An n-gram is a model that iterates over text and takes in “n” words per iteration. The model will take in the first “n” words, then move over one word to take in another group of words. For example, the sentence “I go to the store”, would be iterated with a value of 2 as follows: (I, go), (go, to), (to, the), (the, store). This in turn makes a model of words that often sit next to each other in sentences. The n-gram model will take in these bigrams (in this case) and create a count of how many times it finds said bigram. This allows the model to detect patterns and start to understand how sentences are formed without having to learn grammar rules. N-grams can be used for many language processing applications such as generating sentences, detecting languages, and highlighting common combinations of words in text.

You can find the probabilities of unigrams can be determined by simply dividing the given unigram count by the total number of unigrams in a dictionary. For determining the probability of a bigram, you divide the given bigram count over the number of unigrams that are the first element of the bigram.

The source text is one of the most important parts of the n-gram model, because it determines the complexity and accuracy of the model. The model will take in the n-grams of the original source and apply it to a new source, so it is more accurate when the source material is related to the text analyzed or generated by the n-gram. If you took source material from a children’s book and applied it to a calculus class, the n-gram would not be able to make any connections between the material.

Smoothing is the process of taking a bigram that has no counts from the dictionary and assigning it a low value. This is important because we are able to store the bigram for future use and when using formulas relating to the count of a bigram, it is better to have a low value than a 0 value.

When an n-gram dictionary is filled, you can generate sentences with it. The model is able to look at commonly used n-grams and fit them together. These sentences usually resemble the source material, which is another reason why it’s important to have a complex and varied source. Text generation using this method can be unintelligible because the model has no context for what a word means, just how it is used with other words. We can evaluate how good a language model is by looking at how often it uses certain words. With a lower probability of seeing the same words over and over, the better the perplexity of the model is.

Google has a built in n-gram where you can select any phrase or word and see the usage of the word over time. I looked up the word “Castle” and found that there is a large decrease in the usage of castle over time. The peak was in 1807, and the usage steadily decreased until the 1990s. This is interesting because I can’t think of a reason that this word would raise in popularity recently. Chart, line chart

Description automatically generated