Assignment - 1

Tokenization:

1. **Sentence Tokenizer:**

   Handles sentence boundaries, and avoids splitting on common abbreviations (e.g., "Mr.", "Ms.", "Dr.") by utilizing negative look behinds.

2. **Word Tokenizer:**

   Extracts individual words, allowing alphanumeric characters and hyphens in words. Maintains word boundaries in the tokenized output.

3. **Placeholder Usage:**

   Introduces placeholders ("<NUM>", "<MAILID>", "<URL>", "<HASHTAG>", "<MENTION>") to mask specific types of information (e.g., numbers, email addresses, URLs, hashtags, mentions) in the tokenized text.

4. **Punctuation Tokenizer:**

   Extracts punctuation marks from words and preserves punctuation information in the tokenized output.

5. **Output:**

   The final output is a list of lists, where each inner list represents a sentence. The inner lists contain tokens (words and punctuation marks) with placeholders replacing identified patterns.

N-Grams:

- **Preprocessing of corpus:**
  - Replaces the newline ('\n') with space (' ') and removes contractions using Python's contraction module.
  - Tokenizes the preprocessed corpus and adds n-1 start tokens (<s>) at the beginning of a sentence and 1 end token (<\s>) at the end of the sentence.
  - Replaces the punctuation mark with a space (' '). All the tokens with frequency 1 are replaced by <UNK>.
- **Good Turing Smoothing:**

$$\text{Count}^*(w_1w_2w_3) = r^* = (r+1) * \frac{S(N_{r+1})}{S(N_r)} (r = \text{Count}(w_1w_2w_3))$$

$$S(N_0) = 1$$

$$P(w_3|w_1w_2) = \frac{\text{Count}^*(w_1w_2w_3)}{\sum_{w_i \in V} \text{Count}^*(w_1w_2w_i)}$$

Nr for unknown values is estimated from

$$\log(N_r) = a + b \log(r)$$

a, b are intercept and slope of log(Zr) - log(r) regression line.

- **Interpolation:**

$$P(t_3|t_1,t_2) = \lambda_1 \hat{P}(t_3) + \lambda_2 \hat{P}(t_3|t_2) + \lambda_3 \hat{P}(t_3|t_1,t_2) \tag{6}$$

$\hat{P}$ are maximum likelihood estimates of the probabilities, and $\lambda_1 + \lambda_2 + \lambda_3 = 1$, so P again represent probability distributions.

Unigrams: $\hat{P}(t_3) = \frac{f(t_3)}{N}$

Bigrams: $\hat{P}(t_3|t_2) = \frac{f(t_2,t_3)}{f(t_2)}$

Trigrams: $\hat{P}(t_3|t_1,t_2) = \frac{f(t_1,t_2,t_3)}{f(t_1,t_2)}$

lambda values are estimated as follows:

set $\lambda_1 = \lambda_2 = \lambda_3 = 0$
foreach trigram t_1, t_2, t_3 with $f(t_1,t_2,t_3) > 0$
    depending on the maximum of the following three values:
        case $\frac{f(t_1,t_2,t_3)-1}{f(t_1,t_2)-1}$: increment λ_3 by $f(t_1,t_2,t_3)$
        case $\frac{f(t_2,t_3)-1}{f(t_2)-1}$: increment λ_2 by $f(t_1,t_2,t_3)$
        case $\frac{f(t_3)-1}{N-1}$: increment λ_1 by $f(t_1,t_2,t_3)$
    end
end
normalize $\lambda_1, \lambda_2, \lambda_3$

Average Perplexity Scores:

LM_TYPE	TRAIN SET	TEST SET
LM-1: pride and prejudice - gt	622721364.1088043	8809439.417691033

