Birla Institute of Technology & Science, Pilani
Work Integrated Learning Programmes Division
Second Semester 2023-2024

Mid-Semester Test
(EC-2 Make up)

Course No.	: AIMLCZG537/DSECLZG537
Course Title	: Information Retrieval
Nature of Exam	: Closed Book
Weightage	: 30%
Duration	: 2 Hours
Date of Exam	: 28/07/2024 (FN)

No. of Pages = 2
No. of Questions = 3

Note to Students:
1. Please follow all the *Instructions to Candidates* given on the cover page of the answer book.
2. All parts of a question should be answered consecutively. Each answer should start from a fresh page.
3. Assumptions made, if any, should be stated clearly at the beginning of your answer.

Q.1. (a) In an information retrieval system, which factor is used as a measure of informativeness of a **Term.** Brief with an example.

[2]

Inverse Document Frequency

Document frequency is the numbers of documents in the collection in which the term appears. The less it appears the more useful.

(b) How Conjunctive query optimization is done in Boolean retrieval model?

.

[2]

- **Process in order of increasing term frequency:**
  - **start with smallest set, then keep cutting further.**
  - ***(wind OR fire) AND (thunder OR lightning)***
  - **Get doc. frequencies for all terms**
  - **Estimate the size of each *OR* by the sum of its doc. freq.'s**
  - **Process in increasing order of *OR* sizes.**

(c) The following table lists the terms and the document frequency in a total collection of 1000 documents.

Term	Doc.Freq
book	185
pencil	459
pen	765
paper	284

What will be the optimized order of the query (pencil OR book) AND (paper OR (NOT book)) AND (pencil OR pen)?

[2]

crackbitswilp.in

pencil OR book = 459+185 = 644
paper OR (NOT book) = 284+(1000-185)= 284+815= 1099
pencil OR pen = 459+765 = 1224
The optimized order is (pencil OR book) AND (paper OR (NOT book) AND (pencil OR pen)

The steps has to be shown to get full marks.

(d) How does stemming typically affect recall in Boolean document retrieval? Justify with an example. [1]

Stemming generally increases recall.

Ex: We can find not only words with the form of the word you entered, like "persuade" but also related forms like "persuasion". For boolean queries with negation, recall might be lowered.

Any suitable example

(e) Assume the following 3 documents D1,D2 and D3 on which Text normalization (stop word removal) is done. [Clearly give your assumptions].

D1: applications of artificial intelligence
D2: artificial intelligence and applications
D3: raising applications of artificial intelligence

Generate the Biword index for the above documents after applying the text normalization. [3]

Biwords:
applications artificial, artificial intelligence, intelligence applications, raising applications, applications artificial are the total biwords generated.
[eliminating the common ones]

Inverted index is
applications artificial : d1, d3
artificial intelligence : d1,d2,d3
intelligence applications : d2
raising applications: d3
applications artificial : d3

[2+2+2+1+3=10]

Q.2. (a) What is Zipf's Law? . How is it useful for Inverted Index compression in Information Retrieval? [2]

crackbitswilp.in

Zipf's law: The *i*th most frequent term has frequency proportional to 1/*i*
$cf_i \propto 1/i$

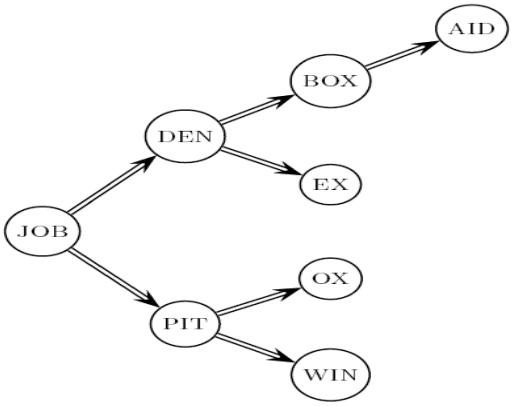
cf_i is collection frequency: the number of occurrences of the term t_i in the collection. – 1 mark

It is used for Modeling the distribution of terms, rare and frequent terms. – 1 mark

(b) Show with an appropriate example, the search done in an uncompressed dictionary and a dictionary compressed by blocking. **[2]**

Dictionary search without blocking

Assuming each dictionary term equally likely in query (not really so in practice!), average number of comparisons = (1+2·2+4·3+4)/8 ~2.6



Dictionary search with blocking

Binary search down to 4-term block;

- Then linear search through terms in block.

