COMP90042 Web Search and Text Analysis
Sample Exam

Semester 1, 2016

Total marks: 50 (roughly)

Students must attempt all questions

In preparing for the exam, please consult the library collection of past exams at:

`http://library.unimelb.edu.au/examination_papers`

Specifically you should look for the exams from:

1. COMP90042 Web Search And Text Analysis, 2014, 2015
2. 433-{4,6}60 Human Language Technology, 2003, 2004, 2008.

Note that the subject content has changed substantially from prior years, and only some of the questions from the above papers are relevant. Last year's exam is the most relevant, and the format will be the same.
I've provided you with some further indicative questions below.

A: Short answer

Expect to answer in a sentence or two, with longer or more thought-out answers corresponding to higher mark allocations.

1. Information retrieval

   a) What is an "information need" and how does this relate to a "query"? Use an example to justify your answer. [2 marks]
   An information need is a conceptual description of the problem that the user is trying to solve with an IR engine, approximated by a query. For example, a user might issue the query `University of Melbourne` when desiring to go to the physical location corresponding to the institution — in which case, they probably want a map — or perhaps the user would like to navigate to the homepage of the institution on the World Wide Web — in which case, they simply want a URL.

   b) Outline an approach to compression used in information retrieval, and state its effect in terms of runtime. [2 marks]
   One approach to compression in IR is compressing the d-gaps (gaps between document identifiers) in a postings list using variable byte compression (1 mark). Since we need to look at the leading bit of each value, along with the data bits, the time complexity of decompression is $\mathcal{O}(n \log n)$, where n is the total number of postings in the collection; as compared to raw integers which is $\mathcal{O}(n)$ (1 mark, big-O nice but not strictly needed).

   c) Describe why recall is important in IR evaluation, and why it is difficult to measure. [2 marks]
   Recall often isn't important in IR evaluation, because there are far too many potentially relevant documents for a given user to realistically look at. However, in some disciplines like Medicine or Law, a user may want to access every single relevant document as any single item could be of critical importance. It also plays a role in numerous desirable evaluation metrics, like MAP (1 mark – either of the above reasons). To assess recall, we would need to examine every single document in the collection for relevance, which would be extremely time-consuming (and exhausting!) for a human adjudicator (1 mark).

d) The "Okapi BM25" model of document retrieval includes two additional components besides term frequency and inverse document frequency. State what these components are, and describe their respective roles in document ranking (no formulae needed). [2 marks]

Okapi BM25 includes components for:

- document length relative to the average document length in the collection, which is used to down-weight the term frequencies of long documents, and up-weight short documents;
- term frequency within the query, so that repeated words in the query receive more importance in the overall calculation;
- modulating the effect of repetitions on scoring between binary occurrence and raw frequency, for Query and Document TF.

(1 mark each for the reasons, max 2 marks)

2. Probabilistic models of language

(a) What are the characteristics of a "Markov chain"? Name an example where "Markov chains" are used in a retrieval or text analysis setting. [2 marks]

A Markov chain is a probabilistic model of sequences, where the probability of the next event depends only on the previous event, or n previous events, for constant n (1 mark). Markov chains are used in various "link analysis" algorithms for IR, like PageRank (not covered), and in HMMs, which are often used to POS tagging, chunk tagging, etc. in NLP (1 mark for any valid example).

(b) Describe the motivation behind the "n-gram" method for "language modelling" and name a limitation of this technique. [2 marks] The n-gram method of language modelling assumes that we can evaluate the joint probability of a sequence $P(w_1 w_2 \cdots w_M)$, according to a Markov chain with $n-1$ words of context (1 mark). It is often the case that other factors outside the $(n-1)$-word context window (further back) are important for predicting the next token. For example, centre-embedding as in

A man that a woman loves ⇒
A man that a woman that a child knows loves ⇒ ...

where each verb at the end depends on its argument at the start of the sentence, which can be made to be $> n$ words away by adding more to the sentence (1 mark — example; doesn't need to be this clever, might just be a lot of adjectives separating a determiner from a noun, or a WH sentence)

(c) Tree-structured models are often used for describing syntax. Describe a property of language that motivates the use of tree based techniques, and provide a supporting example. [2 marks]

The recursive property of languages motivates tree structures (1 mark). For example, we might observe that a noun phrase can be modified by a prepositional phrase, which contains a noun phrase that contains another prepositional phrase (1 mark), and so on: *the cat on the table in the house on the outskirts of the village by the river through the valley*; hence we build rules like `NP->Det N PP`; `PP->P NP`.

