CSE 143 Writeup 2
Aaron Steinberg, Andrew Moshkovich, James Gu

1 Programming: n-gram language modeling

Our implementation starts with tokenizing the data provided. The user can specify the type of n gram by putting in the 1 for unigram, 2 for bigram, and 3 for trigram. If the n is more than 1 the dictionary will hold multiple words for the key. For unigram it will only hold one word. After every word in the data is tokenized, the words with less than 3 instances will be classified under the <UNK> token instead. Our vocabulary should consist of 26602 unique tokens for unigram, 39578 unique tokens for bigram, and 10491 unique tokens for trigram using the training data. And from these dictionaries we can calculate the perplexity with the following formula:

Perplexity = 2^{-l}

l:

$$l = \frac{1}{M}\sum_{i=1}^{N} log(p(x_i))$$

Implementing this formula with MLE should make our perplexities for the unigram, bigram, and trigram be 658, 63.7, 39.5 respectively. These numbers are without smoothing which we will cover in the next section.

2 Programming: additive smoothing

Our implementation uses 1 as our alpha value and M as the value of our vocabulary. For additive smoothing, we added 1/M so the formula was similar to that of Naive Bayes:

$$P(w_i|w_{i-1}, w_{i-2}) = \frac{count(w_i, w_{i-1}, w_{i-2})+\alpha}{count(w_{i-1}, w_{i-2})+\alpha*|V|}$$

3 Programming: smoothing with linear interpolation

The formula used for our interpolation model is as follows:

$$\theta'_{x_j|x_{j-2},x_{j-1}} = \lambda_1\theta_{x_j} + \lambda_2\theta_{x_j|x_{j-1}} + \lambda_3\theta_{x_j|x_{j-1},x_{j-2}}$$