Midterm for CS388: Natural Language Processing (Fall 2022)

Instructions:

- The midterm will be live Thursday, September 29 12pm - Sunday, October 2 6pm. You will have 150 minutes (2.5 hours) to complete and upload the exam once you start it.
- **The first thing you see on Gradescope will be the Honor Code, which you will read and (electronically) sign.** 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 long-answer questions, **please box or circle your final answer** unless it is an explanation.
- Multiple-choice and short-answer questions will be entered directly into Gradescope. Long-answer questions should be uploaded as PDFs. You may type your responses to these questions, handwrite responses on a printed exam, or anything else.
- Because of the asynchronous nature of the exam, **we cannot clarify anything in the exam.**

Grading Sheet (for instructor use only)

Question	Points	Score
1	30	
2	16	
3	12	
4	17	
5	16	
6	9	
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.

Signature: _____

Part 1: Multiple Choice / Short Answer

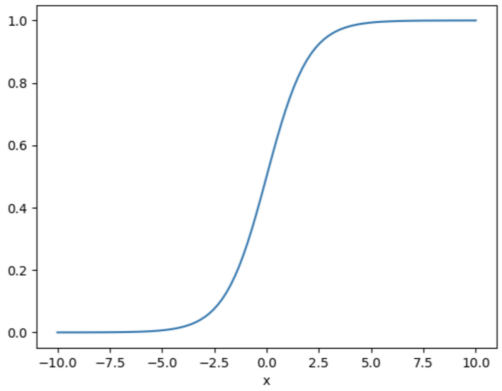
1. (30 points) Answer these questions by giving the option or options corresponding to the answer (2 points each unless otherwise specified). **If given letter options, give exactly one answer. Otherwise, carefully read the instructions on the question.**

_____ C(1) Select the answer that includes all the skip-gram (word, context) training pairs for the sentence *the cat ran away*, for a window size +/- 2 from target.

A. [the, cat], [the, ran], [cat, ran], [cat, away], [ran, away]
B. [the], [cat], [ran], [away]
C. [the, cat], [the, ran], [cat, the], [cat, ran], [cat, away], [ran, the], [ran, cat], [ran, away], [away, cat], [away, ran]
D. [the, ran], [ran, cat], [cat, away]
E. [the, cat ran], [cat, the], [cat, ran away], [ran, the cat], [ran, away], [away, cat ran]

_____ D(2) In an HMM, the tag transition probabilities can be interpreted as:

A. The probability of seeing a word given a POS tag
B. The probability of seeing a POS tag given a word
C. The probability of seeing a POS tag given all prior tags, not conditioned on the words
D. The probability of seeing a POS tag given the preceding tag, not conditioned on the words
E. The probability of seeing the next POS tag given the current word



_____ B(3) What is depicted in the graph above?

A. A tanh function: as x approaches negative infinity, the function converges to 0, and as x approaches infinity, the probability converges to 1.
B. A sigmoid function: as x approaches negative infinity, the function converges to 0, and as x approaches infinity, the function converges to 1.
C. A sigmoid function: as x approaches negative infinity, the function converges to 1, and as x approaches infinity, the function converges to 0.
D. A tanh function: as x approaches negative infinity, the function converges to -1, and as x approaches infinity, the probability converges to 1.

_____ I, III, IV, V(4; 4 points) Suppose you are training a unigram bag-of-words model. Which of the following preprocessing techniques will *reduce* the size of your feature space? **Circle or list all that apply:**

I. Lowercasing the data
II. Using bigram features instead of unigram features
III. Stemming (removing suffixes from words like *-ed* from verbs or *-s* from plural nouns)
IV. Removing stopwords (*the*, *of*, etc.)
V. Tokenizing punctuation (splitting *great!* into *great !*) this removes great! great. great, etc.
VI. Using tf-idf weighting

_____ A(5) In *Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings*, what is the specific kind of gender bias Bolukbasi et al. are seeking to remove? (Reminder that you may look at the paper or lecture videos to answer this.)

