Report

Datasets:

Our datasets consist of two literary works, namely "Pride" and "Ulysses". Models: The language modeling involves the implementation of two distinct types of models, each employing a unique smoothing technique:

- Laplace Smoothing invloves adding a small value (Generally 1) to each count in a frequency distribution to prevent zero probabilities.
- Good Turing: This method involves adjusting the probability of n-grams based on their frequency, assigning a non-zero probability to unseen n-grams.
- Interpolation: This approach combines probabilities from n-grams of all orders, ensuring a more balanced prediction.

1. Objective:

The aim of this report is to assess and contrast the efficiency of language modeling through the Laplace Smoothing, Good-Turing and Linear Interpolation smoothing methods.

1.1 Metric:

The evaluation of model performance utilizes perplexity scores, where lower scores signify superior predictive power and a better understanding of the language structure by the model.

1.2 Results:

(Text files containing perplexity scores for indivudual lines for both datasets using all the 3 smoothing techniques are also included in the folder named **Text Files**)

These are the average train and test perplexity scores for both dataets using all the 3 smoothing techniques for different n.

Corpus	n	Laplace Train Perplexity	Laplace Test Perplexity	Good Turing Train Perplexity	Good Turing Test Perplexity	Interpolation Train Perplexity	Interpolation Test Perplexi
Pride and Prejudice	1	440.585607	1144.337159	418.391064	1006.722106	440.591364	1144.467560
Pride and Prejudice	3	12.881835	11.463579	3.276596	2.684569	5.372355	4.003369
Pride and Prejudice	5	10.204745	10.588221	1.315309	1.980351	2.203966	2.832701
Ulysses	1	907.466333	1383.359552	806.449291	1271.366001	907.522787	1383.369181
Ulysses	3	52.916791	30.530206	4.814811	3.420300	7.778601	4.757329
Ulysses	5	59.042963	33.182554	3.196707	2.482523	5.388709	3.231894

1.3 Analysis

The results indicate how perplexity scores vary across models, smoothing techniques, and corpora:

1. Observations by Corpus and N-gram Order

- **Pride and Prejudice:**
  - For : N=1
    - Good-Turing achieves slightly lower perplexity than Laplace and Interpolation, indicating better handling of rare events at the unigram level.
  - For N=3 and N = 5 :
    - Good-Turing significantly outperforms both Laplace and Interpolation, especially in the test perplexity. This shows its ability to generalize effectively for higher N-grams, reducing overfitting to training data.
    - Interpolation exhibits intermediate performance, balancing context (higher N) and generalization (lower N).
- **Ulysses:**
  - For N = 1:
    - Good-Turing again performs better than Laplace and Interpolation, though all techniques have very high perplexity. This reflects the complexity of the corpus and its vocabulary size.
  - For N = 3 and N = 5:
    - Good-Turing achieves the lowest perplexity, indicating its suitability for handling rare word combinations in complex texts like "Ulysses."
    - Interpolation performs better than Laplace but lags behind Good-Turing. The higher N-gram context in Ulysses may result in overgeneralization by interpolation.

2. Observations Across Techniques

- **Laplace Smoothing:**
  - Results in consistently higher perplexity scores, particularly for higher N-grams, across both corpora. This reflects its tendency to distribute probabilities uniformly, which can lead to poor performance on larger corpora or with rare N-grams.
  - Best suited for simpler, smaller datasets.
- **Good-Turing Smoothing:**
  - Achieves the lowest perplexity scores for both corpora, particularly with higher N-grams. This suggests that it is effective in redistributing probabilities to unseen grams, making it better suited for complex and diverse datasets.
  - However, it may face computational challenges with very large datasets.
- **Interpolation:**
  - Offers intermediate performance, often outperforming Laplace but not matching Good-Turing in test perplexity.
  - Performs better in "Pride and Prejudice" than in "Ulysses," likely due to its ability to balance context when patterns are less sparse or repetitive.

3. Corpus-Specific Observations

- **"Pride and Prejudice"**:
  - Lower perplexity values suggest that this corpus is less complex and more homogeneous, with fewer unique word combinations. Good-Turing excels by redistributing probabilities for rare events, while Interpolation performs well due to its ability to capture broader patterns.
- **"Ulysses"**:
  - The higher perplexity values reflect the text's complexity, with dense, rare word combinations. Good-Turing handles this better, likely because it accounts for unseen events, while Interpolation struggles with the sparsity of high-order grams.

Advantages and Disadvantages of Techniques

Laplace Smoothing

- **Advantages:**
  - Simple to implement and interpret.
  - Guarantees non-zero probabilities, which is useful for small datasets.
- **Disadvantages:**
  - Distributes probabilities too uniformly, leading to higher perplexity on larger datasets or higher N-grams.
  - Does not differentiate well between frequent and rare events.

