Midterm for CS378: Natural Language Processing (Fall 2022)

Instructions:

- You will have 80 minutes to complete the exam.
- This exam is to be completed individually be each student.
- You are allowed one 8.5”x11” double-sided note sheet.
- You are **not** allowed calculators or other electronic devices.
- Partial credit will be given for short-answer and long-answer questions, so please show work in the exam.
- For short-answer and long-answer questions, **please box or circle your final answer** (unless it is an explanation).

Grading Sheet (for instructor use only)

Question	Points	Score
1	40	
2	15	
3	15	
4	16	
5	14	
Total:	100	

Name: ____

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

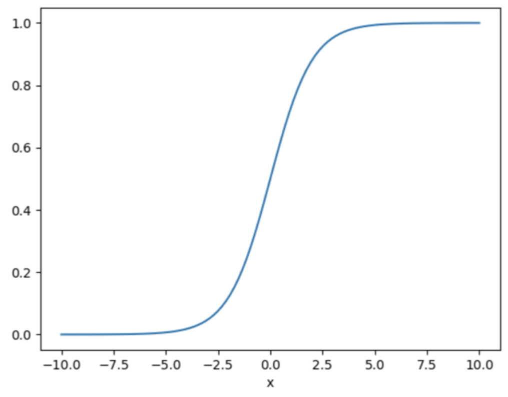
1. (40 points) Answer these questions by giving the option or options corresponding to the answer (3 points each unless otherwise specified). **If given letter options, give exactly one answer. Otherwise, carefully read the instructions on the question.**

_____ 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 $k = 2$ 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]

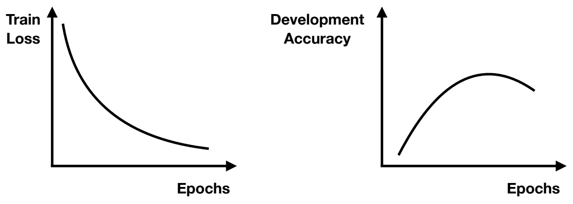
_____ D(2) In an HMM, the tag transition probabilities can be interpreted as:

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.



______ II, III, IV(4) In the figure above, we see train loss and validation accuracy plotted for a model. Which of the following statements are true? **Circle or list all that apply:**

I. The model is underfitting the training dataset
II. The model is overfitting the training dataset
III. Decreasing the step size and retraining the model for the same number of steps could lead to better performance
IV. Changing the model (e.g., from a linear classifier to a neural network) could lead to better performance

______ I, III, IV, V(5; 6 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 !*)
VI. Using tf-idf weighting

______ 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

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(7; 4 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(8) 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(9) 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(10) 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(11) Report which two words are most similar using cosine similarity, which is defined as

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

(the cosine of the angle between the two vectors).
A. (good, bad)
B. (good, the)
C. (bad, the)

_____ A(12) 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. (15 points) Answer the following questions.

a. (5 points) List **one advantage** and **one disadvantage** of treating sequence labeling as structured prediction (e.g., using an HMM) 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. Disadvantage: CRFs have higher computational complexity for training due to needing to use forward-backward to compute gradients

b. (5 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.

The best answers were: 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. 3. Replace the word with a similar negative word from the vocabulary. Decomposing the slang word may not always be possible or easy. Using bigrams won't obviously help. The fact that it's negative does not need to be counterbalanced or anything.

c. (5 points) In 1-2 sentences, contrast the optimization goals of logistic regression and perceptron.

Part 3: Long Answer

3. (15 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, −)

   Here is the perceptron algorithm with step size fixed to 1 (i.e., it's removed from the algorithm):

Algorithm 1 Perceptron

```
Initialize $\mathbf{w} = \mathbf{0}$
for $t = 1$ to $|T|$ do          ▷ Loop over $T$ epochs, or until convergence (an epoch passes with no update)
    for $i = 1$ to $|N|$ do          ▷ Loop over $N$ examples
        $y_{\text{pred}} = 1$ if $\mathbf{w}^\top \mathbf{f}(\mathbf{x}^{(i)})) > 0$, else $-1$          ▷ Make a prediction of 1 or -1
        if $y_{\text{pred}} = -1$ and $y^{(i)} = 1$ then
            $\mathbf{w} \leftarrow \mathbf{w} + \mathbf{f}(\mathbf{x}^{(i)})$          ▷ Update weights if an error was made
        else if $y_{\text{pred}} = 1$ and $y^{(i)} = -1$ then
            $\mathbf{w} \leftarrow \mathbf{w} - \mathbf{f}(\mathbf{x}^{(i)})$          ▷ Update weights if an error was made
        end if
    end for
end for
Return $\mathbf{w}$
```

a. (8 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]

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 (e.g., $x_1 < 5$ and $x_2 > 10$), draw a picture, or describe in words, as long as you are precise. **Note that the perceptron only has the two features given and there is no additional bias term.**

$x_2 \leq x_1$ 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 degree to the right ("1:30"), with negative on the left of the boundary. This accommodates negative points anywhere on the left ($x_1 < 0$) as well as in the first quadrant but above the $x_2 = x_1$ line.

c. (3 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? Answer yes or no and briefly justify your answer.

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

4. (16 points) Consider the following PCFG with S as the root; probabilities are given in parentheses:

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

Non-terminal productions:
(1.0) S $\rightarrow$ PRP VP
(1.0) VP $\rightarrow$ V NP
(1.0) NP $\rightarrow$ J N

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. (6 points) Give **the parse (draw the tree) and its probability** for the sentence *I like green salsa*. You should give either a fraction/decimal (for probability) or a negative number (for log probability) as the answer; either is accepted. **Box your final probability.**

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

b. (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 $\rightarrow$ N J

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

I like salsa red

d. (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

5. (14 points) Suppose you want to use the skip-gram model to learn **bigram embeddings** instead of word embeddings. Recall the basic skip-gram model:

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

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.**

a. (5 points) With $\mathbf{v}$ and $\mathbf{c}$ denoting the word and context vectors respectively, and b denoting a bigram, write out the model for skip-gram with word bigrams and context unigrams:

$P(\text{context} = y \mid \text{word} = b) =$

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

b. (3 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. Hint: taking the dot product of two d-length vectors is an $O(d)$ operation.

$O(Vd)$

c. (3 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)

d. (3 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.

$Bd + Vd$

BONUS (2 points) Give the last name of any author of any paper assigned as reading so far this semester (textbooks don't count).

Manning was the most common correct guess. Other smart guesses: Chen, Wang (common Chinese last names, Danqi Chen and Sida Wang both had papers). Going off of famous CS people, Jeff Dean and Andrew Ng are both represented. We mentioned Iyyer quite a bit for DANs. Lillian Lee is an author on the sentiment paper.