**Study Questions**

CSC/LIN 205, “Computational Linguistics”
Department of Computer Science
Grinnell College
revised December 13, 2018

**For August 31, 2018**

1. What is a *language model*? How does it differ from the *grammar* of a language?
2. Explain the nature of the disparity between natural languages as people actually use them and the formally stated problems for which algorithms provide solutions.
3. Suggest strategies for accommodating the disparity described in the preceding exercise.
4. What kinds of natural-language processing are particularly suited for an introductory course?

**For September 3, 2018**

5. Do the lab “Libraries in R^7RS.” Write a short summary confirming that you completed each step and reporting any difficulties you encountered along the way. Report the results of all exercises that perform numerical computations.

**For September 5, 2018**

6. Do the lab “Records in R^7RS.” Write a short summary confirming that you completed each step and reporting any difficulties you encountered along the way. Include the Scheme code that you wrote.

**For September 7, 2018**

7. What is Unicode? What exactly does the Unicode standard standardize? How is its work useful to computational linguists?
8. What is an encoding form? What reasons might computational linguists have for using or avoiding the UTF-8 encoding form?
9. Does R^7RS Scheme accommodate Unicode and UTF-8 encoding? To what extent does our implementation accurately meet the requirements of the *Revised 7 Report on the Algorithmic Language Scheme*?

**For September 10, 2018**

10. What are the most significant solved or mostly solved problems in computational linguistics?
11. What kinds of string-matching patterns arise frequently in practice and are typically implemented in a library for regular expressions or pattern-matching?
12. Explain the difference between a “mini-language” model of string patterns, as in Python, and a “data-structure” model, as in SNOBOL4, and describe the advantages of each.
13. Informally present a brute-force backtracking algorithm for string pattern matching.

**For September 12, 2018**

14. How do the English interjections *uh* and *um* differ in meaning, (more specifically, in the nature of the expectations that the speaker signals)?
15. What questions does one need to resolve in the course of designing and implementing a tokenizer for English text?
16. What changes would you make in the *tokenize* procedure in the (cl tokenizer) library in order to make the tokenizer recognize hyphens and apostrophes as internal to word tokens when they are immediately preceded and followed by alphabetic characters?
For September 14 and 17, 2018

17. Under what conditions would one use a hash table rather than an association list for storing key-value pairs?

18. Do the lab “Character Spectra.” Write a short summary confirming that you completed each step and reporting any difficulties you encountered along the way. Show the Scheme code that you write and the output from the last exercise in the lab.

For September 19, 2018

19. Give an example of a pair of words that are spelled identically but have different (and differently spelled) roots.

20. Suggest some purely syntactic heuristics for identifying sentence boundaries in English text.

21. The test program for the Porter stemmer at

   /home/reseda/computational-linguistics/code/tests/test-Porter-stemmer.ss

contains a lot of test cases, including many that fail. Find additional examples of “errors of commission” (cases in which the stemmer finds identical stems for unrelated words) and “errors of omission” (cases in which the stemmer finds different stems for different forms of the same word).

22. Try to make improvements in the Porter stemmer so that it passes more of the test cases.

For September 21, 2018

23. Why is dynamic programming an appropriate algorithmic design for computing minimum edit distances?

24. The Min-Edit-Distance algorithm in Figure 2.15 on page 29 of draft third edition of the textbook is implemented in Scheme in the (cl minimum-edit-distance) library at

   /home/reseda/computational-linguistics/code/cl/minimum-edit-distance.ss.

What changes in this implementation would be required if we wanted to change the cost of each deletion to 3, each insertion to 5, and each character replacement to 6?

25. What changes in the Scheme code would be required to store backpointers for each subproblem?

26. What changes in the Scheme code would be required to have minimum-edit-distance return a backtrace (or, better yet, editing instructions) along with the edit distance?

For September 24, 2018

27. The formal specification of the transition function $\delta$ in the definition of finite automata (page 28 of Jurafsky and Martin, second edition) doesn’t quite match the transition table that the authors propose because ordinarily a function cannot have “illegal or missing” values. A function with domain $Q \times \Sigma$ and range $Q$ must produce a state, a member of $Q$, for every combination of a state and a character from the alphabet. How can these different ways of describing transitions be reconciled?

