CPSC 477

Spring 2019

HW3

Due Friday, March 29 by 4 PM

Problem	Grade
QUESTION 1 (Language Modeling)	/10
QUESTION 2 (Recurrent Neural Networks)	/10
QUESTION 3 (Morphology)	/10
QUESTION 4 (Text Similarity)	/10
QUESTION 5 (Dimensionality Reduction	/10
QUESTION 6 (Naïve Bayes Classifier)	/10
QUESTION 7 (Training Neural Networks)	/10
QUESTION 8 (Word Embeddings)	/10
TOTAL	/80

Submission Instructions

1. Submit each solution on a separate sheet.
2. Include your **name**, **netID** and **problem number** on each page.

$p(a|bc) = \frac{p(b|ac)\,p(a|c)}{p(b|c)}$

QUESTION 1 (Language Modeling)

You are in a noisy coffee shop, diligently studying for your midterm, and your friend is trying to get your attention, using only a two words vocabulary. She has said a sentence but you couldn't hear one of the words:

$$(w1 = 'hi'; w2 = 'yo'; w3 = ???; w4 = 'yo')$$

Assume that your friend was generating words from this first-order Markov model:

$$p('hi'|'hi') = 0.7 \qquad p('yo'|'hi') = 0.3$$
$$p('hi'|'yo') = 0.5 \qquad p('yo'|'yo') = 0.5$$

Given these parameters, what is the posterior probability of whether the missing word is 'hi' or 'yo'? Show your work.

want $p(w_3|w_1, w_2, w_4)$, by Markov assumption $p(w_3|w_2, w_4)$

$\propto p(w_3|w_2)\,p(w_4|w_2, w_3)$

(by Markov assumption) $\propto p(w_3|w_2)\,p(w_4|w_3)$

$$\frac{p(w_3 = hi)}{p(w_3 = yo)} = \frac{p(hi|yo)\cdot p(yo|hi)}{p(yo|yo)\cdot p(yo|yo)}$$

$$= \frac{0.5 \times 0.3}{0.5 \times 0.5}$$

$$= \frac{3}{5}$$

$\Rightarrow p(w_3 = hi) = 0.375$

$p(w_3 = yo) = 0.625$

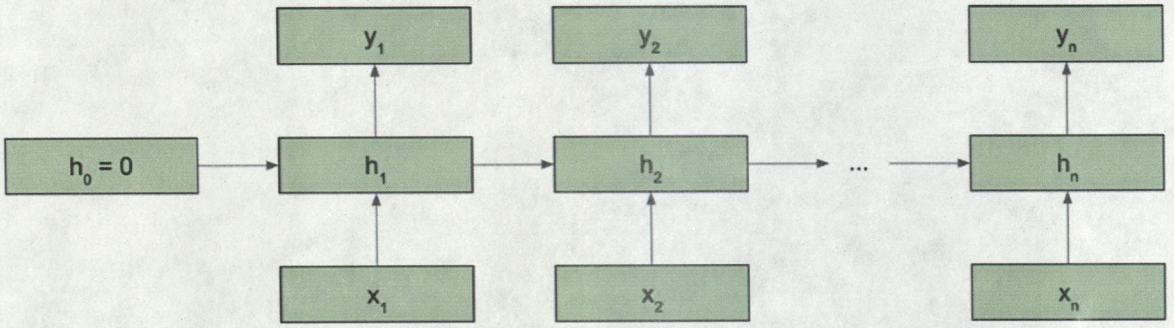
QUESTION 2 (Recurrent Neural Networks)

Your goal will be to set up weights for a simple recurrent network (SRN and also known as an Elman Netowork) to solve two simple sequence tasks. The network reads in a sequence $x_1,..,x_t$and produces outputs $y_1,..,y_t$. We consider the final output y_t to be the answer given by the network.

