1 N-Gram Language Models

1.1 Definition

Applying the markov assumption to text for the purpose of building up a language model (a distribution of probabilities):

$$P(w_i|w_{<i}) = P(w_i|w_{i-1}) \tag{1}$$

The above specific case is a 1st order markov model- a 'bigram' language model; each word's probability is conditioned on the preceding word only. We can extend this to an nth order markov model- an n-gram:

$$P(w_i|w_{i-n+1<i}) \tag{2}$$

1.2 Learning the probabilities - MLE

$$P(w_i|w_{i-n+1<i}) = \frac{\#counts(w_{i-n+1<i}, w_i)}{\#counts(w_{i-n+1<i})} \tag{3}$$

1.3 Evaluating models

Evaluating the probability of a sequence of text based on the language model could be done a bunch of ways, but if it were simply the joint probability then it would be sensitive to the sequence length.

So, commonly, a measure called 'perplexity', the inverse joint probability normalised by the length (# words), is used. The perplexity of a sequence of text W is:

$$PP(W) = \sqrt[N]{\frac{1}{\prod_i^N P(w_i|wi-n+1<i)}} \tag{4}$$

where W is the sequence of text

1.4 Smoothing, Backoff and Interpolation

n-gram combinations that are unseen in the training corpora can cause the joint probability for a sequence to equal zero. A simple way to combat this is by shifting mass (e.g. 'add-one smoothing'):

$$P(w_i|w_{i-n+1<i}) = \frac{\#counts(w_{i-n+1<i}, w_i) + 1}{\#counts(w_{i-n+1<i}) + |V|} \tag{5}$$

An alternative is to 'back off' the n-gram order until the gram is observed in the vocabulary. Alternatively, Interpolation enforces constant backoff by mixing N n-gram models (where N is the order.)

$$P(w_i|w_{<i}) = \sum_{n=1}^{N} \lambda_n P(w_i|w_{i-n+1<i}) \tag{6}$$