Blocks of 4 (binary tree), avg. = (1+2·2+2·3+2·4+5)/8 = 3 compares

1 mark for each

(c) Consider the postings list <4, 10, 11, 12, 15, 62, 63, 265, 268, 270, 400> with a corresponding list of gaps <4, 6, 1, 1, 3, 47, 1, 202, 3, 2, 130> . Assume that the length of the postings list is stored separately, so the system knows when a postings list is complete.

Using Variable byte encoding:

(i) What is the largest gap that can be encoded in 1 byte?

(ii) How many bytes will the above postings list require under this encoding? (Count only space for encoding the sequence of numbers.)

[3]

(i) Largest gap encoded using 1 byte is (2^7)-1 = 127 - 1 mark

(ii) <4, 6, 1, 1, 3, 47, 1, 202, 3, 2, 130> = 9 one bytes and 2*2 bytes(202 and 130)
Total = 13 bytes are needed.

2 marks if proper explanation given for 13 bytes

(d) Decode the Variable byte code given below for the index term **scope** and give the corresponding doc-id of the posting.

scope: 10000111000000111000100110101010000000110010101

gaps 7, 393,42,149 - 2 marks

docid: 7 , 400,442, 591 – 1 mark

crackbitswilp.in

[3]

[2+2+3+3=10]

Q.3. (a) What is the effect of Length normalization in Vector model.? Justify with an example. [2]

- **By doing length normalization makes, Long and short documents have comparable weights - 1 mark**
- **Ex: Effect on the two documents d and d′ (d appended to itself) ; they have identical vectors after length-normalizati**on. – 1 mark

(b) Construct the index using BSBI method for the below 2 documents. State your assumptions **(No Text normalization is done).**

D1: applications of artificial intelligence
D2: artificial intelligence and applications **[2]**

Taking D1 as block 1 and D2 as block 2

Block 1
applications d1
of d1
artificial d1
intelligence d1

block 2
artificial d2
intelligence d2
and d2
applications d2

Sort the term doc id pairs (term id can also be allocated)
Block 1
applications d1
artificial d1
intelligence d1
of d1

block 2
and d2
applications d2
artificial d2
intelligence d2

Merge the sorted blocks
and d2
applications d1,d2
artificial d1,d2
intelligence d1,d2
of d1

crackbitswilp.in

(c) Rank the documents for the below query using vector space model [Use the weighting scheme **ltc.ltc**] .

query: egg butter

d1: bread, milk,milk, butter,butter
d2: egg, bread, bread, bread, butter
d3: butter,egg, egg, egg [6]

idf computation
bread : log(3/2) = 0.17
milk :log(3/1)= 0.48
egg: log(3/2)=0.17
butter: log(3/3)=0

	d1					d2					d3					Query				
Terms	tf-raw	tf-wt	idf	Tf-idf	n'lize	tf-raw	tf-wt	idf	Tf-idf	n'lize	tf-raw	tf-wt	idf	Tf-idf	n'lize	tf-raw	tf-wt	idf	Tf.idf	n'lize
bread	1	1	0.17	0.17	0.23	3	1.48	0.17	0.25	0.83	0	0	0.17	0	0	0	0	0.17	0	0
egg	0	0	0.17	0	0	1	1	0.17	0.17	0.56	3	1.48	0.17	0.25	1	1	1	0.17	0.17	1
butter	2	1.3	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0
milk	2	1.48	0.48	0.71	0.97	0	0	0.48	0	0	0	0	0.48	0	0	0	0	0.48	0	0

D1.query = 0+0+0+0 = 0
D2.query = 0.56
D3.query = 1
So the ranking is d3,d2,d1

crackbitswilp.in

Tf-idf Computation for each document and query (1* 4 = 4)

Similarity computation between each document and query (0.5*3= 1.5)
Ranking the documents – 0.5

[2+2+6=10]

crackbitswilp.in