INTRODUCTION TO NLP (CS7.401, SPRING 2025)

ASSIGNMENT 1

Mayank Mittal (2022101094)

International Institute of Information Technology, Hyderabad

January 2025

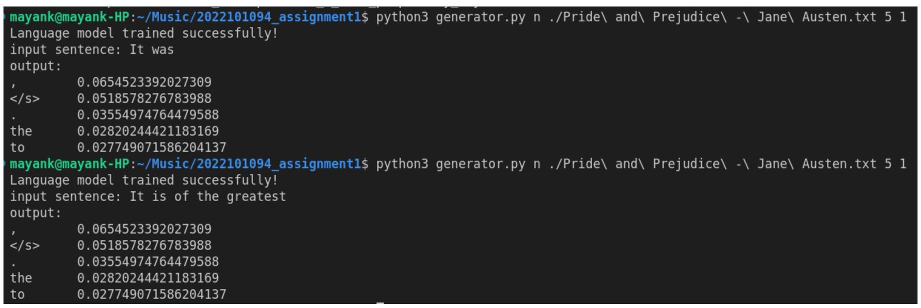
Contents

1 Generation

1.1 No Smoothing

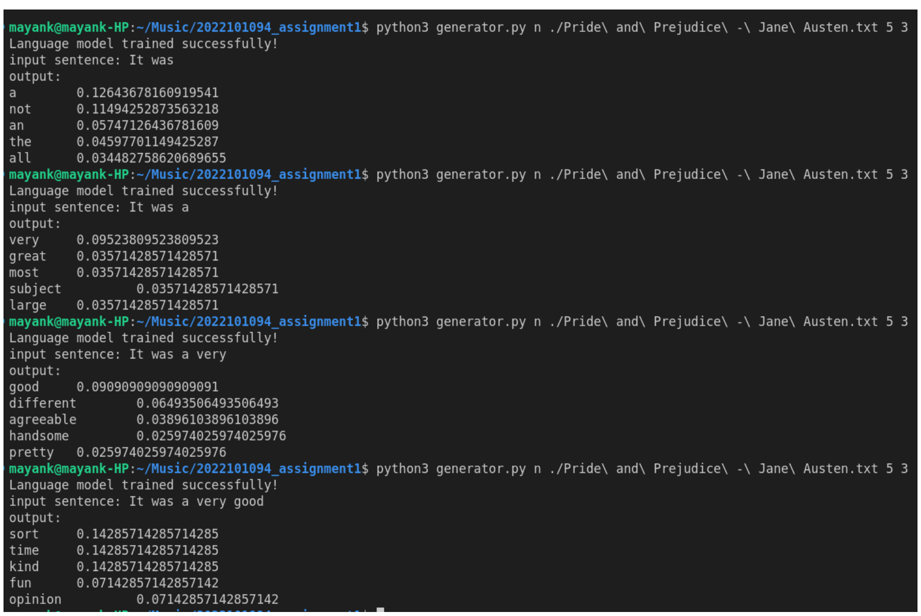
1.1.1 In-Context Text

Task-1: Using the generated N-gram models (without the smoothing techniques), try generating sequences. Experiment with different values of N and report which models perform better in terms of fluency.



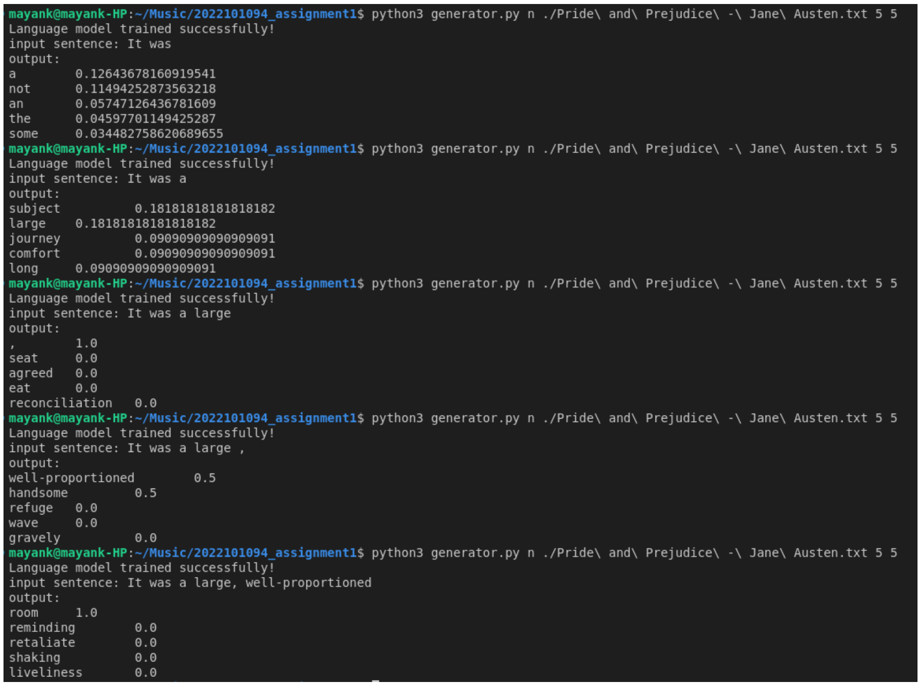
```
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py n ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 1
Language model trained successfully!
input sentence: It was
output:
,        0.0654523392027309
</s>     0.0518578276783988
.        0.03554974764479588
the      0.02820244421183169
to       0.027749071586204137
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py n ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 1
Language model trained successfully!
input sentence: It is of the greatest
output:
,        0.0654523392027309
</s>     0.0518578276783988
.        0.03554974764479588
the      0.02820244421183169
to       0.027749071586204137
```

Figure 1: No Smoothing. N=1



```
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py n ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
Language model trained successfully!
input sentence: It was
output:
a        0.12643678160919541
not      0.11494252873563218
an       0.05747126436781609
the      0.04597701149425287
all      0.034482758620689655
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py n ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
Language model trained successfully!
input sentence: It was a
output:
very     0.09523809523809523
great    0.03571428571428571
most     0.03571428571428571
subject          0.03571428571428571
large    0.03571428571428571
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py n ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
Language model trained successfully!
input sentence: It was a very
output:
good     0.09090909090909091
different        0.06493506493506493
agreeable        0.03896103896103896
handsome         0.025974025974025976
pretty   0.025974025974025976
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py n ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
Language model trained successfully!
input sentence: It was a very good
output:
sort     0.14285714285714285
time     0.14285714285714285
kind     0.14285714285714285
fun      0.07142857142857142
opinion          0.07142857142857142
```

