Introduction to NLP

Assignment - 1

Submitted by - Ishit Bansal (2021101083)

Report

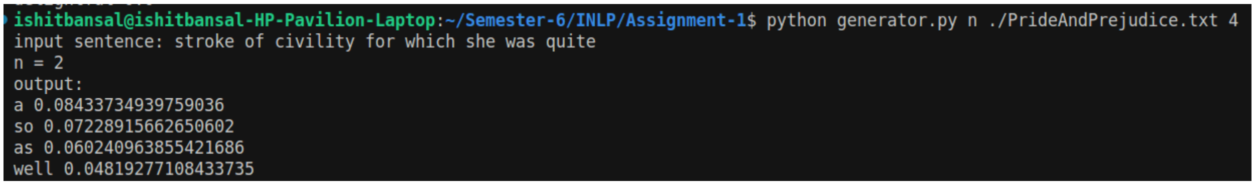
- **Reported the average perplexity and perplexity scores for all sentences in training and test set for each of the corpora in their corresponding text files**

- **Using the generated N-gram models (without the smoothing techniques), generating sequences and experimenting with different values of N.**

Example:-

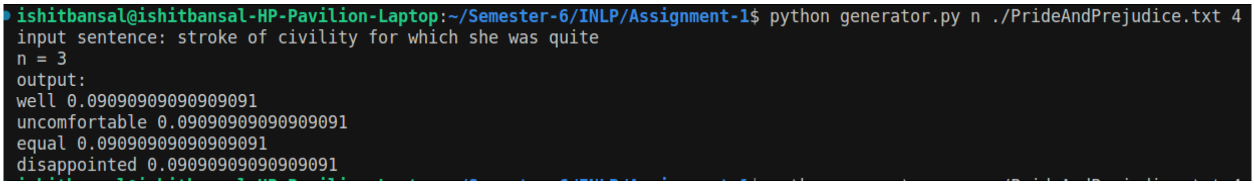
Input sentence: stroke of civility for which she was quite

N=2:



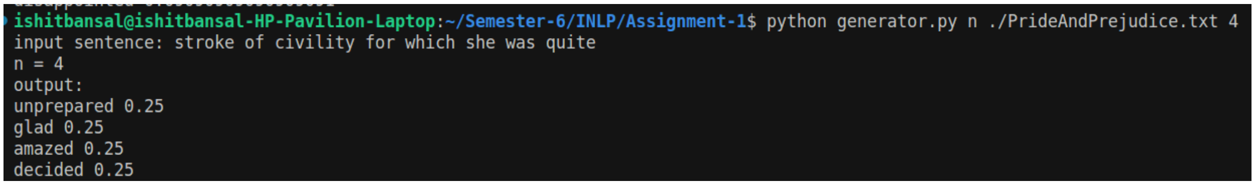
```
ishitbansal@ishitbansal-HP-Pavilion-Laptop:~/Semester-6/INLP/Assignment-1$ python generator.py n ./PrideAndPrejudice.txt 4
input sentence: stroke of civility for which she was quite
n = 2
output:
a 0.08433734939759036
so 0.07228915662650602
as 0.060240963855421686
well 0.04819277108433735
```

N=3:



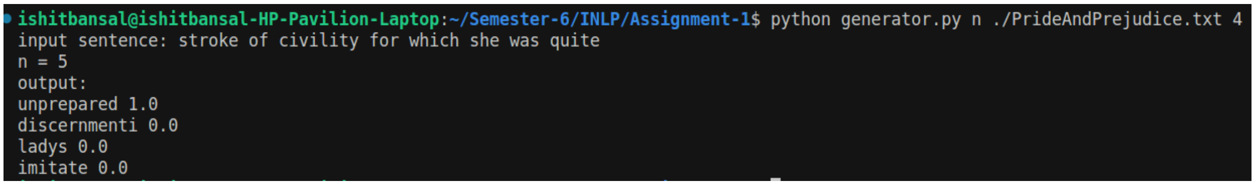
```
ishitbansal@ishitbansal-HP-Pavilion-Laptop:~/Semester-6/INLP/Assignment-1$ python generator.py n ./PrideAndPrejudice.txt 4
input sentence: stroke of civility for which she was quite
n = 3
output:
well 0.09090909090909091
uncomfortable 0.09090909090909091
equal 0.09090909090909091
disappointed 0.09090909090909091
```

N=4:



