4.2 n-gram Language Model

Model Information

	Unigram Model	Bigram Model	Trigram Model
Corpus Length	1622905	1684434	1684433
Unique number of n-grams	26602	510392	1116160

Table 1 – Model information

Model Evaluation – Perplexity Scores

	Unigram Model	Bigram Model	Trigram Model
Train Perplexity	1105.5403	77.0735	7.8729
Validation Perplexity	1009.9143	Infinity	Infinity
Test Perplexity	1015.3158	Infinity	Infinity

Table 2 – Unsmoothed model perplexity scores

The above-mentioned perplexity value makes sense here and follows the general intuition that the larger the history the better the model will perform. In the unigram model, since the UNK token was used there the probability was always non-zero allowing for a finite perplexity value to be found.

In the validation and test datasets for the bigram and trigram models, the perplexity was found to be infinite because the probability of certain bigram/trigrams was found to be 0. Since there was no smoothing added in this example, it is valid for the perplexity to be infinite.

To further validate my test results, I added Lidstone Smoothing (with alpha = 0.5) to my model and obtained the following results:

	Unigram Model	Bigram Model	Trigram Model
Train Perplexity	1106.1765	978.8943	4020.5838
Validation Perplexity	1011.2453	1251.1889	8019.1787
Test Perplexity	1016.5973	1247.5270	7991.0045

Table 3 – Lidstone smoothed model perplexity scores

The results were a little surprising here. For each type of model, the training dataset generally has a lower perplexity value compared to the validation and test sets. However, one surprising element is that the perplexity tends to increase a lot as we move from unigram to the trigram model. Generally, one expects a trigram model to operate better and thus the perplexity to decrease.

One potential explanation for this could be the bias-variance tradeoff that smoothing introduces. Generally, adding smoothing will increase the perplexity of the training dataset (as it did), however it should reduce the perplexity of the validation and test datasets.

4.3 – Smoothing

Model Evaluation – Perplexity Scores

	Train Perplexity	Validation Perplexity	Test Perplexity
Lambda1 = 0.01 **Lambda 2 = 0.05** **Lambda 3 = 0.94**	8.1494	2360.2248	2354.1756
Lambda1 = 0.07 **Lambda 2 = 0.08** **Lambda 3 =0.85**	8.8966	885.6395	884.7960
Lambda1 = 0.06 **Lambda 2 = 0.14** **Lambda 3 = 0.70**	10.6769	978.8043	978.3856
Lambda1 = 0.20 **Lambda 2 = 0.20** **Lambda 3 =0.60**	12.1013	540.2321	540.6785
Lambda1 = 0.20 **Lambda 2 = 0.30** **Lambda 3 =0.50**	14.2190	554.8653	555.6197
Lambda1 = 0.33 **Lambda 2 = 0.33** **Lambda 3 =0.33**	20.6060	491.8504	493.0516

Table 4 – Interpolation smoothing model perplexity scores

The table below shows the perplexity for the specified lambda values, in the assignment

	Train Perplexity	Validation Perplexity	Test Perplexity
Lambda1 = 0.1 **Lambda 2 = 0.3** **Lambda 3 = 0.6**	12.0533	736.4886	736.9305

Table 5 - – Interpolation smoothing model perplexity scores for specific lambda values

If you use half of the training data, would it increase or decrease the perplexity on previously unseen data? Why? Provide empirical experimental evidence if necessary.

Generally, I would assume that reducing the training data would increase the perplexity (make the model worse off) since there is less information and tokens that the model can utilize. After training my model only on ½ the training data I found the following scores using interpolation smoothing:

	Train Perplexity	Validation Perplexity	Test Perplexity
Lambda1 = 0.1 **Lambda 2 = 0.3** **Lambda 3 = 0.6**	Not Calculated	468.3740	471.4108

Table 6 – Experimental perplexity scores after using ½ the training set

Comparing this to the results found from table 5, the perplexity has decreased (the model is performing better). One reason for the explanation could be now there are a lot more UNK tokens. As a result, the model is more general.

If you convert all tokens that appeared less than 5 times to (a special symbol for out-of-vocabulary tokens), would it increase or decrease the perplexity on the previously unseen data compared to an approach that converts only a fraction of words that appeared just once to ? Why? Provide empirical experimental evidence if necessary.

If there are more UNK tokens in the data this would make the model more general, thus reducing the perplexity of the model. After changing the 'UNK Threshold' from 3 to 5, I obtained the following results for the unigram model:

	Unigram Model
Train Perplexity	909.6217
Validation Perplexity	853.7755
Test Perplexity	856.9755

Table 7 - Experimental perplexity scores for UNK Threshold of 5

Compared to Table 2, the perplexity scores after changing the 'UNK Threshold' to 5 it can be observed that the perplexity does go down.