Figure 2: No Smoothing. N=3



```
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py n ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 5
Language model trained successfully!
input sentence: It was
output:
a        0.12643678160919541
not      0.11494252873563218
an       0.05747126436781609
the      0.04597701149425287
some     0.034482758620689655
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py n ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 5
Language model trained successfully!
input sentence: It was a
output:
subject          0.18181818181818182
large    0.18181818181818182
journey          0.09090909090909091
comfort          0.09090909090909091
long     0.09090909090909091
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py n ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 5
Language model trained successfully!
input sentence: It was a large
output:
,        1.0
seat     0.0
agreed   0.0
eat      0.0
reconciliation   0.0
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py n ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 5
Language model trained successfully!
input sentence: It was a large ,
output:
well-proportioned        0.5
handsome         0.5
refuge   0.0
wave     0.0
gravely          0.0
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py n ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 5
Language model trained successfully!
input sentence: It was a large, well-proportioned
output:
room     1.0
reminding        0.0
retaliate        0.0
shaking          0.0
liveliness       0.0
```

Figure 3: No Smoothing. N=5

I attempted to predict a sentence starting with ”It was.” The results show that the unigram model performed the worst, as it does not consider any previous context, leading to random and incoherent predictions. The trigram model performed better, but the sentence still lacked fluency, with inconsistent shifts in the subject and meaning. The 5-gram model outperformed the others, producing the most fluent and coherent sentence. Based on this, we can conclude that, without smoothing, the 5-gram model provides the best performance, while the unigram model is the least effective among the 5-gram, 3-gram, and unigram models.

1.1.2 Out-of-Data (OOD) scenario

Task-2: Attempt to generate a sentence using an Out-of-Data (OOD) scenario with your N-gram models. Analyze and discuss the behavior of N-gram models in OOD contexts.

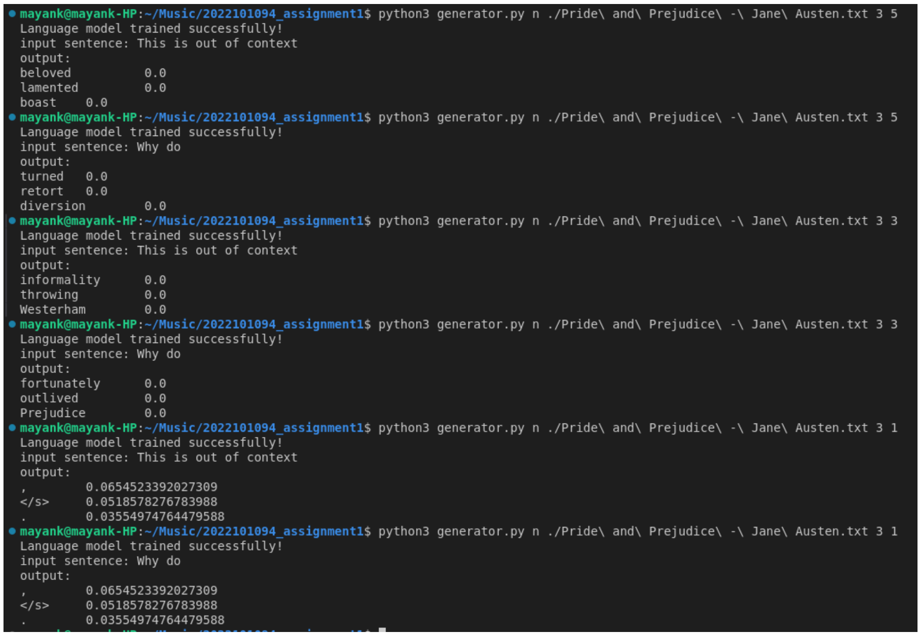
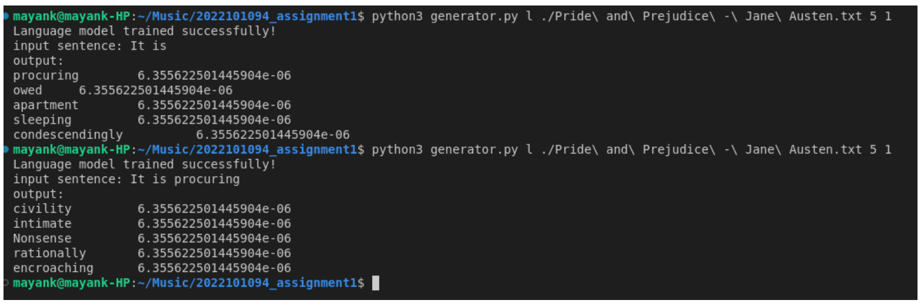


Figure 4: No Smoothing. Out-of-Data (OOD) scenario

This is focused on inputs where the individual words were familiar, but their combination was entirely new and had not appeared in the training data. In such cases, the N-gram models assigned zero probability to predictions because the specific context was missing from the training data, resulting in a zero count. Without this context, the models were unable to make valid predictions. This is a key limitation of N-gram models, as they struggle with new word combinations and cannot generalize effectively without techniques like smoothing.

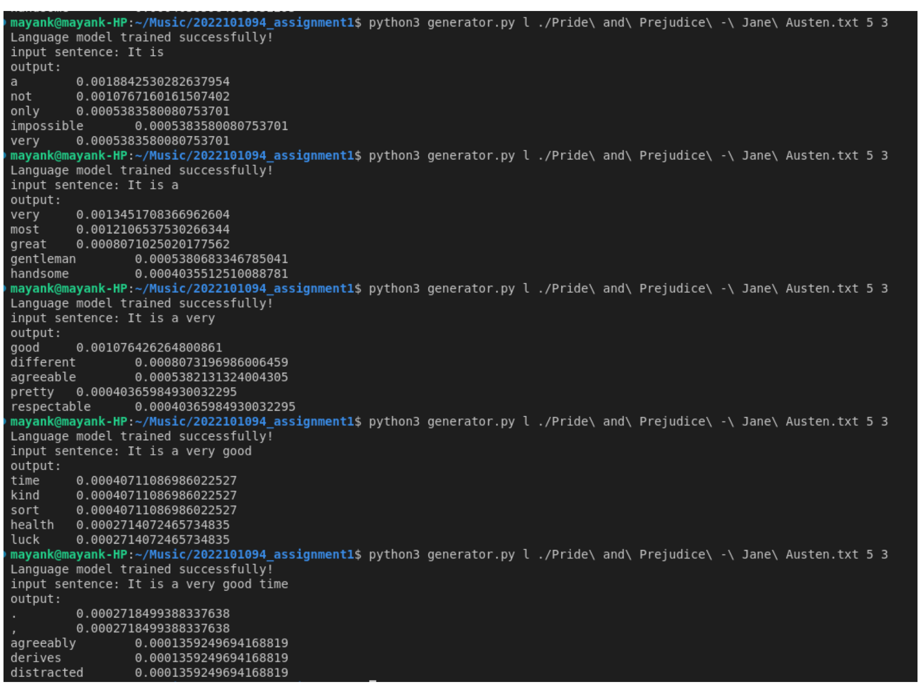
Task-3: Now try to generate text using the models with the smoothing techniques (LM1, LM2, LM3, LM4, LM5, LM6) for N=1,3 and 5 each.