We will assume a threshold activation function:

$$f(x) = 1 \text{ if } x > 0; \text{ otherwise } f(x) = 0.$$

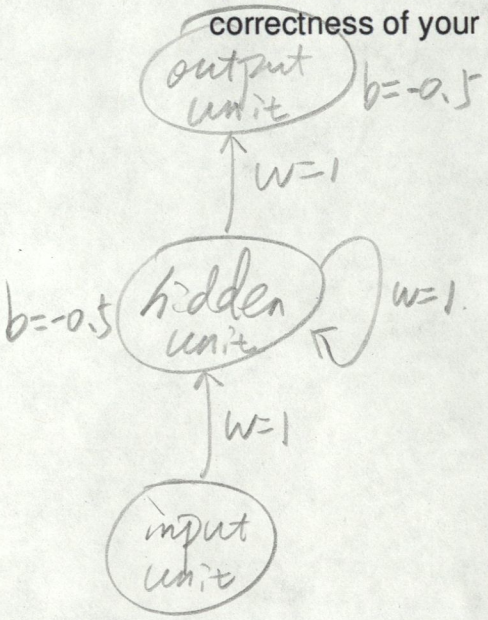
The network has one simple recurrent layer followed by a linear layer (shown in below diagram). The dimension of both the input to the network and its output should be 1.

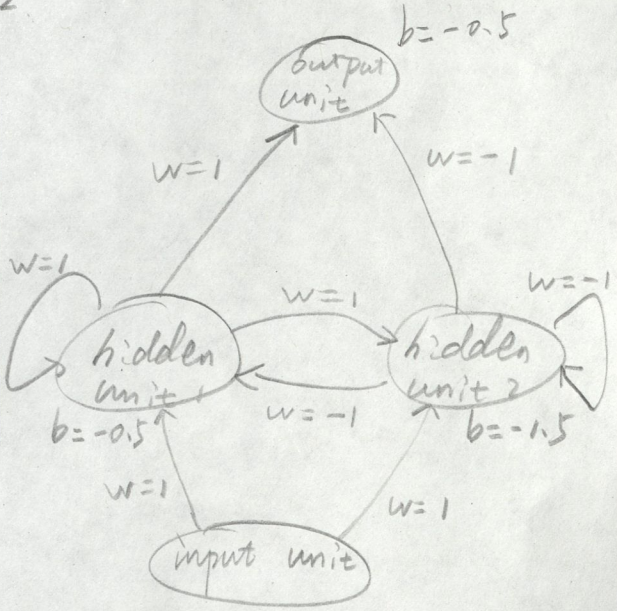


1. Construct an SRN with 1 hidden unit that reads a sequence of 1s and 0s and outputs 1 if any of $x_1, .., x_t$ had the value 1. Justify the correctness of your network.

2. Construct an SRN with 2 hidden units to solve the parity task (read in a sequence of 1s and 0s and return the sum mod 2, shown in the below table). Justify the correctness of your network.

x	y
0	0
1	1
010	1
0110	0
10101	1





Justification (1)

at each time i

$u^i = input^i + u^{i-1} - 0.5$

$output^i = u^i - 0.5$

time							
input	0	0	1	0	1	1	0
unit	0	0	1	1	1	1	1
output	0	0	1	1	1	1	1

"OR" of 0, 00, 001, 0010, 00101, 001011, 0010110, respectively.

∴ verified

Justification: (2)

at each time i,

$u_1^i = input^i + u_1^{i-1} - u_2^{i-1} - 0.5$

$u_2^i = input^i + u_1^{i-1} - u_2^{i-1} - 1.5$

$output^i = u_1^i - u_2^i - 0.5$

time	1	2	3	4	5
input	1	0	1	0	1
unit 1	1	1	1	0	1
unit 2	0	0	1	0	0
output	1	1	0	0	1

Parity of 1, 10, 101, 1010, 10101, respectively.

∴ verified

QUESTION 3 (Morphology)

Refer to the slides for lecture #143 for background on this question.

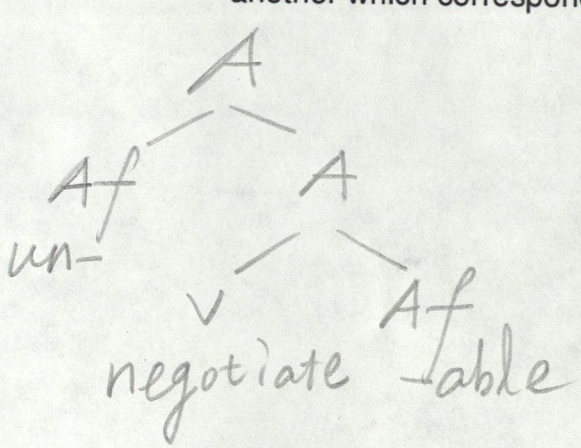
1. Consider the English verb *drink*. Give two inflected forms of *drink*, and two derivational forms of *drink*.

