Mentesinot Cherenet
CS4395-001
Professor Karen Mazidi
March/04/2023

Assignment 4 Narrative about N-grams

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

N-grams are contiguous sequences of N words from a given text. These sequences are used to build a statistical language model that predicts the likelihood of the next word given the previous N-1 words in a sentence. So to build a language model using n-grams, we must tokenize a substantial text corpus into individual words or sub-words. After that, we count the number of times each N-gram appears in the corpus for each sentence, calculate their probabilities, and generate them all. New text generation, text classification, machine translation, and other natural language processing tasks can all be performed with the resulting model.

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

Grammar Checking, Language Translation, Speech recognition, text summarization, used to identify languages or to detect plagiarism.

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

Probabilities are calculated by counting the number of times a word appears in a corpus and dividing by the total number of words in the corpus. The probability of the first word is multiplied by the probability of the second word in the bigram to get the probability of the bigram.

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

It is vital to pick a good text source while building a language model for a particular application since that training data will conclude the probabilities of each word showing up in the test information. It is also important because it is used to calculate the probabilities of words and bigrams. The more words and bigrams in the source text, the more accurate the probabilities will be

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

In order to get rid of outliers in a probability set, smoothing is used. For instance, if a word appears only once in a corpus, its probability is 1. Since the word is unlikely to appear again, this is of little use. Smoothing reduces the likelihood of bigrams and rare words. Laplace smoothing is a common, straightforward method of smoothing that adds one to the count of each corpus word and bigram.

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

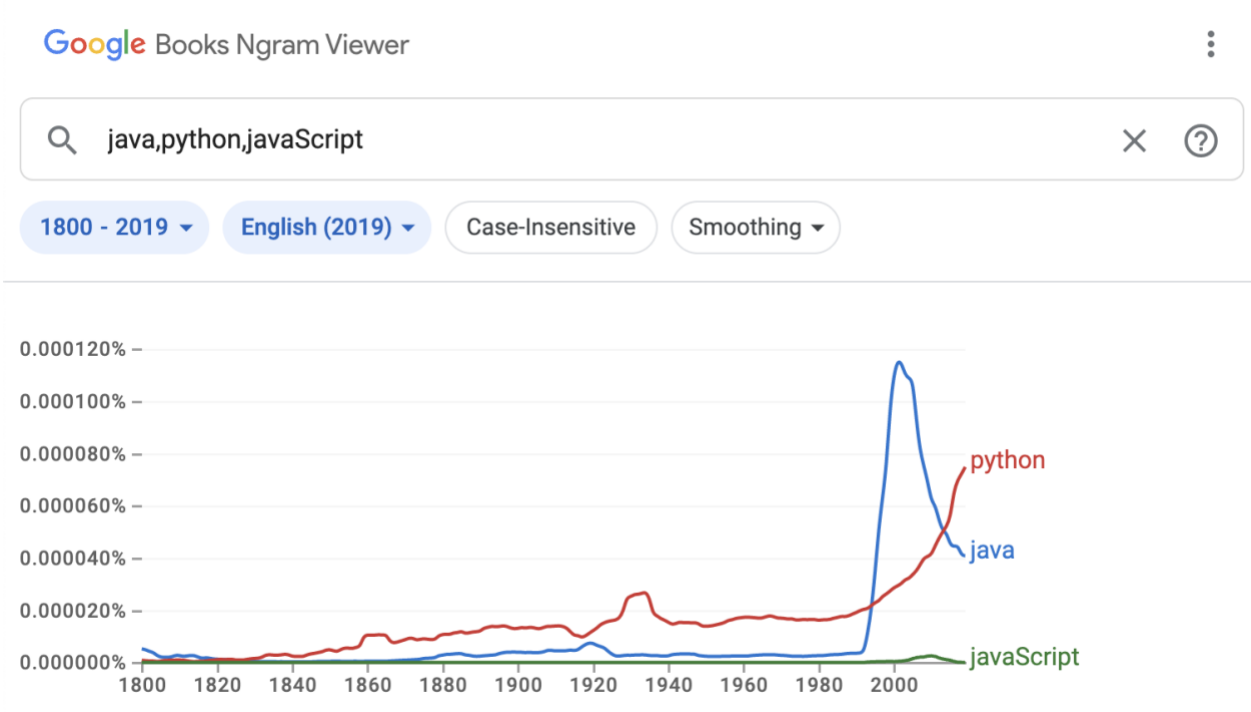
language models can generate text by categorizing the probabilities of words that are close to each other and having a large corpus. That enables you to generate text by selecting at random and adjusting the probabilities' weights. This method has one drawback: you can generate text that initially makes sense but has no meaning as it progresses.

g. **Describe how language models can be evaluated**

Word and bigram dictionaries with probabilities can be created using language models for text generation. A number between 0 and 1 is chosen at random from the probabilities. The word or bigram with the lowest probability of the random number is selected. Until a sentence is produced, this procedure is repeated. The generated text will not make sense because the probabilities are not based on the sentence's context, which is one of this method's limitations. For instance, the word "the" will almost certainly appear in a sentence, but not in the proper context. Because it does not take into account the context of the sentence, this approach will not produce text that is coherent. Grammar rules are also ignored by this.

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

Google's n-gram viewer is used to visualize the relative frequency of words in a corpus. It has a large selection of corpuses to choose from, and will show the probability of each word showing up at different points in time


Google Books Ngram Viewer
java,python,javaScript
1800 - 2019
English (2019)
Case-Insensitive
Smoothing
0.000120%
0.000100%
0.000080%
0.000060%
0.000040%
0.000020%
0.000000%
1800
1820
1840
1860
1880
1900
1920
1940
1960
1980
2000
python
java
javaScript