ANLP ASSIGNMENT 1 REPORT

Perplexity Analysis

- **Neural Network Language Model (5-gram context):**
    - Average Perplexity: **287.5360**
    - This model, using a 5-gram context, shows relatively high perplexity, indicating that it struggles to predict the next word in a sequence accurately. This could be due to the limited context size and the model's inability to capture long-range dependencies effectively.
- **RNN-based Language Model:**
    - Average Perplexity: **226.5984**
    - The RNN-based model improves on the neural network by reducing perplexity. RNNs are better at capturing sequential dependencies, which likely accounts for the performance improvement. However, it still falls short in handling very long sequences due to issues like vanishing gradients.
- **Transformer Decoder-based Language Model:**
    - Average Perplexity: **38.1390**
    - The Transformer model significantly outperforms the other two, with a much lower perplexity score. This suggests that the Transformer can model long-range dependencies more effectively and is better at handling complex language structures.

Reasons for Decreasing Perplexity Across Models

1. **Neural Network Language Model (5-gram context):**
    - **Context Limitation:** The Neural Network Language Model uses a fixed 5-gram context, meaning it can only consider the last five words when predicting the next word. This limited context fails to capture long-range dependencies, leading to higher perplexity. The model is more likely to make inaccurate predictions when the necessary context for the correct prediction extends beyond five words.
    - **Model Simplicity:** This model is typically less complex, with fewer parameters and layers compared to more advanced models like RNNs or Transformers. The simplicity limits its capacity to model complex patterns in language, contributing to higher perplexity

2. **RNN-based Language Model:**
    - **Sequential Processing:** RNNs process sequences of data one step at a time, allowing them to capture dependencies across different time steps. This ability to remember

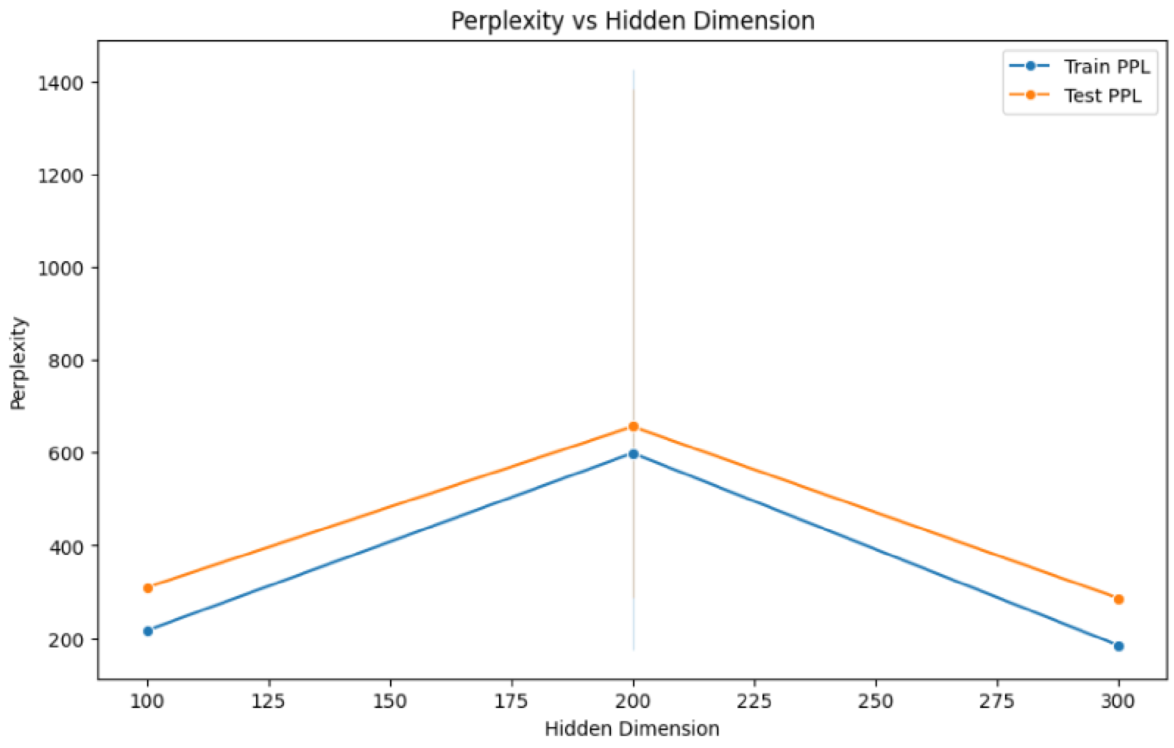
previous words, even beyond a fixed context window, helps in reducing perplexity compared to the 5-gram model. The RNN can theoretically model long-range dependencies, which is why it performs better.

