National University of Computer & Emerging Sciences, Karachi

Spring-2022 Department of Computer Science

Midterm Exam-I (Sol)

March 09, 2022, 10:00 AM – 11:00 AM

Course Code: CS4051	**Course Name:** Information Retrieval
Instructor Name / Names: Muhammad Rafi	
Student Roll No:	**Section:**

Instructions:

- Return the question paper.
- Read each question completely before answering it. There are **3 questions** on **2 pages**.
- In case of any ambiguity, you may make assumption. But your assumption should not contradict any statement in the question paper.

Time: 60 minutes **Max: 40 Marks**

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 is meant by Extended Boolean Model? What are some of its goals?

Extended Boolean Model – is an extension of classical Boolean Model in which term frequency and weighting is used to create a ranked based retrieval. The main goal of Extended Boolean Model is to overcome the drawbacks of the Boolean model like: binary decision of relevance, term weighting, no ranking of the documents.

b. Define the terms Token, Type and Term. Give an example of each.

A token is an instance of a sequence of characters in some particular document that are grouped together as a useful semantic unit for processing.
A type is the class of all tokens containing the same character sequence.
A term is a (perhaps normalized) type that is included in the IR system's dictionary as a feature for retrieval. For example, consider an overly simplified document below:

I eat what I eat, even if I have never eaten it before.

There are 13 tokens. The tokens: eat, eat and eaten all of the same type and can be mapped to a single term eat.

c. What is token normalization? What are its benefits?

Token normalization is the process of canonicalizing tokens so that matches occur despite superficial differences in the character sequences of the tokens. For example, all morphological variation of the term "Operate" like Operation, Operator, Operands mapped to a single feature "Operate". The main benefit of token normalization is reducing the dictionary size.

d. What do we mean by the term "Trucasing"?

Truecasing is a machine learning process through which a parser may decide to keep the true case of a token from a stream. For example, if we want to keep all capital letters word that appears in side a sentence, it may be a Named Entity. So, if we want to keep this intact we may opt for Truecasing in the parsing phase.

e. Define the term Permuterm index. Give one advantage and one disadvantage of it.

The Permuterm index is a type of index used to process wild-card queries (tolerant retrieval). The idea is to prepare all possible rotations of each dictionary term and if a single * query is presented it will be directly fetched with matching the pre-fix and suffix part of the query. For example X*Y on will be looked up in permuterm dictionary as Y$X*. A big advantage of this type of index is a quick retrieval for a single wildcard queries. A major disadvantage is space such an index may need space equal to 4 times of a standard dictionary.

f. How could an IR system combine use of a positional index and k-gram index to give rapid response to simple phrase query like "t1 t2".

The strategies of k-gram indexes and positional indexes can be fruitfully combined. If users commonly query on particular phrases, such as Michael Jackson, it is quite inefficient to keep merging positional postings lists. A combination strategy uses a phrase index, or just a k-word index, for certain frequent queries and uses a positional index for other phrase queries.

g. How context sensitive spelling correction works? What kind of queries it can correct?

In general phrase query it is some time common that individual terms are valid dictionary words but a contextual reference of the query suggests that there is some error. For example: "How to get a peace of cake" – in this query all terms are correct but the term "peace" is wrongly placed – such an error in queries corrected by context sensitive spelling correction techniques. In which each word is replaced by a correct rendition of the dictionary term and all possible phrases are analyzed to get the correction. {peace is a valid word in isolation but an error in this context (should be piece)}

h. Give 3-gram index vocabulary for the terms ‘index’ and ‘independent’?

index ---> $index$ ---> in, ind,nde,dex,ex
Independent ---> $independent$ ---> $in, ind,nde,dep,epe,pen,end,nde,den,ent,nt$

i. How Zipf’s Law is helpful in inverted index processing?

Zipf's law states that the frequency of a token in a text is directly proportional to its rank or position in the sorted list of all tokens. This law describes how tokens are distributed in languages: some tokens occur very frequently, some occur with intermediate frequency, and some tokens rarely occur. This helps in building inverted index for deciding how many posting entries would be there for some frequent terms.

j. How Single-Pass in Memory Indexing (SPIMI) improves over Block-Sort Based Indexing (BSBI)?

