Smoothing, Backoff and Entropy

CS702 B.Tech & M.Tech 2025 ,

Shreya Agarwal

Important points

A higher n-gram value captures **more context** and can lead to **better predictive performance**, but it also introduces the significant challenges of **data sparsity** and **increased computational requirements**.

A balance must be struck to optimize performance for a given application.

Points

1. If the word ‘ham’ never occurs in our training set, but is in the test set? The answer is that although words might be unseen, we normally run the NLP algorithms.

2. With subword tokenization, these algorithms depends not on words but on subword tokens. Any word can be modeled as a sequence of known smaller subwords. The language model vocabulary is normally the set of tokens rather than words, and in this way the test set can never contain unseen tokens.

3. What do we do with words that are in our vocabulary (they are not unknown words) but appear in a test set in an unseen context (for example they appear after a word they never appeared after in training)?

Limitations of Vanilla N-gram Model

1. There is a problem with using maximum likelihood estimates for probabilities: any finite training corpus will be missing some perfectly acceptable English word sequences. That is, cases where a particular n-gram never occurs in the training data but appears in the test set.
2. These unseen sequences or zeros—sequences that don't occur in the training set but do occur in the test set—are a problem for two reasons.

a. The zero probability value means we are underestimating the probability of word sequences that might occur, which hurts the performance of any application we want to run on this data.

b. Perplexity is defined based on the inverse probability of the test set.

Way Forward

Smoothing or Discounting:

Smoothing

Discounting algorithms take-off a bit of probability mass from some more frequent events and give it to unseen events.

1. Laplace (add-one) smoothing
2. Stupid backoff,
3. N-gram interpolation.

Laplace Smoothing

The simplest way to do smoothing is to add one to all the n-gram counts, before

we normalize them into probabilities. Laplace smoothing merely adds one to each count (hence its alternate name addaddone one smoothing). Since there are V words in the vocabulary and each one was incremented, we also need to adjust the denominator to take into account the extra V observations.

$$P_{\text{Laplace}}(w_i) = \frac{c_i + 1}{N + V}$$

Continued….

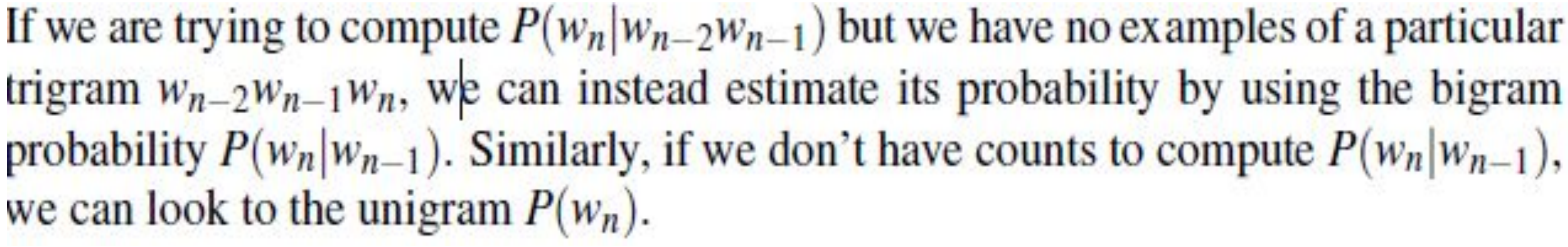
Question

What happens to our P values if we don’t increase the denominator

$$P_{\text{Laplace}}(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)+1}{\sum_w (C(w_{n-1}w)+1)} = \frac{C(w_{n-1}w_n)+1}{C(w_{n-1})+V}$$

Backoff and Interpolation

The discounting can help solve the problem of zero frequency n-grams. In a backoff n-gram model, if the n-gram we need has zero counts, we approximate it by backing off to the (n-1)-gram. We continue backing off until we reach a history that has some counts.


If we are trying to compute $P(w_n|w_{n-2}w_{n-1})$ but we have no examples of a particular trigram $w_{n-2}w_{n-1}w_n$, we can instead estimate its probability by using the bigram probability $P(w_n|w_{n-1})$. Similarly, if we don't have counts to compute $P(w_n|w_{n-1})$, we can look to the unigram $P(w_n)$.

continued….

There are two ways to use this n-gram “hierarchy”. In backoff, we use the trigram if the evidence is sufficient, otherwise we use the bigram, otherwise the unigram. In other words, we only “back off” to a lower-order n-gram if we have zero evidence for a higher-order interpolation n-gram.

In interpolation, we always mix the probability estimates from all the n-gram estimators, weighting and combining the trigram, bigram, and unigram counts.

Interpolation

In simple linear interpolation, we combine different order n-grams by linearly interpolating them.

$$\begin{aligned}\hat{P}(w_n|w_{n-2}w_{n-1}) &= \lambda_1 P(w_n)\\ &+\lambda_2 P(w_n|w_{n-1})\\ &+\lambda_3 P(w_n|w_{n-2}w_{n-1})\end{aligned}$$

How are these lambda values set?

Both the simple interpolation and conditional interpolation are learned from a held-out corpus. A held-out corpus is an additional training corpus, so-called because we hold it out from the training data, that we use to set hyperparameters like these l values.

We do so by choosing the ‘L’ values that maximize the likelihood of the held-out corpus. That is, we fix the n-gram probabilities and then search for the ‘L’ values that give us the highest probability of the held-out set.