UNIVERSITY OF EDINBURGH

COLLEGE OF SCIENCE AND ENGINEERING

SCHOOL OF INFORMATICS

INFR11061 NATURAL LANGUAGE UNDERSTANDING (LEVEL 11)

Friday 29 th April 2016

14:30 to 16:30

INSTRUCTIONS TO CANDIDATES

Answer QUESTION 1 and ONE other question.

Question 1 is COMPULSORY.

All questions carry equal weight.

CALCULATORS MAY NOT BE USED IN THIS EXAMINATION

Year 4 Courses

Convener: I. Stark
External Examiners: A. Burns, A. Cohn, P. Healey, T. Field, T. Norman

THIS EXAMINATION WILL BE MARKED ANONYMOUSLY

1. **You MUST answer this question.**

   (a) In the course, we discussed a neural network architecture called long short-term memory (LSTM).

      i. Briefly describe this architecture using a simple example of an LSTM. Use a diagram as appropriate. [*3 marks*]

      ii. Which problem with standard recurrent neural networks (RNNs) are LSTMs designed to solve? [*2 marks*]

      iii. How could you use an LSTM for constituency parsing, i.e., to turn an input consisting of a sequence of part-of-speech tags into a phrase structure tree? In your answer focus on how you would represent the input and the output of the LSTM. [*5 marks*]

   (b) The technical term *hyperparameters* is used both for neural networks and for Bayesian models. Does it refer to the same concept in both cases? [*3 marks*]

   (c) In the course we discussed convolutional neural networks (CNNs) as a way for inducing feature representations for sentence classification tasks.

      i. How are sentences represented in order to serve as an input to a CNN? [*3 marks*]

      ii. How are feature maps produced using a convolution operation? Give two examples of filters. [*4 marks*]

      iii. Why is it necessary to apply a pooling function to the feature map? What is the most common pooling operator for NLP tasks? [*2 marks*]

   (d) Give a brief description of the gradient descent optimization algorithm. What is the difference between gradient descent and stochastic gradient descent? Which algorithm is more efficient and why? [*3 marks*]

2. Dependency Parsing

In the course, we discussed the *maximum spanning tree* (MST) algorithm for dependency parsing. In this question we will apply the algorithm to an example sentence and explore possible extensions.

(a) Explain how the MST algorithm works. Describe the data structures it uses and the main steps of the computation it performs. [*4 marks*]

(b) What are *non-projective dependency trees*? Explain how the MST algorithm handles them. [*3 marks*]

(c) Apply the MST algorithm to compute the highest scoring dependency tree for the sentence *He saw Peter yesterday*, given the graph in Figure 1 (next page). [*5 marks*]

In relation to MST-based dependency parsing, we distinguish *first order*, *second order*, and *third order* dependency parsing. First-order parsers use properties (features) of the head and the modifier of an edge only. Second-order parsers also use properties of either the sibling or the parent. Third-order parser include both. This is illustrated in Figure 2 (next page).

Assume you want to train a feed-forward neural network that scores dependency edges for the MST algorithm.

(d) Describe how you would represent the input for this neural network. Which features should the input encode, assuming you use first-order features? [*5 marks*]

(e) How does the input to the network change if the you encode second-order features? [*3 marks*]

(f) Describe the output layer of your network. What training data would you require to train the network? [*5 marks*]