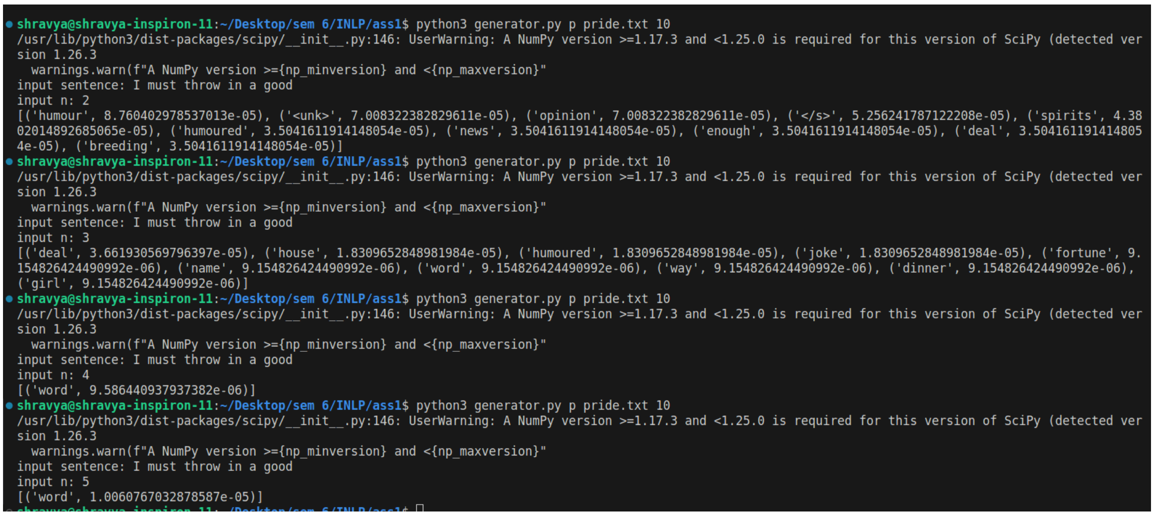
LM_TYPE	TRAIN SET	TEST SET
LM-2: pride and prejudice - i	15.381154391717658	314.95857002977056
LM-3: ulysses - gt	5092318718.764593	136758968.79318216
LM-4: ulysses - i	33.89745816908959	486.1538076733941

Generation:

- For the N-gram model (without smoothing), as the value of N increases it generates the correct word because of long history. (No.of guesses decreases).

  Example:

  Sentence: I must throw in a good word for my little Lizzy.

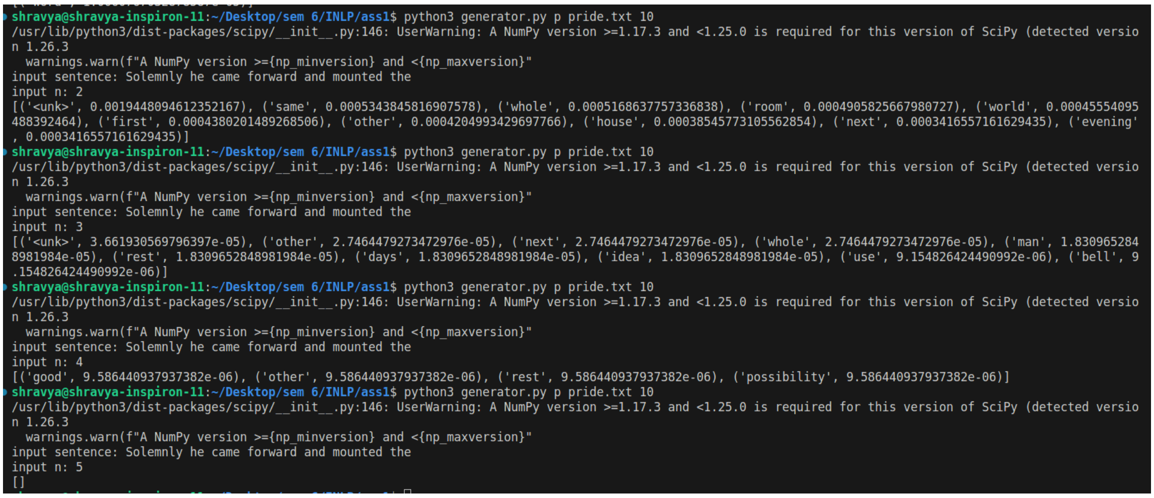


```
shravya@shravya-inspiron-11:~/Desktop/sem 6/INLP/ass1$ python3 generator.py p pride.txt 10
/usr/lib/python3/dist-packages/scipy/__init__.py:146: UserWarning: A NumPy version >=1.17.3 and <1.25.0 is required for this version of SciPy (detected ver
sion 1.26.3
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
input sentence: I must throw in a good
input n: 2
[('humour', 8.760402978537013e-05), ('<unk>', 7.008322382829611e-05), ('opinion', 7.008322382829611e-05), ('</s>', 5.256241787122208e-05), ('spirits', 4.38
02014892685065e-05), ('humoured', 3.5041611914148054e-05), ('news', 3.5041611914148054e-05), ('enough', 3.5041611914148054e-05), ('deal', 3.50416119141480
4e-05), ('breeding', 3.5041611914148054e-05)]
shravya@shravya-inspiron-11:~/Desktop/sem 6/INLP/ass1$ python3 generator.py p pride.txt 10
/usr/lib/python3/dist-packages/scipy/__init__.py:146: UserWarning: A NumPy version >=1.17.3 and <1.25.0 is required for this version of SciPy (detected ver
sion 1.26.3
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
input sentence: I must throw in a good
input n: 3
[('deal', 3.661930569796397e-05), ('house', 1.8309652848981984e-05), ('humoured', 1.8309652848981984e-05), ('joke', 1.8309652848981984e-05), ('fortune', 9.
154826424490992e-06), ('name', 9.154826424490992e-06), ('word', 9.154826424490992e-06), ('way', 9.154826424490992e-06), ('dinner', 9.154826424490992e-06),
('girl', 9.154826424490992e-06)]
shravya@shravya-inspiron-11:~/Desktop/sem 6/INLP/ass1$ python3 generator.py p pride.txt 10
/usr/lib/python3/dist-packages/scipy/__init__.py:146: UserWarning: A NumPy version >=1.17.3 and <1.25.0 is required for this version of SciPy (detected ver
sion 1.26.3
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
input sentence: I must throw in a good
input n: 4
[('word', 9.586440937937382e-06)]
shravya@shravya-inspiron-11:~/Desktop/sem 6/INLP/ass1$ python3 generator.py p pride.txt 10
/usr/lib/python3/dist-packages/scipy/__init__.py:146: UserWarning: A NumPy version >=1.17.3 and <1.25.0 is required for this version of SciPy (detected ver
sion 1.26.3
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
input sentence: I must throw in a good
input n: 5
[('word', 1.0060767032878587e-05)]
```