A. Correlation with a *he-she* axis in word embedding space
B. Greater frequency of masculine-gendered words in text
C. The existence of pronouns in a word embedding space
D. Associations with gendered roles like *king* and *queen*

_____ B(6) You want to use word embeddings and a FFNN (feed-forward neural network) to predict positive or negative sentiment of a sentence. Which of the following is the best implementation of this FFNN assuming that other code around it is written correctly? The brackets [input x output] indicate the dimensions of each transformation.

A. 300-dim embedding layer → [300 x 2] Linear layer → ReLU → [2 x 2] Linear → softmax
B. 50-dim embedding layer → [50 x 75] Linear layer → Tanh → [75 x 2] Linear → log-softmax
C. 300-dim embedding layer → [50 x 100] Linear layer → Tanh → [100 x 2] Linear → softmax
D. 50-dim embedding layer → [50 x 150] Linear layer → [150 x 2] Linear layer → softmax
E. 300-dim embedding layer → [300 x 200] Linear layer → Tanh → [200 x 2] Linear → Tanh → softmax note the Tanh before softmax – this is very bad to do because it squashes your activations and prevents the model from maxing out the probability of the right answer

_____ III, IV(7) In the forward-backward algorithm, which of the following *on its own* can safeguard against double counting emissions? **Circle or list all that apply:**

I. Exponentiating emission and transition features
II. Summing in the recurrence step instead of maxing as we do in the Viterbi algorithm
III. Counting emission for the current step in the forward recurrence and the next step in the backward recurrence
IV. Normalizing the marginal distribution at each timestep once forward and backward are combined quite tricky but this also does do the trick because the double-counting is a constant factor for each term in the marginal
V. Storing values as log probabilities.

The following question parts reference the HMM whose probabilities are given in the following tables:

Initial	
J	2/3
N	1/3

Transitions	J	N	STOP
y_{i-1} = J	1/2	1/4	1/4
y_{i-1} = N	0	1/2	1/2

Emissions	big	red	convertible	car
J	1/2	1/4	1/4	0
N	0	0	1/3	2/3

_____ 1/36(8; 3 points) What is the probability of the sequence *J/big N/car*? Please write your answer as a single fraction or decimal (all terms multiplied together; you may use a calculator). Note that no answer options are given for this question.

_____ 2(9; 3 points) How many possible tag sequences are there for the sentence *big convertible car*? Answer with a single nonnegative integer. Note that no answer options are given for this question.

_____ 1(10) How many possible tag sequences are there for the sentence *big car convertible*? Answer with a single nonnegative integer.

Now consider the following vectors:

good = [0, 0, 3]
bad = [0, 1, 3]
the = [10, 10, 10]

_____ C(11) Report which two words are most similar using **dot product similarity** (taking the dot product of the two vectors).
A. (good, bad)
B. (good, the)
C. (bad, the)

_____ A(12) Report which two words are most similar using cosine similarity, which is defined as

$$\cos(a, b) = \frac{a \cdot b}{|a||b|}$$

(the dot product normalized by the lengths of the vectors).
A. (good, bad)
B. (good, the)
C. (bad, the)

_____ A(13) Now suppose that all of the vectors are normalized to have length 1. Using the new vectors, report which two words are most similar using dot product similarity. You only need to give the word pair.
A. (good, bad)
B. (good, the)
C. (bad, the)

Part 2: Short Answer

2. (16 points) Answer the following questions.

a. (4 points) In one or two sentences, briefly explain the difference between the skip-gram and continuous bag-of-words methods.

In CBOW, surrounding context words are all used at once to predict the current word. In skip-gram, the input word is used to predict each context word one at a time.

b. (4 points) List **one advantage** and **one disadvantage** of treating sequence labeling as structured prediction (e.g., using a CRF) as opposed to treating it as label-wise classification (e.g., using a classifier to predict a tag for each word)?

Advantage: CRFs allow us to capture dependencies between adjacent pairs of labels (this is really the only main advantage). Disadvantage: there are a few possible ones here. The intended response was that CRFs have higher computational complexity for training due to needing to use forward-backward to compute gradients.

c. (4 points) Suppose you train a unigram bag-of-words model. You're running it in production for a few years, then you realize that a new slang word has emerged that your model hasn't seen before but is very negative in sentiment. You come across a few examples of this word being used in negative sentences in the wild. Name **two ways** you can update your model to do better on such sentences.

