Midterm for CS378: Natural Language Processing (Fall 2021)

Instructions:

- The exam is due at **5pm CST on Saturday, October 16. Leave yourself ample time to upload it.**
- **The first thing you do should be to read and sign the honor code. You will upload this with your exam.** This exam is to be completed individually be each student. Again, **you may not collaborate with other students**! If we find out that you have done so, that will be considered a violation of the course Academic Honesty policy.
- This exam is an **open book take-home exam**. You are allowed to consult any resources that are helpful, with the exception of other people.
- Partial credit will be given for short-answer and long-answer questions, so please show work in your answers, but avoid writing essays. **You might be penalized for writing too much if it's incorrect.**
- For short-answer and long-answer questions, **please box or circle your final answer** unless it is an explanation.
- You will scan and upload your exam into Gradescope. You may type your responses, handwrite responses on a printed exam, or a mix of both. If you type your responses, please use the Gradescope interface to indicate where each question part can be found.
- If you have questions during the exam, please use a private Edstem post or directly email the course staff. Important clarifications will be posted as Canvas announcements.

Grading Sheet (for instructor use only)

Question	Points	Score
1	10	
2	24	
3	15	
4	14	
5	13	
6	11	
7	13	
Total:	100	

Name: _____

Honor Code (adapted from Dr. Elaine Rich)

The University and the Department are committed to preserving the reputation of your degree. In order to guarantee that every degree means what it says it means, we must enforce a strict policy that guarantees that the work that you turn in is your own and that the grades you receive measure your personal achievements in your classes:

By turning in this exam with your name on it, you are certifying that this is yours and yours alone. You are responsible for complying with this policy in two ways:

1. You must not turn in work that is not yours or work which constitutes any sort of collaborative effort with other students.
2. You must take all reasonable precautions to prevent your work from being stolen. It is important that you do nothing that would enable someone else to turn in work that is not theirs.

The penalty for academic dishonesty will be a course grade of F and a referral of the case to the Dean of Students Office. Further penalties, including suspension or expulsion from the University may be imposed by that office.

Please sign below to indicate that you have read and understood this honor code. If submitting electronically, you can submit any page with a text version of the honor code and a scanned or drawn signature.

Signature: _____

Part 1: Multiple Choice (10 points)

1. (10 points) Answer these questions by giving the option or options corresponding to the answer (2 points each).

_____ A(1) How many parameters are in there in a DAN when you fine-tune the embeddings? Assume V words, d-length word embeddings, one hidden layer of size h, and a binary classification output layer.

A. $O(Vd + dh)$
B. $O(Vdh)$
C. $O(dh)$
D. $O(Vd + d^2h)$
E. $O(Vd^2 + dh)$

_____ C(2) Which of the following is **NOT** a concept directly involved in parameter learning?

A. Backpropagation
B. Stochastic gradient descent
C. Viterbi algorithm
D. Maximum likelihood estimation
E. Learning rate

_____ IV(3) You are running perceptron with a constant step size (learning rate; also called α in lectures) over all updates and all epochs. What happens if you change the step size from 1 to 2? (So you're rerunning perceptron with a new constant learning rate of 2.) Assume you do not randomize the order of the examples at all and other hyperparameters are the same. **Circle or list all that apply:**

I. The algorithm may converge more quickly
II. The algorithm may converge more slowly
III. The algorithm may give a different decision boundary
IV. The algorithm may give a different set of weights

_____ III(4) Consider the noun phrase *olive oil*. Which of the following are true? **Circle or list all true statements:**

I. 'Olive' is the head of the phrase
II. A correct constituency tree would show that 'olive' is modifying 'oil'
III. A correct dependency tree would show that 'olive' is modifying 'oil'

_____ II, III(5) Which of the following is true of the logistic function in binary logistic regression? **Circle or list all that apply:**

I. The logistic function is continuous but not differentiable everywhere
II. The logistic function accepts any real number as input
III. The logistic function's outputs are in the range $(0, 1)$
IV. The logistic functions outputs are in the range $(0, \infty)$

Part 2: Short Answer (24 points)

2. (24 points) Answer the following short-answer questions. **Showing work may enable you to receive partial credit, but you are not required to show work.**

   a. (4 points) Suppose you are trying to do sentiment analysis in Czech using methods similar to Assignment 1. Czech has two important properties compared to English:

   1. Freer word order: There are fewer grammatical constraints on the order in which words can appear; e.g., certain words can be put at the beginning of the sentence to emphasize them.[1]
   2. Richer morphology: Unlike English words like *arrive* which only have a few conjugations (*arrives*, *arrived*), Czech words can show up in many possible conjugations with different suffixes.