A difference between BSBI and SPIMI is that SPIMI adds a posting directly to its postings list. Instead of first collecting all termID–docID pairs and then sorting them. In SPIMI each postings list is dynamic (i.e., its size is adjusted as it grows) and it is immediately available to collect postings. There are two advantages of this approach:
(1) It is faster because there is no sorting required.
(2) it saves memory because we keep track of the term a postings list belongs to, so the termIDs of postings need not be stored.

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:

accord	11201	dine	760	obtain	7610
benefit	676	engage	1423	policy	1123
choice	12010	establish	634	practice	2130
commit	23410	event	534	stock	23123
concern	1290	issue	13400	zeal	712

Query1: benefit of the choice of event
Query2: stock choice obtain or practice

a. Assuming of, or, and the etc. are stop words. Give the possible Boolean expression queries with best query execution cost. [5]

Query1 in Boolean Expression => (benefits) ^(choice) ^ (event)

On considering the posting size, we will have the best order as

Query1 in Boolean Expression(Optimal) => (event)^(benefits)^(choice)

Query2 in Boolean Expression => (stock) ^(choice) ^ (obtain)^(practice)

On considering the posting size, we will have the best order as

Query2 in Boolean Expression(Optimal) => (practice)^ (obtain) ^(choice)^(stock)

b. For a conjunctive query from the part (a), is processing postings lists in order of size guaranteed to be optimal? Explain why it is, or give an example where it isn't. [5]

This is not necessarily the best order of processing. Suppose from the part (a) query. There are three terms (**(event)^(benefits)^(choice)**) and the size of the list is event=534, benefits=676, choice=12010. Suppose the intersection size of event and benefits is 534, and the intersection size of event and choice is 0. If it is processed in the order of **(event)^(benefits)^(choice)**, it needs 534+676+534+12010=13754. But if it is processed in the order of **(event)^(choice) ^(benefits)** 534+12010+0+0=12544 is needed.

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

Consider the following documents in a collection. Where ***in*** and ***of*** are treated as stop words.

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
Query: label text classification

a. By Assuming a Boolean Model for information retrieval, how would you retrieve the relevant documents for the query form given collection? You can use Jaccard similarity for computing relevance between document and query. [5]

Vocabulary or Features = {multi, label, classification, text, python}
Jaccard's co-efficient (similarity)(X,Y) = $(X \cap Y) / (X \cup Y)$

Doc1: { multi, label, classification, text}
Doc2: { text, classification, python}
Doc3: { label, text, classification}
Doc4: { classification, label, text}
Query: { label, text, classification}

Jaccard's co-efficient (Doc1,Query) = (Doc1 $\cap$ Query) / (Doc1 $\cup$ Query)
= 3/4 = 0.75
Jaccard's co-efficient (Doc2,Query) = 2/4 = 0.5
Jaccard's co-efficient (Doc3,Query) = 3/3= 1
Jaccard's co-efficient (Doc4,Query) = 3/3 = 1

The relevance of the retrieved documents against the given query will be"
Doc3, Doc4 (top documents against query at the same level)
Doc 1
Doc 2

b. Using a Vector Space Model (VSM) for information retrieval, compute a rank order of relevancy for the given set of document against the given query. You can use simple term frequency for vectors and Cosine similarity between document and query. [5]

The vector space of the vocabulary from the documents are as below:

R^5 = <classification, label, multi, python, text>

Now document vectors with simple term frequencies will be as follow:

Doc1: <1,1,1,0,1>

Doc2: <0,1,0,1,1>

Doc3: <1,1,0,0,1>

Doc4: <1,1,0,0,1>

Query: <1,1,0,0,1>

Cos (Doc1, Query) = (Doc1 dot Query) / (| Doc1| X |Query|) = 3 / (2√3)

Cos (Doc2, Query) = (Doc2 dot Query) / (| Doc2| X |Query|) = 2 / 3

Cos (Doc3, Query) = (Doc3 dot Query) / (| Doc3| X |Query|) = 1

Cos (Doc4, Query) = (Doc4 dot Query) / (| Doc4| X |Query|) = 1

The relevance of the retrieved documents against the given query will be"

Doc3, Doc4 (top documents against query at the same level)

Doc 1

Doc 2

BEST OF LUCK