National University of Computer and Emerging Sciences

Information Retrieval (CS4051)

Date: February 28 2024

Course Instructor

Muhammad Rafi, Basit Jasani

Sessional-I Exam

Total Time: 1 Hour

Total Marks: 40

Total Questions: 03

Semester: Spring-2024

Campus: Karachi

Department: AI & DS

CLO # 1: < Understand the basic concepts and techniques in Information Retrieval.>
CLO # 4 < Appreciate the importance of data structures, indexes and retrieval efficiency>

Q1. Answer the following questions to the point. **[Time: 30 mins] [Marks: 2x10]**

a. What is a term-document matrix? What are its limitations?

A term-document matrix represents the relationship between terms and documents, where each row stands for a term and each column for a document, and an entry is the number of occurrences of the term in the document. For a large collection, this matrix is very large, sparser, and not scalable. A better representation is Inverted Index.

b. What is an Inverted Index? What are its components?

Inverted index is a kind of index on text which is composed of vocabulary and a list of occurrences in the documents. It has two main components: (i) dictionary – a collection of all the terms appears in the corpus and (ii) posting list – a list of document ids which contains these terms.

c. What do we mean by Post-Processing in IR? What is it complexity?

The post processing in IR is a specialized phase of result-set processing. In some model of IR there is a chance of getting false positive documents against a given query due to the processing approach. The post processing tries to filter all the false positive documents. The complexity of this phase is directly proportional to number of documents in the result-set.

d. What is meant by tolerant retrieval?

Tolerant retrieval means that the information retrieval system deliver you best possible answer on the submission of query by compensating the error in the query both syntax and semantics and guide the users for effectively retrieving information process through information retrieval system.
- Examples of tolerant retrieval from web searches are:
1. Spelling suggestion for query word
2. Wild card support
3. Context sensitive information retrieval
4. Search options based on statistical results of the systems.

e. What is meant by extended bi-words indexing? How it is useful?

Extended bi-words Index is an index structure that uses Part of Speech tagging to extract the terms for indexing. Extended bi-words terms like: Nouns (N) and articles/prepositions (X). Any text feature of the form NX*N to be an extended bi-word of the form N X* N. Each such extended biword is now made a term in the dictionary. For example: the text "Coins into the Pocket" is index as "Coin Pocket" - this is very useful for the commonly associated nouns.

f. How Heap's Law is useful in inverted index?

Heap's law provides a formula that can be used to estimate the number of unique terms in a collection based upon constants k and b and the number of terms or tokens (T) parsed from all documents. This can give you good idea about how many terms will be there in the dictionary of an inverted index.

g. Define Context-Sensitive Spelling Correction, Give an example query

Isolated-term correction would fail to correct typographical errors such as the query "I want a peace of cake", Although, all terms are correct the entire sentence give a clue that the word "peace" is incorrect. A search engine uses context sensitive spelling correction to identify such a situation. Using the context, it can offer that the right word for this context will be "I want a piece of cake". Using the service as "Did you mean this".

h. Find two differently spelled proper nouns whose soundex codes are the same.

Consider the proper nouns "Smith" and "Smyth" two differently spelled ones but whose Soundex codes are the same. S530 is the required code.

i. How Single-Pass In-memory Indexing (SPIMI) gain advantages over Block-Sort Based Indexing (BSBI) approach?

SPIMI adds a posting directly to its postings list. Instead of first collecting all termID-docID pairs and then sorting them like BSBI, each postings list is dynamic (i.e., its size is adjusted as it grows) and it is immediately available to collect postings. This has two advantages: It is faster because there is no sorting required, and it saves memory because we keep track of the term a postings list belongs to, so the termIDs of postings need not be stored. As a result, the blocks that individual calls of SPIMI-INVERT can process are much larger and the index construction process as a whole is more efficient hence these are some of the advantages over BSBI.

j. Which law is useful in estimating the distribution of terms for an inverted index? Explain how?

A commonly used model of the distribution of terms in a collection is Zipf's law. It states that, if t_1 is the most common term in the collection, t_2 is the next most common, and so on, then the collection frequency cf_i of the ith most common term is proportional to 1/i, that is to say: $cf_i \propto 1/i$ – It give you the estimate about the posting in an inverted index.

CLO # 2: <Understanding how IR Model works >

Q2. Consider the following documents in a collection. **[Time: 15 mins] [Marks: 2x5]**

Doc 1: multi label classification text
Doc 2: text classification in python
Doc 3: label of text in classification
Doc 4: classification of label in text

a. Draw the inverted index that would be built for this collection. Assuming "in" and "of" are the stop words that you can filter. Give both the dictionary and posting list for each term. [5]

Inverted Index

Dictionary	**Posting Lists**
classification	classification->1,2,3,4
is	is -> 2
label	label ->1,3,4
multi	multi ->1
python	python ->2
text	text -> 1,2,3,4

b. How this inverted index can be used to process the query "text AND label AND python", How this query can be optimized? [5]

The query "text AND label AND python" if it is processed in the given order the cardinality of each term will be |term| = 4, |label|=3, and |python|=1 hence the first pair of term intersect and there will be three attempt of matching document Ids. Later the result set will be intersecting with posting list of |python| no document qualifies. If we run the query like: python AND label AND text least effort is required.

CLO # 3: < Appreciate the importance of data structures, indexes and retrieval efficiency >

Q3. Answer the following questions. **[Time: 15 mins] [Marks: 2.5x4]**

a. Construct a bi-gram index for the terms {cool, cooler, coolest}

co	cool	cooler	coolest
er	cooler		
es	coolest		
le	cooler	coolest	
ol	cool	cooler	coolest
oo	cool	cooler	coolest
st	coolest		

b. What will be the permuterm index term, when we try to fetch index for the query "lo*al"?

If we wanted to search for lo*al in a permuterm wildcard index, we would be looking for al$lo* in the index.

c. Consider an inverted index with skip pointer (a skip length of √P, where P is the size of postings list) what are the choice of P and what is it effect on performance?

Inverted index with skip pointers are good for conjunctive queries. The value for which we need to place skip pointers are critical and there is a tradeoff. If we choose a smaller value of P, there will be more skips means shorter skip spans, and that we are more likely to skip. But it also means lots of comparisons to skip pointers, and lots of space storing skip pointers. On the other hand, we select a large value for P, it would give fewer skips means few pointer comparisons, but then long skip spans which means that there will be fewer opportunities to skip.

d. What sort of redundancy is captured by Front Coding? Explain.

In dictionary a common source of redundancy is due to consecutive entries are alphabetically maintained and share common prefixes among these consecutive terms. A common prefix is identified for a subsequence of the term list and then referred to with a special character. In the case of Reuters, front coding saves another 1.2 MB as found by an experiment.