CS371N Assignment 4: Sequence Modeling and Parsing

Academic Honesty: Please see the course syllabus for information about collaboration in this course. While you may discuss the assignment with other students, **all code you write and your writeup must be your own!**

Goals You will gain experience with basic structured prediction, including the Viterbi algorithm. You'll also look at parse trees, analyze syntactic phenomena, and reason about the behavior of PCFGs.

Note that this assignment is only worth 50 points.

Part 1: HMMs and Tagging (20 points)

Q1 (20 points) Consider the following corpus:

their/N raises/N
he/N raises/V
my/N purses/N

a) List the maximum-likelihood initial state probabilities for an HMM estimated from this data. (Hint: this should be a probability distribution over tags.)

b) List the maximum-likelihood transition probabilities for an HMM estimated from this data (including transitions to the STOP symbol).

c) Give an example of a grammatical English sentence **and its tags** using the words above that would be assigned zero probability under this HMM (assuming no smoothing is applied).

For the following question parts, assume that the log probabilities (log base 2) are as follows (these are rounded so they don't quite normalize as you would expect):

Initial	
N	-1
V	-1

	Transitions		
	N	V	STOP
y_{i-1} = N	-1	-1.5	-2
y_{i-1} = V	-2	-2.5	-0.5

			Emissions		
	their	he	my	raises	purses
N	-4	-2	-3	-3	-2
V	-5	-7	-5	-2	-4

Consider the sentence *he raises purses*

d) Draw the Viterbi chart for this data. Be sure to fill in every value. **You do not need to include a final column for the STOP symbol; just fill in the chart up through the third word.**

e) What is the highest posterior probability tag sequence for this sentence (now including the STOP transition)? **Is this sequence consistent with your interpretation of the sentence above?**

Part 2: Syntactic Parsers (30 points)

Q2 (20 points) In this part, you'll look at how parsers work in practice. You'll be using the Stanford CoreNLP constituency and dependency parsers[1] which are strong parsing models available as web demos. The constituency representations should look familiar from examples in class.

Dependency parsers produce relations between *head* words (parents) and *modifier* words (children). There are no intermediate categories like in constituency parsing; the sentence itself and its dependency arcs becomes a directed tree. Verbs are the heads of sentences. Subjects and objects of verbs are then modifiers of those verbs. Adjectives and nouns modify "head" nouns (e.g., for *art museum*, *museum* is the head and *art* is a modifier). These relationships are depicted as arrows going from parents to children.

a) For the sentences *I ate spaghetti with chopsticks* and *I ate spaghetti with meatballs*, describe the following. (1) Which of the two interpretations does the **constituency parser** choose for each sentence? Is it correct? (2) Do the same analysis for the **dependency parser**. (You should be able to understand the behavior by consulting the explanations of dependencies above.) (3) Are you surprised by the model's behavior here? Comment on what you see.

b) Find a new example that the **constituency parser** parses incorrectly. Include the example, its parse, and describe what is incorrect about the parse. Hint: you can think of what makes sentences ambiguous or try to find complicated sentences.

c) Construct (or find) an example of at least 8 words whose **constituency tree** starts as a balanced binary tree: the top production breaks the sentence exactly into two constituents of length $\frac{n}{2}$. Give the example and the top layer of the parse (the part that is balanced); you do not need to give the whole parse.

d) Construct (or find) an example of at least 8 words whose **constituency tree** is completely right-branching: the main "backbone" of the tree is a binary tree. You can disregard unary productions, so if there are unaries in the tree, but otherwise it's right-branching, that's fine. Hint: you may have to drop trailing punctuation to make this work.

e) Find a sentence that has at least five children for a single word in its **dependency parse**; perhaps try some different sentences to get a sense of how this might arise. Report the phrase involving that word and describe why this behavior arises here.

[1] `https://corenlp.run/`

Q3 (10 points) Consider the following PCFG (bracketed numbers are probabilities):

NP → NP CC NP [0.3]
NP → NP PP [0.3]
NP → NNS [0.4]
PP → P NP [1.0]

NNS → cats [1.0]
CC → and [1.0]
P → in [1.0]

Define a PCFG with these rules, the nonterminals {NP, PP, NNS, CC, P}, terminals {*cats, and, in*}, and root symbol NP.

For all question parts, provide justification so we can give partial credit as appropriate.

a) For the sentence *cats and cats*, how many valid syntactic parses (nonzero probability under this grammar) are there?

b) For the sentence *cats and cats in cats*, how many valid syntactic parses (nonzero probability under this grammar) are there?

c) The rules involving tag-word pairs in the grammar are called the lexicon. Suppose we smooth the lexicon so that all (word, tag) pairs have nonzero probability. For example, in this case [NNS → *and*] and [NNS → *in*] would be introduced to the grammar, as would similar extra rules for CC and P. Now, for the sentence *cats and cats*, how many valid syntactic parses (nonzero probability under this grammar) are there?

Deliverables and Submission

You will submit your writeup to Gradescope as a PDF or text file of your answers to the questions.