a. what are n-grams and how are they used to build a language model

An n-gram is an n-sized sliding window used for text. It creates a model for a language based on probability by observing the frequency of n-sized phrases in a corpora. They are used to build language models, using this probability.

b. list a few applications where n-grams could be used

Like in the program, an n-gram can be used to figure out what language is being read. Additionally, since n-grams are generated through certain corpora, they can be used to sort texts, based on the probability generated by frequency of certain n-grams.

c. a description of how probabilities are calculated for unigrams and bigrams

unigram probability = occurrences of word/total words

bigram probabilities = occurrences of bigram/ occurrences of first word in bigram

d. the importance of the source text in building a language model

The source is obviously very important when building a language model. The language model is learning only from the information provided from a source. If we attempted to build a language model based off of very old English texts, but then used that model in applications where modern English was used, we cannot expect the model to be as effective. The text is supposed to be a representation of the language we want the model to be based off of, so the phrases it includes should be relevant to the use case.

e. the importance of smoothing, and describe a simple approach to smoothing

Smoothing addresses the sparsity problem for n-grams, which is that not every n-gram will be in the dicts, meaning the probability for some n-grams will be at 0, causing the language model to be ineffective at times. One approach to smoothing would be Laplace smoothing, which just adds one to the count of every word, and then calculates the probability, adding the extra words to the division, so it still adds up to one. This means that no word has a probability of 0.

f. describe how language models can be used for text generation, and the limitations of this approach

Given a word, we can use the language model generated by the n-grams to put in the most likely next word, and chain these to generate text based on the previous words. However, since these are just sentences generated by probability of what should come next, it is hard to capture any sort of intent or logic with text generated in this method.

g. describe how language models can be evaluated

There are many ways to evaluate a language model, and one of the ways is through perplexity. Perplexity is calculated through the formula PP(W) = P(w1w2….wN)^(-1/N). The lower the perplexity, the better the language model is. Another way of evaluating the model would be something like what was done in this project. Counting how many times a language model predicted correctly helps to gauge the accuracy of a model, which is another way we could evaluate a model.

h. give a quick introduction to Google's n-gram viewer and show an example

Google's n-gram viewer allows you to see how much certain n-grams have been used over time, and gives the probability of the n-gram to come up by looking at texts through many years. Here is an example of an output when searching for amazon and ebay.