- **Handling Variable-Length Contexts:** Unlike the 5-gram model, RNNs can handle inputs of varying lengths, adjusting dynamically based on the sequence. This flexibility helps improve predictions by using the necessary amount of context, leading to a reduction in perplexity.
- **Challenges with Long Sequences:** Despite improvements, RNNs still struggle with very long sequences due to issues like vanishing gradients, which can cause the model to forget earlier parts of the sequence. This limitation keeps the perplexity higher compared to Transformers.

3. **Transformer Decoder-based Language Model:**
    - **Self-Attention Mechanism:** The Transformer architecture introduces a self-attention mechanism, allowing the model to weigh the importance of different words in the input sequence when predicting the next word. This mechanism helps the model capture long-range dependencies more effectively, as it can focus on any part of the sequence, regardless of its position. This comprehensive understanding of the context significantly lowers perplexity.
    - **Parallel Processing:** Unlike RNNs, which process data sequentially, Transformers process sequences in parallel. This allows the model to learn relationships between words more efficiently, leading to faster convergence and lower perplexity.
    - **Positional Encoding:** Transformers use positional encoding to retain the order of words in a sequence. This allows them to model not just relationships between words but also the specific order of these relationships, leading to more accurate predictions and, consequently, lower perplexity.
    - **Scalability and Depth:** Transformers can be scaled with more layers and larger models, allowing for greater capacity to learn intricate patterns in data. This scalability contributes to better performance, further reducing perplexity.

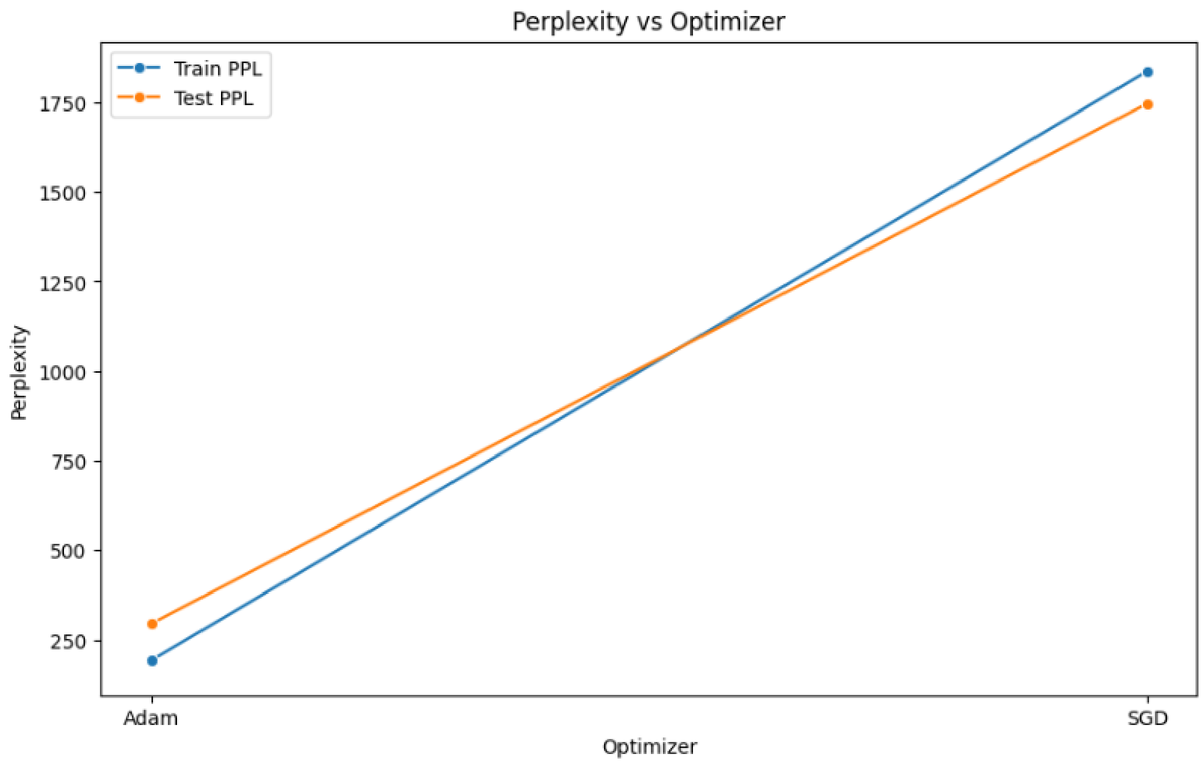
Conclusions:

The decrease in perplexity from the Neural Network Language Model to the RNN-based model and finally to the Transformer Decoder-based model is primarily due to the increasing ability of each subsequent model to capture and utilize context more effectively. While the 5-gram model is limited by a fixed context and simpler architecture, the RNN can process sequences with variable lengths and remember more context. The Transformer goes a step further with its self-attention mechanism, parallel processing, and scalability, allowing it to model complex language structures with much higher accuracy, resulting in the lowest perplexity among the three models.



