# Report for Assignment 1: “Bigram/ Trigram language model for word sequences”

Baltzi Sofia (P3351915), Kafritsas Nikos(p3351905), Mouselinos Spyridon (P3351914)

May 1, 2020

## Introduction

The purpose of this document is to report the results for the 1st Assignment of class Text Analytics, April – June 2020. Code for the assignment can be found by following the link: https://colab.research.google.com/drive/1OVj90zCVb6iJPzqrbKgW6Zkkewy4KQfG?usp=sharing

## Data

The corpus used in the assignment is “The parent's assistant” by author Maria Edgeworth and was downloaded from NLTK Python Package. A splitting schema of training (60%), development (20%) and test set (20%) was applied. Vocabulary as well as OOV words (occurrence limit 5) were chosen from the train portion only.

## Corpus Preprocessing

Since the corpus came from a novel with characters using abbreviations and slangs, a dictionary of the most common contractions as well as their respective switch term was created.

|  |  |
| --- | --- |
| Contractions’ Dictionary | |
| Term | **Switch Term** |
| n’t | not |
| ’t | not |
| ’ve | have |
| ’ll | will |
| ’s | is |
| ’d | would |
| ’m | am |
| ’em | them |

For the preprocessing of the corpus three methods were implemented and used:

1. *special\_char\_remove(corpus: str) -> str*

Gets a string as input and gives a string consisting of letters and the characters “.,?,!” as output.

1. *sentence\_tokenization(corpus:str) -> list*

Gets a large corpus-string as input and gives a list of sentences as output. The nltk.sent\_tokenize() method was used internally for the word tokenization.

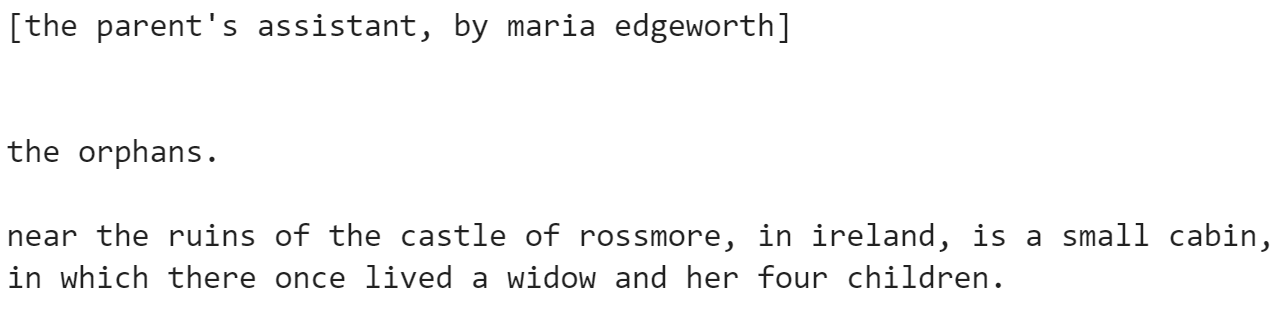
1. *word\_tokenization(text:str) -> list*

Gets a string as input and gives a list of words as output. The nltk.word\_tokenize() method was used internally for the word tokenization. The method also removes specific contractions by utilizing the aforementioned dictionary.

