CS54701 Spring 2016 Midterm solutions, 1 March, 2016
Prof. Chris Clifton

Turn Off Your Cell Phone. Use of any electronic device during the test is prohibited. As previously noted, you are allowed notes: Up to two sheets of 8.5x11 or A4 paper, single-sided (or one sheet double-sided).

Time will be tight. If you spend more than the recommended time on any question, **go on to the next one**. If you can't answer it in the recommended time, you are either giving too much detail or the question is material you don't know well. You can skip one or two parts and still demonstrate what I believe to be an A-level understanding of the material.

Note: It is okay to abbreviate in your answers, as long as the abbreviations are unambiguous and reasonably obvious.

1 Vector Space Model Limitations (12 minutes, 6 points)

Assume that you have an information retrieval system based on the vector-space model. You are asked to extend it to support retrieval of phrases. For example, someone could give a query "information retrieval course", and it would retrieve only documents with those words together as a phrase, in the order given.

Note that phrases can be long and are not known in advance.

A. Describe an approach you could take to do this. Make use of the existing vector-space model system.

Use the existing system to retrieve documents using the phrase as independent words. Then retrieve returned documents and do a string search for the exact phrase. Only the once containing the phrase would be shown to the querier.

(Another common, and more elegant, solution was to create a position index, storing the position of the word in a document. This is used to identify or filter for phrases.)

Two points for solid approach using existing model, one point for an approach that works for some phrases but is not complete. Note that "index all phrases as terms" (all n-grams) is not really a plausible approach, given that phrases can be long and are not known in advance. How many possible phrases of length n are there? Such an answer was worth one point if you gave a good approach for identifying a subset of n-grams to call phrases and recognized the limitations.

B. Would you expect your approach to have good precision? Explain.

Precision would be high, as only documents that contained the phrase would be returned.

1 for showing knowledge of precision and good explanation

C. Would you expect your approach to have good recall? Explain.

Recall would be at best equivalent to the underlying system - if the underlying system failed to return a relevant document, then my approach

would not find it, so the numerator of the equation could not go up (and the denominator is unchanged.)

1 point for showing knowledge of recall, 1 for correct/explanation

D. Give a drawback to your approach.

The system would be slow, as it would have to retrieve all documents returned by the vector-space model system and perform a string search.

1 point for plausible answer

2 Preprocessing (10 minutes, 5 points)

For the following, assume the vector-space model and a cosine similarity retrieval model. Given the following two documents:

Document A

The information retrieval class will be an introduction to retrieving information from text. It will be on Tuesday and Thursday from 3-4:15 in Wang 2555.

Document B

```
<html><title>Information Retrieval</title>
<body> <dl>
<dt>Introduction to Information
    Retrieval</dt>
<dd>Class: Tuesday, Thursday 3-4:15,
           Wang 2555.
Text:  Information Retrieval.</dd>
</dl> </body> </html>
```

and the query Information Retrieval Class :

A. Give an example of preprocessing and weighting steps that you expect would lead to Document A being ranked higher than Document B. Explain why.

Note that the term frequencies of the query terms are higher in Document B, and the lengths are similar. To get Document A ranked higher, we would need to increase the term counts in A, decrease it in B, or decrease the length of Document A more than B. Stopword removal will shorten Document A, and stemming will increase frequency of retriev (retrieving as well as retrieval.) Ignoring information outside the main body of text (e.g., in the title) would also reduce the term frequency in Document B (losing data from the Title). Including the HTML tags in the indexed terms will make Document B longer, and this should make Document A more likely. Position-sensitive weight might help as well.

1 point for showing knowledge of steps in either part, 1 for good choices or formality in explanation, 1 for decent explanation.

B. Give an example of preprocessing and weighting steps that you think would lead to Document B being ranked higher than Document A. (Again, explain why.)

This is easier. Simply removing all HTML tags and no other preprocessing would result in Document B having greater term frequencies and smaller document size. Therefore standard TF-IDF weighting and cosine similarity would rank Document B higher.

1 for good choices or formality in explanation, 1 for decent explanation

3 Cosine Similarity (10 minutes, 5 points)

Given the following term-document weight matrix:

	DocX	*DocY*	*DocZ*
Computer	4	0	2
Science	3	0	5
Exam	0	4	2

and the query Computer Science :

A. Compute the cosine similarity between *DocX* and the query. Assume that the query terms are weighted using Term Frequency only.

$\frac{\vec{q}\cdot\vec{d}}{|\vec{q}|\cdot|\vec{d}|} = \frac{1*4+1*3+0*0}{\sqrt{4^2+3^2+0^2}\cdot\sqrt{1^2+1^2+0^2}} = \frac{7}{5\cdot\sqrt{2}} \approx 1$ (slightly less than 1).

