CS 124/LING 180/LING 280
From Languages to Information
Week 2: Group Exercises on Language Modeling
Winter 2019

Dan Jurafsky

Wednesday, January 23, 2019

Part 2: Group Exercise

We are interested in building a language model over a language with three words: A, B, C. Our training corpus is

`AAABACBABBBCCACBCC`

1. First train a unigram language model using maximum likelihood estimation. What are the probabilities?
   Reminder: We don't need start or end tokens for training a unigram model, since the context of each word doesn't matter. Thus, we won't add any special tokens to our corpus for now.

   Answer:

$$P(A) = \frac{6}{18}$$
$$P(B) = \frac{6}{18}$$
$$P(C) = \frac{6}{18}$$

2. Next train a bigram language model using maximum likelihood estimation. For this problem, we'll add both a start token, $< start >$, and an end token, $< end >$, at the end of the string. These allow us to model the probability of the sentence starting with a particular letter and ending after a particular letter. Fill in the probabilities below. Leave your answers in the form of a fraction.

   Answer:

   Our corpus now becomes: `<start>AAABACBABBBCCACBCC<end>`

$$P(A| < start >) = \frac{count(< start > A)}{count(< start >)} = \frac{1}{1}$$
$$P(A|A) = \frac{count(AA)}{count(A)} = \frac{2}{6}$$
$$P(A|B) = \frac{count(BA)}{count(B)} = \frac{2}{6}$$
$$P(A|C) = \frac{count(CA)}{count(C)} = \frac{1}{6}$$
$$P(A| < end >) = \frac{count(< end > A)}{count(< end >)} = \frac{0}{1}$$
$$P(B| < start >) = \frac{count(< start > B)}{count(< start >)} = \frac{0}{1}$$
$$P(B|A) = \frac{count(AB)}{count(A)} = \frac{2}{6}$$
$$P(B|B) = \frac{count(BB)}{count(B)} = \frac{2}{6}$$
$$P(B|C) = \frac{count(CB)}{count(C)} = \frac{2}{6}$$
$$P(B| < end >) = \frac{count(< end > B)}{count(< end >)} = \frac{0}{1}$$
$$P(C| < start >) = \frac{count(< start > C)}{count(< start >)} = \frac{0}{1}$$
$$P(C|A) = \frac{count(AC)}{count(A)} = \frac{2}{6}$$
$$P(C|B) = \frac{count(BC)}{count(B)} = \frac{2}{6}$$
$$P(C|C) = \frac{count(CC)}{count(C)} = \frac{2}{6}$$
$$P(C| < end >) = \frac{count(< end > C)}{count(< end >)} = \frac{0}{1}$$
$$P(< end > | < start >) = \frac{count(< start >< end >)}{count(< start >)} = \frac{0}{1}$$
$$P(< end > |A) = \frac{count(A < end >)}{count(A)} = \frac{0}{6}$$
$$P(< end > |B) = \frac{count(B < end >)}{count(B)} = \frac{0}{6}$$
$$P(< end > |C) = \frac{count(C < end >)}{count(C)} = \frac{1}{6}$$
$$P(< end > | < end >) = \frac{count(< end >< end >)}{count(< end >)} = \frac{0}{1}$$

3. Now evaluate your language models on the test corpus:

`ABACABB`

What is the perplexity of the unigram language model evaluated on this corpus? Since we didn't add any special start/end tokens when we were training our unigram language model, we won't add any when we evaluate the perplexity of the unigram language model, either, so that we're consistent.

Answer:

$$\sqrt[-7]{(\frac{1}{3})^7} = (\frac{1}{3})^{-1} = 3 \tag{1}$$

What is the perplexity of the bigram language model evaluated on this corpus? Since we added a start and end token when we were training our bigram model, we'll add them to this corpus again before we evaluate perplexity.

Answer: In this case, the corpus becomes: `<start>ABACABB<end>`

The perplexity is therefore:

$$\sqrt[-8]{P(A| < start >)P(B|A)P(A|B)P(C|A)P(A|C)P(B|A)P(B|B)P(< end > |B)} \tag{2}$$

Note that in our unsmoothed bigram model, $P(< end > |B)$ is 0, since $< end >$ never occurs after B in our training set. Therefore, the probability of this corpus is 0, so its perplexity is infinite.