1.2 Laplace Smoothing



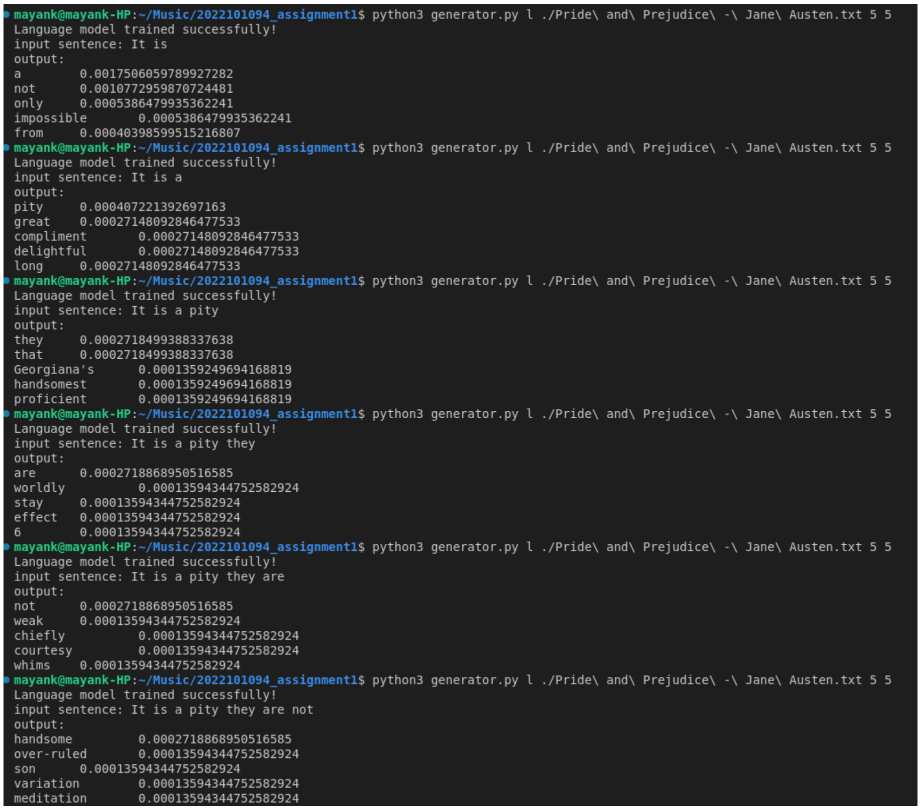
```
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py l ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 1
Language model trained successfully!
input sentence: It is
output:
procuring        6.355622501445904e-06
owed     6.355622501445904e-06
apartment        6.355622501445904e-06
sleeping         6.355622501445904e-06
condescendingly          6.355622501445904e-06
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py l ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 1
Language model trained successfully!
input sentence: It is procuring
output:
civility         6.355622501445904e-06
intimate         6.355622501445904e-06
Nonsense         6.355622501445904e-06
rationally       6.355622501445904e-06
encroaching      6.355622501445904e-06
mayank@mayank-HP:~/Music/2022101094_assignment1$
```

Figure 5: Laplace Smoothing. N=1



```
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py l ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
Language model trained successfully!
input sentence: It is
output:
a        0.0018842530282637954
not      0.0010767160161507402
only     0.0005383580080753701
impossible       0.0005383580080753701
very     0.0005383580080753701
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py l ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
Language model trained successfully!
input sentence: It is a
output:
very     0.0013451708366962604
most     0.0012106537530266344
great    0.0008071025020177562
gentleman        0.0005380683346785041
handsome         0.0004035512510088781
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py l ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
Language model trained successfully!
input sentence: It is a very
output:
good     0.001076426264800861
different        0.0008073196986006459
agreeable        0.0005382131324004305
pretty   0.00040365984930032295
respectable      0.00040365984930032295
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py l ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
Language model trained successfully!
input sentence: It is a very good
output:
time     0.00040711086986022527
kind     0.00040711086986022527
sort     0.00040711086986022527
health   0.0002714072465734835
luck     0.0002714072465734835
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py l ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
Language model trained successfully!
input sentence: It is a very good time
output:
.        0.0002718499388337638
,        0.0002718499388337638
agreeably        0.0001359249694168819
derives          0.0001359249694168819
distracted       0.0001359249694168819
```

Figure 6: Laplace Smoothing. N=3

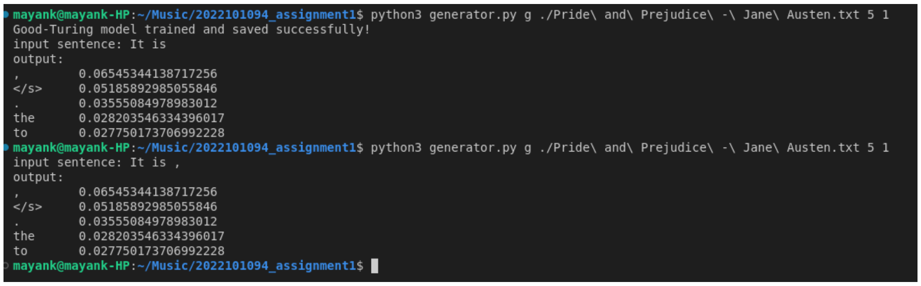


```
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py l ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 5
Language model trained successfully!
input sentence: It is
output:
a       0.0017506059789927282
not     0.0010772959870724481
only    0.0005386479935362241
impossible      0.0005386479935362241
from    0.00040398599515216807
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py l ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 5
Language model trained successfully!
input sentence: It is a
output:
pity    0.000407221392697163
great   0.00027148092846477533
compliment      0.00027148092846477533
delightful      0.00027148092846477533
long    0.00027148092846477533
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py l ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 5
Language model trained successfully!
input sentence: It is a pity
output:
they    0.0002718499388337638
that    0.0002718499388337638
Georgiana's     0.0001359249694168819
handsomest      0.0001359249694168819
proficient      0.0001359249694168819
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py l ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 5
Language model trained successfully!
input sentence: It is a pity they
output:
are     0.0002718868950516585
worldly         0.00013594344752582924
stay    0.00013594344752582924
effect  0.00013594344752582924
6       0.00013594344752582924
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py l ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 5
Language model trained successfully!
input sentence: It is a pity they are
output:
not     0.0002718868950516585
weak    0.00013594344752582924
chiefly         0.00013594344752582924
courtesy        0.00013594344752582924
whims   0.00013594344752582924
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py l ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 5
Language model trained successfully!
input sentence: It is a pity they are not
output:
handsome        0.0002718868950516585
over-ruled      0.00013594344752582924
son     0.00013594344752582924
variation       0.00013594344752582924
meditation      0.00013594344752582924
```