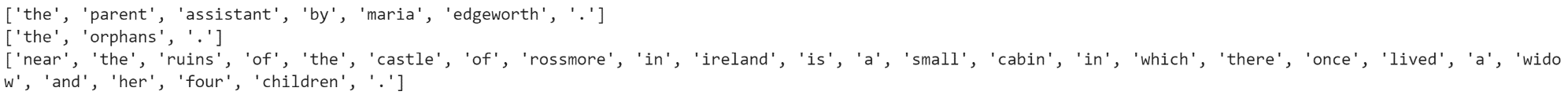
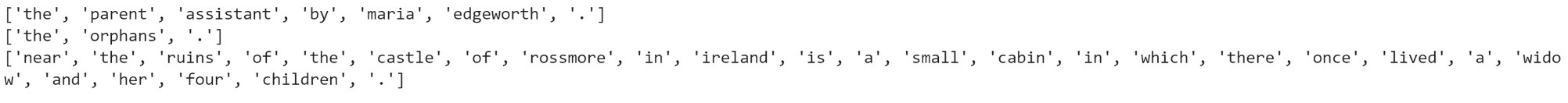
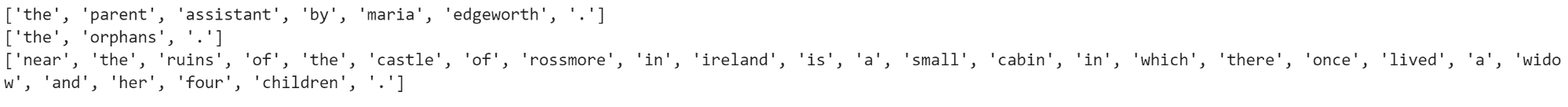
Method is applied repeatedly on lists created by sentence\_tokenization.

After converting the corpus to lowercase, it was processed by the three methods in the order given above, leaving a list of 10,168 list/sentences and a total of 898,088 words (characters “.,?,!” included).

*Figure 1: Part of corpus before preprocessing*



*Figure 2: Same part as "Figure 1" after preprocessing*



## Language Models Creation

In this part a Bigram and a Trigram Language Model were created using only the train split, where OOV words were substituted with token \*UNK\*. For the tuning of Laplace Smoothing alpha parameter the development split was utilized, with both unseen and OOV words replaced with \*UNK\*. Unseen and OOV words were replaced with \*UNK\* in the test split as well.

In total 3 Language Models were implemented: A Unigram, a Bigram and a Trigram model. All of them utilized :

* The NLTK’s ngrams function that could insert start (<s>) and end (<e>) tokens to each sentence.
* The built-in collection’s Counter class capable of creating a frequency mapping of uni/bi/tri - grams in the train corpus.

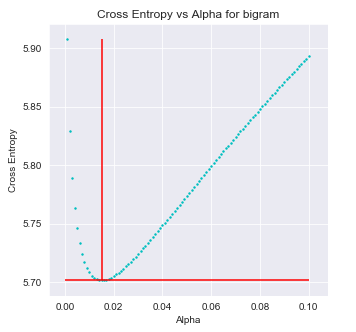
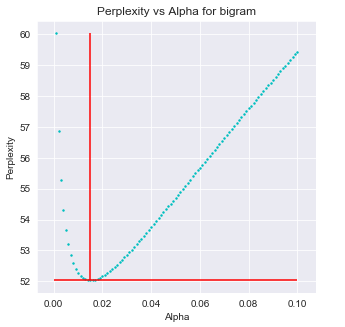
The top 5 most frequent per category are presented below:

|  |  |  |
| --- | --- | --- |
| **Unigrams** | **Bigrams** | **Trigrams** |
|  |  |  |

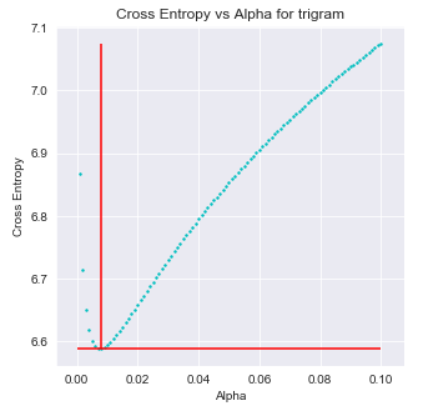
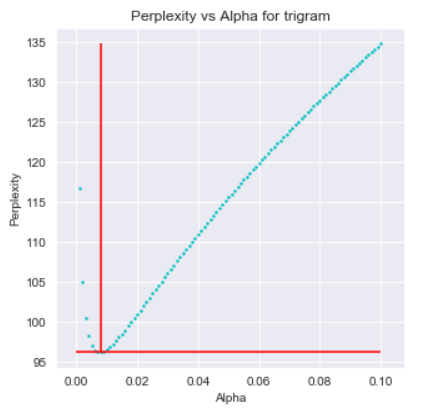
That made all the probability calculations under Markov assumptions possible for the next step.

