Examination in IN3120/IN4120

UNIVERSITY OF OSLO
The Faculty of Mathematics and Natural Sciences
Written examination IN3120/IN4120
2023 Autumn
Duration: November 28, 03:00 p.m. - 07.00 p.m. (4 hours)
Permitted aids: None
It is important that you read this front page before you start.

The different questions have different weights, as indicated.

You can answer in Norwegian or English. Please use the language that you are most comfortable with.

P.S.: Inspera calculator i available

1 VECTOR SPACES [35p]

(a) [10p] There are two main ways to think about vector spaces for text: (A) As a sparse and extremely high-dimensional representation where each unique vocabulary term corresponds to a distinct dimension of your vector space, or (B) as a dense and lower-dimensional representation where each unique word in your vocabulary corresponds to a point in an abstract vector space we can call an “embedding space” (where we beforehand have fixed, say, typically a few hundred dimensions.)

- (i) For case (A) above, briefly discuss how a larger text buffer (e.g., a document) could be placed in this vector space, and outline some pros and cons of working with representation (A).
- (ii) For case (B) above, briefly and at a high level discuss the general ideas behind how a given word gets placed in this embedding space, how we might go about placing a larger text buffer (e.g., a document) in this same embedding space, and outline some pros and cons of working with representation (B).

(b) [5p] Consider the dense vectors x = [0.6, 0.2, 0.8] and y = [1.0, 0.1, 0.9]. Show how to compute the cosine similarity between x and y. (Clearly showing the correct procedure without arriving at a final numerical result will give full marks.)

(c) [5p] Explain what an approximate nearest neighbour (ANN) index is, and why it is useful.

(d) [15p] List at least 5 strategies that an ANN index can employ to efficiently find matches, and succinctly explain the thinking behind each strategy.

Fill in your answer here

Maximum marks: 35

2 MEASURING RELEVANCE [20p]

(a) [5p] Describe what the F_β-score is, and define it in terms of precision P and recall R. What does the β parameter control? If $P = 0.1$ and $R = 0.5$, what is the F_1-score?

(b) [5p] Assume a ranked retrieval context. Describe what a precision-recall curve is and how we generate it. What is an interpolated precision-recall curve?

(c) [5p] Let R denote a relevant document, and let N denote a non-relevant document. Consider a search system that for the query *carrot* produces the ranked result set RRNRNNNR and that for the query *chocolate* produces the ranked result set RNRR. Show how you compute the search system's mean average precision (MAP) score. (Clearly showing the correct steps without arriving at a final numerical result will give full marks.)

(d) [5p] Kendall's tau distance can help us assess how "close" a ranked result set L for a given query is to a given set of pairwise preferences P for that query. Describe the high-level idea behind how this is computed. If L = [A, C, B, D] and P = {(A, B), (A, C), (A, D), (B, C), (B, D), (C, D)}, what is Kendall's tau distance between the two?

Fill in your answer here

Maximum marks: 20

3 MIXED GRILL WITH CARROTS [30p]

(a) [10p] Based on the training data in the table below, show how a multinomial naïve Bayes text classifier would classify an unseen document *d* having the body text *carrot carrot carrot toffee jellybean*. That is, from the training data, show how to estimate all required prior probabilities and conditional probabilities, and show how to combine these to arrive at values proportional to Pr(*healthy* | *d*) and Pr(*unhealthy* | *d*), respectively. Use simple add-one smoothing when estimating the conditional probabilities. (Clearly showing the correct expressions without arriving at a final numerical result will give full marks.)

document_id	body	class
1	*carrot broccoli carrot*	*healthy*
2	*spinach carrot carrot*	*healthy*
3	*carrot mango*	*healthy*
4	*toffee jellybean carrot*	*unhealthy*

(b) [5p] Consider an inverted index produced by indexing the field *body* in the corpus consisting of the 4 documents listed in the table in task (a) above. Assume that each posting contains a document identifier and a term frequency. List all postings in all posting lists in the inverted index.

(c) [10p] Consider the same inverted index as in task (b) above.

- (i) How many bytes would you need to store the compressed posting list for *carrot*, when combining simple gap-encoding with variable-byte encoding? Explain your reasoning.
- (ii) How many bits would you need to store the compressed posting list for *carrot*, when combining simple gap-encoding with Elias gamma-encoding? Explain your reasoning.

(d) [5p] Consider a tiny Bloom filter backed by 16 bits of storage and with 3 hash functions. Assume the hash values shown in the table below.

- (i) Show what the filter's bit array looks like before inserting anything, after inserting *carrot*, and after inserting both *carrot* and *toffee*.
- (ii) Given that only the two values *carrot* and *toffee* have been inserted, explain what the filter will say when queried about the set memberships of *steak* and *carrot*, respectively, and explain the logic for how the filter arrives at these decisions.
- (iii) Outline how you could modify the Bloom filter to reduce the probability of false positives.

Hash function *h*	Value *x*	Hash value *h*(*x*)
h_1	*carrot*	12
h_1	*toffee*	0
h_1	*steak*	7
h_2	*carrot*	7
h_2	*toffee*	12
h_2	*steak*	15
h_3	*carrot*	15
h_3	*toffee*	3
h_3	*steak*	11

Fill in your answer here

Maximum marks: 30

4 APROXXIMAT MATHCING [15p]

(a) [8p] Given a trie encoding a large collection of strings *D* and given a query string *q*, we would like to efficiently find all strings in *D* that are within *k* edits from *q*. Assuming that *k* is a small number, describe a trie-based search algorithm that does this. Make sure to explain the most important insights that contribute to making the search algorithm efficient.

(b) [7p] Propose a way to alter the algorithm and/or data structures from task (a) so that you can combine edit distance with phonetic hashing (such as, e.g., Soundex codes): Your proposed algorithm should be able to efficiently find all strings in a large collection of strings *D* where the phonetic hashes between a string in *D* and a query string *q* differ in at most *k* edits. For example, *richards* and *lichardson* would match with *k* = 1, since their respective Soundex codes only differ in one symbol.

Fill in your answer here

Maximum marks: 15