Language Modeling

In this lab we will explore how to implement different language models for calculating probabilities for a specific sentence based on a defined corpus. Standard preprocessing was applied to the files, which included lowercasing, tokenizing and removing punctuation. Also symbols $< s >$ were included to the beginning and end of each sentence or question.

Unigram Model

Unigram models are quite straightforward to implement, in this case the corpus file was loaded and processed to count each token (word) appearance. Then a dictionary of words and their associated probabilities was created.

Afterwards to get the results for which of the 2 words was the most likely we just compared their probabilities.

The results from the model are **6 correct answers** and **4 wrong**, giving a **60% accuracy**, just a bit better than random choice.

Bigram Model

The bigram model allows for better results, adding more context to the estimation of the probabilities, so we should expect to get a better accuracy when computing the results.

To implement this model, we have to create the bigrams present in each question and also in the corpus.

After creating the bigrams the process is very similar to the unigram model, but instead of looking at the probabilities for the words in the two options, we evaluate the probability of the whole sentence, this is multiplying the probabilities of each bigram to see in total which one is higher (more likely). In this case of predicting a specific word inside a sentence, we don't even need the total sentence bigrams, just the previous word and the word after, because the probabilities for the other bigrams will be the same in each case.

The results from the model are **7 correct answers** and **3 answers with 0 probability** (the model can't choose between the two), we'll explain this further.

Why can't our model make a decision on these 3 options? The explanation is very simple, these questions contain bigrams that our model hasn't seen (they are not present in the corpus) so when it evaluates the whole sentence, at least one of the probabilities is 0, which makes the whole value 0.

For the case of the "serial / cereal" pair, the corpus does not have the bigram "eat serial" or "eat cereal". So both options end up evaluated as 0.

For the pair "chews / choose", the corpus does not contain the bigram pairs "normally chews" or "normally choose". So both options end up evaluated as 0.

And for the pair "sell / cell" the corpus does not contain the bigram for "cell it" or "im going". In this case if we evaluated only on the bigrams that contain the word sell or cell, we could obtain a different result, because the word sell has probabilities ¿ 0 in all those bigrams (and the rest are the same for both)

Bigram Model with Laplace Smoothing

In order to fix the problem with 0 probabilities we need to implement a technique called Laplace Smoothing, in simple words means giving a small probability to unseen bigrams, taking a bit of probability from very frequent words. This helps us because it removes the 0 probability case and allows us to always be able to select a word.

In this case we did a 2 step process which added the bigrams that were in the questions but not in the corpus, and then recalculated the probabilities.

1. Add bigrams from questions NOT IN corpus to total bigrams with value 0
2. Add 1 to every bigram in total bigrams
3. Recalculate probabilities for each bigram

The results of the model are **9 correct and 1 wrong**, with a **90% accuracy**, which is much better than the standard unigram.

The final comparison for the models can be seen in this table.

Model	Correct	Wrong	Equal Prob	Accuracy
Unigram	6	4	0	0.6
Bigram	7	0	3	0.7
Bigram with smoothing	9	1	0	0.9

I think this results are pretty in line with what I expected and show the power of bigrams in adding context for estimating probabilities. I tried to examine the word that none of the models was able to answer correctly ("chews" vs "choose") and it may probably be because our corpus is too small, so more common words still have too much influence in the decision.