Figure 7: Laplace Smoothing. N=5

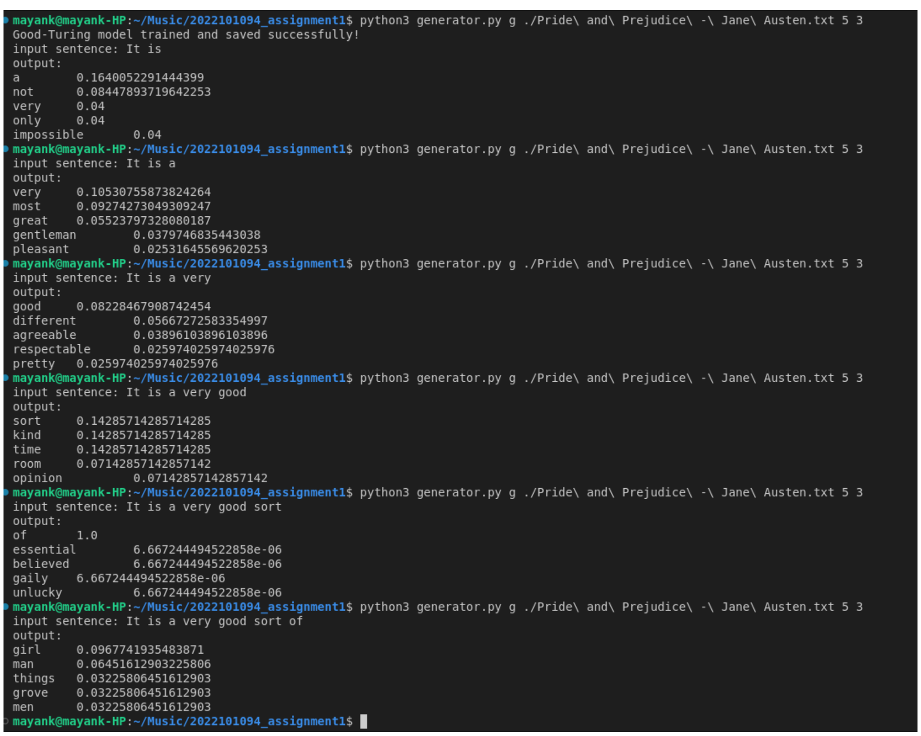
when dealing with unseen context using N=1, we found that the probabilities of the top predictions weren't zero, which was an improvement over the case with no smoothing. Additionally, the N=5 model produced sentences that were more fluent than those generated by the N=3 model. With N=3, the fluency started to drop off after a certain point because it only considered the previous two words. Overall, the N=5 model gave the best results in terms of fluency, while N=1 performed the worst.

1.3 Good Turing



```
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py g ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 1
Good-Turing model trained and saved successfully!
input sentence: It is
output:
,       0.06545344138717256
</s>    0.05185892985055846
.       0.03555084978983012
the     0.028203546334396017
to      0.027750173706992228
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py g ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 1
input sentence: It is ,
output:
,       0.06545344138717256
</s>    0.05185892985055846
.       0.03555084978983012
the     0.028203546334396017
to      0.027750173706992228
mayank@mayank-HP:~/Music/2022101094_assignment1$
```

Figure 8: Good Turing. N=1



```
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py g ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
Good-Turing model trained and saved successfully!
input sentence: It is
output:
a       0.1640052291444399
not     0.08447893719642253
very    0.04
only    0.04
impossible      0.04
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py g ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
input sentence: It is a
output:
very    0.10530755873824264
most    0.09274273049309247
great   0.05523797328080187
gentleman       0.0379746835443038
pleasant        0.02531645569620253
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py g ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
input sentence: It is a very
output:
good    0.08228467908742454
different       0.05667272583354997
agreeable       0.03896103896103896
respectable     0.025974025974025976
pretty  0.025974025974025976
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py g ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
input sentence: It is a very good
output:
sort    0.14285714285714285
kind    0.14285714285714285
time    0.14285714285714285
room    0.07142857142857142
opinion         0.07142857142857142
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py g ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
input sentence: It is a very good sort
output:
of      1.0
essential       6.667244494522858e-06
believed        6.667244494522858e-06
gaily   6.667244494522858e-06
unlucky         6.667244494522858e-06
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py g ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
input sentence: It is a very good sort of
output:
girl    0.0967741935483871
man     0.06451612903225806
things  0.03225806451612903
grove   0.03225806451612903
men     0.03225806451612903
mayank@mayank-HP:~/Music/2022101094_assignment1$
```

Figure 9: Good Turing. N=3

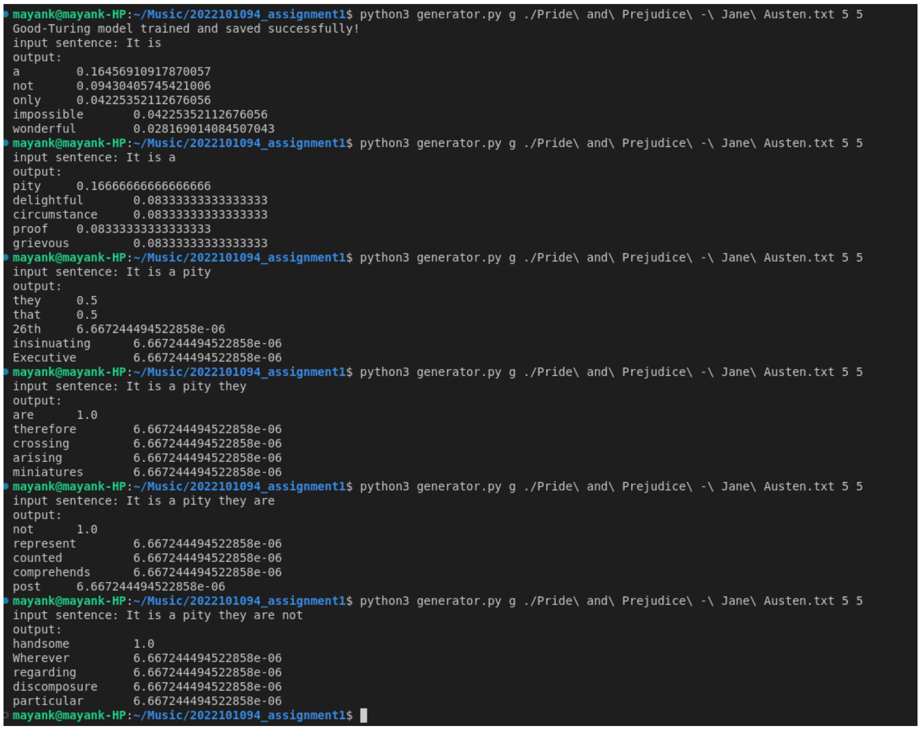


Figure 10: Good Turing. N=5

we can see that the 5-gram model performs the best in terms of fluency, making more confident predictions for the next word. It is followed by the 3-gram model, which also predicts fluently and outperforms the Laplace-smoothed model. The least fluent predictions come from the 1-gram model, which struggles the most.