28. The automaton depicted in Figure 2.16 (page 31 of Jurafsky and Martin, second edition) is non-deterministic, because in state $q_0$, on the input word twenty, there is a transition to $q_1$ and also a transition to $q_2$ (and similarly for input words thirty, forty, fifty, sixty, seventy, eighty, and ninety and for state $q_4$). Show how to construct an equivalent deterministic finite automaton by adding two more states.

29. Use the (cl fsas) and (cl hash-tables) libraries to build the Sheeptalk finite-state automaton in Scheme.
Study Questions — page 3

For September 26, 2018

30. In Figure 2.23 (page 40 of Jurafsky and Martin, second edition), there is a state labelled $q_f$ in $\text{FSA}_1$ and another with the same label in $\text{FSA}_2$. Which of these states are final states in the automaton for the concatenation language?

31. Do the lab “Compiling Finite-State Automata.” Write a short summary confirming that you completed each step and reporting any difficulties you encountered along the way. Show the state diagram that you construct and include any Scheme code that you write and the output from your programs and tests.

For September 28, 2018

32. How would the designer of a text-messaging application use a word-prediction language model?

33. Roughly, by a back-of-the-envelope calculation, how many entries would the full table of observed bigram frequencies for the data in the Berkeley Restaurant Project have, given that the corpus from which it was derived contains 9232 sentences, all of which are user queries about restaurants in Berkeley?

34. Do the lab “Building the Word-Prediction Model.” Write a short summary confirming that you completed each step and reporting any difficulties you encountered along the way. Provide any Scheme code that you write and the output from your programs and tests.

For October 1, 2018

35. How does one compute the probability of a test set under a given language model?

36. Why is the probability $p$ of a test set usually expressed as $\log p$, and calculations about such probabilities usually performed by adding logarithms rather than by multiplying the probabilities themselves?

37. What is the perplexity of the string 00020010000 in the mini-language described on page 43 of the text, in which the words are decimal digits and 0 occurs with probability 10/19 and each of the other digits with probability 1/19?

For October 3, 2018

38. How can we adjust a unigram model to assign a non-zero probability to words not occurring in the training set? (In other words, if we have a tally of the occurrences of each word-type in the training set, how should we extend or transform it in order to avoid estimating the probability of the occurrence of an unknown word as 0?)

39. How could we implement add-k smoothing in our n-gram model? In other words, what changes would we make in the actual hash table that the $\text{n-grams}$ procedure provides or in the procedures that we use to access that hash table in order to get smoothed probability estimates from it?

40. How does simple linear interpolation work (as a smoothing technique)?

For October 5, 2018

41. What are the limitations of rule-based classifiers?

42. What is the difference between generative and discriminative classifiers?

43. In sentiment analysis, where the objective is to sort texts into “positive” and “negative” classes, the classifier described by the algorithm in Figure 4.2 has to learn the polarity of each word entirely from the training set. If we had a list of “marker words” for each class, like the MPQA subjectivity lexicon mentioned in section 4.4, how could we use it to “pretrain” the classifier?
44. Consider the following artificial example: A spam filter is evaluated on twelve thousand e-mail messages, of which nine thousand are actually spam and three thousand are non-spam. Of the nine thousand spam messages, the classifier correctly identifies 8600 (the “true positives”) as spam and incorrectly lets four hundred spam messages pass through (these are the “false negatives”). Of the three thousand non-spam messages, the classifier correctly identifies 2960 as non-spam and incorrectly blocks forty.

(a) Compute the filter’s accuracy, precision, and recall.
(b) Compute the $F_1$-measure for these results.

45. Why might a spam filter be designed in such a way that its precision is greater than its recall?

46. In discussing the methodology for training and evaluating classifiers, the textbook mentions the practice of dividing a large corpus of examples (each marked with its correct classification) into a training set, a development set, and a test set. What is the point of having a development set and keeping it separate from both of the other sets?

47. Why might one want to hold back more than one test set when comparing the performance of different classification algorithms?

48. Do the lab “The Bootstrap Algorithm.” Write a short summary confirming that you completed each step and reporting any difficulties you encountered along the way. Provide any Scheme code that you write and the output from your programs and tests.

49. Use your implementation of the bootstrap algorithm to determine whether the accuracy of the naive Bayes classifier in the (cl sentiment-classifiers) library is significantly higher than the accuracy of the simplistic classifier in that library.

50. Implement the sigmoid function in Scheme and test your implementation.

51. How does one determine which features of a text should be included in the feature representation for the purpose of training a classifier based on the method of logistic regression?