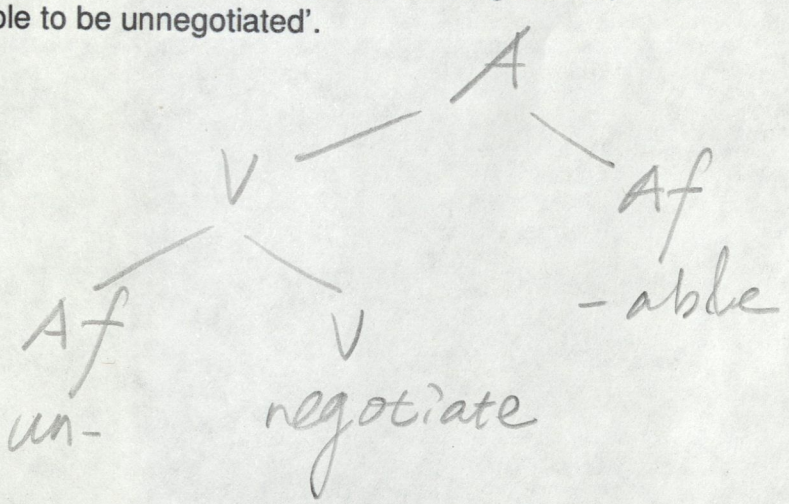
inflextional form: drinks, drank
derivational form: drinkable, undrinkable.

2. Identify all the morphemes in *infallibilities*. For each morpheme, say whether it is a prefix, suffix, or root.

prefix: in-
root: fall
suffix: -ible, -ity, -ies

3. Consider the word *unnegotiable*. Draw and label two morphological trees for this word, one which corresponds to the meaning 'not able to be negotiated', and another which corresponds to 'able to be unnegotiated'.





QUESTION 4 (Text Similarity)

We first need to introduce a function that compares the similarity of two sentences. Recall the definition for the cosine distance between two vectors x and y:

$$cosine(x, y) = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x^2} \sqrt{\sum_{i=1}^{n} y^2}}$$

Part A: How would you define a mapping from sentences s to vectors $f(s)$ such that the cosine distance $cosine(f(s_1), f(s_2))$ is a measure of the similarity between two sentences s_1 and s_2? We will give full marks for any reasonable mapping.

Part A
Let $f(s)$ be a vector of count for each word in the vocabulary.

Part B: Now, we will define a dynamic programming algorithm that computes sentence alignment of two translated documents. The alignment score is the sum of the similarity scores of the aligned sentences. Our goal is to find an alignment with the highest score.

We will consider alignments of the following form:

- a sentence can be aligned to an empty sentence. This happens when one of the translators omits a sentence.
- a sentence can be aligned to exactly one sentence.
- a sentence can be aligned to two sentences. This happens when one of the translators either breaks or joins sentences.

Our sentence alignment algorithm recursively computes the alignment matrix F indexed by i and j, one index for each sentence. The value stored in $F(i, j)$ is the score of the best alignment between the first i sentences of x and the first j sentences of y. $s(x_i, y_j)$ is the similarity between sentence x_i and sentence y_j.

Based on the above information, please

Define $F(0,0)$, $F(i, 0)$ and $F(0, j)$.
Define $F(i, j)$ for $i > 0$ and $j > 0$.

Part C: Next, we will modify the alignment score to be the same as before, but to include a fixed penalty p each time a sentence is aligned to an empty sentence. p is a parameter,

which is >= 0, chosen by the user of the algorithm. Describe a modified dynamic programming method that takes this new penalty into account.