   Describe how your unigram LR model for Assignment 1 needs to change (if it needs to) for a language with freer word order.

   It doesn't: unigram LR does not depend on word order. Answers about lowercasing due to different orders were accepted, although this is a minor difference. Other feature changes or suggestions lost points if they didn't directly address the question (e.g., actually answered question b or something)

   b. (4 points) Continuing on the example from above, describe how your unigram LR model for Assignment 1 needs to change (if it needs to) for a language with richer morphology. That is, in this language, a single base verb might show up spelled with, say, 10 different suffixes depending on the context.

   Stemming is the best way to handle this, as it canonicalizes across the different forms.

[1] See `https://en.wikipedia.org/wiki/Czech_language#Sentence_and_clause_structure` for more information if you're interested.

c. (4 points) Suppose we have these trees:

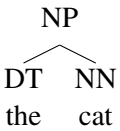
What grammar would be estimated from this? Write down the **rules and their probabilities**. You don't need to write down the nonterminals, terminals, or the start symbol. (Hint: the nonterminals are VP, VB, NP, and NN, and each nonterminal has at least one rule with that nonterminal on the left hand side.)

VP $\rightarrow$ VB (0.5)
VP $\rightarrow$ VB NP (0.5)
NP $\rightarrow$ NN (1)
VB $\rightarrow$ go (1)
NN $\rightarrow$ team (1)

d. (3 points) Suppose we have this grammar:

NP $\rightarrow$ DT NN 0.3
NP $\rightarrow$ NNS 0.7
DT $\rightarrow$ the 0.4
DT $\rightarrow$ a 0.6
NN $\rightarrow$ cat 0.5
NN $\rightarrow$ dog 0.5
NNS $\rightarrow$ cats 0.5
NNS $\rightarrow$ dogs 0.5

What is the probability of



under this grammar? You do not need to simplify.

$0.3 \times 0.4 \times 0.5$

e. (3 points) Continuing on the above example, suppose we apply smoothing to the lexicon in this grammar, but only for the nouns. That is, we allow any of the words with rules for either NN or NNS to be produced by *either* NN or NNS. Give a tree that exists under the new smoothed grammar but not under the original.

[NP [DT the] [NNS cats]]

f. (2 points) Imagine you're building a POS classifier that's also designed to do NER. For example, instead of tagging

```
NNP     NNP
Barack Obama
```

we instead tag

```
NNP-PER NNP-PER
Barack  Obama
```

The **only** tag that we apply these refinements to is the NNP tag. Assume that the NNP tag *only* shows up as part of named entities, and no other tags do. (In reality, other tags can show up in cases like *United States **of** America*, but we'll ignore those for now.)

If there are k POS tags and l NER tags, what is the number of tags in this joint tagging model?

$k + l - 1$. Note that -1 since you lose the base NNP tag.

g. (4 points) Following on the previous question, suppose you have a dataset D_1 of data that is jointly labeled with POS and NER and another dataset D_2 that is only labeled with POS. Describe how you would use **both** datasets while doing parameter estimation (the counting and normalizing procedure) for your HMM. Be as precise as you can; your answer should be 1-3 sentences.

The intended solution was to use counts from D1 and D2 for anything not involving NNP, and use only counts from D1 for tags involving NNP, to get a joint tagger. That is, exclude any transition or emission counts in D2 involving NNP from the counts. Ignoring NER entirely is possible, but this is

not consistent with the goal of (f-g) which is to also do NER. We also accepted answers that involved stapling estimates of NER tags into D_2 *if these estimation procedures were strong.* They should use an actual tagger from D_1 or something like expectation-maximization (although you haven't been taught this) rather than simpler heuristics like distributing probability mass uniformly over NER tags, though we gave significant partial credit for that idea.

Part 3: Long Answer (62 points)

3. (15 points) Suppose you have the following examples for multiclass classification with three classes:

   $\mathbf{x}^{(1)} = (1, 0), y^{(1)} = 1$
   $\mathbf{x}^{(2)} = (0, 1), y^{(2)} = 2$
   $\mathbf{x}^{(3)} = (1, 1), y^{(3)} = 3$

   a. (4 points) Run multiclass perceptron on this data with $\mathbf{w}_1$, $\mathbf{w}_2$, and $\mathbf{w}_3$ initialized to all zeroes. Use a step size of 1. Break ties in the order 1-2-3; that is, if classes 2 and 3 are tied for the highest score, return class 2. What are the weight vectors after one iteration through the data in the order specified above? (Note that you may use code to check your work on this question if you'd like, but you do not need to.)

