

INLS 490-154: Information Retrieval Systems Design & Implementation. Spring 2009.

6. Retrieval models-2

Chirag Shah*

School of Information & Library Science (SILS)

UNC Chapel Hill NC 27599

chirag@unc.edu

1 Introduction

Once again, we will continue our exploration of various retrieval models. This time, we will look at probabilistic and relevance models. More importantly, we will see how relevance feedback (real or pseudo) can be incorporated in the retrieval process.

2 Probabilistic model in Okapi

Probabilistic models of IR are based on treating retrieval as a probabilistic inference. The key idea for matching and retrieval in such models is to evaluate the probability or estimation of a document being “relevant” to a given query. As we saw before, one way of evaluating this “relevance” is by computing the likelihood of a query generated by a given document. This is the *query likelihood model* (a kind of language model). While such a model is mathematically very appealing, we could do something *simpler*.

This is what (Robertson, Walker, Jones, Hancock-Beaulieu, & Gatford, 1994) did with their Okapi system. They came up with a new and improved function for ranking the documents, which was based on earlier work on probabilistic inferencing in IR (Robertson & Jones, 1976). Full details of this model is beyond the scope of this document; we will simply look at the derived formula for document ranking, which is given below.

$$\text{Similarity}(D, Q) = \sum_{i=1}^n TF(q_i, D) \cdot IDF(q_i) \quad (1)$$

$$= \sum_{i=1}^n \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1(1 - b + b \frac{|D|}{\text{avgl}})} \cdot \log \left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} \right) \quad (2)$$

*  These notes for INLS 490-154 Spring 2009 by Chirag Shah (<http://www.unc.edu/~chirags>) are licensed under a Creative Commons Attribution-NonCommercial-No Derivative Works 3.0 United States License.

$f(q_i, D)$ = frequency of query term q_i in document D
 $|D|$ = length of document D
 $avgdl$ = average document length in the collection
 N = total number of documents
 $n(q_i)$ = number of documents in which query term q_i occurs
 k_1, b = constants derived empirically

This formulation is called Okapi BM25 (or a variation of it) and has been one of the most successful term weighing and documents raking functions in IR.

Okapi is provided as one of the retrieval models in Lemur. Code 1 lists a parameter file for *RetEval* application that allows us to use Okapi scoring and raking function for retrieval.

Code 1: Parameter file for RetEval with Okapi retrieval model

```
<parameters>
  <index>myindex</index>
  <textQuery>query_parsed.txt</textQuery>
  <resultFile>results.txt</resultFile>
  <resultFormat>3col</resultFormat>
  <resultCount>100</resultCount>
  <retModel>okapi</retModel>
</parameters>
```

3 Relevance feedback

So far we have kept “relevance” undefined, and for a good purpose. “Relevance” is hard to define and often misunderstood as shown in several works. (Schamber, Eisenberg, & Nilan, 1990; Saracevic, 1996; Mizzaro, 1997, 1998; Borlund, 2003). However, we could consider some information as relevant without defining relevance, if the user says it is relevant! So let us ask the user what is relevant for him. Of course, we cannot show everything to the user and ask for his judgment on relevance. Here is what we can do.

1. Get a query from the user.
2. Run the query and perform retrieval.
3. Return the retrieved set or a part of it to the user and ask him to mark the documents that are relevant to him.
4. Based on the relevance judgments that we received from the user, revise the query and/or re-rank the documents. In case of revising the query, re-run the retrieval.
5. Present the new retrieval set to the user. Hopefully, this set will have more relevant information than that of the first set.

The above process is called obtaining (and using) *relevance feedback*. The hypothesis is - if the user can tell us what is relevant to him for a given query, we could use that information to bias the matching and ranking. While this is theoretically a good idea, there are some problems with this model as listed below.

1. User does not always know what is “relevant”. His misjudgment can make the second set of retrieved documents much worse than the first set.
2. Asking a user to give judgments is an expensive interaction.
3. Re-running a query, with possible expansion, incurs more cost on the system.

Often, systems do implicit feedback loop without actually asking the user. This is called *pseudo relevance feedback*. In this case, the system itself *decides* what could be relevant for the user and uses it as the feedback. For instance, the system can take the top 10 documents from the initial retrieval, assume that they all are relevant, and use this information to re-run the retrieval. At the user’s end such a re-run is not conveyed and so the user is not aware of implicit relevance feedback that took place within a system. The advantages of pseudo relevance feedback is that we could avoid expensive user interaction and that we do not have to worry about misjudgments by the user. The disadvantage, on the other hand, is that we are relying too much on the ranking of our first retrieval. If for some reason (bad retrieval algorithm, bad query or collection), our initial retrieval was not so good, we will end up harming the second run with pseudo feedback.

To use or not to use relevance feedback (real or pseudo) is a design decision.

