CHAPTER

B Kneser-Ney Smoothing

Kneser-Ney

A popular advanced n-gram smoothing method is the interpolated **Kneser-Ney** algorithm (Kneser and Ney 1995, Chen and Goodman 1998).

B.1 Absolute Discounting

Kneser-Ney has its roots in a method called **absolute discounting**. Recall that **discounting** of the counts for frequent n-grams is necessary to save some probability mass for the smoothing algorithm to distribute to the unseen n-grams.

To see this, we can use a clever idea from Church and Gale (1991). Consider an n-gram that has count 4. We need to discount this count by some amount. But how much should we discount it? Church and Gale's clever idea was to look at a held-out corpus and just see what the count is for all those bigrams that had count 4 in the training set. They computed a bigram grammar from 22 million words of AP newswire and then checked the counts of each of these bigrams in another 22 million words. On average, a bigram that occurred 4 times in the first 22 million words occurred 3.23 times in the next 22 million words. Fig. B.1 from Church and Gale (1991) shows these counts for bigrams with c from 0 to 9.

Bigram count in training set	Bigram count in heldout set
0	0.0000270
1	0.448
2	1.25
3	2.24
4	3.23
5	4.21
6	5.23
7	6.21
8	7.21
9	8.26

Figure B.1 For all bigrams in 22 million words of AP newswire of count 0, 1, 2,...,9, the counts of these bigrams in a held-out corpus also of 22 million words.

absolute discounting

Notice in Fig. B.1 that except for the held-out counts for 0 and 1, all the other bigram counts in the held-out set could be estimated pretty well by just subtracting 0.75 from the count in the training set! **Absolute discounting** formalizes this intuition by subtracting a fixed (absolute) discount d from each count. The intuition is that since we have good estimates already for the very high counts, a small discount d won't affect them much. It will mainly modify the smaller counts, for which we don't necessarily trust the estimate anyway, and Fig. B.1 suggests that in practice this discount is actually a good one for bigrams with counts 2 through 9. The

equation for interpolated absolute discounting applied to bigrams:

$$P_{\text{AbsoluteDiscounting}}(w_i|w_{i-1}) = \frac{C(w_{i-1}w_i) - d}{\sum_v C(w_{i-1}\,v)} + \lambda(w_{i-1})P(w_i) \tag{B.1}$$

The first term is the discounted bigram, with $0 \leq d \leq 1$, and the second term is the unigram with an interpolation weight λ. By inspection of Fig. B.1, it looks like just setting all the d values to .75 would work very well, or perhaps keeping a separate second discount value of 0.5 for the bigrams with counts of 1. There are principled methods for setting d. For example, Ney et al. (1994) set d as a function of n_1 and n_2, the number of unigrams that have a count of 1 and a count of 2, respectively:

$$d = \frac{n_1}{n_1 + 2n_2} \tag{B.2}$$

B.2 Kneser-Ney Discounting

Kneser-Ney discounting (Kneser and Ney, 1995) augments absolute discounting with a more sophisticated way to handle the lower-order unigram distribution. Consider the job of predicting the next word in this sentence, assuming we are interpolating a bigram and a unigram model.

I can't see without my reading _____ .

The word *glasses* seems much more likely to follow here than, say, the word *Kong*, so we'd like our unigram model to prefer *glasses*. But in fact it's *Kong* that is more common, since *Hong Kong* is a very frequent word. A standard unigram model will assign *Kong* a higher probability than *glasses*. We would like to capture the intuition that although *Kong* is frequent, it is mainly only frequent in the phrase *Hong Kong*, that is, after the word *Hong*. The word *glasses* has a much wider distribution.

In other words, instead of $P(w)$, which answers the question "How likely is w?", we'd like to create a unigram model that we might call $P_{\text{CONTINUATION}}$, which answers the question "How likely is w to appear as a novel continuation?". How can we estimate this probability of seeing the word w as a novel continuation, in a new unseen context? The Kneser-Ney intuition is to base our estimate of $P_{\text{CONTINUATION}}$ on the *number of different contexts word w has appeared in*, that is, the number of bigram types it completes. Every bigram type was a novel continuation the first time it was seen. We hypothesize that words that have appeared in more contexts in the past are more likely to appear in some new context as well. The number of times a word w appears as a novel continuation can be expressed as:

$$P_{\text{CONTINUATION}}(w) \propto |\{v : C(vw) > 0\}| \tag{B.3}$$

To turn this count into a probability, we normalize by the total number of word bigram types. In summary:

$$P_{\text{CONTINUATION}}(w) = \frac{|\{v : C(vw) > 0\}|}{|\{(u', w') : C(u'w') > 0\}|} \tag{B.4}$$

