CSE556 - Natural Language Processing
Assignment 1 Report

Charvi Jindal (2020045), Chetan (2020046)

Bigram Language Model

1. Incorporating smoothing algorithm
   In our bigram language model, the Add-1 or Laplace smoothing algorithm is used, which is based on the following formula-

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

   Here V stands for the size of our vocabulary. This equation is used to find probabilities of every possible bigram in the model.

Desired Outcomes

Part A

b) The top 4 bigrams are:

```
Top 4 bigrams and their scores:
Bigram = ('cant', 'wait') | Score after smoothing= 0.005936227951153324
Bigram = ('good', 'morning') | Score after smoothing= 0.007133377571333776
Bigram = ('last', 'night') | Score after smoothing= 0.005311857436600411
Bigram = ('look', 'like') | Score after smoothing= 0.004793699708953946
```

c) Accuracy of test set using dataset A for training:

```
0.8835403726708074
```

Part B

a) <u>Including the sentiment component</u>
   We have incorporated the sentiment component in two ways on unigram level. We did so because when we applied both of the following models individually we found that the accuracy was not increased.

1. Using vader to calculate the sentiment of each word in the newly generated sentence
2. Using the ratio of occurrence of each of the words in positive and negative classes of the given dataset
3. Additionally, we have taken log of the probabilities of each of the bigram pairs and the first unigram occurring in the sentence to avoid underflow

The following conditional statement is used to ensure that the generated sentence is sentiment oriented

```
if (abs(bigram_score) > 1 and (ratio_score>1 or ratio_score <-0.1) and sentiment_score!=0 ):
```

We have used the first condition to make more probable sentence using our bigram model.

We have used the second condition to ensure that the accuracy in the trained model for dataset B is improved as the ML model used is based on Naive Bayes model.

Lastly, we have used the third condition to get better results when we run vader analysis on the generated sentences.

c) Average perplexity of the 500 generated sentences:

```
5183.7210848683135
```

d) Generated samples

```
Positives:
['funky knob nyanza haircut disappear reef journey enjoyer', 'goodness
tooth nye gnarly graphic bum koklass auction', 'guru entertainer
manage mistake bunch kitty matta pout', 'vitamin surprise drag juke
heaven moss mell abaze', 'harder beng ghoster waver inlayer thus sadly
enjoy']
Negatives:
['wolverine wese miserable mocha isogonal sumpit alma trade', 'lues
busy single ghat atma deemer ugh development', 'battle graben uily
puff otherwise trip hood imagine', 'waist must woman rapper amount
reality ridiculously democrat', 'restless train dentist luke howso
octoon city ladykin']
```

e) Accuracy of test set using database B for training:

```
0.8913043478260869
```