(d) Define with the aid of an example the term "hypernym". [1 mark]

A hypernym is a specific case of a more general word, for example *cat* is a hypernym of *animal* (1 mark)

(e) Distributional Semantics

i. Give one reason why "PMI" would be preferred over "term frequency" as a measure of word association for computational semantics. [1 mark]

Raw term frequency overemphasizes common words which have very little semantic content, such as function (stop) words such as *the*.

ii. The creation of "word2vec" word vectors (embeddings) applies supervised classification techniques. What is the underlying classification task for each of the two approaches we discussed (that is, "CBOW" and "skip-gram")? [1 mark]
CBOW involves predicting a target word based on the words in the context around them, the Skip-gram method involves predicting the context words based on the target word.

iii. What is "dimensionality reduction"? Give two examples [2 marks]
Dimensionality reduction involves converting a high dimensional vector space to a lower dimensional vector space while (typically) trying to preserving information such as distances between vectors. Truncating the matrices created by a singular value decomposition (SVD) as part of LSA is a kind of dimensionality reduction, as is latent Dirichlet allocation (LDA).

B: Method questions

These tend to require longer answers, again see the mark allocations for approximate answer difficulty and length.

1. Markov models

a) Contrast the use of n-gram Markov models with hidden Markov models. What is it about hidden Markov models that is "hidden"? [2 marks]

A Markov model defines a probability over a sequence of events, typically words, using the most recent $n - 1$ words to predict the next word. This is the standard setting of language modelling. In a Hidden Markov Model, there is an extra "hidden" layer, because some property of the words cannot be directly observed from the word sequence. For example, the part-of-speech tag of the word. In this case, the states are the (hidden) POS tags, the transition probabilities are based around sequences of POS tags, and each POS tag has an emission probability of observing each word, given this POS tag.

b) Present the Viterbi algorithm for a first order Hidden Markov Model with the aid of a simple example sentence, and state its time complexity of inference. [5 marks]

Wordy answer The Viterbi algorithm for HMMs is a dynamic programming algorithm for detemining the most-likely state sequence for a set of observations. For example, the best POS tags for a sequence of words.

Let's consider the sentence `bear house pest`; we have some set of states, that we will assume consists solely of J,N,V for simplicity.

The Viterbi algorithm works by progressively estimating the best joint distribution for the first words of the sentence together with the corresponding tag sequence. In this case, we will begin by estimating the probability of beginning with J for `bear`, and N, and V. We can do this simply by multiplying the initial state probabilities by the corresponding emission probability for `bear`.

We will use these probabilities, along with the transition probabilities from J to J, N to J, and V to J to estimate the best sequence where `house` is J; similarly for N and V for `house`. We can then multiply the probability of the best sequence with the emission probability of `house`, for each of the three states.

These will then be the inputs for the `pest` entries, which will proceed the same way. Whichever probability is highest at this point (combining the best transition from the previous probabilities with the emission probability of `pest`) will correspond to the best sequence. (And we've kept back-pointers to record what this sequence was! Did I mention that? `:)`)

Since, for each word in the sentence, we consider each tag, and the probability that it was immediately preceded by every tag, the complexity is $\mathcal{O}(| w || t |^2)$ (the number of words in the sentence, times the square of the number of states).

(It would be worth also illustrating the sentence as a picture, showing the lattice of states.)

Continued overleaf ...

