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

- An n-gram is a sliding window over a given piece of text, that selects n tokens at a time. For example: a unigram will choose one word at a time, a bigram will choose two words at a time, a trigram will choose three words at a time, and so on. N-grams are used in building language models to help the system predict the probability of a given word appearing in a text, given a list of words preceding it.

B. List a few applications where n-grams could be used

- Auto completion of a sentence, Grammar checking, Spelling Error Detection, Speech Recognition, Language Translation.

C. A description of how probabilities are calculated for unigrams and bigrams

- Unigrams are a list of single tokens, extracted from a piece of text. To calculate the probability of a unigram, we just need to count the number of times a certain token appeared in the text, and divide that by the total number of tokens.

For Bigrams, to calculate the probability of two words appearing at the same time, we need to calculate the probability of the first word, and multiple that by the probability of the second word appearing after the first word. In simple terms, count the number of times the first token appears in the given text, and divide that by the total number of tokens. Then, count the number of times the two tokens appear together, in the given text, and divide that by the number of times the first token appears in the text. Lastly, multiply those two numbers.

D. The importance of the source text in building a language model

- It is very important to choose a good text source when building a language model for a specific application because that training data will decide the probabilities of each word appearing in the test data. If you use literature text source to build a language model, it will not be very useful when applied to scientific textbooks because both of them use very different vocabulary and combination of tokens.

E. The importance of smoothing, and describe a simple approach to smoothing

- The importance of smoothing is that we don’t want probabilities of zero for the algorithms or else it could be hard to determine the probability of how likely something is. So smoothing gives a small probability for unseen circumstances which also makes more frequent events less probable. One smoothing technique would be laplace smoothing algorithm which takes the bigrams and normalize the probabilities and if it where the counts are increased by 1 so 0 is 1.

F. Describe how language models can be used for text generation, and the limitations of this

Approach

- You can use language models to generate text by having a large corpus and classifying probabilities of words that are near each other. With that you can randomly select with weight adjustments for probabilities to use for text generation. The limitation on this approach is that

you can generate text but it makes sense at the start but as it goes you see it doesn't have meaning.

G. Describe how language models can be evaluated

- Language models can be evaluated by humans because we can get other people to annotate or come up with the right solutions and see how good the language model is compared to what other people said. Another metric would be perplexity if we have low perplexity, it is a better language model usually. Low perplexity might be bad as it is confident, but it doesn't have to be accurate it good be inaccurate still.

H. Give a quick introduction to google's n-gram viewer and show an example

- Google's n gram viewers show how often a word or a phrase has been used inside of a corpus of books. Google's n gram viewers only uses published books and has an emphasis on scientific books so it won't be perfect.

Ex: this is an example of unigrams