52. What is the cross-entropy loss function for a classifier that uses $n$ features, with weights $w_1, \ldots, w_n$, and bias $b$, and applies them to a training set comprising $m$ items? How is it computed from the values of the features of each item in the training set and the “training label” for each item (i.e., the correct classification, as determined in advance by human classifiers)?

53. In logistic regression, how does one compute the gradient vector for the weights and the bias gradient given a new training item?

54. What is the significance of the parameter $\eta$, the “learning rate,” in the construction of a logistic-regression classifier?

55. In the definition of the gradient function in the (cl logistic-regression library, explain what the expression ‘(map (section * multiplier <> features)’ computes and what it has to do with minimizing the cross-entropy loss function.

56. State the Distributional Hypothesis and give a plausible justification for it in terms of the pragmatics of discourse.
57. What is the difference between a tf-idf model of word meaning and a model constructed by means of the word2vec algorithm?

58. In constructing a word2vec model, what is the purpose of using logistic regression to train the model? For what classification task is the model being trained?

For October 29, 2018

59. What is the XOR problem, and why can’t it be solved with a single neuron?

60. Do the lab “Neural Network Basics.” Write a short summary confirming that you completed each step and reporting any difficulties you encountered along the way. Provide any Scheme code that you write and the output from your programs and tests.

For October 31, 2018

61. Give an overview of the procedure for training a multilayer neural network to compute some function of its inputs.

62. In sections 7.4.3 and 7.4.4 of the draft third edition of the textbook, the authors demonstrate the backwards-differentiation technique in computation graphs by working through an example of the computation of the function $L(a, b, c) = c(a + 2b)$ for the inputs $a = 3$, $b = 1$, and $c = -2$, determining that in this case $\frac{\partial L}{\partial a} = -2$, $\frac{\partial L}{\partial b} = -4$, and $\frac{\partial L}{\partial c} = 5$. Use the same technique to find the partial derivatives of $L$ with respect to $a$, $b$, and $c$ for the inputs $a = -7$, $b = -1$, and $c = 3$.

(Notes: In introducing their example, the authors incorrectly give the value of $c$ as $-1$ rather than $-2$; the diagrams in Figures 7.9 and 7.10 give the correct value. Also, in Figure 7.10, the authors label the $a$ node in the computation graph with the equation ’$\frac{\partial L}{\partial a} = -2$’, which should be ’$\frac{\partial L}{\partial a} = -2$’.)

63. The authors propose an architecture for a neural-network word-prediction model that includes two hidden layers, one for generating an embedding for each word in the vocabulary and one for producing predictions given the embeddings of the input words. How many neural units are needed in the first of these layers?

For November 2, 2018

64. The authors say that one use for a hidden Markov model in computational linguistics is tagging — marking each word in a text with its syntactic category. What’s the connection? What are the values of the “hidden” variable in a hidden Markov model that is used for tagging, and what are the “observations”?

65. To set up the hidden Markov model, we need the initial probability vector $\pi$, the transition probability matrix $A$, and the observation likelihood matrix $B$. Given a training set of accurately tagged sentences, how could we derive the values of the elements of $\pi$, $A$, and $B$?

66. In the textbook’s example of a hidden Markov model, the initial probability vector is given by

$$\pi[\text{hot}] = 0.8,$$
$$\pi[\text{cold}] = 0.2.$$

The transition probability matrix is given by

$$A[\text{hot, hot}] = 0.7,$$
$$A[\text{hot, cold}] = 0.3,$$
$$A[\text{cold, hot}] = 0.4,$$
$$A[\text{cold, cold}] = 0.6,$$
and the observation-likelihood matrix by

\[
\begin{align*}
B[\text{hot},1] &= 0.2, \\
B[\text{hot},2] &= 0.4, \\
B[\text{hot},3] &= 0.4, \\
B[\text{cold},1] &= 0.5, \\
B[\text{cold},2] &= 0.4, \\
B[\text{cold},3] &= 0.1.
\end{align*}
\]

Using the forward algorithm, compute the probability of the observation sequence \(\langle 2,2,2,1 \rangle\) according to this model.

