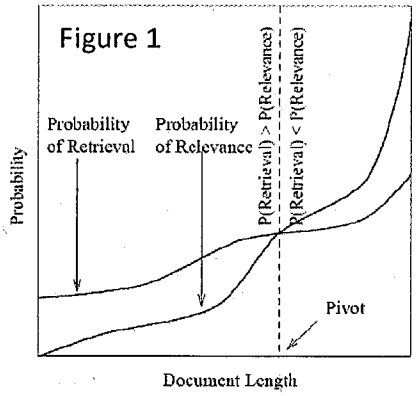
101. 3. 9

資訊網路與多媒體研究所

資格考試科目：資訊檢索與擷取

1. (15 pts) Many retrieval systems use vector space model (VSM) with tf-idf weighting to determine the similarity between documents and queries.
   (a) (5 pts) Explain why the VSM model needs document normalization to normalize term frequency (tf)?
   (b) (10 pts) One way to do normalization is cosine normalization. Figure 1 shows the relationship between document length and probability of retrieval and relevance when cosine normalization is performed. Based on Figure 1, please describe (i) the problem of cosine normalization and (ii) how to fix the problem.



2. (30 pts) Language model (LM) is to estimate the probability of a sequence of words.
   (a) (5 pts) Briefly describe how to apply LM to the ranking problem in IR.
   (b) (10 pts) Specify "maximum likelihood" estimation and "maximum a posteriori" estimation, respectively, and then compare their difference.
   (c) (5 pts) Describe how to calculate the probability of a word w, i.e. $p(w|\theta_D)$, if maximum likelihood estimation is applied to estimate the multinomial document language model.
   (d) (10 pts) Most of existing IR work used maximum likelihood estimation when LM is applied. Please give two applications for "maximum a posteriori" estimation. Explain your answers.

3. (20 pts) F-measure, Mean Average Precision (MAP), and Precision at k (P@k) are three measures for performance evaluation in IR.
   (a) (5 pts) Please specify MAP.
   (b) (15 pts) For each of the three measures, please give an application so that only that measure is appropriate for evaluation but the others are not. Explain your answers.

4. (15 pts) Relevance feedback provides retrieval systems useful information about "what are relevant or not".
   (a) (5 pts) Please give a situation that pseudo relevance feedback might produce worse results.
   (b) (10 pts) Describe the differences between vector space relevance feedback and probabilistic relevance feedback.

5. (20 pts) Suppose a retrieval system collects which documents in the search results (with respect to some query) are viewed and which documents are skipped by a user.

So its query log looks like:

(query1, document1, skipped)
(query1, document2, viewed)
...
(query2, document1, skipped)
(query2, document3, viewed)
...
(query3, document2, skipped)
(query3, document3, viewed)
(query3, document4, skipped)
...

Given query q, if q has appeared in the log, a common method is to simply rank the documents based on how frequently they are viewed for q. However, such method has three major problems: (i) it may return a few results for low-frequency queries, i.e., those queries only appearing one or two times with a few clicks, (ii) it may return no results for new queries, i.e., those queries not appearing in the log, and (iii) it doesn't use the "skipped" information kept in the log, i.e., some documents might be irrelevant to q but ranked high. Please propose three methods to solve the problems, respectively. Explain how and why your methods work.