1.4 Linear Interpolation

```
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py i ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 1
Language model trained successfully!
input sentence: It is
output:
untitled        0.0
earnest         0.0
pieces  0.0
taken   0.0
impelled        0.0
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py i ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 1
Language model trained successfully!
input sentence: It is untitled
output:
contrariwise    0.0
additional      0.0
wearisome       0.0
pressing        0.0
height  0.0
mayank@mayank-HP:~/Music/2022101094_assignment1$
```

Figure 11: Linear Interpolation. N=1

```
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py i ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
Language model trained successfully!
input sentence: It is
output:
a       0.2934313418274028
not     0.2892827156131309
impossible      0.2865169648036163
only    0.2865169648036163
very    0.2865169648036163
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py i ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
Language model trained successfully!
input sentence: It is a
output:
very    0.09290347703052125
most    0.09224704883206052
great   0.09027776423667828
gentleman       0.08896490783975679
pleasant        0.08830847964129604
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py i ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
Language model trained successfully!
input sentence: It is a very
output:
good    0.04232740684184199
different       0.0409804502787667
agreeable       0.0396334937156914
handsome        0.038960015434153755
favourable      0.038960015434153755
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py i ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
Language model trained successfully!
input sentence: It is a very good
output:
time    0.035471665774509074
kind    0.035471665774509074
sort    0.035471665774509074
table   0.03176753522605202
health  0.03176753522605202
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py i ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
Language model trained successfully!
input sentence: It is a very good time
output:
,       0.036522681127708914
.       0.036522681127708914
silently        0.010593767288509512
trim    0.010593767288509512
persevered      0.010593767288509512
mayank@mayank-HP:~/Music/2022101094_assignment1$ python3 generator.py i ./Pride\ and\ Prejudice\ -\ Jane\ Austen.txt 5 3
Language model trained successfully!
input sentence: It is a very good time ,
output:
and     0.22942839231522433
but     0.22501496017238187
without         0.22501496017238187
that    0.22391160213667127
as      0.22391160213667127
```

Figure 12: Linear Interpolation. N=3

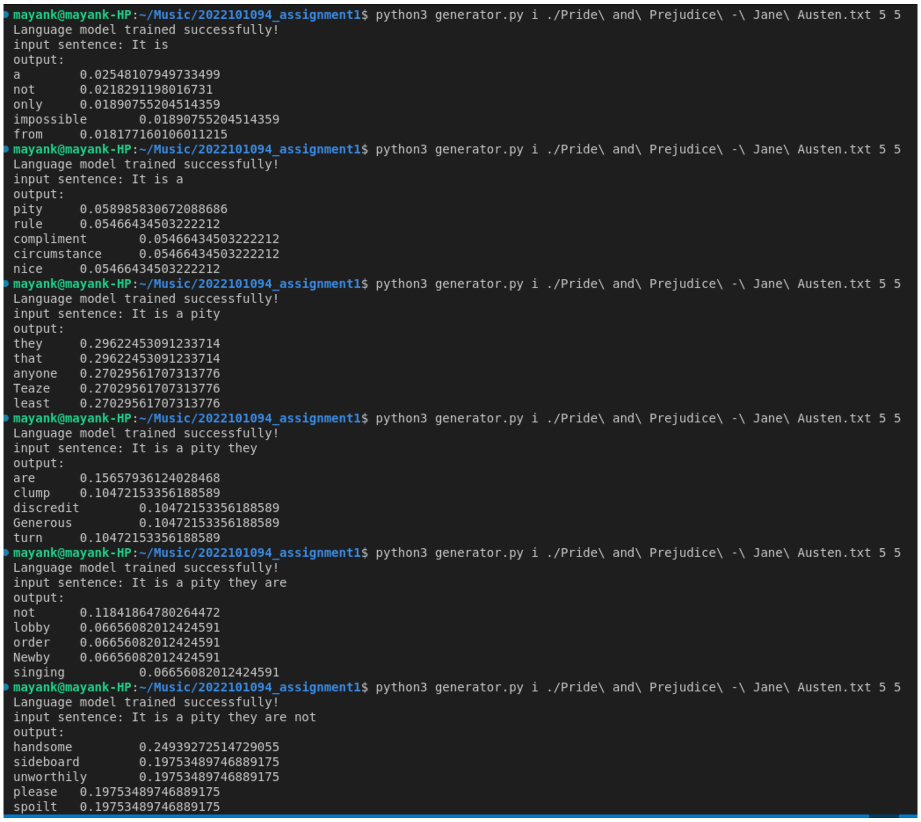


Figure 13: Linear Interpolation. N=5

Here we see that the unigram performs poorly.The trigram produces fluent but not completely fluent sentences. The 5-gram model produces the best fluent predictions out of N=1,3,5.

Among the three smoothing methods, Good-Turing Smoothing produces the most fluent sentences. On the other hand, Laplacian smoothing tends to generate fewer coherent sentences. This may be due to the addition of 1, which might be too large, or the vocabulary size in the denominator, which can make all probabilities small. Linear interpolation, however, can perform better than Good-Turing Smoothing if we find a better way to set the λ weights for each model.