**For November 5, 2018**

67. What is the difference between a preposition and a particle? How can they be distinguished?

68. A garden-path sentence is one in which the first few words seem to imply one syntactic structure, but which, when taken as a whole, can be seen to have a very different one. Using the Penn Treebank tagset, tag the following garden-path sentence:

The soldiers marched past the reviewing stand remained in step, while those who were directed to the parade ground soon fell out.

69. One proposed criterion for distinguishing conjunctions from prepositions (considered as heads of phrases) is that prepositions can take modifiers (such as just in just behind the door) and conjunctions cannot. Can a subordinating conjunction like while or although take modifiers? If so, could most such words be reclassified as prepositions that take sentential complements? (The complementizers that and whether clearly cannot be reclassified in this way because of their special syntactic role.)

**For November 7, 2018**

70. In section 8.4.4 of the draft third edition of our textbook, the authors mention that taggers based on hidden Markov models make two simplifying assumptions about the limited dependence of tags and words on one another. Which of these assumptions is more likely to result in incorrect taggings of garden-path sentences?

71. Explain the role of the backpointer array in the Viterbi algorithm.

72. Apply the Viterbi algorithm to the simplified weather-and-ice-cream HMM described in section 6.2 of the second edition of the textbook to calculate the most probable sequence of weather states given the observation sequence \(\langle 1,3,1,2,1 \rangle\).

**For November 9, 2018**

73. In the learning phase of the construction of a transformation-based tagger, the system determines which of a large number of transformation rules would, if applied, produce the largest net increase in the number of correctly tagged words in the training corpus. What are the possible forms of these rules?

74. At what point is the learning phase of a transformation-based tagger stopped? What kind of configuration information is produced at the end of this phase?

75. How is transformation-based tagging related to early rule-based tagging programs and to taggers based on hidden Markov models? How is it likely to improve on these earlier approaches to tagging?
For November 12, 2018

76. How can a hidden Markov model be used to implement a "noisy channel" model for spelling correction?

77. Define the term confusion matrix and explain how a linguist might determine the values of the elements of a confusion matrix.

78. Some modern spelling correctors collect individual performance data from their users as they type messages and another documents and apply it to retrain their models. How could the accuracy of a "noisy channel model" be improved as one learns observes the user’s interactions with the input interface?

For November 14, 2018

79. Every context-free grammar implicitly specifies a language, that is, a set of strings of terminals that are constructed according to the productions of that language. Give a formal definition of this set. What are the exact conditions under which a particular string of terminals is a member of the language that the grammar specifies?

80. In the section on syntax in LIN 114, we learned that questions (for example) are derived transformationally from deep structures that are generated by a phrase-structure grammar. Jurafsky and Martin ignore the role of transformations, and instead provide a phrase-structure grammar that generates questions directly. Why?

81. Define the terms ‘nominal’, ‘gerundive verb phrase’, and ‘predeterminer’. Give an example (in context) of each of these syntactic categories.

November 16, 2018

82. After examining several examples of productions in the context-free grammar derived from the Penn Treebank, Jurafsky and Martin conclude that “The grammar used to parse the Penn Treebank is relatively flat.” What does that mean, and what might explain the flatness of the grammar?

83. Under what conditions are two context-free grammars “weakly equivalent”? Give an example.

84. Using the procedures exported from the (cl context-free-grammars) library, write a Scheme predicate that takes a context-free grammar as argument and determines whether it is in Chomsky normal form.

November 19, 2018

85. The code for the (cl Chomsky-normal-form) library contains several calls to a zero-argument procedure called gensym. What does this procedure do? What use does the algorithm for converting a context-free grammar to Chomsky normal form make of the values that gensym returns?

86. What is the difference between attachment ambiguity and coordination ambiguity?

87. Why might dynamic programming be a suitable algorithm-design technique for addressing the problem of constructing parse trees for a given sequence of words?

November 21, 2018

88. In the course of parsing a list of \( n \) words, the Cocke–Younger–Kasami algorithm tries to parse every sublist of the list. In what order does it parse the sublists?

89. Show all the parse trees that the Cocke–Younger–Kasami algorithm builds for the sentence ‘does she prefer a meal on the flight from Houston’, using the grammar shown on the right-hand side of Figure 11.3 of the textbook (third edition), along with the lexical rules from Figure 11.1.
90. Suggest a way to adapt the Cocke–Younger–Kasami algorithm so that it can accept grammars containing unit rules.

November 26, 2018

91. Under what conditions does the “completer” procedure in the Earley parsing algorithm add a new entry to the current chart?