Mathsy answer The Viterbi algorithm searches for the best path (sequence of states) in a HMM given a sequence of observations. This is framed as

$$\text{argmax}_{s_1,s_2,\ldots,s_T}\, p(x_1, x_2, \ldots, x_T, s_1, s_2, \ldots, s_T) \tag{1}$$

where s_i is the i^{th} state and x_i is the i^{th} observation. Using the formulation of the HMM, this can be expressed more simply as

$$\begin{aligned} &\max_{s_1,s_2,\ldots,s_T} p(s_1)p(x_1|s_1)p(s_2|s_1)p(x_2|s_2)\cdots p(s_T|s_{T-1})p(x_T|s_T) \\ =&\max_{s_1} p(s_1)p(x_1|s_1)\max_{s_2} p(s_2|s_1)p(x_2|s_2)\cdots \max_{s_T} p(s_T|s_{T-1})p(x_T|s_T) \end{aligned} \tag{2}$$

where we have swapped to $\max$ for simplicity (but see back-pointer discussion below). The Viterbi algorithm uses dynamic programming to represent each component in Equation 2, i.e.,

$$\begin{aligned} \alpha[1, s_1] &= p(s_1)p(x_1|s_1) \\ \alpha[2, s_2] &= \max_{s_1} p(s_2|s_1)p(x_2|s_2) \\ &\ldots \\ \alpha[T, s_T] &= \max_{s_{T-1}} p(s_{T-1}|s_T)p(x_T|s_T) \end{aligned}$$

such that Equation 2 can be expressed as

$$\max_{s_T} \alpha[T, s_T]$$

And the argmax can be recovered by storing back-pointers which record the 'winning' previous state for each cell of $\alpha[i, s_i]$. There are $T \times S$ values in the α dynamic programming matrix (space complexity $O(TS)$), and the computation of each cell involves enumerating S prior states, so the time complexity is $O(TS^2)$. Then give the **bear house pest** sentence showing the states and relate the maximium score for reaching a state to alpha in the maths above.

2. Machine translation

a) Define "word alignment", and explain why it is often a neccessary step in learning a machine translation system. [2 marks]

Word alignment is the process of automatically identifying how the words of a sentence in the source language map to the words of the translated sentence in the target language, for a given parallel (sentence-aligned) corpus (1 mark). It is often a necessary step in learning a machine translation system, because it allows us to ascertain how words are translated between the two languages which is necessary if we wish to translate novel sentences (1 mark).

b) Contrast word-based and phrase-based translation, and provide a reason why the phrase-based approach is more effective. [2 marks]

Phrase–based translation treats word n-grams as the translation units; word–based translation is phrase–based translation for the special case where $n = 1$ (1 mark). Taking larger fragments of the sentence to translate allows us to find common translation fragments (like multi-word expressions or idioms, which are usually not well-handled in word–based systems), as well as common re-ordering patterns (like different pronoun or negative-particle orders in some languages). Avoiding the need for many decisions about reordering and often ambiguous word translation improves the system predictions. Larger fragments also have the benefit of being internally coherent, leading to more coherent overall output (1 mark for either of the reasons, with justification).

3. Parsing

(a) What is the difference between the "CYK" and "Earley" parsing algorithms with respect to the expectations of the form of the input context free grammar. You should give the explicit restrictions. [2 marks]

Whereas Earley parsing can be applied to any CFG, the CYK algorithm requires that the CFG be in Chomsky Normal Form, which is to say that all rules must be of the form $A \rightarrow a$ (where a is a terminal) or $A \rightarrow BC$, which is to say that the right hand side of any rule is either one nonterminal, or two terminals.

(b) Give a slightly modified version of the more-restricted algorithm which does not have these restrictions (this will involve an extreme increase in the time complexity of the algorithm). In doing this, you should explain the key part of the original algorithm [2 marks]

In order to fill in cell of the table corresponding to the interval $[n, m]$, the algorithm looks for nonterminal nodes in intervals $[n, i]$ and $[i, m]$ for all indicies i, $n < i < m$ which correspond to the right hand side (RHS) of some production. First, in order to allow terminals on the right hand side in more general cases, terminals must be added to the cells (a terminal at index i in the sentence should be put in the cell $[i, i+1]$). Suppose that the longest RHS among all productions in the grammar is of length k; the most straightforward (though extremely inefficient) modification involves considering all possible values of $k-1$ indicies $i_1, i_2 \ldots, i_{k-1}$, where $n \leq i_1 \leq i_2 \ldots \leq i_{k-1} \leq m$, and whether the terminals in the cells corresponding to the intervals created by these indicies for the RHS of some production in the grammar.