Several answers are possible. The intended answers were something like 1. Add those sentences as labeled data and retrain on them for a few epochs. 2. Simply add in the appropriate feature with a high negative weight.

d. (4 points) You come across text in a new dialect of English. In addition to modifications to spelling and morphology, sentences are written by composing each sentence, then alphabetizing them. Which type of parsing, dependency or constituency, is most appropriate for analyzing its syntax? Explain in 1-2 sentences.

Dependency parsing is better. The word of the sentences will be highly variable and won't respect the rigid constraints that constituency expects, whereas dependency can be relaxed to support arbitrary word orders quite easily.

Part 3: Long Answer

3. (12 points) Suppose you have the following training points for classification; points are listed as (x_1, x_2, y):

   (1, 0, +)
   (1, 1, +)
   (0, 1, −)

   a. (6 points) Execute one pass of perceptron on this data, starting from a weight vector initialized at 0 and using the decision rule of $\mathbf{w}^\top \mathbf{x} > 0$ (that is, classifying as positive if the score is greater than 0). What is the final weight vector you get? **Box your final answer.**

   [1, 0]. We get an update on the first one and that's it.

   b. (4 points) Suppose we add a 4th point to the training set with a negative class label. Mathematically describe the region in the $\mathbf{x}$ feature space where, if this example is added, the perceptron *will not converge*. You may either give your answer as a set of mathematical constraints, draw a picture, or describe in words, as long as you are precise.

   $x_2 \leq x_1$ and $x_1 > 0$. Think about rotating the decision boundary from the origin like a "clock". It basically can go between vertical ("noon") and 45 degrees to the right ("1:30"), with negative on the left of the boundary. This accommodates negative points anywhere on the left ($x_1 \leq 0$) as well as in the first quadrant but above the $x_2 = x_1$ line.

   c. (2 points) Now suppose we add the point (4, 2, -). Does there exist a neural network (with any depth, any nonlinearity, and any width) that would classify all of these points correctly? Briefly justify your answer.

   Yes, neural networks are universal function approximators and there exist functions that separate these

4. (17 points) Consider the following PCFG with S as the root; probabilities are given in parentheses:

   Non-terminal productions:
   (1.0) S → PRP VP
   (1.0) VP → V NP
   (1.0) NP → J N

   Pre-terminal productions:
   (1.0) PRP → I
   (0.5) V → like
   (0.5) V → eat
   (0.25) J → green
   (0.25) J → red
   (0.25) J → verde
   (0.25) J → roja
   (1.0) N → salsa

   Note that *roja* and *verde* (both Spanish) mean *red* and *green*, respectively. *salsa* is originally a Spanish word but we will consider it as both English and Spanish for the purposes of this problem.

   a. (4 points) Give the parse for the sentence *I like green salsa*. (We strongly prefer if you can draw the tree, but if you prefer you can write it with brackets.)

   (S (PRP I) (VP (V like) (NP (J green) (N salsa))))

   b. (3 points) What probability does the grammar assign to this parse? You should give either a fraction or a decimal as the answer. **Box your final answer.**

   1/8

   c. (3 points) Add one rule to the grammar that allows you to parse the sentence *I like salsa verde*. You should add a **nonterminal** production, not a pre-terminal production (i.e., do not add a rule that ends in a word). You do not need to update probabilities in the grammar.

   NP → N J

d. (3 points) Give an ungrammatical sentence that this grammar now generates that was not generated previously.

I like salsa red

e. (4 points) We need to make bigger changes to the grammar to make it work for both Spanish (*salsa verde*, *salsa roja*) and English (*green salsa*, *red salsa*). Rewrite more of the grammar (both pre-terminal productions and non-terminals) to split out NP-EN and NP-ES (English and Spanish noun phrases) to accept the English and Spanish examples correctly.

