**Hrishav Aryal**

1. **How many word types (unique words) are there in the training corpus? Please include the padding symbols and the unknown token.**

* Total Word Types in training corpus: **41740**

1. **How many word tokens are there in the training corpus?**

* Total Word Tokens in training corpus: **2668210**

1. **What percentage of word tokens and word types in the test corpus did not occur in training (before you mapped the unknown words to <unk> in training and test data? Please include the padding symbols in your calculations.**

* Percentage of word tokens in the test corpus which did not occur in training

= **1.6382014639247124 %**

* Percentage of word types in the test corpus which did not occur in training

= **3.682946357085669 %**

1. **Now replace singletons in the training data with <unk> symbol and map words (in the test corpus) not observed in training to <unk>. What percentage of bigrams (bigram types and bigram tokens) in the test corpus did not occur in training (treat <unk> as a regular token that has been observed)**

* Percentage of bigrams types in the test corpus which did not occur in training

= **24.498977505112475 %**

* Percentage of bigrams tokens in the test corpus which did not occur in training

= **23.70399683419074 %**

1. **Compute the log probability of the following sentence under the three models (ignore capitalization and pad each sentence as described above). Please list all of the parameters required to compute the probabilities and show the complete calculation. Which of the parameters have zero values under each model? Use log base 2 in your calculations. Map words not observed in the training corpus to the <unk> token.**

* I look forward to hearing your reply .

Note:

* log used below is base 2.
* p(term) = probability of the word ‘term ’in training corpus
* c(term) = total count the word ‘term’ in training corpus
* N = total word count in training corpus
* V = total word types (unique tokens) in training corpus
* **Unigram Model:**
  + p (<s> i look forward to hearing your reply . </s>)

=

=

= log ( 7339/41740 ) + log ( 613/41740 ) + log ( 474/41740 ) + log ( 53048/41740 ) + log ( 209/41740 ) + log ( 1217/41740 ) + log ( 13/41740 ) + log ( 87894/41740 ) + log ( 100000/41740 ) +

= -8.506073004952297 + -12.087697523143856 + -12.458697537936377 + -5.652430047181399 + -13.640081654753754 + -11.098327334120908 + -17.647001068693545 + -4.923963722754861 + -4.737800312397825

**= -90.75207220593482**

**Parameters with 0 values: none**

* **Bigram Model:**
  + p (<s> I look forward to hearing your reply . </s>)

=

=

= log ( 2006/100000 ) + log ( 15/7339 ) + log ( 34/613 ) + log ( 100/474 )

+ log ( 6/53048 ) + 0 + 0 + 0 + log ( 82888/87894 )

= -5.639534583824631 + -8.93447718627382 + -4.172280422440442 +

-2.2448870591235344 + -13.110048238932082 + 0 + 0 + 0 +

-0.08460143194821208

= **-34.185828922542726**

**Parameters with 0 values:**

* **Bigram Model with Add-One Smoothing:**
  + p (<s> I look forward to hearing your reply . </s>)

=

=

= log ( 2007/141740 ) + log ( 16/49079 ) + log ( 35/42353 ) + log ( 101/42214 )

+ log ( 7/94788 ) + log ( 1/41949 ) + log ( 1/42957 ) + log ( 1/41753 )

+ log ( 82889/129634 )

= -6.142062527223568 + -11.582818233486321 + -10.240893526538176 +

-8.707222435563736 + -13.725061885430772 + -15.356348798981255

-15.390605622494517 + -15.349592240121291 + -0.6451915904499356

= -97.1397968602896

**Parameters with 0 values: none**

1. **Compute the perplexity of the sentence above under each of the models.**

* Perplexity of the sentence above:
  + Unigram Model – **539.3983209831056**
  + Bigram Model– **10.692912013032808**
  + Bigram Model with Add one smoothing– **839.8452354810934**

1. **Compute the perplexity of the entire test corpus under each of the models. Discuss the differences in the results you obtained.**

* Perplexity of the entire test corpus:
  + Unigram Model – **820.7875246008351**
  + Bigram Model– **24.82407302566233**
  + Bigram Model with Add one smoothing– **1431.647474821038**

Perplexity is a measurement of how well a probability model predicts a test data. In the context of Natural Language Processing, perplexity is one way to evaluate language models. Regarding this project, perplexity of Bigram Model is the least among three. This means bigram model is the suitable model for the purpose of this project. Unigram model has a higher perplexity value which is not surprising as Unigram Models perform poorly compared to bigram models. Perplexity of bigram model with add one smoothing is the worst of all. Since we add one to the count of bigrams even its 0, too much probability mass is assigned to unseen events. Hence, this is a poor method of smoothing, Therefore, the perplexity is high, which signifies it’s a poor model to use.