CS630 Lecture Notes

Lecturer: Lillian Lee
Scribes: Chris Danis (cgd3) & Brian Rogan (bcr6)

Lecture 11: 2 March 2006

1 Introduction / recap

Today we will finish our coverage of explicit relevance feedback; later, we will move on to implicit relevance feedback.

Relevance feedback (RF) is a process that involves the user "judging" documents as to whether they are relevant or not after submitting an initial query to the system. The system can use this relevance information for term reweighting and automatic query expansion (e.g. automatically noticing that "car" and "auto" are the same concept). We will explore a few current RF research issues.[1]

Query expansion can be very confusing to the user if the system strangely or wrongly interprets the relevance feedback. There is a technique called "interactive query expansion" or IQE, which uses an automatic query expansion (AQE) technique to rank terms, and suggests new query terms to the user[2]. Advantages: IQE allows more user control, it makes better use of the user's knowledge, and the system can apply relevance data to the suggested query expansions. Disadvantages: IQE adds extra work for the user, and the user can make poor choices for new terms, without realizing said choices were poor[3] .

[1] A good survey of RF methods (and of much of IR itself) can be found in [2].

[2] Compare with Google's "Did you mean..." feature.

[3] Not only can terms be misunderstood, but there may very well be a disparity between the terms a user would use to describe a concept, and the terms that will make the system successful in retrieving documents about that concept.

2 AQE vs IQE

Fowkes and Beaulieu, in their 2000 paper [3], studied when people prefer AQE versus IQE. In their study, they defined "easy" and "hard" search tasks (e.g., "Find Cornell's homepage" versus "Find when Lillian Lee was born"). They then checked users' preferences for AQE versus IQE for both classes of search tasks. Unsurprisingly, people preferred AQE for the easy tasks and IQE for the hard tasks. For the easy tasks, automatic query expansion works reasonably well, and interactive query expansion is too much trouble. For the hard tasks, AQE may not work well, and the IQE affords the user some extra control.

Ruthven's 2003 paper (best paper at SIGIR '03, [1]) instead did a competitive analysis of AQE and IQE. The driving idea was to compare performance between a fairly good AQE system versus the *best possible* performance with an IQE version. The study is heavily biased in favor of IQE; however, the findings were that only a very small subset of potential IQE queries would do better than the best corresponding AQE query[4]. This is, of course, somewhat surprising.

Thus, given that users have a low chance of selecting a good query at random, the next question was: can users find those results deliberately? Ruthven generated expansion terms from the top 25 relevant documents for each of the initial queries, ranked those expansion terms by how much they helped performance on average, and then asked *users* to rate those expansion terms. This stage of the study was de-

[4] Against the "most realistic" default AQE approach, 9 to 12% of the possible IQE variants would have yielded significant improvement.

signed to simulate an IQE system providing expansion terms, the difference being that Ruthven knew which expansion terms were the best and sought to find whether users of the system could determine which terms would lead to the best performance. Depending on the person, 32-73% of the good expansion terms were identified; however, 26-54% of the terms that decreased performance were identified as "good" by users. Again, this shows a disparity between terms that represent the user's idea of a topic and terms that will make the system perform well and retrieve documents about that topic.

3 Active RF

A very recent topic in RF is "active RF"[5] (Shen & Zhai SIGIR '05, [4]). Instead of just presenting the top k documents to the user for judging, try presenting some other set of documents and getting feedback on those; maybe we can do something "spiffier". How do we decide which documents to present for judgment? Top-k doesn't necessarily provide the user with a diverse set of documents: they can all be not relevant, or possibly can all be the *same* relevant document. Non-top-k has a greater chance of non-relevance, thus degrading the user experience; also, systems are often not able to do much with only negative feedback, which is a possibility if non-top-k presents only non-relevant documents to the user.

Shen and Zhai evaluated several different approaches for picking other sets of documents to be judged by the user, including: "gapped" top-k documents (i.e., for a gap of n, pick every nth document in your ranked list, until you have k documents), and "cluster centroid": take the top n documents, throw away your rankings, and try to cluster them within some space[6] . After clustering, pick a "representative" document from each cluster and present those documents to the user for judging. Gapped top-k may return a diverse set; cluster centroid seems like it probably will (for reasonable n).

[5]"Active" is a reference to "active learning," a machine learning technique.

[6]Clustering can be expensive, but depending on our scheme, we may be able to do this offline.

Overall, their approaches got *better* performance with fewer documents judged by the user – more payoff for less work!