(c) Explain how the other algorithm (the one without restrictions) avoids the blow up in complexity seen in the proposed modification above. [1 mark]

Rather than trying to complete entire production rules in single step as CYK does, Earley parsing represents partially completed rules in the chart, and completes each rule one nonterminal at a time; In essence, despite productions of unlimited length, the Earley algorithm manages to keep things binary (and therefore tractable) by having each completion step consist of one (uncompleted) edge, and one nonterminal.

C: Algorithmic Questions

Rocchio algorithm Given the following term-document matrix

	Term frequency, $w_{t,d}$				
doc	"soccer"	"football"	"pitch"	"hockey"	"tournament"
d1	3	0	4	0	0
d2	0	6	8	0	0
d3	1	0	0	2	2

a) compute the cosine similarity between the query "soccer" and each document, using the vectors above (no need to include IDF term), and show the ranked order of documents. You are not required to simplify fractions. [3 marks]

First, we need a representation of the query. We're told not to use IDF; for a single query term, IDF is irrelevant anyway. Assuming a VSM for $\langle$`soccer,football,pitch,hockey,tournament`$\rangle$, our query is $\langle 1, 0, 0, 0, 0 \rangle$.

We will also need the document lengths. These are easy here:

$$\begin{aligned} \mid d_1 \mid &= \sqrt{3^2+0^2+4^2+0^2+0^2} \\ &= \sqrt{9+16} = \sqrt{25} = 5 \\ \mid d_2 \mid &= \sqrt{0^2+6^2+8^2+0^2+0^2} \\ &= \sqrt{36+64} = \sqrt{100} = 10 \\ \mid d_3 \mid &= \sqrt{1^2+0^2+0^2+2^2+2^2} \\ &= \sqrt{1+4+4} = \sqrt{9} = 3 \end{aligned}$$

(1 mark)
Now we can apply our cosine similarity model; we're going to calculate the dot product of the query vector with the given document vectors ($\langle 3,0,4,0,0\rangle$ for d_1 etc.) — but in this case, that is just going to be the value of the `soccer` dimension. We're then going to divide by the document length (the query length is 1):

$$\begin{aligned} \cos(q,d_1) &= \frac{q \cdot d_1}{\mid q \mid\mid d_1 \mid} = \frac{3}{(1)(5)} = 0.6 \\ \cos(q,d_2) &= \frac{0}{10} = 0 \\ \cos(q,d_3) &= \frac{1}{3} \approx 0.33 \end{aligned}$$

(1 mark)
The ranked order of the documents, from higher assumed relevance to lowest, is 1,3,2. (1 mark)

b) using the Rocchio algorithm, defined as

$$q_e = \alpha q_0 + \beta \frac{1}{|D_r|} \sum_{d_i \in D_r} d_i - \gamma \frac{1}{|D_{nr}|} \sum_{d_i \in D_{nr}} d_i$$

compute the new query vector for "soccer" using the top ranked document for pseudo relevance feedback (with $\alpha = \beta = 0.5, \gamma = 0$) and compute the new document ranking. You are not required to simplify fractions. [3 marks]

The query representation is as above $\langle 1,0,0,0,0\rangle$. Assuming that the top-ranked document is relevant, d_1 is in our D_r set. Since $\gamma = 0$, we can ignore the D_{nr} set.
Just before we substitute, we want the document vector to be of unit length, so we divide through by the length that we calculated above: $d_1 = \langle 0.6, 0, 0.8, 0, 0\rangle$. Now:

$$\begin{aligned} q_e &= \alpha q + \beta \frac{1}{1} d_1 + 0 \cdots \\ &= (0.5)\langle 1,0,0,0,0\rangle + (0.5)\langle 0.6,0,0.8,0,0\rangle \\ &= \langle 0.8, 0, 0.4, 0, 0\rangle \end{aligned}$$