2 Perplexity Analysis

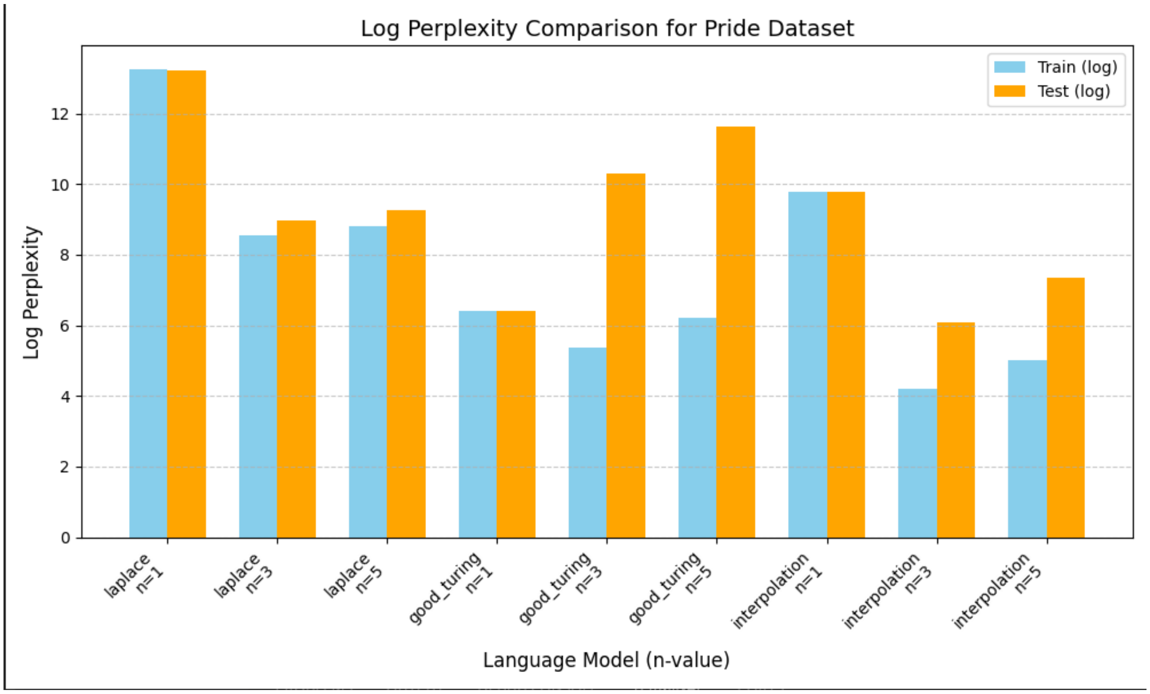


Figure 14: Average perplexity scores for Pride and Prejudice

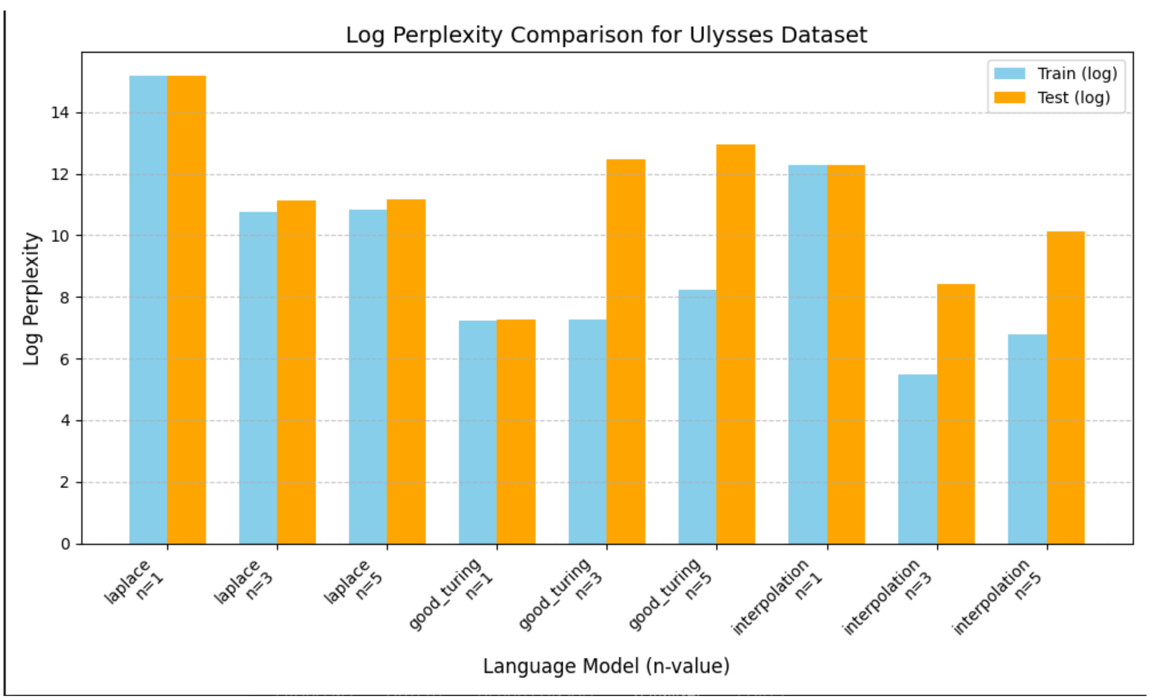


Figure 15: Average perplexity scores for Ulysses

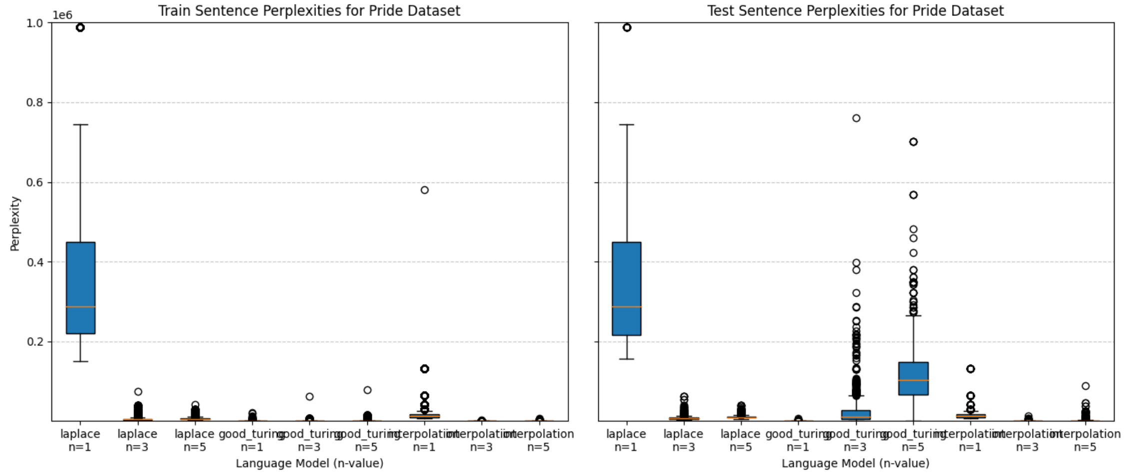


Figure 16: Box plot for perplexity scores for Pride and Prejudice

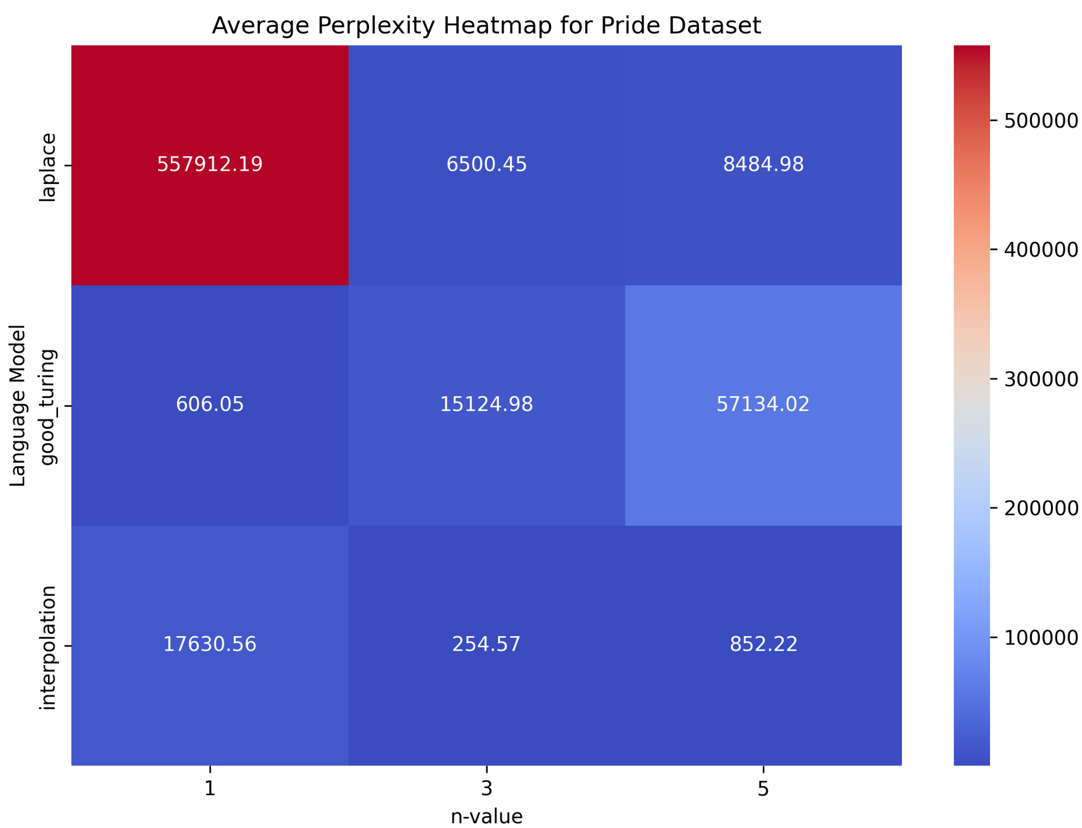


Figure 17: Heatmap of avg perplexity scores for Pride and Prejudice

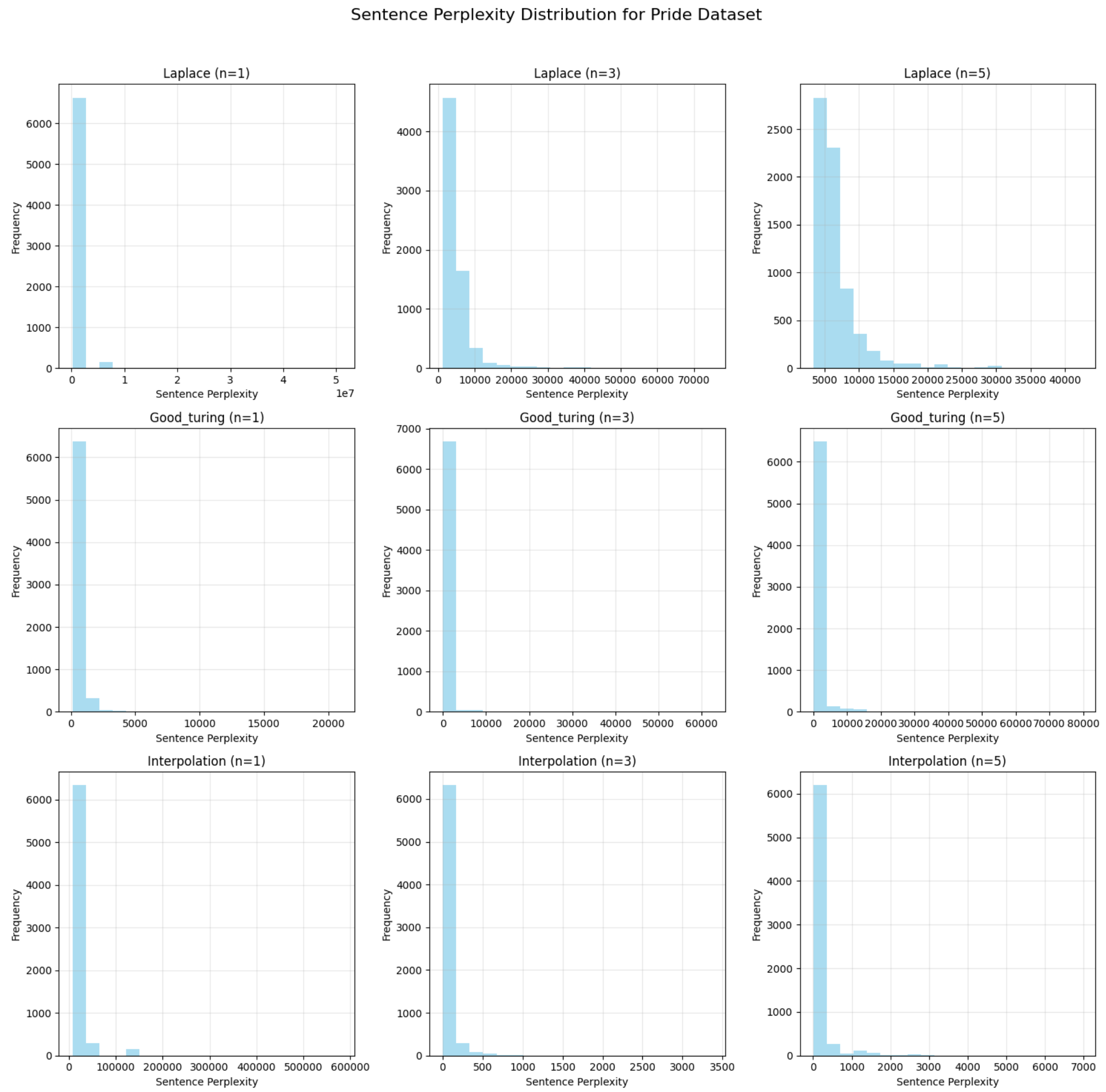


Figure 18: Frequency distribution of perplexity scores for Pride and Prejudice

