**Assignment #3 N-Gram Language Model**

yuweic3

2017/9/17

**#1 N-Gram Language Model**

1. Markov assumption: the probability of seeing any give word is contingent on the last N-1 words.
2. It is reasonable as the probability of each word can be estimated based on its prior context. In fact, the words we speak and write at any time are largely dependent on the precious 3-4 words.
3. Since we cannot keep track of all possible histories of all words, by applying Markov Assumption we can limit history to fixed number of words N.

**#2 Unigram Language Model**

1. Since words in the constitute the entire vocabulary, we need at least 7 parameters to specify this Unigram Language model.

“count()” stands for total # words in corpus.

Such 8 parameters:

, ,, , ,

, ,

However, we only need any 7 parameters out of these 8 parameters for necessity as the sum of all probabilities is 1.

1. We need 12 parameters for specification.

, ,, , ,

, ,,

, , ,

Similarly, we only need any 11 parameters out of these 12 parameters for necessity as the sum of all probabilities is 1.

**#3 Bigram Language Model**

1. After we add to particularly denote the start of the sentence, there are 13 denotations in our vocabulary list. We need necessary parameters for specification (actually, it is a table).

, ,,

, ,

, ,

1. [1] Sparse probability matrix. As we do not know bigram of “”, . Similar to , , and

[2] Yes. Even we add four new parameters, their probabilities are still 0.

[3] Bigram Language model is more severe. The more complex the -gram model is (the larger the is), the severer the problem of-gram training matrix sparsity is.

[4] We can smooth it by estimating the likelihood of unseen -grams

**#4 Smoothing**

1. [1] In unigram language model, for a fixed , when , we can ignore the effect of and then have and , then

[2] In unigram language model, for a fixed , when , we have and , then . This implies that each word is equally likely in the background language model.

1. Jelinek-Mercer (interpolation) smoothing adjusts probabilities for all seen and unseen events, while Katz (back-off) smoothing only adjusts for unseen events.

On the other hand, Jelinek-Mercer performs better for low(nonzero) counts than Katz, because interpolated models use information from lower-order models while Katz-backoff models do not.

**#5 Application of Smoothing**

1. According to the Dirchlet Prior Smoothing,

then we have

,

,

,

,

,

,

,

,

,

,

,

.

Before smoothing, the probability of some events is 0. After smoothing, every event has its probability.

1. When ,

,

,

,

,

,

,

,

,

, ,

,

When , we have

,

,

,

,

,

,

,

,

, ,

,

Yes, it matches my intuition. When is relatively too small for (which means is large enough), a unigram language model smoothed with a Dirichlet prior becomes equivalent to its maximum likelihood estimate. On the other hand, when is large enough, a unigram language model smoothed with a Dirichlet prior will get close to its background language model.

1. For Jelinek-Mercer smoothing, we have

when , we have :

,

,

,

,

,

,

,

, ,

,

when

,

,

,

,

,

,

,

, ,

,

when

,

,

,

,

,

,

,

,

, ,

,

Similarities: We can find that the larger is, the more smoothing we have, putting more weight on the background probabilities. This is similar to the Dirichlet prior as gets larger, it gets more equivalent to its background language model.

Difference: In Dirichlet prior smoothing, is dynamic and will actually depend on the current document being scored because is used in the smoothed probability. However, in the Jelinek-Mercer smoothing, is a constant. The Dirichlet prior smoothing is able to capture the document length normalization than Jelinek-Mercder smoothing.