1 point for correct term weighting, 1 for correct formula, 1 for correct value.

B. Give a rank ordering of the documents based on cosine similarity. You do not need to calculate all the cosine similarities, but if you do not calculate the cosine similarities, you should explain why your ordering is correct.

(DocX $\succ$ DocZ $\succ$ DocY). DocY is clearly 0 so last. DocX and DocZ would give the same numerator in the above calculation, but the denominator (length) is smaller for DocX, so it would clearly be first.

1 for correct order, 1 for calculating or explanation.

4 Relevance Feedback (10 minutes, 4 points)

You are given a retrieval system, and asked to incorporate some form of relevance feedback. Assume that you get the query results, and the user provides a set of "good" and "bad" documents from the result. How might you go about implementing relevance feedback, *without modifying the existing retrieval system.* For full credit, demonstrate with an example how this would be done.

One approach is query expansion or generation: incorporate terms that occur in the good documents but not the bad documents into the query, and then submit as a new query. For example, we could learn a generative language model for all the "good" documents and add all of the terms to the query, weighted by the generation probability in the model. We could do the same for "bad" documents, but *invert* the weights.

For example, given the query information retrieval, the "good" document information retrieval class, and the "bad" document information retrieval system, the language models would each have three words with probability 1/3. The weights for information and retrieval would cancel each other out, so our new terms would be class +1/3 and system -1/3. Adding to the original query would give information 1 retrieval 1 class 1/3 system -1/3 as the new query.

1 for idea (presumably query expansion), 1 for clear description, 1 for formal/algorithmic description, 1 for example (or extremely complete/clear formal description.

5 Probabilistic Language Models (20 minutes, 9 points)

Given the following documents:

Document 1:
Jack and Jill went up the hill

Document 2:
To fetch a pail of water

Document 3:
Jack fell down and broke his crown

Document 4:
And Jill came tumbling after

You may assume that "a", "and", "of", "the", and "to" are stopwords.

A. Compute the unigram term probabilities for each term for Documents 1 and 2

	Doc1	**Doc2**
Jack	**1/5**	**0**
Jill	**1/5**	**0**
went	**1/5**	**0**
up	**1/5**	**0**
hill	**1/5**	**0**
fetch	**0**	**1/3**
pail	**0**	**1/3**
water	**0**	**1/3**
fell	**0**	**0**
down	**0**	**0**
broke	**0**	**0**
his	**0**	**0**
crown	**0**	**0**
came	**0**	**0**
tumbling	**0**	**0**
after	**0**	**0**

1 point for weights for terms, 1 for probabilities, 1 for correct.

B. Given the query Jack Jill , what is the probability the query was generated by:

- Document 1
  $P(\vec{q}|\vec{Doc1}) = P(Jack|\vec{Doc1}) \times P(Jill|\vec{Doc1}) = 1/5 \times 1/5 = 1/25$

- Document 2
  $P(\vec{q}|D\vec{o}c2) = P(Jack|D\vec{o}c2) \times P(Jill|D\vec{o}c2) = 0 \times 0 = 0$

1 for looking at probability of each word, 1 each for correct values.

C. Term probabilities of 0 can cause problems, as the probability of a query containing that term being generated by the document becomes 0. Using the above examples, demonstrate why this is a problem, and explain what can be done to fix it. For full credit, demonstrate how this would work and how it would affect the query result (i.e., you could repeat the answer to part 2 in a way that addresses this problem, or demonstrate by showing what would happen with Document 3 or 4. If your answer to part 2 already deals with this issue, just explain the issue and how you dealt with it.)

Doc2 could never match a query containing Jack or Jill, even a query q_2 = Jill fetch pail water, as $P(Jill|D\vec{o}c2) = 0$. When the probabilities are multiplied, the result is 0 regardless of the value of other probabilities.

Smoothing is a way of dealing with this issue. In smoothing, we essentially introduce a possibility that any word can be generated by the language model. A simple example is Jelinek Mercer smoothing: have the probability for a word (e.g., Jill) be $\lambda P(Jill|D\vec{o}c2) + (1-\lambda)P(Jill|Corpus)$. This makes the language model for Jill for Doc2 be $\lambda * 0 + (1-\lambda)P(Jill) = 0 + (1-\lambda) * 2/18$. Assume $\lambda = 0.9$, this would now give $P(q_2|Doc2) = 0.2/18 * (.9/3 + .1/18) * (.9/3 + .1/18) * (.9/3 + .1/18)$, which while small, is going to be higher than for other documents.

1 for right idea of issue, 1 for right idea of fix, 1 for demonstration of dealing with issue.