Part B.

$$F(0,0)=0,\quad F(i,0)=0,\quad F(0,j)=0$$

$$F(i,j)=\max\{F(i-1,j-1)+S(i,j),$$
$$F(i-1,j),$$
$$F(i,j-1),$$
$$F(i-2,j-1)+S(i-1,j)+S(i,j),$$
$$F(i-1,j-2)+S(i,j-1)+S(i,j)\}$$

Part C

$$F(0,0)=0,\quad F(i,0)=-i\cdot p,\quad F(0,j)=-j\cdot p$$

$$F(i,j)=\max\{F(i-1,j-1)+S(i,j),$$
$$F(i-1,j)-p,$$
$$F(i,j-1)-p,$$
$$F(i-2,j-1)+S(i-1,j)+S(i,j),$$
$$F(i-1,j-2)+S(i,j-1)+S(i,j)\}$$

QUESTION 5 (Dimensionality reduction and SVD)

(a) Prove that the left singular vectors of a matrix A are the right singular vectors of A^T

let $A = U\Sigma V^T$ ⟶ U is the left singular vectors of A

then $A^T = V\Sigma^T U^T$ ⟶ U is the right singular vectors of A^T

(b) Calculate the covariance matrix for the matrix M shown below.

$$M = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

for each column, minus each value by the column mean

$$A = \begin{pmatrix} \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} \\ -\frac{2}{3} & -\frac{2}{3} & 0 & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & 0 & -\frac{2}{3} \end{pmatrix}$$

x_1 x_2 x_3 x_4

mean $\frac{2}{3}$ $\frac{2}{3}$ 0 $\frac{2}{3}$

∴ the covariance matrix $= \frac{1}{n}A^TA$, where $n=3$

covariance matrix

$$= \begin{pmatrix} \frac{2}{9} & \frac{2}{9} & 0 & -\frac{1}{9} \\ \frac{2}{9} & \frac{2}{9} & 0 & -\frac{1}{9} \\ 0 & 0 & 0 & 0 \\ -\frac{1}{9} & -\frac{1}{9} & 0 & \frac{2}{9} \end{pmatrix}$$

(c) Find the singular values of the matrix M

$$M^TM = \begin{pmatrix} 2 & 2 & 0 & 1 \\ 2 & 2 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 2 \end{pmatrix}$$

let $\det(M^TM - \lambda I) = 0$,

$\lambda = 0,\ 3+\sqrt{3},\ 3-\sqrt{3}$

∴ singular values of M is $0, \sqrt{3+\sqrt{3}}, \sqrt{3-\sqrt{3}}$.

or equivalently, $0, 2.175, 1.126$

QUESTION 6 (Naïve Bayes Classifier)

Suppose you are given the following set of data with three Boolean input variables a, b, and c, and a single Boolean output variable K.

a	b	c	K
1	0	1	0
1	1	1	1
0	1	1	0
1	1	0	0
1	0	1	0
0	0	0	1
0	0	0	1
0	0	1	0

	a		b		c	
	0	1	0	1	0	1
K=0	2/5	3/5	3/5	2/5	1/5	4/5
K=1	2/3	1/3	2/3	1/3	2/3	1/3

a. $P(K=1 \mid a=1, b=1, c=0)$

$= \frac{P(a=1 \mid K=1) \cdot P(b=1 \mid K=1) \cdot P(c=0 \mid K=1) \cdot P(K=1)}{P(a=1, b=1, c=0)}$

$= \frac{\frac{1}{3} \times \frac{1}{3} \times \frac{2}{3} \times \frac{3}{8}}{\frac{1}{3} \times \frac{1}{3} \times \frac{2}{3} \times \frac{3}{8} + \frac{3}{5} \times \frac{2}{5} \times \frac{1}{5} \times \frac{5}{8}}$

$= \frac{25}{52}$

$= 0.48$

b. $P(K=0 \mid a=1, b=1)$

$= \frac{P(a=1 \mid K=0) \cdot P(b=1 \mid K=0) \cdot P(K=0)}{P(a=1, b=1)}$

$= \frac{\frac{3}{5} \times \frac{2}{5} \times \frac{5}{8}}{\frac{3}{5} \times \frac{2}{5} \times \frac{5}{8} + \frac{1}{3} \times \frac{1}{3} \times \frac{3}{8}}$

$= \frac{18}{23}$

$= 0.78$

(For parts (a) and (b), assume we are using a Naïve Bayes classifier to predict the value of K from the values of the other variables.)

