National University of Computer & Emerging Sciences

Midterm Examination – Fall 2016 sol

Course: IR&TM (CS567) Time Allowed: 1 Hour
Date: September 24, 2016 Max. Marks: 40

Instructions: Attempt all question. Be to the point. Draw neat and clean diagram/code where necessary. Answer each question on the new page of the answer book, no marks for junk explanations. You must address all inquires in a question.

Question No. 1 **[Time: 25 Min] [Marks: 20]**

Answer the following questions briefly using 4-5 lines of answer book. Be precise, accurate and to the point, only answer genuine query in the question. Each question is of 2 marks.

a. What are some of the drawbacks of Vector-Space Model (VSM)?

- Order of the term from the document is lost when it is represented in VSM
- All terms are treated statistically independent to each other, human languages have a context and the use of words are generally based on the same context.
- Suffers from synonym and polysemy, documents that contains different vocabulary for the same context are treated as different documents.

b. It is pointed out in the text-book that the vocabulary terms in the postings are lexicographically ordered. Why is this ordering useful?

Vocabulary terms in the posting list should be lexicographically ordered, this can be very useful in efficiently computing intersection operation between 2 or more posting lists, if the posting list is ordered the processing will take linear time. This ordering is also beneficial for VSM as it offer standard "standard space" for representing vectors.

c. Illustrate at least three problems while performing "Tokenization" of documents.

Tokenization is the process of breaking a stream of text up into words, phrases, symbols, or other meaningful elements called tokens. It is very challenging aspect of information retrieval, the tokenization process need to decide about a lot of different aspect of a natural language like:

1. Direction of parsing for tokenization.
2. Should it treat space as separator for token e.g. Les Vegas a single or two tokens.
3. Treatment of punctuation characters like hyphen(-) co-ordinated.

4. If there is no space in between word boundary, how it will decide about tokens like in Japanese or German compound nouns.

d. What do we mean by token normalization? How it is done? Give a simple example.

Given a character sequence and a defined document unit, tokenization is the task of chopping it up into pieces, called tokens. Token normalization is the process of canonicalizing tokens so that matches occur despite superficial differences in the character sequences of the tokens. The most standard way to normalize is to implicitly create equivalence classes, which are normally named after one member of the set.

Example: operator, operation, operational all mapped to a root word "operate"

e. What is Truecasing? What are its benefits in IR?

Truecasing is the process of restoring case information to badly-cased or non-cased text, for index in IR systems. It is the process of restoring case information to raw text. The advantages of Truecasing is rEadaBiLITY and it also enhances the quality of case-carrying data.

f. Why sometimes post-processing is required when wild-card queries are processed with k-gram indexing? Give an example.

Consider using the 3-gram index for the query red*. The process first issue the Boolean query $re AND red to the 3-gram index. This leads to a match on terms such as retired, which contain the conjunction of the two 3-grams $re and red, yet do not match the original wildcard query red*. A post-filtering step, in which the terms enumerated by the Boolean query on the 3-gram index are checked individually against the original query red* to find the actual documents.

g. Define what do we mean by trailing wildcard query? Give an example. Suggest a data structures that best handle such queries.

A query such as mon* is known as a trailing wildcard query, because the * symbol occurs only once, at the end of the search string. A search tree on the dictionary is a convenient way of handling trailing wildcard queries.

h. Write down the entries in the permuterm index dictionary that are generated by the term "tata".

The entries in permuterm index for term “tata” will be as follow:

tata$, ata$t, tata, atat, $tata.

i. A bigram index is used to retrieve document for wildcard query “te*ti*al”. What Boolean query on a bigram index would be generated for this query? Give an example of term that may contain in the result-set.

The Boolean query using bi-grams will be

$t AND te AND ti AND al AND l$

Example is: testimonial

j. What is the limitation of isolated term correction approach to spelling correction?

In isolated-term correction, we attempt to correct each term independent of the actual number of query terms. For example query like "flew form Heathrow" would be try to correct each term "flew" "form" and "Heathrow". This type of correction is applicable to all lexical terms. For example a single query term "carot" can easily be corrected to "carrot". There are two techniques for this method (i) Edit distance and (ii) k-gram overlap.