Good-Turing Smoothing

- **Advantages:**
  - Redistributes probabilities effectively to unseen N-grams, improving generalization.
  - Performs well on corpora with rare word combinations (e.g., "Ulysses").
- **Disadvantages:**
  - Computationally intensive for large datasets.
  - May produce unreliable estimates for very low-frequency N-grams.

Interpolation

- **Advantages:**
  - Combines lower and higher-order n-grams, balancing generalization and context.
  - Adapts well to less sparse datasets with moderate complexity (e.g., "Pride and Prejudice").
- **Disadvantages:**
  - May overgeneralize by including probabilities from lower-order N-grams.
  - Struggles with very sparse or complex datasets where higher-order grams dominate.

Key Insights

1. **Choice of Technique Depends on Corpus:**
   - **Good-Turing** performs best for complex, sparse datasets (e.g., "Ulysses").
   - **Interpolation** is effective for more balanced, less sparse corpora (e.g., "Pride and Prejudice").
   - **Laplace** is generally outperformed by both techniques but may still be useful for smaller datasets or as a baseline.
2. **Impact of NNgram Order:**
   - Higher grams (N=3,N=5) benefit from techniques like Good-Turing, which handle sparsity well.
   - Lower grams (N=1) tend to favor simpler techniques like Laplace, though perplexity remains high due to limited context.
3. **Importance of Dataset Size and Diversity:**
   - Larger, diverse corpora reduce the limitations of all techniques, though the choice of smoothing still matters for high-order N-grams.

---

2. Objective:

To evaluate the effectiveness of N-gram models with and without smoothing in text generation and assess their ability to handle unseen data (Out-of-Distribution or OOD scenarios).

2.1 Generating Sequences with N-gram Models (Without Smoothing)

N-gram models predict the next word in a sequence based on the frequency of N-word combinations in the training corpus. Without smoothing, these models assign a probability of zero to any N-gram that does not exist in the training data.

Key Observations

1. **Bigrams (N=2)**:
   - Bigrams incorporate basic context by considering word pairs (e.g., "I am," "am sure").
   - The text becomes marginally more coherent compared to unigrams, but predictions are still limited to immediate predecessors.
   - Sentences may appear jumbled due to a lack of broader context.
2. **Trigrams (N=3)**:
   - Adding a third word (e.g., "I am sure") improves fluency and coherence significantly.
   - The model strikes a good balance between randomness and overfitting, capturing more meaningful patterns from the training corpus.
3. **Higher N-grams (N=4 and above)**:
   - Models with higher N rely heavily on specific word combinations in the training data.
   - This leads to problems of **sparsity** (not enough data to cover all possible combinations) and **overfitting** (model becomes too dependent on training data).
   - Unseen N-grams are assigned a probability of zero, leading to "zero-probability" issues and random word predictions.

```
========== N: 2 ==========
('sure', 0.19076923076923077)
('not', 0.09538461538461539)
('afraid', 0.052307692307692305)
('i', 0.04)
('sorry', 0.03076923076923077)
========== N: 3 ==========
('sure', 0.20394736842105263)
('not', 0.09868421052631579)
('afraid', 0.05592105263157895)
('sorry', 0.03289473684210526)
('very', 0.03289473684210526)
========== N: 4 ==========
('not', 0.14285714285714285)
('sure', 0.12087912087912088)
('sorry', 0.06593406593406594)
('afraid', 0.06593406593406594)
('very', 0.054945054945054944)
```

```
========== N: 5 ==========
('not', 0.14285714285714285)
('sure', 0.12087912087912088)
('sorry', 0.06593406593406594)
('afraid', 0.06593406593406594)
('very', 0.054945054945054944)
========== N: 6 ==========
('not', 0.14285714285714285)
('sure', 0.12087912087912088)
('sorry', 0.06593406593406594)
('afraid', 0.06593406593406594)
('very', 0.054945054945054944)
========== N: 7 ==========
('not', 0.14285714285714285)
('sure', 0.12087912087912088)
('sorry', 0.06593406593406594)
('afraid', 0.06593406593406594)
('very', 0.054945054945054944)
```

Conclusion for N-grams without Smoothing

- **Trigrams (N=3)** offer the best balance of fluency and generalization.
- Models with N=4 and above suffer from diminishing returns, sparsity, and the zero-probability problem, which limits their utility.

2.2 Out-of-Distribution (OOD) Scenarios Without Smoothing:

When encountering unseen word sequences (e.g., *"Quantum entanglement revolutionizes technological paradigms"*), the model:

- Fails to find any matching N-grams in the training data.

