Homework 3: Masked Language Models, BERT and Perplexity

INFO 159 / 259, Spring 2024

Due: February 22, 2023 (11:59 PM)

This homework is designed to help you:

1. familiarize yourself with masking approach for transformer-based language models [Part 1]

2. understand the concept of perplexity, implement pseudo-perplexity for BERT [Part 2]

This document contains the instructions for this homework, and the accompanying notebook can be found here:

https://github.com/ucbnlp24/hws4nlp24/blob/main/HW3/HW3.ipynb

To use it, like in HW1&2, hit the button at the top, and be sure to save a copy in your Google Drive or Github before you start to work on the file.

Submit your work on Gradescope. For code questions (Q1-4), replace `# to-do` with your solutions, and as in HW1&2, keep your code between the `# BEGIN SOLUTION` and `# END SOLUTION` flags. For the short-answer question (Q5) simply put your responses in the collab notebook.

1 Part 1: Masking

The first deliverable can be seen as a tutorial of masking, which is one of the core ideas for transformer-based language models.

QUESTION 1 [20 pts]: **BERT mask.**

BERT training relies on the concept of *masked language modeling*: masking a random set of input tokens in a sequence and attempting to predict them. Remember that BERT is *bidirectional*, so that it can use all of the other non-masked tokens in a sentence to make that prediction. In this function, **Create a mask that always masks token positions 2 and 7 for a size 10 sequence input,** as detailed in the notebook.

QUESTION 2 [20 pts]: **Casual mask.**

The GPT class of models acts as a traditional left-to-right language model (sometimes called a "causal" LM). This family also uses self-attention based transformers—but, when making a prediction for the word w_i at position i, it can only use information about input words $w_1, \ldots, w_{i-1}$ to do so. All of the other tokens following position i must be *masked* (hidden from view). **Encode a representation that masks input from $[w_{i+1},\ldots,w_n]$.**

QUESTION 3 [20 pts]: **Input/outputs for causal language model.**

We provide the training structure for a left-to-right causal language model; all that is needed is the correct inputs (x) and outputs (y). **Write a function that takes in a sequence of token ids $[w_1, \ldots, w_n]$ and segments it into 8-token chunks – e.g., $x_1 = [w_1, \ldots, w_8]$, $x_2 = [w_9, \ldots, w_{16}]$, etc. For each x_i, also create its corresponding y_i.** Given this language modeling specification, each y_i should also contain 8 values (for each token in x_i). Keep in mind this is a left-to-right causal language model; your job is to figure out the values of y that respect this design. At token position i, when a model has access to $[w_1, \ldots, w_i]$, which is the true y_i for that position? Each element in y should be a word ID (i.e., an integer).

2 Part 2: Perplexity

To evaluate how good our language model, we use a metric called perplexity. The perplexity of a language model (PP) on a test set is the inverse probability of the test set, normalized by the number of words. Let $W = w_1 w_2 \ldots w_N$. Then,

$$PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_1 \ldots w_{i-1})}}$$

However, since these probabilities are often small, taking the inverse and multiplying can be numerically unstable, so we often first compute these values in the log domain and then convert back. So this equation looks like:

$$\ln PP(W) = \frac{1}{N} \sum_{i=1}^{N} -\ln P(w_i|w_1 \ldots w_{i-1})$$

$$\implies PP(W) = e^{\frac{1}{N} \sum_{i=1}^{N} -\ln P(w_i|w_1 \ldots w_{i-1})}$$

Pseudo-perplexity for BERT. The perplexity calculation above presumes a left-to-right causal language model (note that predicting the probability of w_i conditioning only on words from position 1 to position $i - 1$. To calculate an comparable metric for bidirectional models like BERT, we'll use a pseudo-perplexity calculation:

$$PP(W) = e^{\frac{1}{N} \sum_{i=1}^{N} -\ln P(w_i|w_1 \ldots w_{i-1}, w_{i+1}, \ldots, w_n)}$$

In other words, to calculate the probability of a word at position i given all of the other words in the sentence, we'll simply mask that word from the input and run the entire sentence through BERT to predict the probability of that masked word. We'll do this for each word i in a sentence, one at a time.

QUESTION 4 [20 pts]: Implement Pseudo-Perplexity Calculation.

Implement the above pseudo-perplexity calculation for BERT, as detailed in the notebook. The function calculates the average probability of each token in the sentence given all the other tokens. We need to predict the probability of each word in a sentence by masking the one word to predict. Note that you should not include the probabilities of the

[CLS] and [SEP] tokens in your perplexity equation -- those tokens are not part of the original test sentence.

QUESTION 5 [20 pts]: **Write up - Difference in Perplexity between BERT Models**

Observe the results and highlight patterns that reflect characteristics of the BERT models. For each observation, use the observed pattern to justify characteristics of one specific model or a group of models. You should provide 3 observations with 50 words each. One example is provided for you.

NOTE: Include your short answer response in the collab notebook.

4 How to submit

Submit your work to Gradescope. There is one Gradescope submission link for the notebook.

- Submit to: “Gradescope HW3 code (.ipynb)”
  - Make sure you have replaced all `# to-do` with your solutions, and that they are placed, as in HW1&2, between the `# BEGIN SOLUTION` and `# END SOLUTION` flags.
  - Download your Colab notebook as an *.ipynb* file (File →Download *.ipynb*)
  - Submit *HW3.ipynb*. The file must be named in this way for the Gradescope autograder.