CPSC 477/577

Spring 2020

HW3

Due Monday, March 30

Problem	Grade
Q1. Language Modeling	/10
Q2. Morphological Analysis	/10
Q3. Document Representation	/10
Q4. Tweaking Neural Networks	/10
Q5. Multi-Layer Perceptron	/10
Q6. Training Neural Networks	/10
Q7. Perplexity	/10
Q8. Sentence Alignment	/10
TOTAL	/80

Q1. Language Modeling

1. For the 520 million word Corpus of Contemporary American English, we have the following counts of unigrams and bigrams: your, 883,614; rights, 80,891; doorposts, 21; your rights, 378; your doorposts, 0.

a) Estimate the probabilities P(rights) and P(rights | your) using Maximum Likelihood estimation (no smoothing). Answer with fractions.

b) Estimate the bigram probability P(doorposts | your) using Maximum Likelihood estimation and add-k smoothing with k = 0.05. Assume that the vocabulary consists of 1,254,193 unique words. Answer with a fraction.

c) Suppose that we train three n-gram models on 38 million words of newspaper text: a unigram model, a bigram model, and a trigram model. Suppose further that we evaluate the trained models on 1.5 million words of text with the same vocabulary and obtain the following perplexity scores: 170, 109, and 962. Which perplexity belongs to which model? Provide a short explanation.

Q2. Morphological Analysis

For each of the following words, please:

A. Draw the morphological tree, labeling the part of speech at each node

B. Identify one derivational affix (all of these have multiple derivational affixes). What does this affix "mean," or signify? (in your own words, not from the dictionary)

C. Give another word with a different root but the same derivational affix as in (B). Make sure the affix retains the same meaning in the new word. Do you notice a pattern in what happens to the starting word when this affix is added? (hint: POS)

Words:

1. *institutionalization*
2. *unhappiness*
3. *wirelessly*

Finally, please describe an application of derivational morphology in NLP.

Q3. Document Representation

Calculate the TF*IDF for the terms listed below for documents 1 to 4. There are 10,000 documents in a collection. The number of times each of these terms occur in documents 1 to 4 as well as the number of documents in the collections are listed below. Use this information to fill in the TF*IDF scores in the table below.

Number of Documents Containing Terms:
• *reverse cascade*: 3
• *full shower*: 50
• *half bath*: 10
• *multiplex*: 3

Term Frequencies				
	Documents			
	Doc 1	Doc 2	Doc 3	Doc 4
reverse cascade	8	10	0	0
full shower	3	1	2	2
half bath	0	0	8	7
multiplex	2	2	2	9

First calculate the IDF of the mentioned terms; then fill the following table:

TFIDF for terms in documents				
	Documents			
	Doc 1	Doc 2	Doc 3	Doc 4
reverse cascade				
full shower				
half bath				
multiplex				

Q4. Tweaking Neural Networks

Question: For each of the following neural network model changes, describe how the change is likely to impact the model's capacity to overfit the training data and give one reason why.

A. More layers
B. Fewer epochs
C. More neurons in each layer
D. Using linear activations instead of nonlinear
E. Less training data

Q5. Multi-layer perceptron

Question: Design a multilayer perception that does two bit binary addition (it takes 4 binary inputs and outputs three digits for the sum, including the two sum bits and the one-bit carry bit). Show all hyperparameters and edge weights.

Example input/output: 01 + 11 = 100

Q6. 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:

$$h_1 = ReLU(W_1X + b_1)$$
$$h_2 = ReLU(W_2h_1 + b_2)$$
$$p = Softmax(W_3h_2 + b_3)$$

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.

Q7. Perplexity

Recall that the perplexity of a language model on a test corpus is defined as 2^{-l}
Where

$$l = \frac{1}{M}\sum_{i=1}^{m} log_2 p(x^{(i)})$$

and m is the number of sentences in the corpus, M is the total number of words in the corpus, log2 is log base 2, $x^{(i)}$ is the i'th sentence in the corpus, and $p(x^{(i)})$is the probability of the i'th sentence in the corpus under the language model?

Part A: What is the maximum value that the perplexity can take?
Part B: What is the minimum value that the perplexity can take?
Part C: Assume that we have a bigram language model, where

$$p(w_1 \dots w_n) = \prod_{i=1}^{n} q(w_i|w_{i-1})$$

and w_0= *, and w_n = STOP. We estimate the parameters as

$$q(w|v) = \frac{Count(v,w)}{count(v)}$$

Write down a training corpus and a test corpus such that the perplexity of the model trained on the training corpus takes the maximum possible value on the test corpus.

Part D: Write down a training corpus and a test corpus such that the perplexity of the model trained on the training corpus takes the minimum possible value on the test corpus. (Assume that we use a bigram language model, as in part C.)

Q8. Sentence Alignment

We first need to introduce a function that compares similarity between 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?

Part B: Now, we will define a dynamic programming algorithm that computes sentence alignment of two translations. 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.

• 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'll 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 which takes this new penalty into account.