Jimmy Harvin & Charles Wallis
3/3/2023

N Gram & Language Models

An n-gram is a series of n contiguous words in a text; for example, a unigram would be a single token, and a 4-gram would be a collection of 4 words. Collections of all n-grams in a text can be used to create probabilistic models for both text processing and generation. In processing, seeing how the n-grams in a given text match up with n-grams on a training corpus can be used for classification, such as identifying authors or languages. For generation, one can start with an arbitrary word and continually find the most likely proceeding word based on n-grams of various lengths; a naive approach of this can be used to suggest upcoming words, which is done on most mobile devices. N-grams and language models created using the technique have many applications, from classifying and text generating, to machine translation, and sentiment analysis. Machine translation could utilize n-grams to determine what the most likely translation for a given sequence of text is best appropriate.

The probability of encountering any given unigram given a training text is very simple, merely the count of that specific word divided by the total number of words in the text. Bigram frequency must be calculated using conditional probabilities; the probability of encountering a given bigram is the probability of encountering the first word, which is calculated using the unigram method, multiplied by the probability of encountering the second word immediately after the first. This would be the count of that specific bigram in the text divided by the count of the first word. This formula can be generalized to larger n-grams by chaining conditional probabilities together for each word in the n-gram. Since these probabilities depend exclusively on the training corpus, language generation models will tend to produce sentences similar to the content of the corpus. It is important to select training text that is sufficiently large and matches the style that one would want to identify or replicate.

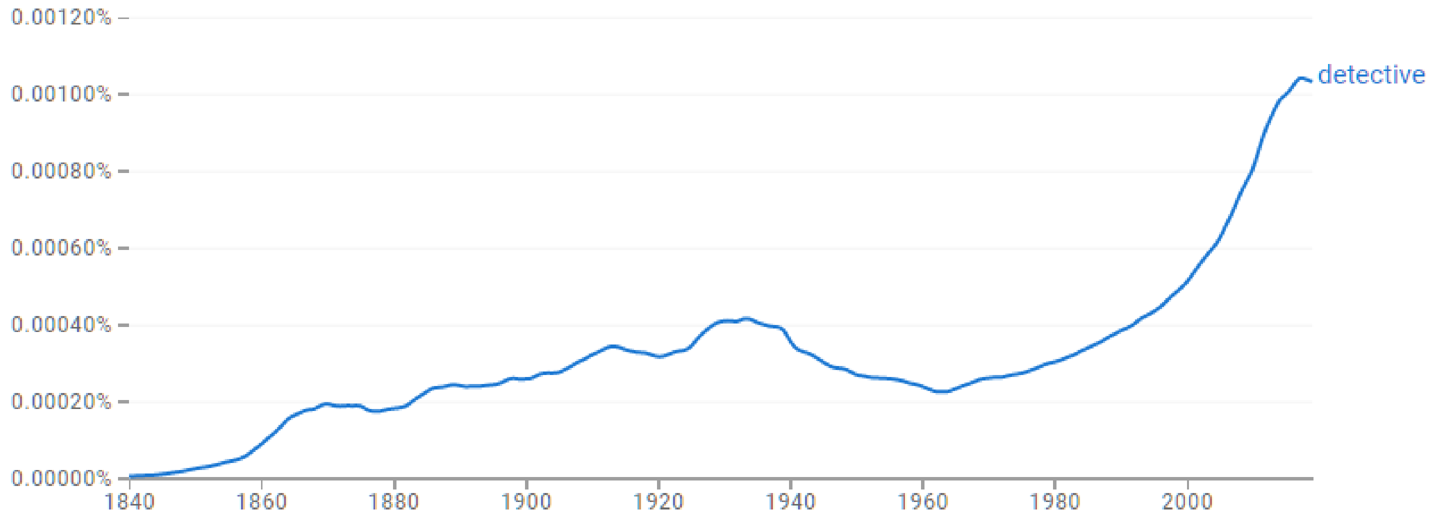
The source text is also important in building a language model, since it determines the context and style of what the language model is trying to target. A language model trained with a particular style will tend to generate text in a similar style, and even similar context. So for a large language model, it is important to find a large training set that is diverse in style. Also, the quality of the model also depends on the source text. Since computers aren't really generating something new, their outputs will always depend on what input quality. The training set needs to be well structured with no errors for a desired model.

Smoothing is also an essential technique that is used to work around with n-grams that the model was not trained with, because not every possible n-gram will be in our dictionary during training. Smoothing will fill in the 0 values of these unseen n-grams with a bit of probability mass. The add-1, or LaPlace smoothing, adds one to all n-gram counts in the training data before calculating the probability, to ensure that all of them have a non zero probability. This is a simple approach yet effective way of smoothing. During the assignment and language prediction model, this method of smoothing was used as well. There are other smoothing methods such as good-turing, which replaces zero counts, probabilities with counts, probabilities of words that occur only once.

Language models can be used for text generation as well, by predicting the likelihood of a given sequence of words. When writing emails, sometimes there are automatic suggestions, or on your smartphone you will get next word suggestions based on what's written. For a long text generation, the most likely word is chosen as the next word, and this process is repeated until a sentence is ended or until the desired length is reached. This approach works best with higher n-grams, and a large corpus helps with smooth text generations. Limitations apply to this method though, to the language and style of the training set. The generated text may sound repetitive to the previous text since it is based solely on the patterns that it sees during the training corpus. Another issue that is seen often on smartphone auto suggests, is that words are often semantically incorrect even when it may be grammatically correct.

Evaluating a language model can be done in several ways. Extrinsic evaluation is done by human annotators, and intrinsic evaluation may be done by an internal metric like perplexity. Perplexity is the inverse probability of seeing the words we observe, which is normalized by the number of words. Lower perplexity means that we have a more ideal language model, as it better predicts the test set.

Google's Ngram viewer is an online search engine that graphs the frequency of annual n-grams found in printed sources from 1500 to 2019, based on Google's digitized books collection in English, Simplified Chinese, French, German, and more. Specialized English corpora such as American English, British English, and English Fiction also exists. For example, when I search the term "Detective", I can see that there is an increase of word usage starting from the late 1850s to 1870, then a steady increase until its decline starting from the late 1930s. Then, we see another rapid increase from the 1960s until the early 2010s.


detective