Manuel Salado Alvarado

MXS170018

CS 4395.001

N-Grams Narrative

Processing text is an ability we learn in our formative years to understand the real world through abstract symbols, words. Getting a computer to read in a string of letters is easy but getting it to understand that string of letters is what natural language processing aims to achieve. N-grams are an interesting tool in the Natural Language Tool Kit where they are not just a sequence of letters but a contiguous sequence of words in a text [1]. To elaborate, N-Grams are more than just substrings, they represent “word occurrences and mostly ignore word order” [1]. For example, a bi-gram extractor would flag “Merry Christmas” as a prevalent bigram in A Christmas Carol because the two words are very often found together in that story. Scrooge and Tiny Tim wouldn’t be considered a trigram because although a human would recognize they have a relation, they aren’t literally found next to each other in the text often. Tiny Tim, on the other hand would be an excellent bigram because those two words are found very often together even if Tiny and/or Tim are found independent of each other elsewhere in the text. N-Grams are one of the simplest methods to build a language model which can “estimate the probability of the last word of an n-gram given the previous words, and also to assign probabilities to entire sequences” [1]. Fundamentally, finding the probability that a given N-Gram is valid stems from the occurrence of the N-gram divided by a looser interpretation of the N-gram. If we wanted to check that “Lucy sells her sea shells” is valid we could divide it by a relaxed version of the problem so: Occurrence(“Lucy sells her sea shells”)/ Occurrence(“Sells her sea shells) [1]. However, this method would likely need an extensive reference document, especially since there is near infinite sentence possibilities. Source text is vital for this reason, we need a diverse, HUGE, sensible corpus to be able to efficiently observe a proper number of occurrences. It is probable that new words will be encountered in test sets, and we don’t want to assign them 0 probability, so we modify them so that they are at least included once in the text. LaPlace smoothing is very simple in that it “adds one to all the n-gram counts” before normalizing them into probabilities. In order to use language models as a mode of text generation, the model would need to have a very good understanding of how the language works. This is the process our brain plays when we create new sentences that have never been explicitly taught to us. If you ask a language model that has been trained on conversational oriented novels, they are likely to provide a good response to the question “How are you?”. The Language Model, however, wouldn’t be able to respond an entirely unique response because it can only generate what it has seen, and variations of what it has seen. Language Models can be evaluated by getting a test file and a solution file, where the solution file is the correct evaluations of the sentences in the test file. The test file is run into the language model, producing a prediction file which is then compared to the solution file for accuracy. Google Books N-Gram Viewer extracts N-Grams from every book available in Google books in search for the provided text. Figure 1 is an example of Artificial Intelligence. It seems to check out in accordance to when journals were being published about Artificial Intelligence. It seems to be stumped when Minecraft is passed in (figure 2), because Minecraft was created in 2011, but it shows occurrences in the 1930’s. This might be a misunderstanding where Google Books’ Ngram Viewer perceives Minecraft as a German word. Interesting how such a powerful tool is still susceptible to many of quirks of language.

Chart

Description automatically generated

*Figure 1: Google Books Ngram Viewer; Artificial Intelligence*

*Chart, line chart

Description automatically generated*

*Figure 2: Google Books Ngram Viewer; Minecraft*

Works Cited

[1] “Speech and Language Processing (3rd Ed. Draft) Dan Jurafsky and James H. Martin.” *Speech and Language Processing*, Stanford, https://web.stanford.edu/~jurafsky/slp3/.