NP_en → J_en N_en
NP_es → N_es J_es
J_en → green
J_en → red
J_es → verde
J_es → roja
N_en → salsa
N_es → salsa

Splitting out the N isn't necessary; we accepted both solutions

5. (16 points) Suppose you want to embed bigrams instead of words. Recall the skip-gram model:

$$P(\text{context} = y \mid \text{word} = x) = \frac{\exp(\mathbf{v}_x^\top \mathbf{c}_y)}{\sum_{y' \in \text{vocab}} \exp(\mathbf{v}_x^\top \mathbf{c}_{y'})}$$

a. (3 points) In your first attempt, you are going to learn a bigram vector for every bigram. That is, for a sentence like *the cat saw*, you will form word-context pairs (b(the cat), u(saw)), (b(cat saw), u(the)), where b and u denote bigram and unigram respectively (for maximal clarity). **Note that contexts are still unigrams.**

With $\mathbf{v}$ and $\mathbf{c}$ denoting the word and context vectors respectively, write out the model for skip-gram with word bigrams and context unigrams: $P(\text{context} = y \mid \text{word} = b)$ (b denotes a bigram).

$P(y|b) = \frac{\exp(v_b^T c_y)}{\sum_{y' \in V} \exp(v_b^T c_{y'})}$

b. (2 points) What is the big-O runtime of computing the probability for a single bigram-context pair? Express this answer in terms of $|V|$, the vocabulary size, $|B|$, the number of bigrams attested in the data, and d, the dimension of the vectors.

Big-O notation involves giving the runtime independent of small constants: for example, taking the dot product of two d-length vectors is an $O(d)$ operation because the time it takes is proportional to the length of the vectors, as each pair of numbers has to be multiplied and then summed together. Assume you are describing the total number of operations and there is no parallelization available.

$O(|V|d)$ ($|V|$ terms in the denominator each taking $O(d)$ time)

c. (2 points) Assume that making a single gradient step on a single example takes the same time as what you reported in part (b). What is the big-O runtime of model training for M (bigram, context) examples on N epochs? Express this in terms of your answer to part (b), which you can denote as E. (You may also use the other quantities defined in the part (b) question as needed.)

$O(MNE)$: MN gradient updates each of complexity E (gradient takes the same time as computing a single probability because the denominator is such a bottleneck)

d. (2 points) How many parameters are in the model? Express this in terms of the quantities in part (b).

Your answer to this part should be exact.

$Bd + Vd$; bigram vectors and context vectors

e. (3 points) In your second attempt, you are going to instead model each bigram as a sum of the two words. That is, instead of having a vector for b(cat, saw), we instead represent this as v_{cat} + v_{saw}. Contexts are still unigrams.

Modify the skip-gram formula accordingly and write the formula below, introducing notation as needed.

$P(y|b) = \frac{\exp\left((v_{b_1} + v_{b_2})^\top c_y\right)}{\sum_{y' \in V} \exp\left((v_{b_1} + v_{b_2})^T c_{y'}\right)}$

f. (2 points) What is the big-O runtime of computing the probability for a single bigram-context pair under this new scheme? Express this in terms of the quantities in part (b).

$O(Vd)$

g. (2 points) How many parameters are in the model? Express this in terms of the quantities in part (b).

$2Vd$. Fewer parameters in this one.

6. (9 points) Suppose you have trained a Conditional Random Field model, but you realize one of the test sentences is unlabeled.

   a. (3 points) Tag the following sentence using NER and the BIO tagset and the classes PERSON, LOCATION, ORGANIZATION, and OTHER:

   It has been 53 years since NASA astronauts Neil Armstrong and Buzz Aldrin landed on the Moon

   O - O - O - O - O - O - B-ORG - O - B-PER - I-PER - O - B-PER - I-PER - O - O - O - B-LOC

   Now suppose you run the sentence from part a through the model, it returns the following prediction:

```
It has been 53 years since NASA astronauts Neil Armstrong and Buzz Aldrin landed on the Moon
O   O   O   O    O     O    O       O     B-PER  I-PER    O  O    B-PER     O    O  O  B-ORG
```

   b. (3 points) Calculate the **precision and recall** of named entity **spans** on this example. (This is the standard NER evaluation.)

   precision = 1/3, recall = 1/4. Note that these are defined in terms of *spans*, not in terms of individual tags. So the question is how many spans are predicted correctly, in the prediction, and in the ground truth.

   c. (3 points) Name **two ways** that you could modify your CRF model to increase **recall** (in general, not necessarily on this example). These could be changes to the model or to how it's trained.

   Weight positives more highly in the objective during training, add a bias to O after training. Many answers were accepted here, but generic suggestions like "change learning rate" were not.