4. Now repeat everything above for add-1 smoothing.

   Answer: Add-one smoothing is simple for the unigram model. We simply add 1 to each numerator, and 3 (the vocabulary size) to each denominator. For this particular corpus, the actual probabilities are the same as without smoothing; all unigram probabilities are equal to one-third. Note that this is not true for most corpuses, as smoothing usually does in fact change the probabilities in your language model.

$$P(A) = \frac{7}{21}$$
$$P(B) = \frac{7}{21}$$
$$P(C) = \frac{7}{21}$$

   Add-one smoothing is slightly more complicated for the bigram model. Since we use both an end-token $< end >$, our vocabulary size is 4 in this case, so we add 1 to each numerator and 4 to each denominator. Note that the ¡start¿ token is not considered part of our vocabulary as we don't expect it to every appear again after the first token in the corpus.

$$P(A| < start >) = \frac{count(< start > A) + 1}{count(< start >) + 4} = \frac{2}{5}$$
$$P(B| < start >) = \frac{count(< start > B) + 1}{count(< start >) + 4} = \frac{1}{5}$$
$$P(C| < start >) = \frac{count(< start > C) + 1}{count(< start >) + 4} = \frac{1}{5}$$
$$P(< end > | < start >) = \frac{count(< start >< end >) + 1}{count(< start >) + 4} = \frac{1}{5}$$
$$P(A|A) = \frac{count(AA) + 1}{count(A) + 4} = \frac{3}{10}$$
$$P(A|B) = \frac{count(BA) + 1}{count(B) + 4} = \frac{3}{10}$$
$$P(A|C) = \frac{count(CA) + 1}{count(C) + 4} = \frac{2}{10}$$
$$P(A| < end >) = \frac{count(< end > A) + 1}{count(< end >) + 4} = \frac{1}{5}$$
$$P(B|A) = \frac{count(AB) + 1}{count(A) + 4} = \frac{3}{10}$$
$$P(B|B) = \frac{count(BB) + 1}{count(B) + 4} = \frac{3}{10}$$
$$P(B|C) = \frac{count(CB) + 1}{count(C) + 4} = \frac{3}{10}$$
$$P(B| < end >) = \frac{count(< end > B) + 1}{count(< end >) + 4} = \frac{1}{5}$$
$$P(C|A) = \frac{count(AC) + 1}{count(A) + 4} = \frac{3}{10}$$
$$P(C|B) = \frac{count(BC) + 1}{count(B) + 4} = \frac{3}{10}$$
$$P(C|C) = \frac{count(CC) + 1}{count(C) + 4} = \frac{3}{10}$$
$$P(C| < end >) = \frac{count(< end > C) + 1}{count(< end >) + 4} = \frac{1}{5}$$
$$P(< end > |A) = \frac{count(A < end >) + 1}{count(A) + 4} = \frac{1}{10}$$
$$P(< end > |B) = \frac{count(B < end >) + 1}{count(B) + 4} = \frac{1}{10}$$
$$P(< end > |C) = \frac{count(C < end >) + 1}{count(C) + 4} = \frac{2}{10}$$
$$P(< end > | < end >) = \frac{count(< end >< end >) + 1}{count(< end >) + 4} = \frac{1}{5}$$

The unigram perplexity ends up being the same as before (since the

smoothed unigram probabilities are the same as the unsmoothed unigram probabilities for this specific corpus):

$$\sqrt[-7]{(\frac{1}{3})^7} = (\frac{1}{3})^{-1} = 3 \tag{3}$$

Bigram perplexity is a little bit different with smoothing. We use the same formula as before, but this time our probability is no longer 0, so we can compute a finite perplexity:

$$\sqrt[-8]{P(A|<start>)P(B|A)P(A|B)P(C|A)P(A|C)P(B|A)P(B|B)P(<end>|B)}$$
$$= \sqrt[-8]{\frac{2}{5}(\frac{3}{10})^5\frac{2}{10}\frac{1}{10}}$$
$$\approx 3.88$$

5. What is the difference between using an UNK token (for unknown words) and smoothing? In what situations would you use one versus the other?

   Answer: UNK tokens are used when unknown words are observed in the dataset. This can happen in both train and test datasets - for some language model applications, it is desirable to have a fixed vocabulary with the UNK token representing the presence of other words during training and testing. If that is the case, at training and test time, any token not belonging to the fixed vocabulary would be converted into an UNK token before counting even starts.

   Smoothing, on the other hand, is a way to redistribute the probabilities of observed occurrences from the training dataset. For almost all smoothing methods, the events (unigrams, bigrams ... etc) with higher probability would be discounted and rare events would be boosted. This is a good remedy for small datasets, where the probability distribution between unigrams and bigrams may be skewed due to the small size. However, in some cases this also causes the higher probability events to be discounted too much. Smoothing could also potentially be used to deal with unknown words - essentially treating unknown words as tokens that appear 0 times in the training corpus with their probabilities estimated from the smoothing addition.