Report on the baseline implementation and experiments

In this progress report, we introduce the baseline n-gram language model (M1) that was developed and evaluated using preprocessed train and test datasets. We go over the perplexing findings from the implementation and offer an explanation of the findings. For both the training and testing phases, we also include performance metrics like runtime and memory utilization.

1. Baseline Language Model Implementation:

The Natural Language Toolkit (NLTK) package and Python were used to create the baseline language model M1, a 2-gram model. A preprocessed test dataset was used to evaluate the model after it had been trained on a training dataset. The text was preprocessed by changing it to lowercase, swapping out specific patterns (such numerals and unidentified words), and tokenizing it with the NLTK word tokenizer.

2. Calculating perplexity:

Perplexity is a popular metric for assessing how well language models function. It evaluates a language model's capacity to forecast a certain dataset. To determine the perplexity of our model on the test dataset, we implemented the NLTK library's perplexity() method.

3. Findings and Interpretation:

We achieved the following results after training our baseline language model M1 on the preprocessed train dataset without using any smoothing techniques and testing it on the preprocessed test dataset.

- Training time: 12.814022541046143 seconds
- Training memory usage: 72,482,816 bytes
- Test perplexity: inf
- Test time: 0.9356515407562256 seconds
- Test memory usage: 11,763,712 bytes

The resources needed to construct the language model using the training dataset are shown by the training time and memory use. Given that smaller perplexity values are desirable, the test perplexity value of infinity indicates that the model is unable to reliably predict the test dataset. This might be because the baseline model doesn't use any smoothing techniques, which would provide unknown n-grams non-zero probabilities and enhance the generalization abilities of the model.

4. Performance Metrics:

The solution has taken into account performance metrics like runtime and memory use. They shed light on the language model's computing needs during the training and testing phases.

Conclusion:

A baseline 2-gram language model (M1) without any smoothing techniques was implemented and evaluated on preprocessed train and test datasets in this progress report. The perplexity results show that the model's prediction capabilities might be significantly enhanced. In subsequent work, we will investigate the use of smoothing techniques like Laplace smoothing or backoff approaches to improve the language model's generalization skills and lessen the ambiguity on the test dataset.