92. Write out the charts for the sentence “I include a flight through Houston,” parsed according to the $L_1$ grammar using the Earley algorithm.

November 28, 2018

93. What is the purpose of dependency parsing? What is the result of a successful parsing operation of this kind?

94. Give an example of a sentence in which wh-movement produces a surface structure with a non-projective dependency tree.

95. Step through the operation of a dependency parser as it processes the sentence Does she prefer Houston to Denver? (You may act as the oracle during the parsing operation.)

96. In practice, dependency parsing uses a neural-network classifier as an oracle to determine which operation to apply at each stage. How are such classifiers trained? In particular, how is an appropriate training set prepared, even given a treebank of correctly constructed dependency trees?

November 30, 2018

97. Define the term ‘constraint-based formalism’, as it is used in the introductory section of chapter 15 in the second edition of our textbook.

98. Feature structures as a data type are defined recursively, in the sense that one feature structure can be a component of another feature structure, but the graph of the relationships among feature structures in the interior of a feature structure is not necessarily a tree. Explain how such a graph could fail to have a tree-like structure.

99. How would one construct the feature structure shown at the bottom of page 515 of the second edition of the textbook in Scheme, using the resources of the (cl feature-structures) library?

December 3, 2018

100. Review the examples of unification in the textbook and compare them with the test cases in the program

$\text{/home/reseda/computational-linguistics/code/testing/test-unification.ss}$

Briefly explain what is demonstrated by each of those examples.

101. How does the unification algorithm contribute to the enforcement of constraints in a constraint-based formalism based on context-free grammatical rules?

102. The feature structure on page 505 suggests that the value of a feature can be a list, in this case a list of feature structures. Revise the unification algorithm in the (cl unification) library to accommodate this possibility. (Hint: In order to enable the algorithm to deal with re-entrant structures that different elements of a list can share, the elements of the list must all be boxes.)

December 5, 2018

103. In the (cl unification) library, the comments integrated with the definition of the unify! procedure mention that, for performance reasons, it does not implement the “occurs check.”
What precondition does the occurs check detect? What will happen if this precondition is not satisfied when the unify procedure is invoked?

104. What is “IOB tagging”? Describe how it might be used in a named-entity recognition system.

105. Describe the Resource Description Format (RDF) and explain how it might be used to record the results of a relation-extraction system.

December 7, 2018

106. Use the command-line version of WordNet to list the synsets for the noun fortune and to find any hyponyms of the most common sense of that noun.

107. WordNet provides a gloss for each sense of a word. Describe how a rule-based disambiguator might use these glosses to determine which sense of the word is being used, given the sentence or paragraph in which it occurs.

108. (a) Exercise 19.1 in the third edition of the textbook challenges the reader to find the number of senses associated with every open-class word in a randomly chosen text and to calculate the total number of distinct combinations of senses. Carry out this exercise for the following sentence:

Making money from stolen tax records presents a slightly different set of obstacles than does profiting off stolen payment card numbers (Josephine Wolff, You’ll See This Message When It Is Too Late, MIT Press, 2018, p. 53).

(b) Exercise 19.2 then challenges the reader to identify which sense of each open-class word is the correct one in the given context. Carry out this exercise for the given sentence.

For December 10, 2018

109. Describe the use of frames in task-directed conversational agents.

110. When a task-directed conversational agent is implemented as a finite-state automaton, what do the states of the automaton represent? What do the transitions represent?

111. What are the advantages and disadvantages of an interactive task-directed conversational agent over a Web form that allows the user to supply the same information by filling in text fields, selecting items from menus, and clicking on radio buttons?

For December 12, 2018

112. Why might syntactic analysis and parsing be advantageous even in a direct-translation system in which each lexeme has its own translation procedure?

113. What demands does the “interlingua” model of machine translation make on meaning representations? What kind of a data structure might be used to represent the meaning of a sentence in such a model?

114. In preparing a training set for a machine-translation system using the “transfer” model, why is it important to infer and model the alignments between the positions of words in the source-language text and the positions of the semantically equivalent words in the target-language text?

For December 14, 2018

115. List the most important methods and concepts discussed in this course (up to a maximum of five) and briefly summarize the most important thing you have learned about each one.

116. Suggest one or more study questions that, in your judgement, should have appeared on this list but did not.

117. Suggest ways in which this course could be improved the next time it is offered.