National University of Singapore
School of Computing
CS3245: Information Retrieval
Tutorial 4A/B

**The Vector Space Model and Complete Search Systems**

**Contents:**
Tutorial 4A: Vector Space Model (Questions 1 and 2)

Tutorial 4B: Complete Search System (Questions 3, 4 and 5)

**Readings:** IIR Chapters 6 & 7

1. **TF×IDF.** Term frequency and inverse document frequency are the most well-known heuristics for grading document relevance that everyone who knows a little bit of IR should know. Let's look at this formula in more detail.

   (a) Look at Figure 6.15 in the textbook (pg. 118). Describe the minimum and maximum values for each of the term frequency measure variants. Do the same for document frequency variants. Do you have a preference for which formulation you'd like to use, from an intuitive viewpoint or from an efficiency viewpoint? (Hint: There's no one right answer for this).

   (b) Calculate the $tf \times idf$ values for the following terms, using the simple logarithmic variant of $tf$ and $idf$ (the second line in the table). Assume the document collection consists of 1,000,000 documents.

| | df | doc 1 | 2 | 3 |
|---|---|---|---|---|
| national | 500 | 3 | 2 | 22 |
| university | 700 | 3 | 0 | 10 |
| of | 750,000 | 3 | 10 | 12 |
| singapore | 100 | 3 | 5 | 7 |

   (c) Describe the effect that stemming has on ranking documents when a query word matches a common stem. (Hint: perhaps it's easier to think in concrete terms. Say we have two terms $a$ and $b$ with document frequencies $df_a$ and $df_b$, have a common stem. What happens to documents that contain $a$ if a query like $a$ $c$ is given, in IR engine that uses stemming versus an IR engine that doesn't?)

   (d) In Malay (*Bahasa Melayu* and *Bahasa Indonesia* among others), plurals are sometimes formed by duplicating a word (cat = "kucing", cats = "kucing-kucing"; book = "buku", books = "buku-buku")[1]. Discuss how this affects $tf$ if the duplication is treated as two separate words, and whether such an outcome is desirable or not.

2. **Vector Space Model.** The VSM framework gives a nice mathematical model for ranking documents, ranking documents by virtue of their similarity in direction to a query.

[1] Duplication can also be used to give emphasis or yield related or different meaning; varies per word.

(a) Calculate the relevance of Documents 1, 2 and 3 (from Question 1, part (b)) to the query "singapore" and to the query "national university". Show all work.

(b) So far we have just discussed vectors that have positive quantities (e.g., if a term $a$ occurs in a document, its component in the dimension representing $a$ is some positive quantity, may be proportional to $a$'s $idf$). Are there any reasons to use negative quantities? What effect would that have on the bounds of cosine scoring?

(c) Given a query $q$, in the vector space model, what would the highest possibly ranked document look like (e.g., a document with cosine score 1 with respect to $q$)? Do you think that such a document would be the best document to return to the user for this search? Why or why not?

(d) Let's examine two terms $t_1$ (say "automobile") and $t_2$ ("car"), which are largely synonymous. Does the VSM handle such types of synonymy?

3. **Champion, High-Low Lists and Tiered Indices.** These three mechanisms can be regarded as the same idea, just cast with different names. The general idea is to segregate productive (high-scoring) postings from less productive ones.

(a) How do Champion Lists relate to Index Elimination? Can they be used together?

(b) Your friend, Varth Dader, sees you struggling with your IR project, trying to implement champion lists for individual terms. He says there is an easy way out; that is to just take (up to) the first 100 document IDs for a particular term and treat them as a champion list. What do you think?

(c) Your other friend, Hakashi Katake, sees you later on, still struggling with your IR project, trying to implement tiered indices (pg. 132 in your textbook). He says there is an easy way out; rather than use strict $tf$ thresholds, you should first sort the postings by $tf$ and put the first top $n$ on the Tier 1 index, the second $2n$ on the Tier 2 index, the third $4n$ on the Tier 3 index, etc. What do you think?

4. **Query Term Proximity.** Our textbook describes query term proximity as an important consideration for ranking in free text search.

(a) Given two documents with the same $tf$ for a term, describe how (query) term proximity might be used to order these two documents in a postings list.

(b) The book describes simulated query term proximity weighting by using multiple queries where only phrasal search is supported (the steps on pg. 134). Describe how we could change the our scoring method to consider query proximity directly (A sketch of the algorithm that describes how positional information from multiple terms is used is sufficient).

5. **Zones and Fields.** In zone and field indexing, we can store the zone information either at the dictionary level (by having different terms for each zone; e.g., *term.title* and *term.body*) or at the postings level (by storing which zone the occurrence appears in within the document; e.g., *doc*1.*title*, *doc*2.*body*). Give arguments in favor of both possibilities. Which do you think is better in your personal opinion? (Again, there's no particular right or wrong answer)