Exercise: Understanding Language Models

Goals The main goal of this worksheet is for you to understand language models a bit more before you implement them.

Question 1 Consider the prefix "*LeBron James talked about* _". Think about the different n-gram orders here: $n = 2$ (1 word of context for the language model, just *about*), through $n = 5$ (all four words of context). **Do you think $n = 3$ or $n = 4$ will yield the same distribution over next words as $n = 5$? Why or why not?**

It will be different because you really need to see the whole context to see LeBron. Even "James talked about" leads to a quite distribution of following words (who is James?).

Question 2 Consider the following corpus (collection of sentences), extended from the one in the video:

I like to eat cake but I want to eat pizza right now. Mary told her brother to eat pizza too. He went to Pizza Hut to get some.

What is the probability distribution of words following *to* under a 2-gram model? That is, what is $P(y \mid \text{to})$? Hint: this should be a list of words, each one associated with a probability. You don't need to explicitly write down all of the words with zero probability.

$P(\text{eat} \mid \text{to}) = \frac{3}{5}$, $P(\text{Pizza} \mid \text{to}) = \frac{1}{5}$, $P(\text{get} \mid \text{to}) = \frac{1}{5}$, all other words in the vocabulary have 0 probability.

Question 3 What data structure or data structures would you use to store the words and probabilities for $P(y \mid \text{to})$?

The best answers are two parallel arrays (one for words, one for probabilities) or two parallel ArrayLists/Lists. A Map from String to Double (Java) or dict (Python) is also acceptable.

(Optional) Question 4 Now suppose you were going to store the entire 2-gram model: the words and probabilities $P(y \mid x)$ for every (x, y) pair. What data structure or data structures would you use for this?

This is similar to Question 3 but scaled up. One way to do it is to have two Lists of Lists (or two 2D arrays), where each row stores a distribution $P(y \mid x)$ for a certain x following the method in Question 3. However, you still need a way of knowing which row (which index in the outer list) refers to which word x. The best solution is to have an "indexer," or a separate list that associates each word with a canonical index. If you have this, you can actually get away with a single 2D array called `probabilities` for everything: you can put x and y through the indexer and then access `probabilities[x][y]`.

There are many possible solutions and they get a bit tricky, hence why this part is optional.