UNIVERSITY OF EDINBURGH

COLLEGE OF SCIENCE AND ENGINEERING

SCHOOL OF INFORMATICS

INFR09028 FOUNDATIONS OF NATURAL LANGUAGE PROCESSING

Monday 9th May 2016

14:30 to 16:30

INSTRUCTIONS TO CANDIDATES

Answer all of Part A and TWO questions from Part B.

Part A is COMPULSORY.

The short answer questions in Part A are each worth 3 marks, 24 marks in total. Each of the three questions in part B is worth 13 marks — answer any TWO of these.

CALCULATORS MAY NOT BE USED IN THIS EXAMINATION

Year 3 Courses

Convener: C. Stirling
External Examiners: A. Cohn, T. Field

THIS EXAMINATION WILL BE MARKED ANONYMOUSLY

Part A

Answer ALL questions in Part A.

Your answers should be thorough but they need not be lengthy: from a few words in some cases up to a paragraph in others. Each question is worth three marks, 24 marks in total for this section.

1. Describe or illustrate the essence of **Zipf's Law** as it pertains to word frequencies. What are the implications of Zipf's law for developing statistical models in NLP?

2. For an **N-gram language model** (using word N-grams to model sentence probabilities), is it *always* better, *sometimes* better, or *never* better to choose $N \geq 3$? Explain your reasoning.

3. Explain what is meant by the **bag-of-words assumption**. Give an example of a probabilistic model that makes this assumption, and a task for which that model might be appropriate.

4. In implementing probabilistic models, we often compute with **negative log probabilities** (costs) instead of actual probabilities. What are the benefits to doing so? Why *can't* we do this when implementing the forward algorithm for HMMs?

5. Suppose we train two different **character trigram models** on English text.

   The first model is trained *without padding.* That is, it defines the probability of a token w consisting of characters $c_1 \dots c_n$ as:

$$P(w = c_1 \dots c_n) = P(c_1)P(c_2|c_1)\prod_{i=3}^{n} P(c_i|c_{i-2}, c_{i-1})$$

   The second model is trained *with padding* (i.e., using explicit begin- and end-of sequence markers).

   Using each model, we then compute the **per-character cross-entropy** of tokens (1) and (2) below:

   (1) `acknowledge`
   (2) `cknowledge`

   (a) For the model *with padding*, which of these tokens is likely to have higher per-character cross-entropy? Explain your reasoning.

   (b) For token (2), *which model* is likely to have higher per-character cross-entropy? Explain your reasoning.

QUESTION CONTINUES ON NEXT PAGE

6. Consider the following two sentences:

   (1) I ate the soup in the restaurant.
   (2) I ate the soup in the pot.

   What type of **syntactic ambiguity** do these sentences illustrate? Explain how these sentences motivate the use of a **lexicalized parsing model**. (Hint: what happens if you use a non-lexicalized PCFG to parse these sentences? Compare that to either a lexicalized PCFG or dependency grammar.)

7. What is the difference between a **constituency parse** and a **dependency parse**? Provide a structural analysis of the following sentence using each method:

   The boy gave Sarah a book

   You do not need to include labels, only the structures themselves.

8. Fill in the blanks: (Write the answers in your script book, not on this paper!)

   (a) In terms of **lexical semantic** relationships, “toddler” is a _______ of “child”.

   (b) In WordNet, “child” and “kid” belong to the same _______ because in many contexts, they are _______.

Part B

Answer two questions in Part B.

1. **Language models and HMMs**

   (a) Suppose we train a **bigram language model** on a corpus where there are no occurrences of the bigram `perfume and`.

   i. What is the maximum-likelihood estimate for $P(\texttt{and}|\texttt{perfume})$? What problem(s) will this cause when using the LM? Illustrate your answer with examples. [*3 marks*]

   ii. Now consider two other estimation methods: Laplace (add-1) or backoff. Which is likely to estimate a higher value for $P(\texttt{and}|\texttt{perfume})$? Why? [*2 marks*]

   (b) Suppose we train an N-gram language model with smoothing, and then compute the per-word cross-entropy of the model on both the *training* set and on a separate *test set*. Which data set is likely to have higher cross-entropy? Why? [*3 marks*]

   (c) HMMs are sometimes used for *chunking*: identifying short sequences of words (chunks) within a text that are relevant for a particular task. For example, if we want to identify all the person names in a text, we could train an HMM using annotated data similar to the following:

   > On/**O** Tuesday/**O** ,/**O** Mr/**B** Cameron/**I** met/**O** with/**O** Chancellor/**B** Angela/**I** Merkel/**I** ./**O**

   There are three possible tags for each word: **B** marks the beginning (first word) of a chunk, **I** marks words inside a chunk, and **O** marks words outside a chunk. We also use **SS** and **ES** tags to mark the start and end of each sentence.