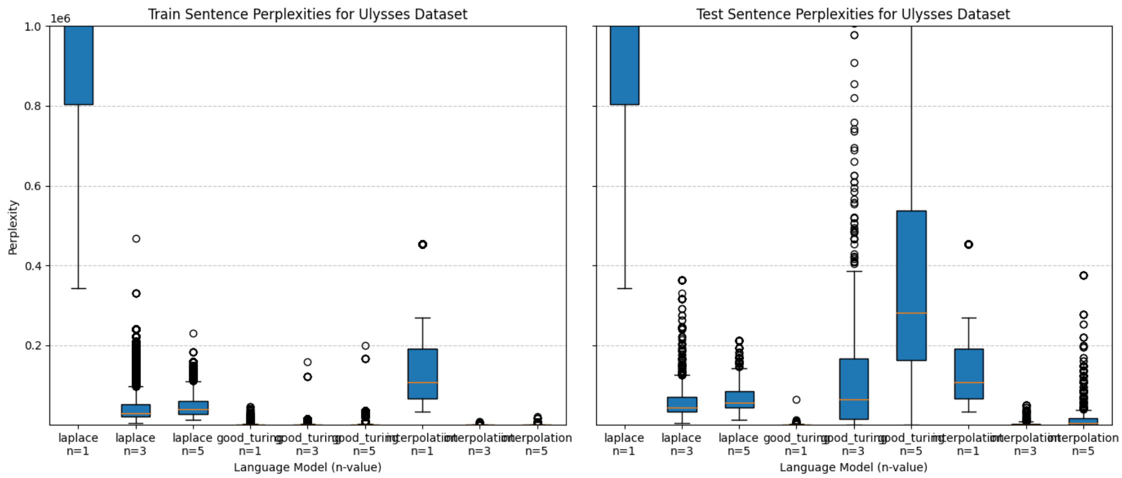


Figure 19: Box plot for perplexity scores for Ulysses

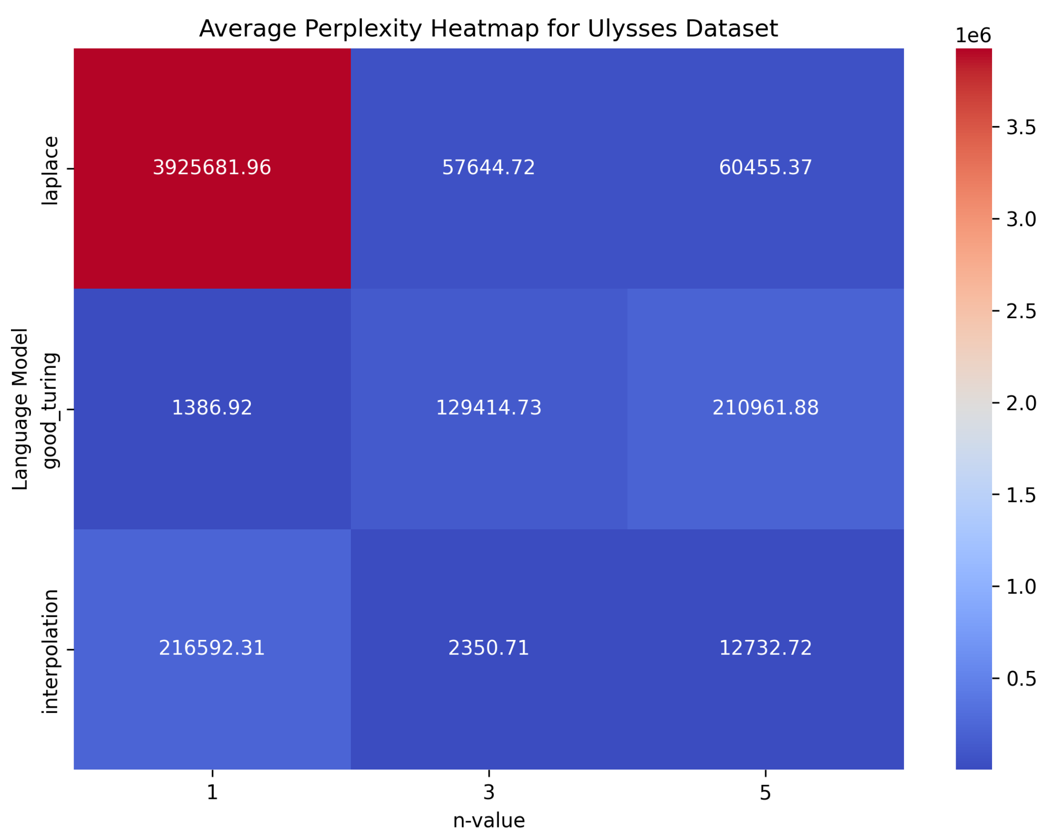


Figure 20: Heatmap of avg perplexity scores for Ulysses

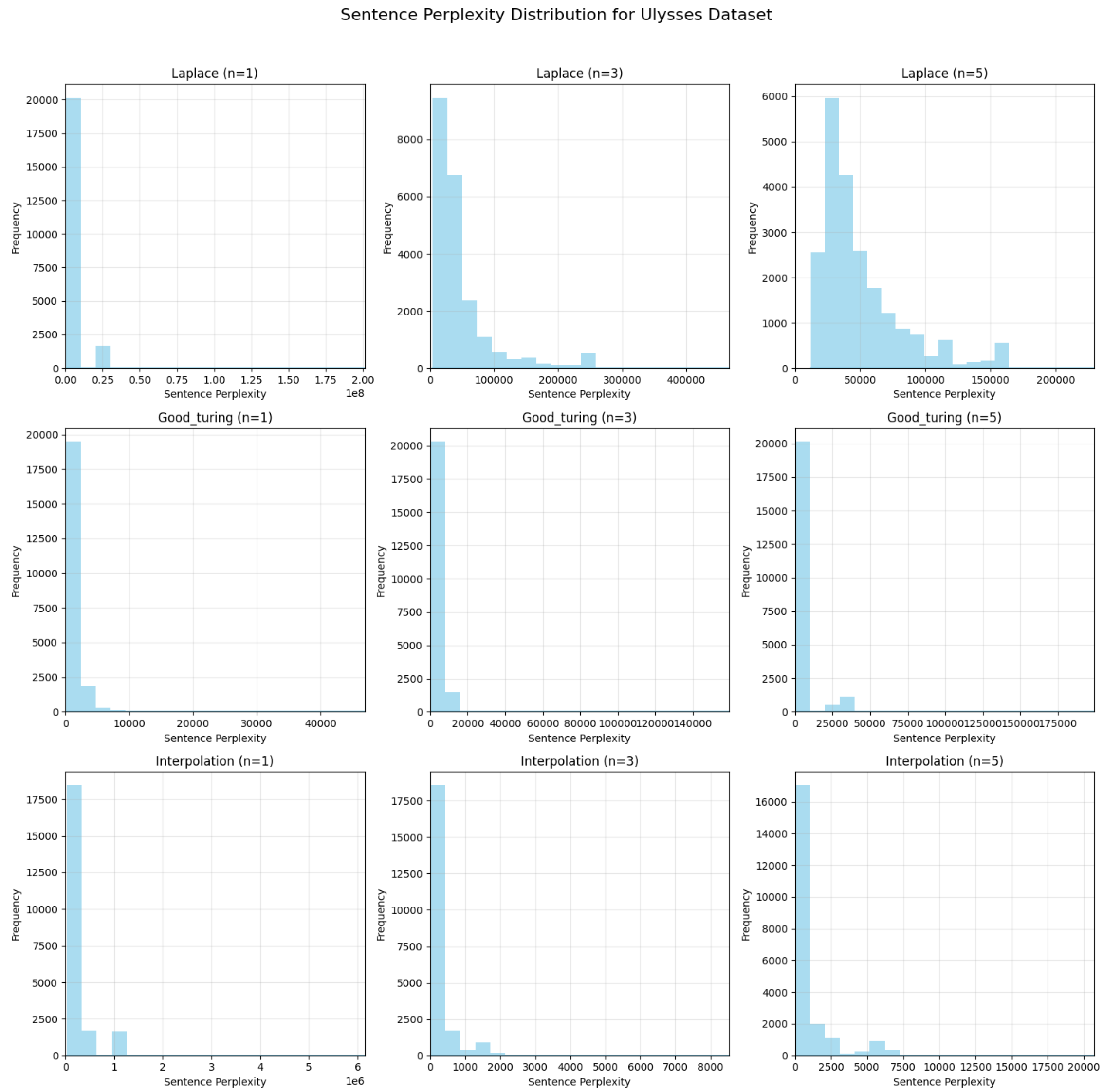


Figure 21: Frequency distribution of perplexity scores for Ulysses

- The test set perplexity is consistently higher than the training set perplexity, indicating overfitting in higher-order n-grams due to data sparsity.
- Good-Turing smoothing outperforms Laplace smoothing for lower-order n-grams, as seen in its lower perplexity for N=1 compared to Laplace.
- Linear interpolation shows a significant increase in test set perplexity for N=3 and N=5, suggesting difficulties in managing data sparsity with higher-order n-grams.
- Good-Turing achieves balanced performance across different n values but struggles with generalization for N=5.

- Laplace smoothing consistently produces higher perplexity than other methods, reflecting its ineffectiveness in reducing overfitting.