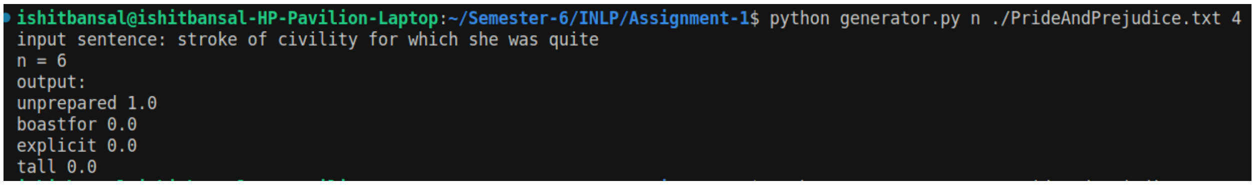
```
ishitbansal@ishitbansal-HP-Pavilion-Laptop:~/Semester-6/INLP/Assignment-1$ python generator.py n ./PrideAndPrejudice.txt 4
input sentence: stroke of civility for which she was quite
n = 4
output:
unprepared 0.25
glad 0.25
amazed 0.25
decided 0.25
```

N=5:



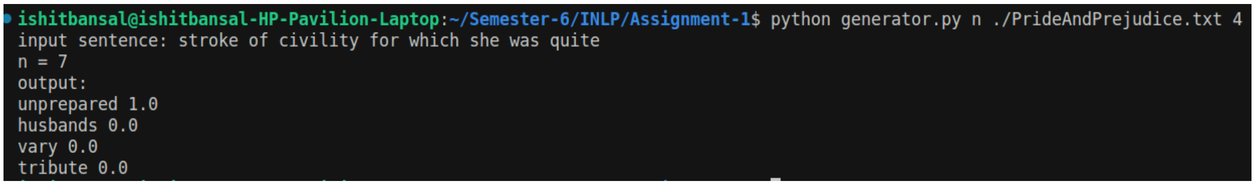
```
ishitbansal@ishitbansal-HP-Pavilion-Laptop:~/Semester-6/INLP/Assignment-1$ python generator.py n ./PrideAndPrejudice.txt 4
input sentence: stroke of civility for which she was quite
n = 5
output:
unprepared 1.0
discernmenti 0.0
ladys 0.0
imitate 0.0
```

N=6:



```
ishitbansal@ishitbansal-HP-Pavilion-Laptop:~/Semester-6/INLP/Assignment-1$ python generator.py n ./PrideAndPrejudice.txt 4
input sentence: stroke of civility for which she was quite
n = 6
output:
unprepared 1.0
boastfor 0.0
explicit 0.0
tall 0.0
```

N=7:



```
ishitbansal@ishitbansal-HP-Pavilion-Laptop:~/Semester-6/INLP/Assignment-1$ python generator.py n ./PrideAndPrejudice.txt 4
input sentence: stroke of civility for which she was quite
n = 7
output:
unprepared 1.0
husbands 0.0
vary 0.0
tribute 0.0
```

We observe that for smaller values of N, we get phrases like ‘quite a’ or ‘quite so’ or ‘was quite well’ which are most probable. This however does not relate to the overall context, as the language model captures short dependencies only.

As we increase N, we get more relevant phrases to the text as the language model captures long range contexts and thus improves fluency.

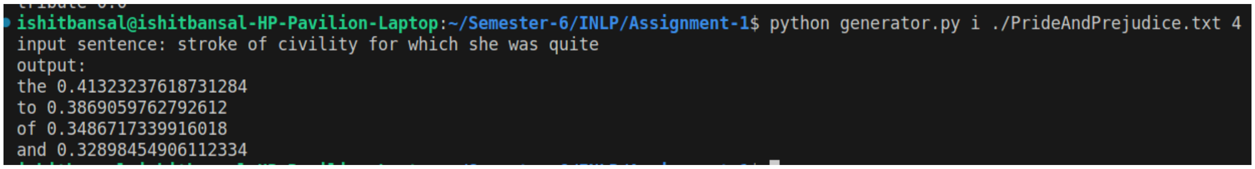
In general, as N increases fluency with respect to context increases.

- **Generating text using models with smoothing techniques (linear interpolation)**

  Example:-

  Input sentence: stroke of civility for which she was quite

  N=3:



```
ishitbansal@ishitbansal-HP-Pavilion-Laptop:~/Semester-6/INLP/Assignment-1$ python generator.py i ./PrideAndPrejudice.txt 4
input sentence: stroke of civility for which she was quite
output:
the 0.41323237618731284
to 0.3869059762792612
of 0.3486717339916018
and 0.32898454906112334
```

  We observe that, stop words like ‘the’, ‘and’, ‘to’ and ‘of’ are most probable to be the next generated word despite having no relevance. This is because the weight obtained for unigram is relatively high and the frequency of stop words is also high.

- **Attempting to generate a sentence using an Out-of-Data (OOD) scenario with your N-gram models.**

While generating an Out-of-Data (OOD) scenario, I observed that for small values of N, the next most probable words were stop words like ‘the’ and ‘of’. For intermediate values of N, I got words which were out of context. For higher values of N, I got 0 probability for all words.