CS 4395.01 Human Language Technologies

Portfolio Chapter8: Ngrams

Instructor: ***Dr. Karen Mazidi***

Organized by:

Ishaaq Razack - imr180000

Savishwa Gaur – sxg180113

Date: Oct 2, 2022

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

An n-gram is a sliding window of size n over text. For example, a unigram would be of size n=1, and a bigram would be of size n=2. Taking the sentence “I am Sam,” we see that the unigrams are [I, am, Sam], while the bigrams are [I am, am Sam].

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

N-grams can be used to create a probabilistic model of a language. For example, in this assignment we demonstrated how we could use n-grams to predict whether a document is in English, French, or Italian.

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

To calculate the probability of a sequence of words $w_1, w_2, ..., w_n$ for unigrams, we can use the equation $P(w_1, w_2, ..., w_n) = P(w_1)P(w_2|w_1)... P(w_n|w_{n-1})$, where $P(w_i)$ is the probability of the unigram (or word) w_1 appearing in the corpus. For bigrams we do use a similar process, but instead of w_i representing each individual word, it represents each bigram.

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

The choice of corpus used in training a language model using n-grams can highly influence the model’s accuracy. Therefore, it is best to use a training corpus that is similar to the data that the model will be trying to predict.

e. The importance of smoothing

There exists a sparsity problem when calculating probabilities which is that not every possible n-gram will be in our dictionaries. Smoothing fills in 0 values with a bit of probability mass. One example is LaPlace smoothing which adjusts the probabilities based on the total number of unigrams and the vocabulary size. The unsmoothed probability is $P(w_i) = \frac{C(w_i)}{N}$, where N is the total number of unigrams, and the smoothed probability is $P(w_i) = \frac{C(w_i) + 1}{N + V}$, where V is the vocabulary size.

f. How language models can be used for text generation, and the limitations

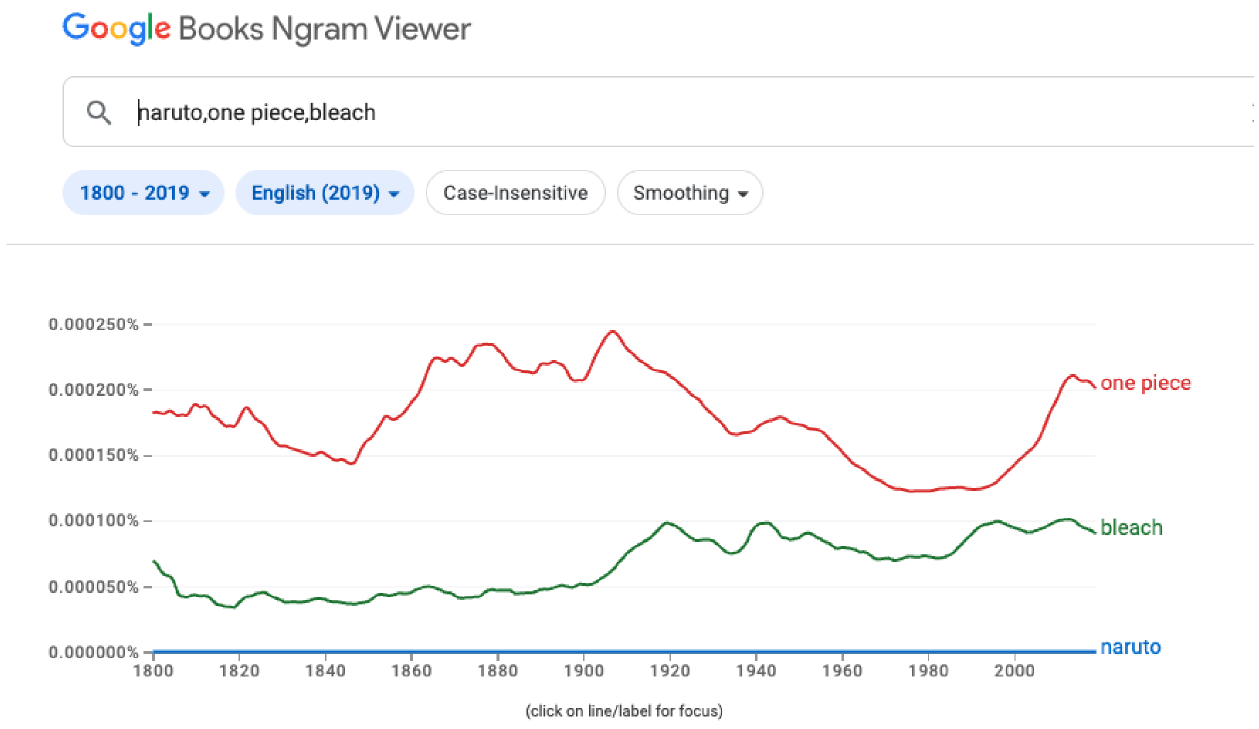
First, create probability dictionaries from the corpus. Then we take a very naive approach to language generation: given a start word, find the most likely next word and continue until we get a sentence end. In this approach, we mainly use unigrams, but trigrams will end up working better. In general, one limitation is that we need higher n-grams to make generation work better, and for even better results we need a large corpus.

g. How a language model can be evaluated

There are two evaluation approaches we can choose from: extrinsic (human annotators) and intrinsic (an internal metric like perplexity). Perplexity (PP) is the inverse probability of seeing the words we observe, normalized by the number of words, and is defined by the following equation: $PP(W) = P(w_1 w_2 ... w_N)^{\frac{-1}{N}}$. We aim for a low perplexity score

h. Introduction to Google's n-gram viewer

Google n-gram viewer is a search engine that displays the use frequencies of the search parameters over time using the count of n-grams in published sources starting from 1500 until 2019. The tool supports English, Chinese, French, German, Hebrew, Italian, Russian, and Spanish. The screenshot below shows the tool being run to compare the frequencies of the Big 3 anime: Naruto, One Piece, and Bleach. One Piece looks to have the most frequency.


Google Books Ngram Viewer
naruto,one piece,bleach
1800 - 2019
English (2019)
Case-Insensitive
Smoothing
0.000250%
0.000200%
0.000150%
0.000100%
0.000050%
0.000000%
1800
1820
1840
1860
1880
1900
1920
1940
1960
1980
2000
one piece
bleach
naruto
(click on line/label for focus)