David Nguyen

dxn180015

Dr. Karen Mazidi

CS 4395.001

HW4 - Ngrams

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

N-grams each contain text that consists of N words or tokens and is a sliding window of size N over the entire text. For example, given the sentence “a dog is there”, the unigrams would be [a, dog, is, there] and the bigrams would be [a dog, is there]. N-grams are used to build a language model by creating a probabilistic model of language. These models are used to predict the possibility of a sequence of N words, depending on the size and number of n-gram models, given the previous set of previous words.

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

A few applications where n-grams could be used are in cases where a user is working with text and may want to generate text. This can be used in many applications such as spell-check, word prediction / autocomplete, and even identifying languages.

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

For unigrams, they assume the probability of each word is independent of any words before it, so the probability of a word is calculated by dividing the number of occurrences of that word in the training text over the total number of words in the training text. While for bigrams, they use the formula "P(w1, w2) = P(w1)P(w2|w1)" where P(w1) is the count of w1 / number of tokens in text. The probability depends on and looks at the previous word(s).

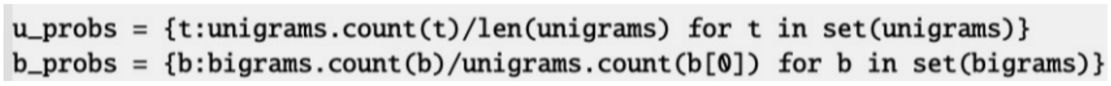
D. The importance of the source text in building a language model

The source text is important in building a language model because if it is not big enough or if the source is not suitable, the accuracy will be lower when used for specific applications. Say you can't use a textbook as a source to predict how people talk casually.

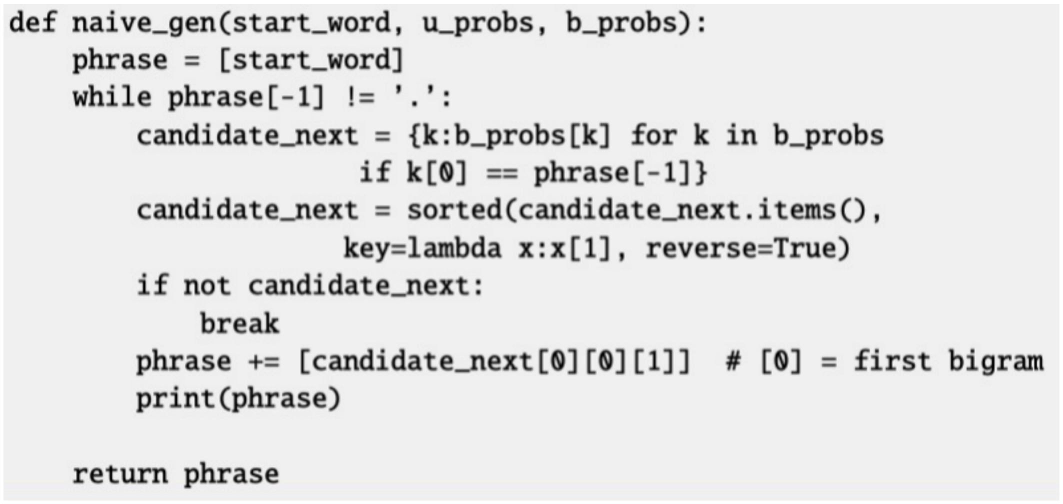
E. The importance of smoothing, and describe a simple approach to smoothing

Smoothing is important because not every possible n-gram will be in our dictionary, so we "smooth" that out by filling in 0 values with a bit of probability mass. A simple approach to smoothing is the Modified Good-Turing smoothing which replaces 0 probabilities with counts/probabilities of words that occur only once ($P(W_0) = N_1 / N$).

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



```
u_probs = {t:unigrams.count(t)/len(unigrams) for t in set(unigrams)}
b_probs = {b:bigrams.count(b)/unigrams.count(b[0]) for b in set(bigrams)}
```



```
def naive_gen(start_word, u_probs, b_probs):
    phrase = [start_word]
    while phrase[-1] != '.':
        candidate_next = {k:b_probs[k] for k in b_probs
                          if k[0] == phrase[-1]}
        candidate_next = sorted(candidate_next.items(),
                         key=lambda x:x[1], reverse=True)
        if not candidate_next:
            break
        phrase += [candidate_next[0][0][1]]  # [0] = first bigram
        print(phrase)

    return phrase
```

They can generate text by creating probability dictionaries or a naïve approach and using large n-gram language models to predict the next sequences of words when given the previous word(s) by using the largest probability. However, the limitations of this approach is that you need a large corpus for it to work the best and the higher n-grams will work better.

G. Describe how language models can be evaluated

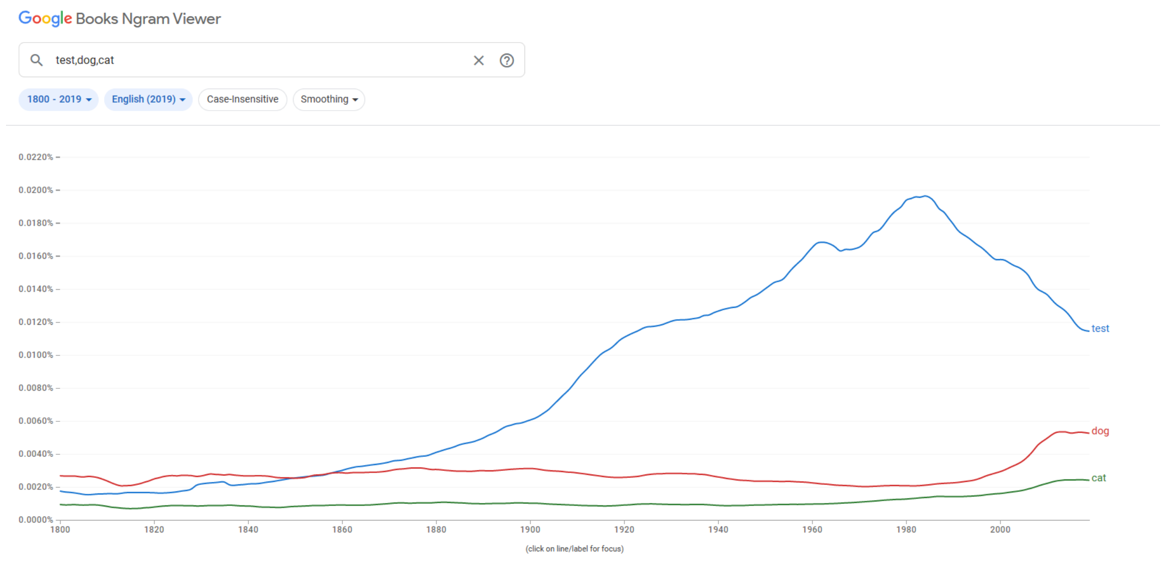
Language models can be evaluated either extrinsic (human annotators) or intrinsic (by using a formula like perplexity). Where Perplexity (PP) is the inverse probability of seeing the words we observe and normalized by the number of words.

$$PP(W) = P(w_1 w_2 \ldots w_N)^{-\frac{1}{N}}$$

The lower the perplexity, the better it is.

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

Google's Ngram Viewer is a search engine that allows you to type strings or n-grams and see the frequencies of that set. This is possible as Google uses a yearly count of n-grams found in Google's text corpora which contain sources between 1500 and 2019.



Here you can see "dog" is more popular than "Cat" but "test" eclipses both of them after 1850.