

# Homework 3: Masked Language Models, BERT and Perplexity

INFO 159 / 259, Spring 2023

**Due:** February 21, 2023 (11:59 PM)


---

This homework is designed to help you:

1. familiarize yourself with masking approach for transformer-based language models [deliverable 1]
2. understand the concept of perplexity, implement pseudo-perplexity for BERT [deliverable 2]

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

<https://github.com/dbamman/nlp23/blob/main/HW3/HW3.ipynb>

To use it, hit the “Open in Colab” 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, Q2, Q3, Q5), replace `# to-do` with your solutions, and as in HW1, keep your code between the `# BEGIN SOLUTION` and `# END SOLUTION` flags. For the short-answer question (Q4), submit your answer as a .pdf file separately (see §3 for instructions on how to submit).

## 1 Deliverable 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**  
[10 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 randomly masks token positions 2 and 7 (for an input sequence length of 10 tokens), as detailed in the notebook.

**QUESTION 2**  
[10 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 words  $w_i, \dots, w_{i-1}$  to do so. All of the other tokens following position  $i - 1$  must be *\*masked\** (hidden from view). Implement this mask.

**QUESTION 3**  
[10 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  $[w_1, \dots, w_n]$  and segments it into 8-token chunks – e.g.,  $x_1 = [w_1, \dots, w_8]$ ,  $x_2 = [w_9, \dots, w_{16}]$ , 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  $[w_1, \dots, w_i]$ , which is the true  $y_i$  for that position? Each element in  $y$  should be a word ID (i.e., an integer).

**QUESTION 4**  
[10 pts]

**Concept Understanding** (writeup only).

In this model, as implemented, does the following equivalence hold?

$$P(y_4 \mid w_1 = \text{go}, w_2 = \text{ahead}, w_3 = \text{make}, w_4 = \text{my}) = P(y_4 \mid w_1 = \text{ahead}, w_2 = \text{my}, w_3 = \text{make}, w_4 = \text{go})$$

Why or why not? Discuss in no more than 100 words.

## 2 Deliverable 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  $W = w_1 w_2 \dots w_N$ . Then,

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

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:

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

**QUESTION 5**  
[10 pts]

**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:

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

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. Implement this pseudo-perplexity calculation for BERT, as detailed in the notebook.

### 3 How to submit

Submit your work to Gradescope. There are two different Gradescope submission links, one for the PDF writeup and one for code:

- Submit to: “Gradescope HW3 writeup”
  - Submit your answer to Q4 as one PDF file to Gradescope. As long as you submit a PDF, there’s no additional requirements for formatting. There is no template for writeups.
- 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, between the `# BEGIN SOLUTION` and `# END SOLUTION` flags.
  - Download your Colab notebook as an `.ipynb` file (File → Download `.ipynb`)
  - Submit `HW_3.ipynb`. The file must be named in this way for the Gradescope autograder.