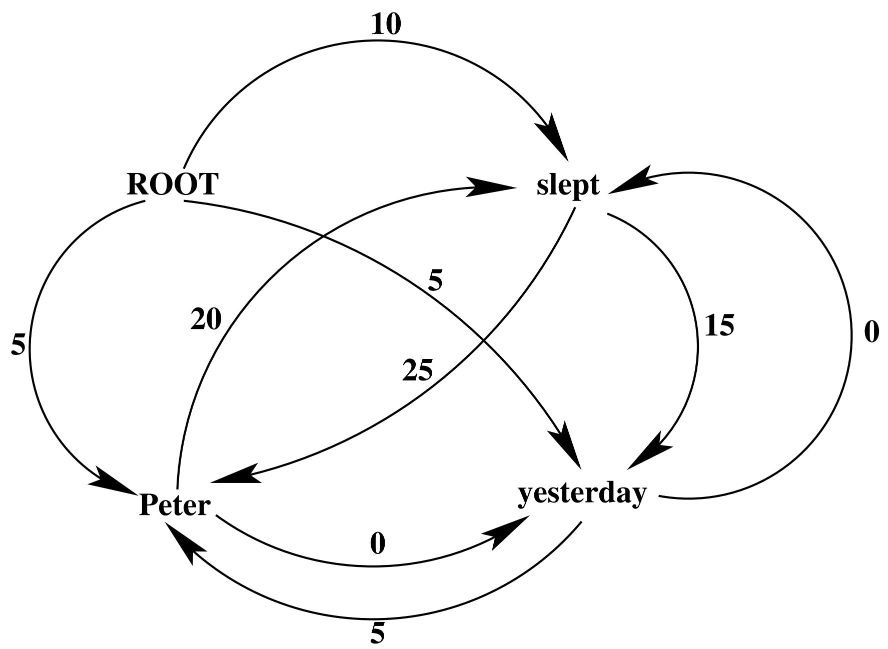

Figure 1: Graph representations of all dependency trees for the sentence *He saw Peter yesterday.*

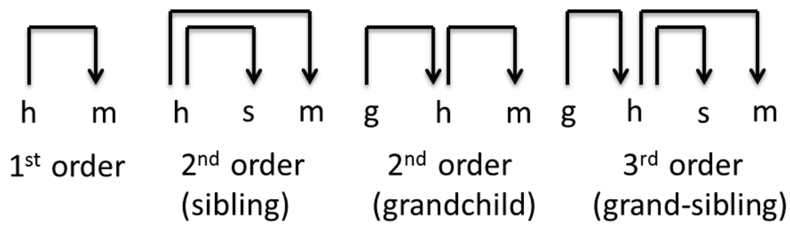

Figure 2: First, second, and third order dependency parsing; h: head, m: modifier (argument), s: sibling of the modifier, g: grandparent or the modifier.

3. The skip-gram model learns embeddings for individual words. In this model we are given a corpus of words w and their contexts c. We consider the conditional probabilities $p(c|w)$, and our goal is to set the parameters θ of $p(c|w;\theta)$ so as to maximize the probability:

$$\underset{\theta}{\operatorname{argmax}} \prod_{(w,c)\in D} p(c|w;\theta) \tag{1}$$

where D is the set of all word and context pairs we extract from the text.

(a) Write down how $p(c|w;\theta)$ is estimated using a softmax classifier. What are the parameters of this model? [*4 marks*]

(b) Why is it computationally expensive to compute the softmax objective? Give a high level description of why the negative sampling approach is a more efficient way of deriving word embeddings. [*4 marks*]

(c) What is the objective of the skip-gram model with negative sampling? Write down the appropriate formulas. [*4 marks*]

(d) Skim-gram learns representations for individual words. In many NLP applications however, we work with sentences, paragraphs, or even entire documents. How could you modify skip-gram in order to learn representations for entire documents *in addition* to embeddings for context and target words? Draw a picture of your modified skip-gram model. Now write down the negative sampling objective with your new skip-gram model. [*6 marks*]

(e) Assume you have obtained vector-based representations for document sentences e.g., using recursive autoencoders or CNNs. What is the simplest way of obtaining a document vector? [*3 marks*]

(f) Can you think of a neural network architecture which would learn compositional document representations? Specifically, this model is given sentence vectors as input and produces a fixed length document vector as output. Briefly describe the architecture of your model. Use a diagram as appropriate. [*4 marks*]