## Parameter Tuning

Tuning was required for the choice of alpha, since the ideal value would provide the minimum Cross Entropy / Perplexity in the dev split. The different values of alpha are plotted against the corresponding values of Perplexity and Cross Entropy, in the diagrams below.



The lowest Perplexity/Entropy regarding Bigram model is: 51.991/5.7 at alpha=0.015.



The lowest Perplexity/Entropy regarding Trigram model is: 96.134 / 6.587 at alpha=0.008

All the following steps assume that the bi/tri-gram alpha values are set to those optimums.

## Model Correctness

This part tested the validity of the implemented language model by observing how sentences from the test set scored against ill-formed (fake) sentences. Both Bigram and Trigram language models were used.

Specifically, 5 fake sentences were used, where each one had the same length (in words) with a sample of 5 sentences from the test corpus. However, the fake sentences consisted of random words from the vocabulary. Log probabilities of each sentence were used as a means of comparison.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Correct – Fake Sentences Log Probability Comparison for Bigram model | | | |
| Correct Sentence | | **Probability** | **Fake Sentence** | **Probability** | |
| ['dr.', 'middleton', 'UNK', 'me', 'and', 'now', 'he', 'will', 'UNK', 'of', 'his', 'confidence', 'in', 'me', '.'] | | -75,58 | ['corner', 'understand', 'market', 'twelve', 'english', 'dress', 'men', 'downs', 'last', 'how', 'worked', 'told', 'gathered', 'running', 'fond'] | -182,25 | |
| ['repeated', 'he', 'UNK', 'the', 'light', 'to', 'his', 'face', '.'] | | -51,12 | ['arthur', 'expected', 'sophy', 'where', 'halfpenny', 'has', 'reasonable', 'safe', 'opened'] | -100,88 | |
| ['grant', 'had', 'some', 'UNK', 'fine', 'raspberries', '.'] | | -43,64 | ['early', 'saw', 'returned', 'landlady', 'bid', 'dark', 'charming'] | -90,84 | |
| ['and', 'louisa', 'also', 'desired', 'me', 'said', 'she', 'to', 'return', 'to', 'you', 'this', 'flora', '.'] | | -104,34 | ['command', 'beg', 'mischief', 'think', 'nonsense', 'smiling', 'fixed', 'dark', 'master', 'arms', 'caught', 'throw', 'either', 'public'] | -167,56 | |
| ['you', 'may', 'go', 'back', 'and', 'tell', 'her', 'so', '.'] | | -46,32 | ['safe', 'flora', 'been', "'", 'friends', 'many', 'then', 'recollect', 'cry'] | -113,20 | |

|  |  |  |  |
| --- | --- | --- | --- |
| Correct – Fake Sentences Log Probability Comparison for Trigram model | | | |
| Correct Sentence | **Probability** | **Fake Sentence** | **Probability** |
| ['dr.', 'middleton', 'UNK', 'me', 'and', 'now', 'he', 'will', 'UNK', 'of', 'his', 'confidence', 'in', 'me', '.'] | -108,52 | ['corner', 'understand', 'market', 'twelve', 'english', 'dress', 'men', 'downs', 'last', 'how', 'worked', 'told', 'gathered', 'running', 'fond'] | -164,64 |
| ['repeated', 'he', 'UNK', 'the', 'light', 'to', 'his', 'face', '.'] | -65,27 | ['arthur', 'expected', 'sophy', 'where', 'halfpenny', 'has', 'reasonable', 'safe', 'opened'] | -101,91 |
| ['grant', 'had', 'some', 'UNK', 'fine', 'raspberries', '.'] | -61,81 | ['early', 'saw', 'returned', 'landlady', 'bid', 'dark', 'charming'] | -90,87 |
| ['and', 'louisa', 'also', 'desired', 'me', 'said', 'she', 'to', 'return', 'to', 'you', 'this', 'flora', '.'] | -119,26 | ['command', 'beg', 'mischief', 'think', 'nonsense', 'smiling', 'fixed', 'dark', 'master', 'arms', 'caught', 'throw', 'either', 'public'] | -161,55 |
| ['you', 'may', 'go', 'back', 'and', 'tell', 'her', 'so', '.'] | -48,75 | ['safe', 'flora', 'been', "'", 'friends', 'many', 'then', 'recollect', 'cry'] | -104,24 |