   Ex 1 correct, Ex 2 wrong yielding w1 = [0, -1], w2 = [0, 1]; Ex 3 wrong yielding w2 = [-1, 0], w3 = [1, 1].

   Final vectors: w1 = [0, -1], w2 = [-1, 0], w3 = [1, 1]

b. (2 points) Do these weight vectors obtained after one epoch of updates correctly classify all three examples in the training data?

No

c. (5 points) Suppose we change the multiclass perceptron update so that it only adds the current feature vector to the correct class, but does *not* subtract the feature vector from the erroneously predicted class.[2]

Run multiclass perceptron on this data starting again from $\mathbf{w}_1$, $\mathbf{w}_2$, and $\mathbf{w}_3$ initialized to all zeroes. Run the algorithm until convergence and report the final weight vectors **or** state that it does not converge.

Takes one iteration + revisiting example 1 on the second iteration. Final weight vectors equal the points from each corresponding class (this is a coincidence, not something fundamental about this variant of the algorithm).

d. (4 points) Now consider doing **binary** perceptron (the basic, unmodified algorithm, not using the change in part(c)) with the points:

$\mathbf{x}^{(1)} = (1, 0), y^{(1)} = +$
$\mathbf{x}^{(2)} = (0, 1), y^{(2)} = -$

Initialize the weight vector to zeros. Consider ties to be broken in favor of the **negative class**. Modify $\mathbf{x}^{(1)}$ so that the algorithm takes more than 10 passes through the data to converge when using a constant step size of 1.

$\mathbf{x}^{(1)} = (1, 100)$ works. A big update then has to be slowly scaled back over many epochs.

4. (14 points) Here are several instances of verbs followed by either prepositions (IN) or particles (RP), with the correct tag in brackets following the word of interest:

[2] See `https://www.cs.utexas.edu/~gdurrett/courses/fa2021/lectures/lec4-notes.pdf`; delete the $\mathbf{w}_{y_{\text{pred}}}$ line of that update to get the algorithm we're talking about here.

```
made up[RP] the story
bought up[IN] to $5 million in real estate
rolling out[RP] a new policy
rolling out[IN] of the car
pick up[RP] the pace
pick up[IN] to three toppings
fight back[RP] a headache
fight back[IN] to back with Bruce Lee
```

Suppose that we want to classify these indicated words as either prepositions or particles. This can be formulated as a binary classification problem: the inputs are $\mathbf{x}$, the sentence in question (not including the bracketed gold label), and i, the index of the preposition or particle. Features are extracted by a function $f(\mathbf{x}, i)$ for that particular index.

a. (3 points) We discussed using a "current word" indicator in class, where given the input $f(\mathbf{x}, i)$, the feature `CurrWord=[`x_i`]` is extracted and can be treated like a "word" in a bag-of-words model. Based on the data shown above, do you think a model using *only* this current word feature (i.e., an indicator on the preposition/particle) will do well at this task? Explain why or why not in 1-2 sentences.

No; the current word is just the preposition/particle by itself and has no context to disambiguate which it is

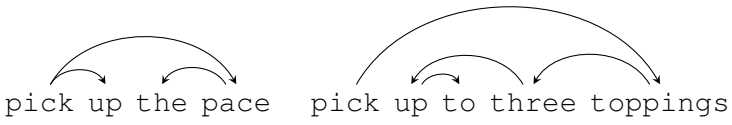
b. (3 points) Suppose we have a "previous word" feature as well. Based on the data shown, do you think adding this feature to the model will substantially improve over using only the current word feature? Explain why or why not in 1-2 sentences.

No; the verbs themselves aren't enough to disambiguate

c. (4 points) Feature *conjunctions* are when a single feature references multiple parts of the input beyond a single word, similar to bigram features in Assignment 1. Describe **two** specific feature conjunctions that you believe can be effective features for this problem, and provide a 1-sentence justification for each.

