CS4248 Natural Language Processing Assignment 3

Distributed on 19 Mar 2024

Due in Canvas `Assignment` by 02 Apr 2024 23:59 SGT

This assignment contributes 10 marks towards your final mark for the class, and is graded out of a rubric of 100 points.

Integrity Note. Since this assignment is similar to other assignments for Natural Language Processing courses at other institutions, there are (undoubtedly) solutions posted somewhere. Under the NUS Code of Conduct, you must follow class policy in working on this individual assignment. When in doubt of whether an action would constitute a violation of policy, please ask us on `Discussion` on the `Canvas` or by emailing the instructors, **before** attempting the action.

To discuss (mostly with your peers in class) the assignment, or to ask topical questions about the assignment itself, pleas use the `Discussion` on the `Canvas`.

Submission. Submit your independently authored solutions to each of the four parts as a PDF file, named `A000000X.pdf`, named after your correct Student ID — `A0000000X` (all letters in CAPITALS). There is no fixed format for PDF file (it is suggested that you use a word processor and key in equations where appropriate, but you may use legible photos/scans of written work), except that each question–part should start on a new, separate page and appropriately numbered, to facilitate grading. For example, a submission which uses 1 page per question–part would have 1 (declaration) + 1 (neural networks) + 2 (word embeddings) + 2 (hmms) = 6 pages. You may slightly exceed the one-page limit for each question part, but do not exceed by too much; too lengthy answers may cause additional confusion. Submit this file by the assigned deadline to Canvas assignment for *Assignment 3*. Do not include your name or personally identifiable information in your submission.

You must include the text of the two statements below in your submitted work and digitally sign your homework using your Student ID number (starting with `A...`; N.B., not your NUSNET email identifier). Make sure you have attached this statement to your submission either in written or typed form, as the first page. Delete (and where appropriate, fill in) one of the two forms of Statement 1:

1A. Declaration of Original Work. *By entering my Student ID below, I certify that I completed my assignment independently of all others (except where sanctioned during in-class sessions), obeying the class policy outlined in the introductory lecture. In particular, I am allowed to discuss the problems and solutions in this assignment, but have waited at least 30 minutes by doing other activities unrelated to class before attempting to complete or modify my answers as per the Pokémon Go rule.*

1B. Exception to the Class Policy. *I did not follow the CS4248 Class Policy in doing this assignment. This text explains why and how I believe I should be assessed for this assignment given the circumstances explained.*

Signed, [Enter your A0000000X Student ID here]

2. References. *I give credit where credit is due. I acknowledge that I used the following websites or contacts to complete this assignment (but please note that many uses of Web search and detailed discussion are not allowed:*

- *Sample. Website 1, for following mathematical proofs.*
- *Sample. My friend, A0000001Y, whom helped me figure out the course deadlines, ... [Fill in appropriately]*

Acknowledgements. Parts of this assignment were adapted from Prof. Christopher D. Manning's original assignment at Stanford University.

1 Neural Network Basis (30 points)

Figure 1 shows a one-hidden-layer neural network. This neural network employs the sigmoid activation function for the hidden layer **h**, and softmax for the output layer. Assume that the one-hot label vector is **y**, and a cross entropy loss is used.

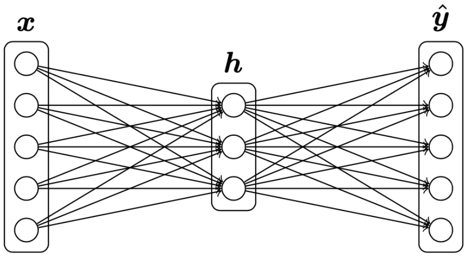

Figure 1: An example of a one hidden layer neural network.

Recall that the forward propagation works by propagating the observations **x** through to the hidden layer **h**, and then to the output layer $\hat{\mathbf{y}}$ as follows:

$$\mathbf{h} = \text{sigmoid}(\mathbf{x}\mathbf{W}^{[1]} + \mathbf{b}^{[1]}) \qquad \hat{\mathbf{y}} = \text{softmax}(\mathbf{h}\mathbf{W}^{[2]} + \mathbf{b}^{[2]}) \tag{1}$$

where $\mathbf{W}^{[i]}$ and $\mathbf{b}^{[i]}$ $(i \in \{1, 2\})$ are the weights and biases, respectively, of the two layers.[1] *(N.B.: For both Questions 2 and 3, you may look up / search for common function derivatives, but please do the derivations without referring to websites that explicitly gives the entire answer to the question; that would constitute a violation of class policy).*

1. **(10 points)** How many parameters are there in this neural network, assuming the input is D_x-dimensional, the output is D_y-dimensional, and there are H hidden units? Show your work.