Question No. 2	**[Time: 15 Min] [Marks: 10]**

The Boolean retrieval model has several drawbacks, one is the query formulation, which is not straight forward. Below are some of the information needs from user's, given in plain English, you need to transform these into Boolean Model queries. Using the term statistics from the collection. Suggest a best order for your queries processing, using your knowledge on inverted index and related concepts.

Some of the term statistics from the dictionary / inverted indexes are given:

Benefit	167	Hot	319	Reduce	980
Cholesterol	6430	Information	23100	Seeds	12323
Cold	767	Local	23900	Success	23451
Eat	37623	Oats	12398	Swim	32414
Effective	231653	Optimize	879	Technique	387912
Exam	21345	Prepare	1213	Water	1998721
Flax	3124	Rate	982345		
Global	21342				

Q1: Eating flax seeds is more effective at reducing cholesterol than eating oats.

Eat AND Flax AND Seeds AND Effective AND Reduce AND Cholesterol AND Oats
Posting sizes: (37623) ∩ (3124) ∩ (12323) ∩ (231653) ∩ (980) ∩ (6430) ∩ (12398)
The most effective query processing order will be:
Reduce AND Flax AND Cholesterol AND Seeds AND Oats AND Eat AND Effective

Q2: Swimming benefits in hot water or cold water

(Swim AND Benefit AND Water) AND (Hot OR Cold)
Posting sizes: (32414) ∩ (167) ∩ (1998721) ∩ (319 U 767)
The most effective query processing order will be:
Benefits AND (Hot OR Cold) AND Swim AND Water

Q3: success rate of exam preparation techniques

Success AND Rate AND Exam AND Prepare AND Technique
Posting sizes: (23451) ∩ (982345) ∩ (21345) ∩ 1213 ∩ 387912
The most effective query processing order will be:
Prepare AND Exam AND Success AND Technique AND Rate

Q4: local or global optimization techniques

(Local OR global) AND Optimize AND Technique
Posting sizes: (23900 U 21342) ∩ 879 ∩ 387912
The most effective query processing order will be:
Optimize AND (Local OR global) AND Technique

Question No. 3	**[Time: 15 Min] [Marks: 10]**

Consider a corpus of three documents, which comprises of Vocabulary = {w_1 w_2 w_3 w_4 w_5}, assume that the subscript dictate order of dimension:

d_1= {w_1 w_2 w_3 w_4 w_3 w_2 w_1 w_5}
d_2= {w_3 w_2 w_1 w_4}
d_3= {w_1 w_2}
q = {w_1 w_2 w_1}

Assume that we use TF= 1+ log ($tf_{t,d}$) for computing the term frequency of term t in document d. The Inverse document frequency (IDF) is define as log (N/df_t), where N=3 you can use TF*IDF to represent every term in VSM. Identify the ranking order of the documents with respect to query q.

Consider the term document matrix for the given case:

Words	d_1	d_2	d_3	q
W_1	2	1	1	2
W_2	2	1	1	1
W_3	2	1	0	0
W_4	1	1	0	0
W_5	1	0	0	0

Words	Tf d1	Tf d2	Tf d3	Tfq	IDF	Tf *idf d1	Tf *idf d2	Tf *idf d3	Tf *idf q
W_1	1.3	1	1	1.3	0	0	0	0	0
W_2	1.3	1	1	1	0	0	0	0	0
W_3	1.3	1	0	0	0.176	0.176	0.229	0	0
W_4	1	1	0	0	0.176	0.176	0.176	0	0
W_5	1	0	0	0	0.176	0.477	0	0	0

$\cos(d_1, q)=0= \cos(d_2, q)= \cos(d_3, q)$

The weighting scheme provided for this case is TF= 1+ log ($tf_{t,d}$) and (IDF) is define as log (N/df_t), TF*IDF is used as weighting coefficient for each term. All documents get the same score which is zero because the IDF weighting remove the effect of these common terms from the documents and these also occur from query.