4 Relevance feedback with Okapi

To incorporate the notion of relevance feedback, Robertson proposed a selection value, later called Robertson Selection Value (RSV) (Robertson et al., 1994). According to this structure, a term’s weight is redefined as the following

$$w(p - q) \tag{3}$$

Here, w is a term’s original weight, probably computing by something like Equation (2), p is the probability of that term occurring in a relevant document, and q is the probability of that term occurring in a non-relevant document. Once again, such a formulation helps biasing a term’s weight according to its relevance to the information need.

Without worrying about the details of this, let us now see how we can use this in Lemur. Since we will not have a real user for these exercise, we will use pseudo relevance feedback. First off, we will run the *RetEval* application to do regular retrieval. We will then feed its output as an input (feedback) to another application called *RelFBEval*. A sample parameter file for this application is given in Code 2.

Code 2: Parameter file for RelFBEval with Okapi retrieval model

```
<parameters>
  <index>myindex</index>
  <textQuery>query_parsed.txt</textQuery>
  <resultFile>results_fb.txt</resultFile>
  <resultFormat>3col</resultFormat>
  <resultCount>100</resultCount>
  <retModel>okapi</retModel>
  <feedbackDocuments>results.txt</feedbackDocuments>
  <feedbackDocCount>10</feedbackDocCount>
  <feedbackTermCount>20</feedbackTermCount>
</parameters>
```

Here, we are asking *RelFBEval* to consider top 20 terms from the top 10 retrieved documents as the feedback terms. *RelFBEval* revises the original query with these terms and re-runs the retrieval.

5 Relevance feedback with language models

In their seminal work (Lavrenko & W.B.Croft, 2001) showed how the probabilistic relevance feedback ideas can be mapped to language models. They used the findings from (Robertson, 1997) that *a document can be ranked by its odd of being observed in the relevant class*, and estimated it by the joint probability distribution of words in that document.

$$\frac{P(D|R)}{P(D|N)} = \prod_{w \in D} \frac{P(w|R)}{P(w|N)} \quad (4)$$

Here, R is the set of relevant documents and N is the set of non-relevant set of documents. Once again, we will leave out the details of the full derivation and talk about how to use relevance feedback with language models in Lemur. A sample parameter file for *RelFBEval* application is shown in Code 3. Notice that there is an additional parameter here - `<queryUpdateMethod>`. This allows us to specify which of the relevance models (1 or 2) we want to use. Details of these two kinds of relevance models can be found in (Lavrenko & W.B.Croft, 2001).

Code 3: Parameter file for RelFBEval with KL retrieval model

```
<parameters>
  <index>myindex</index>
  <textQuery>query_parsed.txt</textQuery>
  <resultFile>results_fb.txt</resultFile>
  <resultFormat>3col</resultFormat>
  <resultCount>100</resultCount>
  <retModel>kl</retModel>
  <feedbackDocuments>results.txt</feedbackDocuments>
```

```
<feedbackDocCount>10</feedbackDocCount>
<feedbackTermCount>20</feedbackTermCount>
<queryUpdateMethod>rm1</queryUpdateMethod>
</parameters>
```

6 Summary

- We saw probabilistic model and relevance model for retrieval.
- Okapi BM25 is one of the most effective scoring and ranking functions in IR.
- Feedback (real or pseudo) can help in improving the retrieval performance.
- User interaction is expensive. Pseudo-relevance feedback often gives good results, but with added expense.
- To use relevance feedback (real or pseudo) or not, and which model to use are design choices.

References

- Borlund, P. (2003). The concept of relevance in IR. *Journal of the American Society for Information Science and Technology*, 54(10), 913–925.
- Lavrenko, V., & W.B.Croft. (2001). Relevance-based language models. In *Proceedings of the 24th annual international ACM SIGIR conference* (p. 120-127).
- Mizzaro, S. (1997). Relevance: The whole history. *Journal of the American Society for Information Science*, 48, 810–832.
- Mizzaro, S. (1998). How many relevances in information retrieval? *Interacting with Computers*, 10, 303–320.
- Robertson, S. E. (1997). The probability ranking principle in IR. In (pp. 281–286). San Francisco, California: Morgan Kaufmann Publishers, Inc.
- Robertson, S. E., & Jones, K. S. (1976). Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27, 129–146.
- Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M. M., & Gatford, M. (1994, November). Okapi at TREC-3. In *Proceedings of the Third Text REtrieval Conference (TREC)*.
- Saracevic, T. (1996). Relevance reconsidered. In *Proceedings of colis 2* (pp. 201–218).
- Schamber, L., Eisenberg, M. B., & Nilan, M. S. (1990). A re-examination of relevance: Toward a dynamic, situational definition. *Information Processing and Management*, 26, 755–776.