relevant documents contain similar terms (so I can “hop” from one relevant document to a different one when giving relevance feedback).
Violation of A1

- Assumption A1: The user knows the terms in the collection well enough for an initial query.

- Violation: Mismatch of searcher’s vocabulary and collection vocabulary

- Example: cosmonaut / astronaut
Violation of A2

- Assumption A2: Relevant documents are similar.
- Example for violation: [contradictory government policies]
- Several unrelated “prototypes”
  - Subsidies for tobacco farmers vs. anti-smoking campaigns
  - Aid for developing countries vs. high tariffs on imports from developing countries
- Relevance feedback on tobacco docs will not help with finding docs on developing countries.
Relevance feedback: Evaluation

- Pick an evaluation measure, e.g., precision in top 10: $P@10$
- Compute $P@10$ for original query $q_0$
- Compute $P@10$ for modified relevance feedback query $q_1$
- In most cases: $q_1$ is spectacularly better than $q_0$!
- Is this a fair evaluation?
Relevance feedback: Evaluation

- Fair evaluation must be on “residual” collection: docs not yet judged by user.

- Studies have shown that relevance feedback is successful when evaluated this way.

- Empirically, one round of relevance feedback is often very useful. Two rounds are marginally useful.
Evaluation: Caveat

- True evaluation of usefulness must compare to other methods taking the same amount of time.

- Alternative to relevance feedback: User revises and resubmits query.

- Users may prefer revision/resubmission to having to judge relevance of documents.

- There is no clear evidence that relevance feedback is the “best use” of the user’s time.
Exercise

- Do search engines use relevance feedback?
- Why?
Relevance feedback: Problems

- Relevance feedback is expensive.
  - Relevance feedback creates long modified queries.
  - Long queries are expensive to process.

- Users are reluctant to provide explicit feedback.

- It’s often hard to understand why a particular document was retrieved after applying relevance feedback.

- The search engine Excite had full relevance feedback at one point, but abandoned it later.
Other use of relevance feedback

- Maintaining a **standing query**

- Example: “multicore computer chips”

- I want to receive each morning a list of news articles published in the previous 24 hours on “multicore computer chips”.

- Relevance feedback can refine this standing query over time.

- Many spam filters offer a similar functionality.

- For standing queries, relevance feedback is more practical than in web search.
Pseudo-relevance feedback

- Pseudo-relevance feedback automates the “manual” part of true relevance feedback.

- Pseudo-relevance algorithm:
  - Retrieve a ranked list of hits for the user’s query
  - Assume that the top $k$ documents are relevant.
  - Do relevance feedback (e.g., Rocchio)

- Works very well on average

- But can go horribly wrong for some queries.

- Several iterations can cause query drift.
Pseudo-relevance feedback at TREC4

- Cornell SMART system

- Results show number of relevant documents out of top 100 for 50 queries (so total number of documents is 5000):

<table>
<thead>
<tr>
<th>method</th>
<th>number of relevant documents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inc.ltc</td>
<td>3210</td>
</tr>
<tr>
<td>Inc.ltc-PsRF</td>
<td>3634</td>
</tr>
<tr>
<td>Lnu.ltu</td>
<td>3709</td>
</tr>
<tr>
<td>Lnu.ltu-PsRF</td>
<td>4350</td>
</tr>
</tbody>
</table>

- Results contrast two length normalization schemes (L vs. l) and pseudo-relevance feedback (PsRF).

- The pseudo-relevance feedback method used added only 20 terms to the query (Rocchio will add many more).

- This demonstrates that pseudo-relevance feedback is effective on average.
Query expansion
Query expansion

- Query expansion is another method for increasing recall.

- We use “global query expansion” to refer to “global methods for query reformulation”.

- In global query expansion, the query is modified based on some global resource, i.e. a resource that is not query-dependent.

- Main information we use: (near-)synonymy

- A database that collects (near-)synonyms is called a thesaurus.

- We will look at two types of thesauri: manually created and automatically created.
Types of user feedback

- User gives feedback on documents.
  - More common in relevance feedback

- User gives feedback on words or phrases.
  - More common in query expansion
Types of query expansion resources

- Manual thesaurus (maintained by editors, e.g., PubMed)
- Automatically derived thesaurus (e.g., based on co-occurrence statistics)
- Query-equivalence based on query log mining (common on the web as in the “palm” example)
Thesaurus-based query expansion

- For each term \( t \) in the query, expand the query with words the thesaurus lists as semantically related with \( t \).

- Example from earlier: HOSPITAL \( \rightarrow \) MEDICAL

- Generally increases recall

- May decrease precision, particularly with ambiguous terms
  - INTEREST RATE \( \rightarrow \) INTEREST RATE FASCINATE

- Widely used in specialized search engines for science and engineering

- Expensive to create a manual thesaurus and to maintain it over time.

- A manual thesaurus has an effect roughly equivalent to annotation with a controlled vocabulary.
Example for manual thesaurus: PubMed

PubMed Query:

("neoplasms"[MeSH Terms] OR cancer[Text Word])
Automatic thesaurus generation

- Attempt to generate a thesaurus automatically by analyzing the distribution of words in documents

- Fundamental notion: similarity between two words

- Def 1: Two words are similar if they co-occur with similar words.
  - “car” ≈ “motorcycle” because both occur with “road”, “gas” and “license”, so they must be similar.

- Def 2: Two words are similar if the occur in a given grammatical relation with the same words.
  - You can harvest, peel, eat, prepare, etc. apples and pears, so apples and pears must be similar.

- Co-occurrence is more robust, grammatical relations are more accurate.
Co-occurrence-based thesaurus: Examples

<table>
<thead>
<tr>
<th>Word</th>
<th>Nearest neighbors</th>
</tr>
</thead>
<tbody>
<tr>
<td>absolutely</td>
<td>absurd whatsoever totally exactly nothing dip copper drops topped slide trimmed</td>
</tr>
<tr>
<td>bottomed</td>
<td>shimmer stunningly superbly plucky witty dog porch crawling beside downstairs</td>
</tr>
<tr>
<td>captivating</td>
<td>repellant lotion glossy sunscreen skin gel renciliation negotiate case conciliation</td>
</tr>
<tr>
<td>doghouse</td>
<td>hoping bring wiping could some would drawings Picasso Dali sculptures Gauguin</td>
</tr>
<tr>
<td>makeup</td>
<td>toxins bacteria organisms bacterial parasite grasp psyche truly clumsy naive innate</td>
</tr>
<tr>
<td>mediating</td>
<td></td>
</tr>
<tr>
<td>keeping</td>
<td></td>
</tr>
<tr>
<td>lithographs</td>
<td></td>
</tr>
<tr>
<td>pathogens</td>
<td></td>
</tr>
<tr>
<td>senses</td>
<td></td>
</tr>
</tbody>
</table>
Query expansion at search engines

- Main source of query expansion at search engines: query logs

- Example 1: After issuing the query [herbs], users frequently search for [herbal remedies].
  - “herbal remedies” is potential expansion of “herb”.

- Example 2: Users searching for [flower pix] frequently click on the URL photobucket.com/flower. Users searching for [flower clipart] frequently click on the same URL.
  - “flower clipart” / “flower pix” are potential expansions of each other.