- Assigns a probability of zero, leading to no predictions.
- Highlights the fundamental limitation of N-gram models that depend solely on the training set.

```
Testing Sentence: Quantum entanglement revolutionizes technological paradigms
N-gram Model (N=2):
  No matching prefix found.
N-gram Model (N=3):
  No matching prefix found.
N-gram Model (N=4):
  No matching prefix found.

Testing Sentence: Blockchain cryptocurrency decentralizes financial ecosystems
N-gram Model (N=2):
  No matching prefix found.
N-gram Model (N=3):
  No matching prefix found.
N-gram Model (N=4):
  No matching prefix found.

Testing Sentence: Machine learning algorithms simulate neural networks
N-gram Model (N=2):
  No matching prefix found.
N-gram Model (N=3):
  No matching prefix found.
N-gram Model (N=4):
  No matching prefix found.
```

2.3 Generating Text with Smoothing Techniques

Smoothing techniques address the zero-probability problem in N-gram models by redistributing probability mass to unseen word sequences. This ensures robust generalization and adaptability in text generation. Here, we evaluate the performance of three smoothing techniques across different N-gram levels (N = 1, 3, 5).

Smoothing Techniques Evaluated

1. **L Smoothing**
   - Redistributes probabilities among seen and unseen word sequences, ensuring all possibilities have non-zero probabilities.
   - Shows consistent performance across different N values, but predictions lack the nuanced adjustment observed in other methods.
2. **G Smoothing (Good-Turing)**
   - Allocates probabilities to unseen N-grams based on observed frequencies.
   - Provides slightly higher probabilities to frequent words compared to L Smoothing, improving prediction confidence for common terms.
3. **I Smoothing (Interpolation)**
   - Combines probabilities across N-gram levels (e.g., unigrams, bigrams, trigrams).
   - Balances context from higher-order and lower-order N-grams, resulting in more contextually relevant predictions.

Results and Key Observations

For N = 1 (Unigrams)

At this level, all three smoothing techniques result in basic word predictions without contextual dependencies.

- **L Smoothing**: Basic word distributions are generated.

- **G Smoothing**: Slightly better allocation to frequent words.
- **I Smoothing**: Performs similarly but offers no distinct advantage at the unigram level.

For N = 3 (Trigrams)

Smoothing techniques improve the coherence and fluency of predictions by considering context from two preceding words.

- **L Smoothing**:
  - Top Predictions:*sure: 0.1347, not: 0.1123, sorry: 0.0674, afraid: 0.0561, glad: 0.0561*
  - **Observation**: Provides balanced probabilities but lacks fine-tuned adjustments.
- **G Smoothing**:
  - Top Predictions:*sure: 0.1628, not: 0.1386, sorry: 0.0903, afraid: 0.0784, glad: 0.0784*
  - **Observation**: Higher probabilities for frequent words (e.g., "sure," "not"), enhancing prediction confidence.
- **I Smoothing**:
  - Top Predictions:*sure: 0.1463, not: 0.1220, sorry: 0.0732, afraid: 0.0610, glad: 0.0610*
  - **Observation**: Balances probabilities effectively, offering both contextual relevance and flexibility.

For N = 5 (Higher-order N-grams)

Higher-order N-grams allow for richer context but face sparsity issues without smoothing.

- **L Smoothing**:
  - Top Predictions:*sure: 0.1347, not: 0.1123, sorry: 0.0674, afraid: 0.0561, glad: 0.0561*
  - **Observation**: Similar to trigrams, probabilities are uniformly distributed without significant emphasis on contextual adjustments.
- **G Smoothing**:
  - Top Predictions:*sure: 0.1628, not: 0.1385, sorry: 0.0902, afraid: 0.0783, glad: 0.0783*
  - **Observation**: Excels in identifying frequent patterns, maintaining prediction fluency and relevance.
- **I Smoothing**:
  - Top Predictions:*sure: 0.1463, not: 0.1220, sorry: 0.0732, afraid: 0.0610, glad: 0.0610*
  - **Observation**: Balances probabilities across all N-gram levels, ensuring adaptability in unseen contexts.

Comparative Analysis

Technique	Strengths	Limitations
L Smoothing	Simple and consistent performance across all N values.	Lacks fine-tuned adjustments, leading to generic predictions.
G Smoothing	Excels in handling frequent terms and unseen combinations.	May overemphasize frequent words, reducing diversity.
I Smoothing	Balances fluency and flexibility with dynamic context use.	Computationally more intensive than other methods.

Conclusion

1. **G Smoothing** offers the highest confidence in frequent predictions, making it ideal for scenarios with predictable patterns.
2. **I Smoothing** provides the most adaptable and coherent text generation, excelling in both in-distribution and out-of-distribution contexts.
3. **L Smoothing** is straightforward but less effective in generating diverse or contextually nuanced predictions.