- **OOD scenario:**
  - N-gram models struggle in out-of-data contexts as they rely heavily on the training data leading to poor generations.
  - N-gram models may face challenges in capturing long-term dependencies between words, especially when the context spans a considerable distance.

  Example:

  Sentence: Solemnly he came forward and mounted the round gunrest.



```
shravya@shravya-inspiron-11:~/Desktop/sem 6/INLP/ass1$ python3 generator.py p pride.txt 10
/usr/lib/python3/dist-packages/scipy/__init__.py:146: UserWarning: A NumPy version >=1.17.3 and <1.25.0 is required for this version of SciPy (detected versio
n 1.26.3
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
input sentence: Solemnly he came forward and mounted the
input n: 2
[('<unk>', 0.0019448094612352167), ('same', 0.0005343845816907578), ('whole', 0.0005168637757336838), ('room', 0.00049058256679807 27), ('world', 0.00045554095
488392464), ('first', 0.0004380201489268506), ('other', 0.0004204993429697766), ('house', 0.00038545773105562854), ('next', 0.0003416557161629435), ('evening'
, 0.0003416557161629435)]
shravya@shravya-inspiron-11:~/Desktop/sem 6/INLP/ass1$ python3 generator.py p pride.txt 10
/usr/lib/python3/dist-packages/scipy/__init__.py:146: UserWarning: A NumPy version >=1.17.3 and <1.25.0 is required for this version of SciPy (detected versio
n 1.26.3
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
input sentence: Solemnly he came forward and mounted the
input n: 3
[('<unk>', 3.661930569796397e-05), ('other', 2.7464479273472976e-05), ('next', 2.7464479273472976e-05), ('whole', 2.7464479273472976e-05), ('man', 1.830965284
8981984e-05), ('rest', 1.8309652848981984e-05), ('days', 1.8309652848981984e-05), ('idea', 1.8309652848981984e-05), ('use', 9.154826424490992e-06), ('bell', 9
.154826424490992e-06)]
shravya@shravya-inspiron-11:~/Desktop/sem 6/INLP/ass1$ python3 generator.py p pride.txt 10
/usr/lib/python3/dist-packages/scipy/__init__.py:146: UserWarning: A NumPy version >=1.17.3 and <1.25.0 is required for this version of SciPy (detected versio
n 1.26.3
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
input sentence: Solemnly he came forward and mounted the
input n: 4
[('good', 9.586440937937382e-06), ('other', 9.586440937937382e-06), ('rest', 9.586440937937382e-06), ('possibility', 9.586440937937382e-06)]
shravya@shravya-inspiron-11:~/Desktop/sem 6/INLP/ass1$ python3 generator.py p pride.txt 10
/usr/lib/python3/dist-packages/scipy/__init__.py:146: UserWarning: A NumPy version >=1.17.3 and <1.25.0 is required for this version of SciPy (detected versio
n 1.26.3
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
input sentence: Solemnly he came forward and mounted the
input n: 5
[]
```

- **Using Smoothing Techniques:**

  Applying these smoothing techniques enhances the adaptability of N-gram models to out-of-data contexts by mitigating issues related to zero probabilities and promoting more robust language modeling.