4 Evaluation?

How do we evaluate performance of RF systems[7]? We can't just use precision/recall at the end of the process – this allows a sort of "cheating", as we can simply place the documents the user told us were relevant at the top of the list. (Term reweighting will tend to do this, but certainly won't always do this.) An easy way to prevent this type of cheating is to remove the documents a user has judged from the rankings under evaluation. This is called "residual ranking"; unfortunately, it is problematic. Residual ranking penalizes systems that perform very well early on in the RF process (especially those that perform well before doing any re-ranking at all). Also, we can't compare performance before/after, as we aren't comparing performance on the same corpus! This last problem has an easy fix – throw out the judged documents when you compute the metrics for the "before" case.

Still, residual ranking seems troublesome. There's an alternative: "freezing" of judged documents' ranks in the result list. The problem with freezing is that it penalizes systems that are "slow" (but eventually effective) learners.

Another approach is "split data", wherein we split the data into two disjoint sets. Set 1 is for query modification and producing the documents for RF; set 2 is where you actually do the evaluation of performance. The problem with this is we're looking at unseen documents; we probably should be looking at unseen queries, but query transference is a very delicate issue. The other problem is that this approach doesn't simulate the user experience for RF.

Unfortunately, these ideas are practically all the ideas for evaluation that the field has had. There's something odd about the RF setting that's difficult to capture in an evaluation scheme.

We're looking for a balance – it seems like we could really help our performance with relevance feedback,

[7]This section follows [5] as presented in [2].

but it's expensive and troublesome to get. Implicit RF tries to strike this balance, and will be covered next lecture.

References

[1] I. Ruthven. Re-examining the potential effectiveness of interactive query expansion. In *Proceedings of the Twenty-Sixth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.*, pages 213–220, Toronto, 2003. ACM.

[2] I. Ruthven and M. Lalmas. A survey on the use of relevance feedback for information access systems. *The Knowledge Engineering Review*, 18(2):95–145, June 2003.

[3] M. Beaulieu and H. Fowkes. Interactive searching behavior: Okapi experiment for TREC-8. In *Proceedings of the BCS-IRSG 22nd Annual Colloquium on IR*, Cambridge, UK, 2000.

[4] X. Shen and CX Zhai. Active feedback in ad-hoc information retrieval. In *Proceedings of the 28th annual ACM SIGIR*, 2005.

[5] Y.K. Chang, C. Cirillo, and J. Razon. *Evaluation of feedback retrieval using modified freezing, residual collection, and test and control groups. In The SMART Retrival System – Experiments in Automatic Document Processing (G. Salton, ed.)*, chapter 17, pages 355–370. Prentice-Hall, 1971.

CS630 Lecture Practice Problems

Lecturer: Lillian Lee
Scribes: Chris Danis (cgd3) & Brian Rogan (bcr6)

Lecture 11: 2 March 2006

1. We began the class by suggesting some reasons that we might prefer IQE (interactive query expansion) to AQE (automatic query expansion) and vice versa. Compare the benefits of both techniques.

   ANSWER:
   Recall that the primary advantage of IQE was that the user has extensive control over the actual query that is entered. There is a strong argument to favor IQE because ultimately the users are the judges of which documents are relevant to their query, so they should decide what terms are relevant to that query. Also, users understand semantically what they are looking for: they understand how different query terms are related with respect to meaning. The primary argument in support of AQE rests on the fact that the system has far more knowledge of the corpus than the user, and it may be that the query-terms which produce the most relevant documents are not those that the user would use to describe their query. There is a careful distinction here between terms which the user finds useful to describe what he is looking for, and the terms correlated with relevance in the corpus. They may be quite different.

2. Ruthven argues in the conclusion to his paper that AQE is not necessarily implicitly more effective than IQE, but that in order to perform effective IQE users need more "support", i.e. the system should display the relationship between potential expansion terms and relevant material before the user decides which expansions to select. How does an approach like this complement the strengths of IQE in general?

   ANSWER:
   One of the main arguments for the use of IQE is that users have a semantic understanding of the relationship between query terms and material they are searching for that the system cannot replicate. Thus users see terms as connected if this connection makes sense with respect to the meanings of the terms used. On the other hand, one of the the primary attractions of AQE is the fact that the system has a statistical understanding of the relationships between query terms and relevance. Thus a system

which could succinctly and clearly display the statistical relationship that has been inferred between an expansion and relevance would allow the user to weigh the semantic relationship to his query with the statistical relationship to relevance in the corpus.