Curr + next word, curr + following two words, trigram centered on current word were the most popular answers. These will help because they can capture the following context of the RP/IN, which is very important.

c. (4 points) Suppose you had an unlabeled dependency parse (just the structure, no labels or POS tags). Here are some examples:



Describe **two** additional features using the dependency parse that could help for our classification task. State these an indicator features: for example, `PrevWord=[word at i-1]` would be how we describe the previous word feature. To reference neighbors in the dependency parse, you can use `parent(i)` (returns a single index for the parent of word i), `children(i)` (returns a potentially empty set of child indices), and `siblings(i)` (returns `children(parent(i))`).

siblings(i) is reasonable, children(i) or num children. parent(i) also helps in these cases

5. (13 points) Suppose we have the following data to learn skip-gram embeddings from:

```
a longhorn
a great
an aggie
an annoying
```

a. (5 points) Suppose we are learning embeddings with dimension $d = 4$ and have fixed the **word** embedddings to be a = (1, 0, 0, 0), an = (0, 1, 0, 0), longhorn = great = (0, 0, 1, 0), aggie = annoying = (0, 0, 0, 1).

Write down a set of context embeddings that optimize the skip-gram objective on this data.

longhorn = great = (100, 0, 0, 0), annoying = aggie = (0, 100, 0, 0), a = (0, 0, 100, 0), an = (0, 0, 0, 100). Follows the similar example in A2.

b. (2 points) Describe in a sentence why these word and context embeddings could be considered biased.

High similarity between annoying and aggie.

Now suppose we have n-dimensional embedding vectors where

```
a = [1, 0, 0, ...]
an = [-1, 0, 0, 0...]
Texas = [1, 0.5, 0, ...]
A&M = [-1, -0.5, 0, ...]
longhorn = [0.5, 0.5, 0.5, 0, 0, 1]
great = [0.5, 0.5, 0.5, 0, 1, 0]
aggie = [-0.5, -0.5, -0.5, 1, 0, 0]
annoying = [-0.5, -0.5, -0.5, 0, 1, 0]
```

One method of removing bias is to project the vectors onto an "unbiased" subspace.

c. (3 points) Suppose we project the model onto the orthogonal subspace of the a/an vectors. That is, we zero out the first coordinate. Do these vectors still exhibit biased correlations by your criterion in part (b)? Why or why not?

Still biased, there's still similarity between annoying and aggie.

d. (3 points) Now suppose we zero out the first two coordinates. Do these vectors still exhibit biased correlations? Why or why not?

Still biased, there's still similarity between annoying and aggie (third coordinate).

6. (11 points) Imagine you are trying to use an HMM to POS tag sentences, but you are running your tagger over a very large dataset and you need your system to be very fast. You want to return an answer as quickly as possible.

   Let n denote the length of a sentence being tagged and T denote the number of possible tags.

   a. (2 points) What is the asymptotic big-O runtime of the greedy algorithm (beam search with beam size = 1)?

   $O(nT)$

   When there are a large number of tags, even the greedy algorithm could be slow. Suppose you have a fast scoring function s that quickly returns a set of tags that are possible for a given word; essentially, this gives a coarse estimate of the emission score. Assume there are $k < T$ tags returned for each word and that s operates in $O(k)$ time.

   b. (3 points) Describe in one or two sentences how you incorporate s into the greedy algorithm.

   Only consider the tags that are returned by s

   c. (2 point) What is the asymptotic big-O runtime of the greedy algorithm if you incorporate s in an optimal fashion?

   $O(nk)$

   d. (2 points) What is the asymptotic big-O runtime of the Viterbi algorithm if you incorporate s in an optimal fashion?

   $O(nk^2)$

   e. (2 points) Assume that there is some probability p that the optimal tag is deleted at each timestep. For a sequence of length n, which is the probability that the hypothesis returned by greedy inference remains the same?

   $(1-p)^n$

7. (13 points) Consider the following grammar:

DT → the 1.0
NN → cat 0.5
NN → rat 0.5
VBD → chased 0.25
VBD → attacked 0.25
VBD → scared 0.25
VBD → ate 0.25

S → NP VBD 1.0
NP → DT NN 0.5
NP → NP S 0.5

a. (5 points) What is the parse of *the rat the cat chased ate*? Hint: there is only one valid parse of that sentence under this grammar. (Note that this is actually a grammatical English sentence; read it as *the rat that the cat chased ate*, so a rat is eating something, with the *that* removed to form a reduced relative clause.)

Everyone got this right, so I won't type out the tree

b. (4 points) What is the parse of *the rat the cat the cat scared chased attacked*? Hint: instead of building the whole CKY chart, try instead reasoning about the grammar in order to parse this.

Take the parse from part (a) and just repeat the structure again

c. (4 points) Suppose we remove the last three rules from the grammar and make them:

S → NP VBD
S2 → NP2 VBD
NP → DT NN
NP → NP2 S2
NP2 → DT NN

List one sentence that is permitted under the original grammar but not this modified one.

The sentence from part (b) works. This grammar limits the recursion depth by introducing the new symbols.