2. **(10 points)** Derive the gradients of the sigmoid function and show that it can be rewritten as a function of the function value (*i.e.* in some expression where only $\sigma(x)$, but not x, is present). Assume that the input x is a scalar for this question. Recall that the sigmoid function takes the form:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{12}$$

3. **(10 points)** Derive the gradient with regard to the inputs of a softmax function when cross entropy loss is used for evaluation; *i.e.* find the gradients with respect to the softmax input vector θ, when the prediction is made by $\hat{y} = \text{softmax}(\theta)$. Recall that the cross entropy function takes the form:

$$CE(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_i y_i \log(\hat{y}_i) \tag{13}$$

where **y** is the one-hot label vector, and $\hat{\mathbf{y}}$ is the predicted probability vector for all classes. **Hint**: you may consider the fact that all but one of the elements of **y** are 0s; and assume that only the k-th dimension of **y** is 1.

[1] To be clear in lecture, we used different notation (θ for **W**), and folded the bias into the weight matrix by assuming an artificial observation $x_0^{[i]} = 1$, per layer. Both notations are equivalent.

2 Word Embeddings (40 points)

Let's have a quick refresher on the `word2vec` algorithm. The key insight behind `word2vec` is that *"a word is known by the company it keeps"*. Concretely, suppose we have a "center" word c and a contextual window surrounding c. We shall refer to words that lie in this contextual window as "outside words". For example, in Figure 2 we see that the center word c is "banking". Since the context window size is 2, the outside words are "turning", "into", "crises", and "as", for time step t.

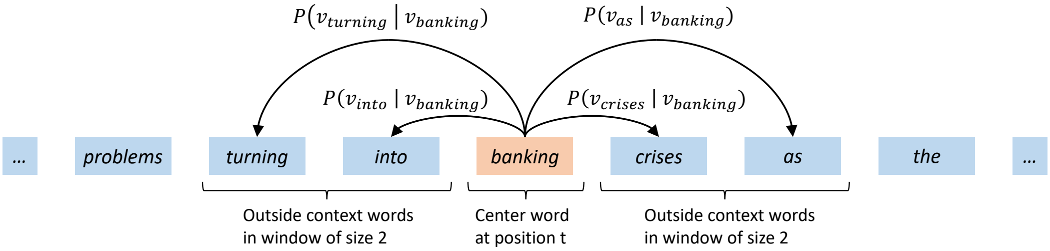
$P(v_{turning} \mid v_{banking})$
$P(v_{as} \mid v_{banking})$
$P(v_{into} \mid v_{banking})$
$P(v_{crises} \mid v_{banking})$
...
problems
turning
into
banking
crises
as
the
...
Outside context words in window of size 2
Center word at position t
Outside context words in window of size 2

Figure 2: The `word2vec` skip-gram prediction model with window size 2.

The goal of the **skip-gram** `word2vec` algorithm is to accurately learn the probability distribution $P(O|C)$. Intuitively, the task is similar to cloze question answering, where a blank (or mask) is left to be filled in, given knowledge of the contextual window. Given a specific word o and a specific word c, we want to calculate $P(O = o|C = c)$, which is the probability that word o is an context word for c, *i.e.*, the probability that o falls within the contextual window of c.

In `word2vec`, the conditional probability distribution is given by taking vector dot products and then applying the softmax function:

$$P(O = o|C = c) = \frac{\exp(v_o^\top v_c)}{\sum_{w \in \text{Vocab}} \exp(v_w^\top v_c)} \tag{14}$$

where v_o and v_c are the (current) word vectors representing the outside word o and the center word c, respectively.

1. **(5 points)** Figure 2 shows all 4 conditional probabilities needing to be calculated in the current time step (*e.g.*, $P(v_{turning}|v_{banking})$). What are the conditional probabilities that need to be calculated in the next time step? Please write down all of them.

   Hint: the contextual window shifts one word to the right for each subsequent time step.

2. **(5 points)** Suppose we now use a contextual window of size 1. What are the conditional probabilities that need to be calculated when the center word is "turning"?

3. **(10 points)** Suppose we have 100 different words in the vocabulary. To calculate all the conditional probabilities shown in Figure 2, what are the minimal number of vector dot product operations need to be performed? Show your calculation, state your numeric answer explicitly, and explain why. You can assume that the word embedding matrix is given, and each row is an embedding for a word. Accessing the embedding for a particular word does not need any computation (such as the dot products of input word's one-hot vector and the word embedding matrix).

4. **(10 points)** The training objective for skip-gram is to minimize the following loss function for each pair of (center word, context word):

$$L_{\text{naive-softmax}}(v_o, v_c) = -\log P(O = o|C = c) \tag{25}$$

Compute the partial derivative of $L_{\text{naive-softmax}}(v_o, v_c)$ with respect to v_c. Please write your answer in terms of v_o, v_c and v_w. Based on your calculation, please explain why it is *inefficient* to optimize this naïve version of loss function $L_{\text{naive-softmax}}$.