3. Recall that in lecture we discussed several schemes for presenting items of varied relevance to the user for review, including gapped top-k and cluster-centroid.

   (a) Does either scheme give us a guarantee that it will produce documents of varied relevance? Does this matter?

   ANSWER:
   Strictly speaking, neither scheme seems guaranteed to give us clusters in "relevance-space." A clustering scheme would likely be based on some metrics that are more easy to measure, and at best might produce something that includes different topics. This is, in a way, more useful then things of varied relevance: knowing that a user is interested in a certain "topic," where that topic corresponds to a certain sub-collection of documents, allows us to return a list of documents that is likely on topic. Gapped top-k does guaranteed that we will output documents with (what the system believes is) varied relevance. It may be that even with a large gap, top documents are still very similar.

   [Editor's Note: One might take a look at the work on Vivisimo or the older Scatter/Gather work, if interested in this]

   (b) Suppose, for some very general query, we use a clustering scheme to decide which documents to display. If we have a large number of clusters (larger than can be displayed to the user), it is not entirely clear which clusters we should show the user. How can we rank clusters by relevance? How can we decide which to display? Propose a scheme, and discuss its merits and drawbacks.

   ANSWER:
   It is not entirely clear how to rank clusters by presumed relevance. Average relevance of the documents contained in the cluster is an obvious choice, but this means that a cluster with many relevant documents, but some documents rated as very irrelevant may do poorly (the average is not a robust metric). Also, this metric does not take into account the size of the cluster: presumably the size of the cluster provides some indication of its relevance (if we accept that topics about which much have been written seem to have a higher probability of relevance a-priori). One could imagine using some sort of size based normalization to correct this problem (if it is found to actually be one).

As for which clusters to display, this is (perhaps upsettingly) just a restatement of the original problem. Instead of deciding what are good documents to display, we're trying to decide what clusters (or cluster representative documents) to display. A scheme like gapped top-k might be interesting, although it is not clear that it will be useful in returning documents of different relevances. It may be that a simple top-k scheme is appropriate under these circumstances. This is a question without an obvious answer, and it is left to the reader to weigh the pros and cons.

4. Recall that in lecture we discussed a variety of evaluation schemes for RF based systems. We discussed residual-ranking, freezing, and the split-data approach but found none to be particularly satisfying. For each of the following ideas, list some benefits and drawbacks:

   (a) A system that used split-data without using disjoint sets (i.e. certain documents could be in the RF set and also the evaluation set)

   ANSWER:
   This scheme requires less data to get the same-sized RF and evaluation sets. This scheme does help to mitigate the problem of cheating somewhat by removing some (possibly many) of the documents with user-relevance labeling, but it does not address the problem which we selected split-data sets to avoid. There is nothing to stop a system from cheating by still putting those documents with high user-judged relevance first.

   (b) A scheme like residual-ranking, but instead of removing documents which were in the RF set, we simply provide them less weight in our performance metric, so that simply listing all the user-supplied relevant documents first is not sufficient to provide a high score. Documents found to be relevant after using RF must also be judged highly.

   ANSWER:
   This scheme does provide less of an incentive for cheating: there is less value to putting pre-judged documents first, and more incentive for making good (new) predictions of what the user will find relevant. However, there becomes an incentive to try to work the evaluation system: rather than strictly try to return the most relevant documents, the evaluation scheme encourages a system to present new documents, even if documents which have already been judged by the user truly are the most relevant. Furthermore, this scheme does not correct the defect that it over-penalizes systems which give good pre-RF performance.

   (c) A scheme which involves soft-freezing, that is, documents are not actually frozen in their original positions but those positions are re-

membered, and the systems' cumulative score comes from taking the average (or possibly weighted average) of its performance across all RF iterations.

ANSWER:
This scheme is admittedly a bit more open-ended then the ones which preceded it, because based on the weighted averaging function you use, you can bias the system in various ways. The system is largely biased towards fast learners: a weighted average which takes earlier iterations to count for more is certainly biased that way. Even a strict average is biased towards fast learners because good systems will quickly discern which documents are best, and will have higher scores for more iterations. That being said, the system is not without appeal: if the weighted average is chosen to weight later iterations higher, some of the fast-learner bias can be erased, but it may still exist (fast-learners still spend more iterations with higher scores). It is not even clear though that this bias is entirely undesirable: as long as both systems can obtain the same performance on their final iteration, fast learning systems are more desirable. With a late iteration weight bias, systems that eventually produce very good results will still receive high scores.