Kinsey Mellon
Ksm180006

N-grams Assignment Report

An N-gram is like a sliding box over some sort of text. For example, there are unigrams which are a single word, bigrams which are 2 words, trigrams which are 3 words, and then "n-grams' above 3 such as 4-grams and 5-grams. N-grams are mainly used to make a probabilistic model of language, but they need a corpus, or body of text, to do that. To make a language model the corpus needs to be divided by sentence and then a list of the unigrams and bigrams needs to be made. The probability comes from the probability of the bigram happening in the specific corpus. For example, the probability of "sherry walked" is the probability of the unigram sherry times the probability of walked given sherry. Dictionaries are used to make keeping the count easier to use when finding the probabilities. Some examples of applications of n-grams are in language generation, spelling correction, machine translation, speech recognition, and auto suggestion typing,

The probabilities of each unigram and bigram are calculated a little bit differently. For a unigram, the probability is the count of that unigram in the corpus divided by the total number of words. On the other hand, the probability for bigrams is the count of that bigram in the corpus divided by the count of the first word.

Choosing the source text is an important part of building a language model and greatly influences the resulting model as well. Models will be completely different if they are made from opposing source texts because they are based off the source material. One model made from a classic novel will be different from a model made from a college level math textbook because the content in the two texts is not similar at all. There will be overlap of the more common English words, but the majority will be different.

Another important part of building a language from n-grams is to include smoothing. The basic concept of smoothing is to account for any cases of zero and fill it in with another value so it doesn't ruin the overall probability. One simple type of smoothing is laplace smoothing or also known as add-one smoothing. The idea behind laplace smoothing is to add 1 to the 0 count so that it's not 0 anymore and doesn't run the risk of zeroing out the probability. To do that though, we need to add 1 to all counts because we don't know where/when a 0 will show up a head of time. Although this approach is simple it's very aggressive and does not perform that well.

One example of the usage for n-grams language models was in language generation, let's dive a little deeper into that concept. N-gram language generation starts with a start word and then looks for the bigram with the highest probability with the start word in the first position and continues in the same fashion until the last token was added, which can be a period. This approach to language generation is limited to the size of the text. A smaller corpus will not produce very good results, but a larger corpus takes more time. Another limitation is using simple unigrams or bigrams, the higher the n-gram it will work better.

One of the main ways language models can be evaluated is by the perplexity metric. Perplexity measures how well the model can guess the text in the test data. In math terms this is the inverse probability of finding the observed words normalized by the total number of words. A good model will have a low perplexity score meaning there is not a lot of chaos in the text. Another way to evaluate language models is to have human annotators analyze the results but that takes a lot of time and money.

Kinsey Mellon
Ksm180006

Google's n-gram viewer shows a graph displaying how often specified phrases have occurred in a corpus of books over a set number of years. Here's an example with the books Harry Potter, Alice in Wonderland, and Percy Jackson.