An equivalent formulation based on a different metaphor is to use the number of word types seen to precede w (Eq. B.3 repeated):

$$P_{\text{CONTINUATION}}(w) \propto |\{v : C(vw) > 0\}| \tag{B.5}$$

normalized by the number of words preceding all words, as follows:

$$P_{\text{CONTINUATION}}(w) = \frac{|\{v : C(vw) > 0\}|}{\sum_{w'} |\{v : C(vw') > 0\}|} \tag{B.6}$$

A frequent word (Kong) occurring in only one context (Hong) will have a low continuation probability.

Interpolated Kneser-Ney

The final equation for **Interpolated Kneser-Ney** smoothing for bigrams is then:

$$P_{\text{KN}}(w_i|w_{i-1}) = \frac{\max(C(w_{i-1}w_i) - d, 0)}{C(w_{i-1})} + \lambda(w_{i-1}) P_{\text{CONTINUATION}}(w_i) \tag{B.7}$$

The λ is a normalizing constant that is used to distribute the probability mass we've discounted:

$$\lambda(w_{i-1}) = \frac{d}{\sum_v C(w_{i-1}v)} |\{w : C(w_{i-1}w) > 0\}| \tag{B.8}$$

The first term, $\frac{d}{\sum_v C(w_{i-1}v)}$, is the normalized discount (the discount d, $0 \le d \le 1$, was introduced in the absolute discounting section above). The second term, $|\{w : C(w_{i-1}w) > 0\}|$, is the number of word types that can follow w_{i-1} or, equivalently, the number of word types that we discounted; in other words, the number of times we applied the normalized discount.

The general recursive formulation is as follows:

$$P_{\text{KN}}(w_i|w_{i-n+1:i-1}) = \frac{\max(c_{KN}(w_{i-n+1:i}) - d, 0)}{\sum_v c_{KN}(w_{i-n+1:i-1}\ v)} + \lambda(w_{i-n+1:i-1}) P_{KN}(w_i|w_{i-n+2:i-1}) \tag{B.9}$$

where the definition of the count c_{KN} depends on whether we are counting the highest-order n-gram being interpolated (for example trigram if we are interpolating trigram, bigram, and unigram) or one of the lower-order n-grams (bigram or unigram if we are interpolating trigram, bigram, and unigram):

$$c_{KN}(\cdot) = \begin{cases} \text{count}(\cdot) & \text{for the highest order} \\ \text{continuationcount}(\cdot) & \text{for lower orders} \end{cases} \tag{B.10}$$

The continuation count of a string $\cdot$ is the number of unique single word contexts for that string $\cdot$.

At the termination of the recursion, unigrams are interpolated with the uniform distribution, where the parameter ϵ is the empty string:

$$P_{\text{KN}}(w) = \frac{\max(c_{KN}(w) - d, 0)}{\sum_{w'} c_{KN}(w')} + \lambda(\epsilon)\frac{1}{V} \tag{B.11}$$

If we want to include an unknown word <UNK>, it's just included as a regular vocabulary entry with count zero, and hence its probability will be a lambda-weighted uniform distribution $\frac{\lambda(\epsilon)}{V}$.

modified Kneser-Ney

The best performing version of Kneser-Ney smoothing is called **modified Kneser-Ney** smoothing, and is due to Chen and Goodman (1998). Rather than use a single fixed discount d, modified Kneser-Ney uses three different discounts d_1, d_2, and d_{3+} for n-grams with counts of 1, 2 and three or more, respectively. See Chen and Goodman (1998, p. 19) or Heafield et al. (2013) for the details.

Chen, S. F. and J. Goodman. 1998. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Computer Science Group, Harvard University.

Church, K. W. and W. A. Gale. 1991. A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams. *Computer Speech and Language*, 5:19–54.

Heafield, K., I. Pouzyrevsky, J. H. Clark, and P. Koehn. 2013. Scalable modified Kneser-Ney language model estimation. *ACL*.

Kneser, R. and H. Ney. 1995. Improved backing-off for M-gram language modeling. *ICASSP*, volume 1.

Ney, H., U. Essen, and R. Kneser. 1994. On structuring probabilistic dependencies in stochastic language modelling. *Computer Speech and Language*, 8:1–38.