(a) According to the naive Bayes Classifier, what is $p(K = 1 | a = 1, b = 1, c = 0)$? 0.48

(b) According to the naive Bayes Classifier, what is $p(K = 0 | a = 1, b = 1)$? 0.78

(c) Given $P(Z|X) = 0.7$ and $P(Z|Y) = 0.4$.

Do you have enough information to compute $P(Z|X, Y)$? If not, write "not enough info". If so, compute the value of $P(Z|X, Y)$ from the above information Not enough info.

(e) Given $P(Z, X) = 0.2$, $P(X) = 0.3$, $P(Y) = 1$

Do you now have enough information to compute $P(Z|X, Y)$? If not, write "not enough info". If so, compute the value of $P(Z|X, Y)$from the above information. Enough info

$P(Z|X, Y) = P(Z|X)$ since $P(Y) = 1$.

$= \frac{P(Z, X)}{P(X)} = \frac{2}{3}$

QUESTION 7 (Training Neural Networks)

This question is to train a three-layer neural network for classification task. Let $H1$ denote the number of hidden units, let D be the dimension of the input X, and let K be the number of classes. The three-layer network has the following form:

$$h_1 = ReLU(W_1 X + b_1)$$
$$h_2 = ReLU(W_2 h_1 + b_2)$$
$$p = Softmax(W_3 h_2 + b_3)$$

where the parameters of the network are $W_1 \in R^{H_1 * D}$, $b_1 \in R^{H_1}$, $W_2 \in R^{H_2 * H_1}$, $b_2 \in R^{H_2}$, $W_3 \in R^{K * H_2}$, $b_3 \in R^K$.

$ReLU$ is the "rectified linear unit" $ReLU(x) = max\{0, x\}$ applied componentwise, and SoftMax maps a d-vector to the probability simplex according to

$$Softmax(v)_k = \frac{exp(v_k)}{\sum_{j=1}^{K} exp(v_j)}$$

For a given input X, this specifies how to calculate the probabilities $P(Y = k|X)$ for $k = 1,2,\ldots,k$.

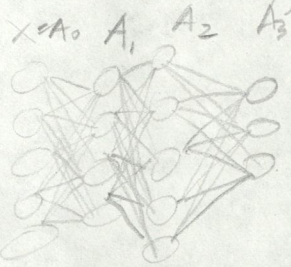
For a given training set $\{(X_i, Y_i)\}$, consider the loss function:

$$L = \frac{1}{n}\sum_{i=1}^{n} - log\, p(Y = Y_i|X_i) + \frac{\lambda}{2}(\|W_1\|^2 + \|W_2\|^2 + \|W_3\|^2)$$

where $\|A\|^2$ means the sum of the squares of the entries of the matrix A.

Give formulas for the derivatives for each layer of the network$j = 1,2,3$.

$$\frac{\partial L}{\partial W_j}, \frac{\partial L}{\partial b_j}$$



Hints:

i. You should ignore the fact that $ReLU(x)$ is not differentiable at $x_i = 0$.

ii. You might try to first do the calculations with $ReLU()$ replaced by the identity function.

$P_k = \text{softmax}(v)_k = \frac{e^{v_k}}{\sum_{j=1}^{K} e^{v_j}}$

$h_1 = ReLU(W_1 X + b_1)$

$h_2 = ReLU(W_2 h_1 + b_2)$

$v = \text{softmax}(W_3 h_2 + b_3)$

$$L = \frac{1}{n}\sum_{i=1}^{n}\left\{-\log P(Y = Y_i(k) \mid X_i)\right\} + \frac{\lambda}{2}\left(\|W_1\|^2 + \|W_2\|^2 + \|W_3\|^2\right)$$

$$\frac{\partial L}{\partial v_k} = \frac{\partial L}{\partial P_k}\frac{\partial P_k}{\partial v_k} = \frac{1}{n}\sum_{k=1}^{n}\left(-\frac{1}{P_k}\right)\cdot P_k(1-P_k) = \frac{1}{n}\sum_{k=1}^{n}(P_k - 1)$$