5. **(10 points)** Now we shall consider the *Negative Sampling* loss, which is an alternative to the Naïve Softmax loss. Assume that K negative samples (words) are drawn from the vocabulary. For simplicity of notation we shall refer to them as $w_1, w_2, \cdots, w_K$ and their corresponding vectors as $v_1, \cdots, v_K$. Note that $o \notin \{w_1, w_2, \cdots, w_K\}$. For a center word c and an outside word o, the negative sampling loss function is given by:

$$L_{\text{neg-sample}}(v_o, v_c) = -\log(\sigma(v_o^T v_c)) - \sum_{k=1}^{K} \log(\sigma(-v_k^T v_c)) \tag{26}$$

where $\sigma(\cdot)$ is the sigmoid function. When $K = 5$, what are the number of vector dot product operations need to be performed in order to calculate $L_{\text{neg-sample}}(v_{into}, v_{banking})$?

3 Hidden Markov Models (30 points)

Given a *state sequence* $S = \{s_0, s_1, s_2, \cdots, s_T\}$ and an *observation sequence* $O = \{o_0, o_1, o_2, \cdots, o_T\}$, the Hidden Markov Model (HMM) models the joint probability $P(O, S)$ with two sets of parameters:

- The *transition probabilities* $P(s_t|s_{t-1})$, denoted by matrix A.
- The *emission probabilities* $P(o_t|s_t)$, denoted by matrix B.

Figure 3 illustrates an Hidden Markov Model (HMM) with $T = 5$.

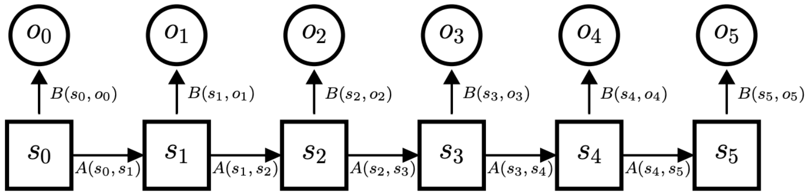

Figure 3: An illustration of a Hidden Markov Model unrolled for 6 steps.

One biological application of HMMs is to determine the secondary structure (*i.e.* the general three-dimensional shape) of a protein. This general shape is made up of alpha helices, beta sheets, and other structures. In this problem, we will assume that the amino acid composition of these regions is governed by an HMM. Under this setting, we have the following transition probabilities A and emission probabilities B. We assume that the start state (or s_0) is always "other" for any sequence. This assumption applies to all sub-questions.

To clarify on this assumption: 1) o_0 is not fixed as a consequence of s_0 being fixed; 2) you do not need consider probability $P(other|other)$ for s_0 as s_0 is already the start state.

	alpha	beta	other
alpha	0.7	0.1	0.2
beta	0.2	0.6	0.2
other	0.3	0.3	0.4

e.g. *P*(Alpha Helix → Beta Sheet) = 0.1

amino acid	alpha	beta	other
M	0.35	0.10	0.05
L	0.30	0.05	0.15
N	0.15	0.30	0.20
E	0.10	0.40	0.15
A	0.05	0.00	0.20
G	0.05	0.15	0.25

Figure 4: Our problem's transition probabilities (left) and emission probabilities (right).

1. **(5 points)** Represent the joint probability $P(O, S) = P(o_0, o_1, \cdots, o_T, s_0, s_1, \cdots, s_T)$ in terms of $A(s_i, s_j)$ and $B(s_i, o_i)$. Please write out detailed derivations. Note detailed derivations only mean that you don't directly give the answer. You should give some intermediate steps about how you arrive at the answer. But this doesn't mean that you have to multiply everything one by one; just use the notations that you are comfortable with.
2. **(5 points)** Compute the probability of the following sequence: $[\langle s_0 = \text{other}, o_0 = \text{M}\rangle, \langle s_1 = \text{alpha}, o_1 = \text{L}\rangle, \langle s_2 = \text{beta}, o_2 = \text{N}\rangle]$.
3. **(10 points)** How many state paths could give rise to the sequence $O = [o_o = \text{M}, o_1 = \text{L}, o_2 = \text{N}]$? Calculate the total probability $P(O)$. Show your work. You can assume start state is always "other" hold.

4. **(10 points)** Give the most likely state transition path S^* for the amino acid sequence $[o_o = \text{M}, o_1 = \text{L}, o_2 = \text{N}]$ using the Viterbi algorithm. Calculate $P(S^*, O)$. Show your work.