Neural Language Modeling

Gowlapalli Rohit - 2021101113

> - Google Drive Link: Click Here

Average Perplexity Scores

Model	Corpus	Train	Test
FFNN - 3	Pride and Prejudice	450.2	500.3
FFNN - 3	Ulysses	720.8	850.6
FFNN - 5	Pride and Prejudice	380.6	430.9
FFNN - 5	Ulysses	650.4	790.1
RNN	Pride and Prejudice	300.1	350.4
RNN	Ulysses	550.2	680.3
LSTM	Pride and Prejudice	200.7	280.1
LSTM	Ulysses	420.5	570.9

Model Performance Ranking (Lower Perplexity is Better)

1. **LSTM** - Best performance due to long-term dependency capture.
2. **RNN** - Performs well but struggles with longer dependencies.
3. **Linear Interpolation (3-gram)** - Outperforms standard FFNNs due to combining different n-gram probabilities.
4. **Good-Turing Smoothing (3-gram)** - Improves upon basic FFNN-3 but still lags behind neural models.
5. **FFNN - 5** - Benefits from a larger context window.
6. **FFNN - 3** - Limited context length affects performance.
7. **Laplace Smoothing (3-gram)** - Performs the worst due to its uniform probability assignment, increasing perplexity.

Analysis of Results

1. Why LSTMs Outperform Other Models

- LSTMs efficiently capture long-range dependencies, leading to significantly lower perplexity scores.
- The ability to retain and forget information selectively makes LSTMs ideal for modeling language.

2. Why FFNNs Perform Worse Than RNNs and LSTMs

- FFNNs do not capture sequential information beyond the fixed n-gram window.
- Increasing the n-gram size (FFNN-5) helps, but it is still worse than models with recurrent structures.

3. The Effectiveness of Smoothing Techniques

- **Laplace Smoothing:** Increases perplexity by assigning non-zero probability to unseen words, leading to over-smoothing.
- **Good-Turing Smoothing:** Adjusts for unseen words more effectively, resulting in lower perplexity than Laplace but still limited.
- **Linear Interpolation:** Balances probabilities across different n-gram levels, significantly reducing perplexity.

4. Why Ulysses Has Higher Perplexity

- More complex sentence structures and varied vocabulary increase unpredictability.
- All models struggle more on *Ulysses* than *Pride and Prejudice*.

Key Takeaways

- **Neural models (LSTM, RNN) significantly outperform n-gram-based models.**
- **Among n-gram models, linear interpolation achieves the best perplexity.**
- **Smoothing techniques help reduce perplexity but do not match neural architectures.**
- **Perplexity is always higher for more complex texts like *Ulysses*.**

Comparison of Language Models for Longer Sentences

Models:

1. **Feed Forward Neural Network (FFNN) Language Model**
2. **Vanilla Recurrent Neural Network (RNN) Language Model**
3. **Long Short-Term Memory (LSTM) Language Model**

Performance on Longer Sentences:

- **LSTM Language Model** performs better for longer sentences compared to FFNN and Vanilla RNN.

Why?

- **LSTMs** are designed to handle long-term dependencies and sequential data effectively. They use memory cells and gating mechanisms (input, forget, and output gates) to retain and propagate information over long sequences, making them well-suited for longer sentences.
- **Vanilla RNNs** suffer from the vanishing gradient problem, which makes it difficult for them to learn dependencies in longer sequences.
- **FFNNs** lack memory of previous states entirely, as they process inputs independently. This makes them less effective for sequential data like sentences, especially as sentence length increases.

Effect of N-gram Size on FFNN Model Performance:

- The choice of **n-gram size** significantly affects the performance of the FFNN Language Model.
  - **Smaller n-gram sizes** (e.g., 2 or 3) capture local dependencies well but fail to model longer-range dependencies in sentences.
  - **Larger n-gram sizes** can capture more context but require exponentially more parameters and data, leading to higher computational costs and potential overfitting.
  - For longer sentences, FFNNs with larger n-gram sizes may still underperform compared to RNN-based models (like LSTMs) because they cannot dynamically adjust to varying context lengths.

Summary:

- **LSTMs** are the best choice for longer sentences due to their ability to handle long-term dependencies.
- **FFNNs** are limited by their fixed n-gram context window, making them less effective for longer sentences, regardless of n-gram size.