The total log probabilities of fake sentences in both Bigram and Trigram models are lower compared to the sample of correct sentences from the test corpus, which was expected. Moreover, the above tables show that the Bigram model performs better than the Trigram one, because the total log probabilities in general are higher.

## Language Cross Entropy and Perplexity

In this section, the Cross Entropy and Perplexity of both models are computed based on the test split. Test subset is treated as a single sequence, having start and end tokens at the beginning and end of each sentence.

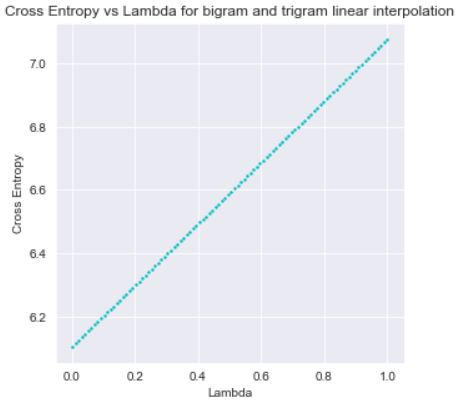
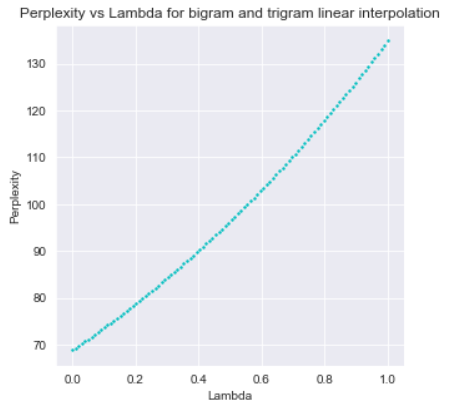
|  |  |  |  |
| --- | --- | --- | --- |
| Language Model Evaluation | | | |
|  | **Cross Entropy** | **Perplexity** |
| Bigram | 5.654 | 50.349 |
| Trigram | 6.547 | 93.538 |

Results imply that Bigram model performs better than Trigram. That was to be expected in the case of language modeling, since Trigrams estimate probabilities for longer sequences than Bigrams, which are intrinsically rarer to find in a corpus.

## Linear Interpolation

For the final question, the Bigram and Trigram models were combined using Linear Interpolation. The model used was . It was evaluated by tuning its hyperparameter λ between 0.001 and 1. For values of the hyperparameter close to 0 the Bigram model is favored, while for values close to 1 the Trigram model is favored.

For the probability calculations of the two models, the bigram\_prob() and the trigram\_prob() methods were used. Tuning of λ was performed based on the Development Set. Finally, the different values of λ were plotted against the corresponding values of Perplexity and Cross Entropy, giving the diagrams below.



As it can be observed, as λ gets higher values the Perplexity and Cross Entropy get higher values too, meaning that the best interpolated model is the one that has a zero λ value. This is to be expected, since it was found that Bigram model performs better (in terms of Perplexity and Cross Entropy) than Trigram in the scope of language modeling. By eliminating the latter form the interpolation, the model becomes the same as the Bigram model and thus, better.