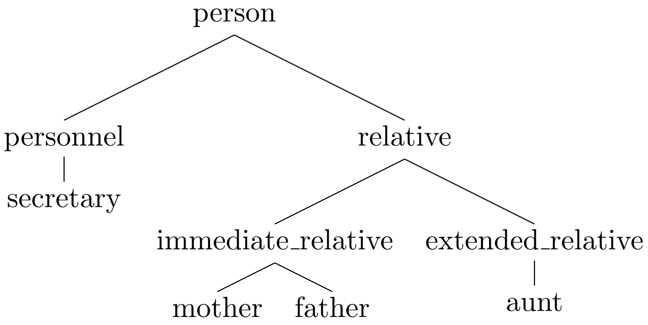
(1 mark)
As we care only about ranking, we won't bother with computing the query length (it only scales the outputs by a constant, thus ranks are not affected.)

Next compute the cosine values

$$\begin{aligned}
\cos(q_e, d_1) &\propto \frac{q_e \cdot d_1}{\mid d_1 \mid} = \frac{0.6 \times 3 + 0.8 \times 4}{5} = 1 \\
\cos(q_e, d_2) &\propto \frac{0.6 \times 0 + 0.8 \times 8}{10} = 0.64 \\
\cos(q_e, d_3) &\propto \frac{0.6 \times 1 + 0.8 \times 0}{3} = 0.2
\end{aligned}$$

(1 mark)
The ranked order of the documents is 1,2,3. (1 mark)
Where errors are made in one step, this will not be penalised in subsequent stages. E.g., getting q_e wrong but the cosine and ranking correct for that q_e can result in 2 marks.

Lexical Semantics This question is based on the following lexical hierarchy.



1. Rank the all other terms with respect to their path similarity to *relative*, and mention one result that doesn't seem reasonable. [1 mark]

   Just counting path distances in the tree, we get *immediate_relative*, *person*, *extended_relative* > *personnel*, *father*, *mother*, *aunt* > *secretary*. The word *personnel* doesn't seem as closely related to *relative* as *mother*, but they are ranked the same using this method.

2. Calculate Wu-Palmer similarity between all the leaves of the tree. You can leave the results in fraction form. [2 marks]

   To calculate Wu-Palmer, we use the depths of the nodes and the depth of their Lowest Common Subsumer (LCS), the equation is $\frac{2 \cdot depth(LCS(w_1,w_2))}{depth(w_1)+depth(w_2)}$. Note that similarity is symmetric, and many of the calculations are identical.

$$\begin{aligned}
simwup(mother, father) &= \frac{2 \times 3}{4+4} = \frac{3}{4} \\
simwup(aunt, mother|father) &= \frac{2 \times 2}{4+4} = \frac{1}{2} \\
simwup(secretary, aunt|mother|father) &= \frac{2 \times 1}{4+3} = \frac{2}{7}
\end{aligned}$$

3. In a corpus, we find that 1 out of every 16 words is a *person*, 1 out of every 128 words is a *relative*, 1 out of every 512 words is a *relative*. 1 out of every 1024 is a *mother*, and 1 out of every 512 is a *personnel*. What's the Lin distance between *mother* and *personnel* based the corpus statistics and the given hierarchy? [2 marks]

   We only care about the two words whose similarity we are calculating and their LCS, which is *person*. First we calculate the information content for these three words (the IC is equal to the negative log (base 2)) and then plug the IC into the Lin equation, which is is the same as Wu-Palmer except with IC substituted for depth:

$$\begin{aligned}
IC(person) &= -\log 1/16 = 4 \\
IC(mother) &= -\log 1/1024 = 10 \\
IC(personnel) &= -\log 1/512 = 9 \\
simlin(mother, personnel) &= \frac{2 \times 4}{9 + 10} = \frac{8}{19}
\end{aligned}$$

D: Essay

You'll be required to write about a page on one of four options. See last year's exam for some ideas.

Concluding remarks

Note that the above questions don't cover every topic in the subject (some whole areas are missing!), so please don't read too much into the areas covered. You will need to prepare on the full range of topics covered in the lectures and workshops. Note also that the exam will be significantly longer than this, particularly in parts B and C. You can get a ballpark estimate by looking at the number of marks assigned to each question versus the section totals on last years' exam (worth XX/50).

— End of Exam —