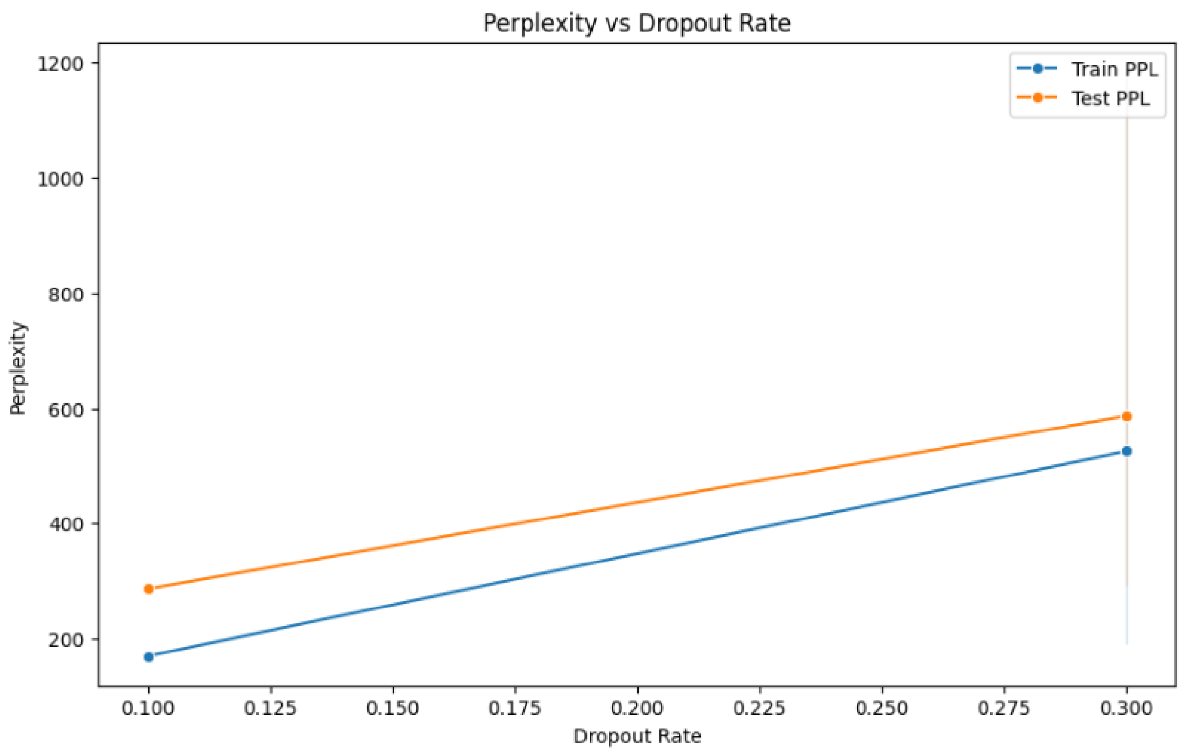
Observation:

- The perplexity is lowest at the boundaries (lower and higher hidden dimensions).
- The model has the worst performance (highest perplexity) at the middle hidden dimension (~200), suggesting a non-linear relationship between hidden dimension size and perplexity.
- A larger hidden dimension (> 200) decreases perplexity on both training and test sets, with the test perplexity being higher than the training perplexity.



Observation:

- Adam achieves much lower perplexity than SGD for both the training and test sets.
- There is a large gap in performance, suggesting that the Adam optimizer significantly outperforms SGD in this context.
- Both training and test perplexities are higher with SGD, meaning the model struggles more to generalize when using this optimizer.



Observation:

- Perplexity increases as the dropout rate increases.
- A lower dropout rate (around 0.1) results in the lowest perplexity, and as the dropout rate increases, the perplexity also increases for both training and test sets.
- The model likely benefits from a lower dropout rate, where the model retains more of its capacity for learning.

Conclusion:

Hidden dimension size, optimizer choice, and dropout rate all affect perplexity significantly. The best results (lowest perplexity) seem to occur when using a larger hidden dimension, the Adam optimizer, and a smaller dropout rate.

Link for pth files:
https://drive.google.com/drive/folders/1e4LEC7Q4R1GGQVJsAgerF788RSBBnIMJ?usp=drive_link