

These are the solutions to the exam questions. The exam itself is repeated here; solutions are in this font.

This is an in-class midterm exam. You have 75 minutes to complete it. The exam is open-book. You may use class notes, textbooks, papers, or other written resources. You may not use any device that is capable of being connected to the internet.

This exam includes four (4) questions on one (1) page. Good luck.

Problem A. (10 points)

Assume that you have an evaluation program that calculates average precision by looking only at the top 1,000 documents in the ranked list. Any relevant documents not found in the top 1,000 are assumed to occur at rank infinity—i.e., precision for that relevant document will be zero. (This is the way that average precision is calculated at TREC.)

Assume that a query is run against a collection of one million (1,000,000) documents and that of the 10 relevant documents, only 6 were found in the top 1,000. For that query, what is the largest and smallest possible error in average precision caused by that assumption?

You do not have to compute the actual number, provided you show an equation that will result in the correct number if calculated. (For example, if the answer were found by adding 4 and 5 [it isn't], you could either answer "4+5" or "9" and either would be counted as correct.)

Let $\{P_i\}_{i=1}^{10}$ be precision at the 10 points of recall (that is, the precision at the 10 positions in the ranked list where recall changes: where relevant documents are retrieved). The Mean Average Precision (MAP) for the entire ranked list consisting of 1,000,000 documents is

$$MAP = \frac{1}{10} \sum_{i=1}^{10} P_i$$

$$MAP = \frac{1}{10} \sum_{i=1}^6 P_i + \frac{1}{10} \sum_{i=7}^{10} P_i$$

When only 1000 documents are retrieved, since only six documents occur in the top 1000, the second term on the right hand side of the above equation is zero. Hence the error is

$$Error = \frac{1}{10} \sum_{i=7}^{10} P_i$$

Maximum error occurs when the 4 relevant documents that were not in the 1000 actually occur at ranks 1001 to 1004. That is, $P_7=7/1001$, $P_8=8/1002$, $P_9=9/1003$, and $P_{10}=10/1004$.

Minimum error occurs when the 4 relevant documents that have not been retrieved occur at the last four ranks in the collection. That is: $P_{10}=10/10^6$, $P_9=9/(10^6-1)$, $P_8=8/(10^6-2)$, and $P_7=7/(10^6-3)$.

As long as you got this idea you received full credit. No calculations were expected.

Precision at 1 is unaffected, as we know that the first relevant document was retrieved in the top 1000 documents.

Problem B. (20 points)

Borrowing ideas from query expansion (e.g., LCA), statistical stemming, and LSI, explain how you might use clustering of terms to address the term independence problem. (You may certainly borrow ideas from other areas, too.) How likely do you think it is that such a system will improve effectiveness? Why do you believe that?

This problem was much more difficult than intended. The expected idea was that a system could look at word co-occurrences to find dependencies between words (using, e.g., EMIM) and use that as a basis of clustering the words into highly-dependent groups of terms. Note that this type of clustering is the same thing done in statistical stemming. A nice modification would be to do the analysis on the top-ranked documents or passages (à la LCA) and find dependencies on-the-fly: it'd put more cost on the system at query time, but might give better results.

At any rate, the clusters could be used in several ways. They could be used in the same way that stem classes are used: any term would be replaced by its group. Or they could be used to form the basis of probability estimates with dependencies as in the sentence forest approach mentioned in class.

Several people suggested that they could use the term classes created implicitly by LSI. That's plausible, though it doesn't borrow any of the other ideas, so it didn't receive full credit. You'd also have to do some additional processing since LSI doesn't really cluster the terms: it represents each term as a linear combination of the basis vectors.

How likely would it be to work? (This was worth 6 of the points.) You could answer either way and get credit provided you supported your answer. The "best" answer was to say that you didn't think it would work since decades of work have yet to find a way to make explicit representation of dependencies help for IR.

Problem C. (15 points)

How does (1) stopping and (2) stemming reduce the size of inverted lists as stored on disk? Be as specific as you can.

Part (1) was intended to be very simple, so was worth only 3 of the 15 points. Stopwords are removed from the inverted file, so their inverted lists do not exist, so the file will necessarily be smaller.

A correct answer to part (2) focused on compression—that's why the question asked about storing them on disk and not in memory (though compression can be used there, too). Some people pointed out that there is a small savings that comes from having slightly less bookkeeping information to store when you merge lists because of stemming. Also, if no position information is stored, there would be some savings from overlapping document information. Those were worth a point or three, depending what you said. The major savings comes when lists are merged, meaning that "words" (stems) occur more often in a given document so delta encoding results in many smaller numbers and that means that the lists will compress better, and that means that the inverted file will be smaller.

Problem D. (15 points)

1. What is a probable problem with query expansion (i.e., assumed-relevance feedback) in a signature-file system?

There were two major points that you should be making (most people just picked up the first one):

- a. Retrieval in a signature file system has the problem of false alarms or false matches. Using such false alarm documents for pseudo relevance feedback will cause problems
- b. Query terms are ANDed in a signature file system which has a big impact on query expansion. If a query contained the terms A and B, a document with only the term A would never be retrieved. Expanding the *query* will never add B to the

document, so it will continue to not be retrieved.

2. Describe a situation in which the zero-frequency problem would arise (the estimation slides in the language modeling lecture). Give a *specific* example of how it might occur.

Suppose you are using Maximum Likelihood Estimation in the language modeling framework. The probability of a document D is:

$$P(Q | D) = P(w_1 | D)P(w_2 | D)\dots P(w_i | D)\dots P(w_d | D)$$

If some term w_i in the query does not occur in D , then $P(w_i | D) = 0$ and by the above, $P(Q | D) = 0$, even if D contained every other query term.

3. It is necessary for a distributed system to have the same stemming algorithm at every collection's site? Why or why not?

No, it is not necessary. Just think of it as two different systems in a distributed system whose scores are not comparable. You can use various kinds of score normalization techniques to merge the scores if required.