Chris Straschewski

CMS180014

CS 4395.001

Narrative: N Grams

a. An n-gram can be described as a sliding window, reading text one word at a time, two words at time, or three or more words at a time. For example, in the sentence, “John went to school,” the unigrams are “John”, “went”, “to”, and “school”, and the bigrams are “John went”, “went to”, and “to school”. N-grams are used to build a language model by taking a certain body of text, or corpus, and extracting information from it. Once you have extracted information from it, calculations can be made, such as the probability that certain phrases will appear within the text.

b. The applications of N-grams involve anything where calculating probability of words and phrases in text is present, so mainly things like text prediction, language identification, and text compression, etc.

c. There are a number of different ways probabilities are calculated regarding unigrams and bigrams, but it is most commonly like this: unigrams: number of times the unigram appears divided by the total number of words in the text. Bigrams: number of times the bigram appears divided by the number of times the first unigram in the bigram appears.

d. The source text is obviously very important when building a language model. Everything you do with it is built from it. It’s the foundation. All of the calculated probabilities are only as accurate as the text itself.

e. Smoothing is important because it eliminates the possibility of getting zeroed-out probabilities. A simple approach used is called Laplace smoothing. For unigrams, instead of having the count of a specific unigram divided by the total number of unigrams, 1 is added to the numerator and the total vocabulary count is added to the denominator.

f. Language models can be used for text generation by applying all of the probabilities calculated with a language model into an approach that predicts text. Basically, trying to predict the next word given the previous words. Some limitations of this are that the generator will only be as good as whatever approach is used, and a large corpus must be used to get any kind of accurate result. However, operating with dictionaries from huge corpuses takes a long time. There is a way around this at least, as after you have created the dictionaries once, you can load them into pickle files and open them up in a separate program nice and fast.

g. Language models are usually evaluated by experts using some kind of advanced evaluation methods. However, this is an expensive and time-consuming thing to do, so it isn’t done often. One specific method used is the calculation of perplexity, which is the inverse probability of seeing the words we observe, normalized by the number of words.

h. Google’s N-gram viewer pulls word frequencies from a massive corpus of books spanning a period of over 500 years, and allows the user to freely test these word frequencies. Basically, you can type different words or phrases into it and get an estimate of how often the respective words were used over a certain time period. In the example below, I tried combining something I would only expect to see more of recently (social media) with some more common words.