   Crucially, the **O** and **SS** tags may not be followed by **I** because we need to have a **B** first indicating the beginning of a chunk.

   i. Write down an expression for the probability of generating the sentence `Sally met Dr Singh` tagged with the sequence **B O B I**. [*2 marks*]

   ii. What, if any, changes would you need to make to the Viterbi algorithm in order to use it for tagging sentences with this **BIO** scheme? How can you incorporate the constraint on which tags can follow which others? [*3 marks*]

2. **Spelling correction**

(a) Give the formulation of the spelling correction task as a **noisy channel model**. Use x to represent the word that was actually typed and y to represent the intended word, and give the name for each component of the model. Which component is typically more task-specific and which part is more similar across different tasks that use noisy channel models? [*3 marks*]

(b) In class we discussed a particular noise model that was formulated as follows:

$$P(x|y) = \prod_{i=1}^{n} P(x_i|y_i)$$

where x is the observed word, y is the intended word, and the x_i and y_i are the individual characters in x and y (including the empty character) that have been aligned to each other.

What independence assumptions does this noise model make? [*2 marks*]

(c) Suppose we want to perform the related task of *text normalization* on Twitter data to make it easier for standard NLP tools to further process the data. For example, we'd like to convert a tweet like (1a) into something like (1b).

(1a) `I hate manny ppl but mi twitter ppl lovh ya !`

(1b) `I hate many people but my twitter people love you !`

Using at least two examples from this pair, explain why the independence assumptions in the simple noise model above are too strong to properly capture the changes needed for twitter normalization. [*4 marks*]

(d) Now consider the following two tweets, and imagine you are trying to get annotators to produce "normalized" versions of them.

(2) `@SwizzOnaRampage lol no comment bro... can't say if I disagree or agree. lol`

(3) `Really wish 1 of the #sixxtards won. Haha mayb a book? Wld b my 3rd copy. My 1st copy got worn out had to buy a new1`

Give two examples of difficult annotation decisions from this data: that is, tokens where annotators might disagree about the correct normalized form unless very specific guidelines are given. For each example explain briefly why it is difficult (i.e. provide two or more alternative normalizations and say why each might be reasonable). [*4 marks*]

3. **Lexical semantics**

The word “Jobs” is ambiguous between the employment sense and the name of former Apple CEO Steve Jobs. Suppose you want to be able to automatically classify whether a tweet that mentions “Jobs” is talking about Steve Jobs. For example, the first tweet below should be classified as +, and the second should be classified as −.

(+) `Apple had a third founder besides Jobs and Wozniak. Ronald Wayne had a 10% stake. He forfeited his share for a total of $2,300`

(−) `it was amazing. I had two Jobs working like 30hrs a week while attending school raised my grades 10%`

(a) One way to solve the problem would be to use an off-the-shelf (already trained) named entity recognizer or supersense tagger. If you used one of these systems with its standard inventory of tags, which tag would uniquely identify all the cases where the tagger thinks “Jobs” refers to Steve Jobs? [*1 mark*]

(b) Give one reason why this kind of off-the-shelf system might not work very well for this task. [*2 marks*]

(c) Alternatively, you could train your own supervised classifier for this task. You would need a collection of tweets containing the word “Jobs” where each tweet is annotated either as + or −. Would it be reasonable to use crowdsourcing as a way of annotating data to train the classifier? If not, explain why not, providing examples where possible. If so, explain why and give two annotation strategies that would ensure the dataset is useful. [*4 marks*]

(d) Name a supervised classification model that you could train on the labeled dataset. Name a corresponding technique to keep the model from overfitting to your training data. [*3 marks*]

(e) Rather than using supervised learning, you could treat the problem as an unsupervised distributional clustering task.

i. Show the nonzero entries of a sparse context vector for the word “Jobs” in each of the above sentences. Lowercase the words and use a window of 5 words on each side (ignoring punctuation).
Circle the features that you think would be most discriminative, i.e., would tend to occur for one sense of “Jobs” but not the other. [*2 marks*]

ii. Compute the cosine similarity between the two vectors. (You do not need to reduce your answer to a single number; answers including unreduced fractions and the like are fine.) [*1 mark*]