$$\frac{\partial L}{\partial W_3} = \frac{\partial L}{\partial v_k}\frac{\partial v_k}{\partial W_3} + \lambda\|W_3\| = \frac{1}{n}\cdot\sum_{k=1}^{n}(P_k - 1)\,h_2 + \lambda\cdot\|W_3\| \quad (1)$$

$$\frac{\partial L}{\partial b_3} = \frac{\partial L}{\partial v_k}\frac{\partial v_k}{\partial b_3} = \frac{1}{n}\sum_{k=1}^{n}(P_k - 1) \quad (2)$$

$$\frac{\partial L}{\partial h_2} = \frac{\partial L}{\partial v_k}\frac{\partial v_k}{\partial h_2} = \begin{cases} \frac{1}{n}\cdot\sum_{k=1}^{n} W_3^T\cdot(P_k - 1), & \text{otherwise} \\ 0, & h_2 < 0 \end{cases}$$

$$\frac{\partial L}{\partial W_2} = \frac{\partial L}{\partial h_2}\frac{\partial h_2}{\partial W_2} + \lambda\cdot\|W_2\| = \frac{1}{n}\cdot\sum_{k=1}^{n} W_3^T h_1^T(P_k - 1) + \lambda\|W_2\| \quad (3)$$

$$\frac{\partial L}{\partial b_2} = \frac{\partial L}{\partial h_2}\frac{\partial h_2}{\partial b_2} = \frac{1}{n}\sum_{k=1}^{n} W_3^T\cdot(P_k - 1) \quad (4)$$

$$\frac{\partial L}{\partial h_1} = \frac{\partial L}{\partial h_2}\frac{\partial h_2}{\partial h_1} = \begin{cases} \frac{1}{n}\sum_{k=1}^{n} W_2^T W_3^T(P_k - 1), & \text{otherwise} \\ 0, & h_1 < 0 \end{cases}$$

$$\frac{\partial L}{\partial W_1} = \frac{\partial L}{\partial h_1}\frac{\partial h_1}{\partial W_1} + \lambda\|W_1\| = \frac{1}{n}\sum_{k=1}^{n} W_2^T W_3^T\cdot X^T(P_k - 1) + \lambda\cdot\|W_1\| \quad (5)$$

$$\frac{\partial L}{\partial b_1} = \frac{\partial L}{\partial h_1}\frac{\partial h_1}{\partial b_1} = \frac{1}{n}\cdot\sum_{k=1}^{n} W_2^T W_3^T(P_k - 1) \quad (6)$$

QUESTION 8 (Word Embeddings[1])

Word2Vec represents a family of word-embedding algorithms that are commonly used in a variety of contexts.

(a) Let's consider a recommender system for an online music-streaming service. It includes information about a set of songs on how frequently users play them together (e.g., song X is commonly played together with song Y). Explain how you would use ideas similar to Word2Vec to recommend the next song to a user who has played some songs from this list. Treat user's listening queue as a sentence, with each word being a song. Then, train word2vec model to get high-dimensional vectors so that similar songs have weights that are closer together

(b) Although pre-trained word vectors work quite well in many applications, it is sometimes better to "re-train" (i.e. continue to learn) the word vectors as parameters of our neural network. Explain why retraining the word vectors may hurt the model performance if the dataset for the specific task is too small.

Pre-trained word vectors produce semantically related words to be located in the same part of the word space. When retrained with a small dataset, these words are shifted in the word space, thus hurting the model performance.

(c) Give two examples of how we can evaluate word-embedding vectors.

① word vector analogies: (a:b) :: (c:?)

② correlation evaluation: compare cosine similarity of word vectors with human's assessment on the similarity between the words

(d) A simple way to produce word embeddings for a vocabulary of words is to create one-hot vectors for each word - a vector with 0s everywhere and only a single 1 in the position corresponding to the word's index in the vocabulary. Give at least two reasons why word2vec-produced word embeddings are better than these one-hot vectors.

① More computationally efficient, since word2vec word embeddings are lower dimensional, dense and doesn't grow with the vocabulary size. One-hot vectors are higher dimensional, sparse, and grows with the vocabulary size.

② word2vec word embeddings preserve the semantic meaning of words, i.e., semantically similar words have similar vectors. This is not the case with one-hot vectors.