Narrative on Assignment 4

N-grams are windows that slide over text. Given a value, N-grams capture N words at a time. There exists different variations of N-grams. For instance, unigrams and bigrams respectively record one and two words at a time. Given a corpus, N-grams can generate probabilistic language models. They model the corpus and measure the probabilities of word occurrences. Different corpuses will produce new language models. These models leverage unigrams and bigrams for contextual understanding and word prediction. N-grams support different NLP applications. Some use cases are auto spell check and auto completion of sentences.

N-grams multiply probabilities to determine the probability of word occurrences. They assume that each probability depends only on previous words. Regarding unigrams, their probabilities are calculated by dividing the number of specific unigrams by the total count of words. For bigrams, their probabilities are calculated by dividing the number of specific bigrams by the count of the bigram's first word.

Smoothing solves scenarios that involve data sparsity. This problem occurs when N-grams cannot find words in the corpus. Without smoothing, the probability formulas will zero-out. Smoothing helps generate new probabilities for sparse words. It replaces zero values with some probability of the overall distribution. This helps produce a smoother distribution of words. One example is Laplace smoothing. Specifically, it increments every N-gram count by 1. In order to offset this change, Laplace smoothing increases the denominator value by adding the total vocabulary count.

Regarding text generation, language models create probability dictionaries of N-grams. When provided with a start word, the program keeps appending a sentence with new bigrams. The program will iterate until it selects a terminating token. This approach is limited by its simplicity and small corpus size.

Language models can undergo different forms of evaluation. These evaluations range from extrinsic to intrinsic metrics. Regarding extrinsic evaluations, human annotators use a predefined metric to score language models. For intrinsic evaluations, programmers can compare language models with the measurement — perplexity.

Google's N-gram viewer displays the frequency of given N-grams over a certain timeframe. Users can select different corpuses of books to analyze N-grams. Users can interact with the charts and focus on different N-grams. One example would be the unigram — "robot". The N-gram viewer displayed this unigram's frequency from 1800 to 2019. Although dormant from 1800 to the 1970s, the unigram spiked in popularity around the 1980s. Its popularity decreased in the early 2000s, but rose again in the 2010s.