**Portfolio Chapter 8 N-grams Narrative**

An n-gram is a sliding window over text where the size of the window is n characters, thus capturing n words at a time. N-grams can be used to create a probabilistic language model. The language model can be learned from a corpus. The corpus used to create the language model can greatly influence the model. The corpus is then transformed into a dictionary of counts, which is the language model.

Some use cases for n-grams include spelling correction, machine translation, speech recognition, and auto suggestion when typing/messaging/searching.

Generally, probability of unigrams in a corpus is calculated by dividing the count of occurrences of the unigram in the corpus divided by the total vocabulary size of the corpus. Additionally, the probability of bigrams in a corpus is calculated by probability of the 1st word occurring divided by probability of the 2nd word occurring with 1st word occurring before the 2nd word. Smoothing is used to deal with zero probabilities.

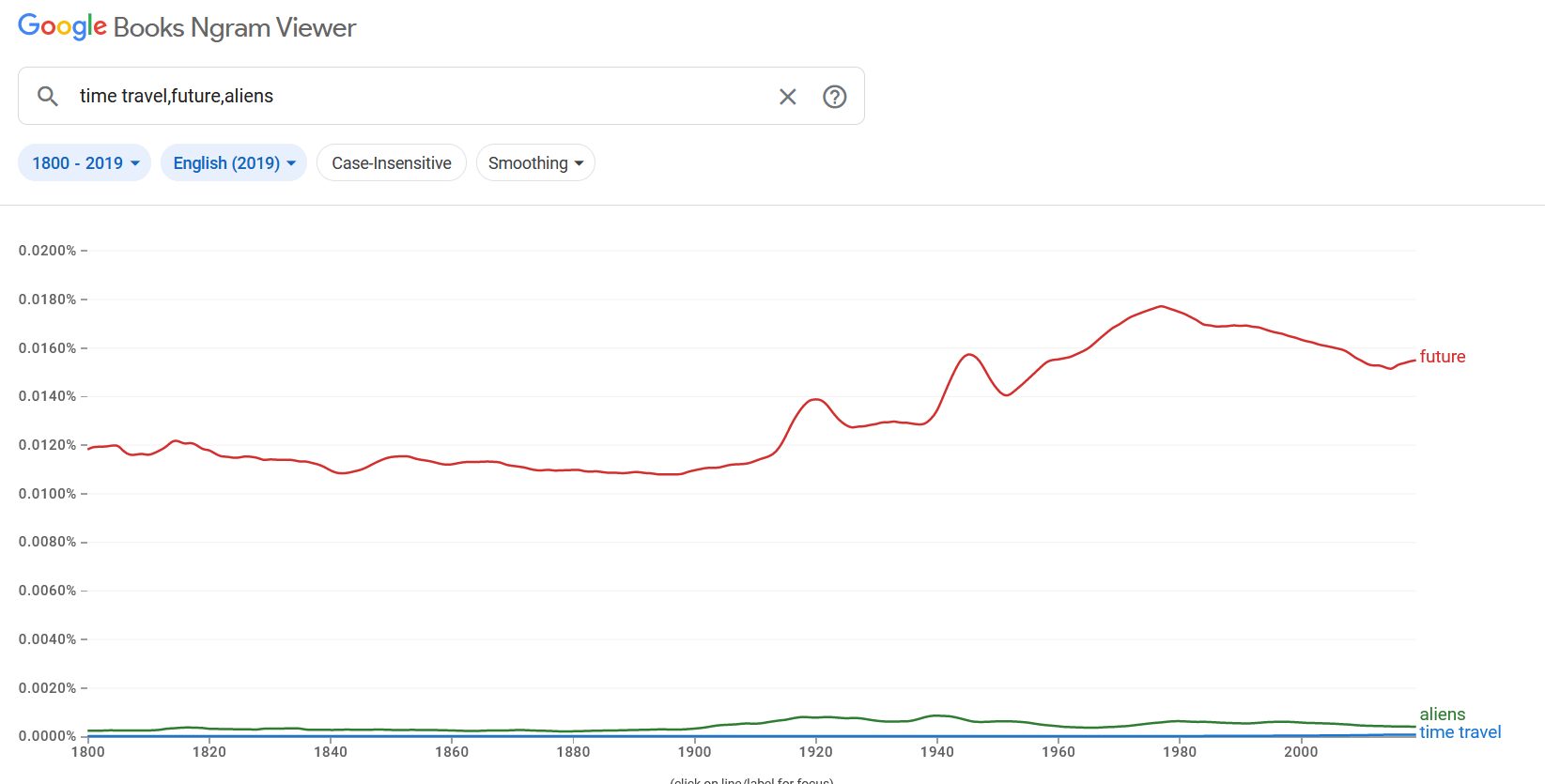
The source text will have a great influence in the creation of the language model. To create the language model, the source text is converted into a dictionary of counts to form the language model. Due to this, using a different source text can result in different probabilities since words and their sequences are different in different source texts.

Smoothing is used to deal with zero probabilities. Calculating probabilities using either unigram or bigram dictionaries involve just multiplication and division operations. With multiplication and division, if even one of the operands is zero, then the resulting probability will be zero or undefined. A simple approach to smoothing is Laplace smoothing. Basically, it adds 1 to the zero probability so that the outcome won’t be zero or undefined. It will add 1 to all the dictionary counts and total vocabulary size is added as a offset in the denominator.

Language models can be used for text generation. Given a start word, a process can look through a list of bigram probabilities to find the bigram with the start word in the first position that has the highest probability. The bigram is concatenated to the overall phrase this process is trying to generate. This will keep going until the last token added is a period. Bigrams get worst results than trigrams. The higher order the n-gram, the better the result. Dictionary counts are needed to create language models used for text generation. Creating the dictionary over large corpus can be very time-consuming. The corpus used to create the dictionary is limiting the effectiveness of the model that is used for text generation, since the corpus being does not contain every sequence of words that exists, which is why complete accuracy cannot be achieved when using language models for text generation.

Language models can be evaluated by having humans evaluated the results using predefined metrics, or compare language models using some metric.

Google n-gram viewer is an online search engine that provides a graphical representation of any set of search strings using a yearly count of n-grams found in printed sources published between 1500 and 2019 [1].



**References**

[1] Wikipedia Contributors, “Google Ngram Viewer,” Wikipedia, Mar. 30, 2019. <https://en.wikipedia.org/wiki/Google_Ngram_Viewer>

[2] Google, “Google Ngram Viewer,” *Google.com*, 2012. <https://books.google.com/ngrams/>

‌

‌