IA-327:
Large Scale Generative Models for NLP and Speech Processing

Lab 1

Student:

Benjamin TERNOT

29/01/2025

Q1: What is the size of the vocabulary?

The vocabulary contains `743` words.

Q2: What do you expect the following probabilities to be? Why?

- The first probability $p(< eos > | [.])$ is expected to be `1`, because the end of sentence token is always the last token in a sentence, so it always follows a point.
- The second probability $p(< eos > | [< bos >])$ is expected to be `0`, because the beginning of sentence token is never followed by the end of sentence token.

Q3: Run this code to produce a plot. What does this plot show? What is on the x-axis, what is on the y-axis?

The plot (cf. *Figure 1*) shows the frequency of n-grams in the dataset. The x-axis represents the rank of the n-grams, and the y-axis represents their corresponding frequency.

Q4: According to this plot, is the ngram assumption justified?

The plot shows that the ngram assumption may not be justified in our dataset, as the frequencies of n-grams do not follow a Zipf's law distribution (we don't see straight lines in the log-log plot).

Q5: Why does this throw an error?

The error occurred because the `NgramCounter` was initialized without smoothing, which caused some n-grams to have a probability of zero.

Q6: Why doesn't the dev perplexity keep decreasing as n increases?

The perplexity doesn't keep decreasing as `n` increases because the model becomes more complex and starts to overfit the training data.

Q7: What are the best values for n and smoothing?

The best values for `n` and smoothing are `2` and `1`, respectively.

Q8: How many parameters does the best ngram model have? Explain how you compute this quantity.

- The best ngram model has `11 831` parameters.
- We can compute it by counting the number of unique n-grams in the training data because each n-gram has a parameter associated with it.

Q9: What do you notice?

We notice that the perplexity on the 'gen' split is higher compared to the other splits, indicating that the model is not generalizing as well. It may be due to differences in the data distribution between the 'gen' split and the other splits, because perplexity on other splits don't show overfitting.

Q10: How does this RNN LM deal with words outside of its vocabulary?

The RNN LM uses a special `<unk>` token to handle out-of-vocabulary words.

Q11: How do the number of parameters of this model compare to the number of parameters of the best ngram model?

The number of parameters of this model is slightly smaller than the number of parameters of the best ngram model. It has `11 615` parameters.

Q12: How does the following compare to the best ngram model?

- The perplexity of the RNN LM is lower than that of the best ngram model on all splits, indicating that the RNN LM performs better than the ngram model.
- The results are significantly better on the 'train', 'dev' and 'test' splits, but also slightly on 'gen' which suggests that the RNN LM generalizes better than the ngram model.

Q13: What else could we have done to further push the dev loss down?

In order to further push the 'dev' loss down, in addition to proposed techniques, we could :

- use regularization techniques such as dropout or L2 regularization. It could be implemented to limit overfitting.
- experimenting with different hyperparameters (learning rate, batch size, number of layers, etc.).

Q14: How does the following compare to the other models?

- The perplexity of the larger RNN LM is quite the same than that of the smaller RNN LM on 'train', 'dev' and 'test' splits.
- However, the perplexity on the 'gen' split is lower, indicating that the larger RNN LM generalize even better as the smaller RNN LM did.

Q15: What do you notice? Is there a phrase that describes this mismatch between distributions?

- We notice that sentences with low perplexity are generally simpler and follow common patterns, while sentences with high perplexity are more complex or contain rare words, making them harder for the models to predict accurately.
- The mismatch between distributions (cf. *Figure 2*) can be described as a *domain shift* or *covariate shift*, where the 'test' data and the 'gen' data come from different distributions, leading to poorer model performance on the 'gen' data that is not well-represented by the training data.

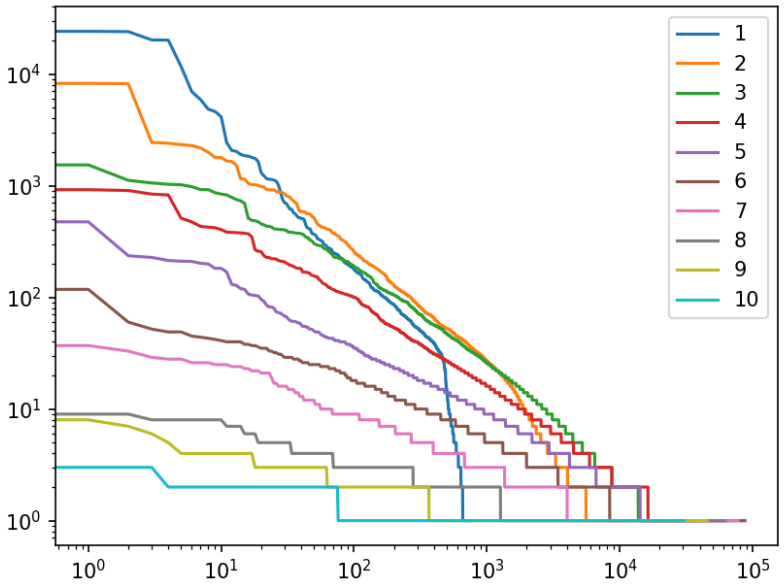

Figure 1: Frequency of n-grams regarding their rank

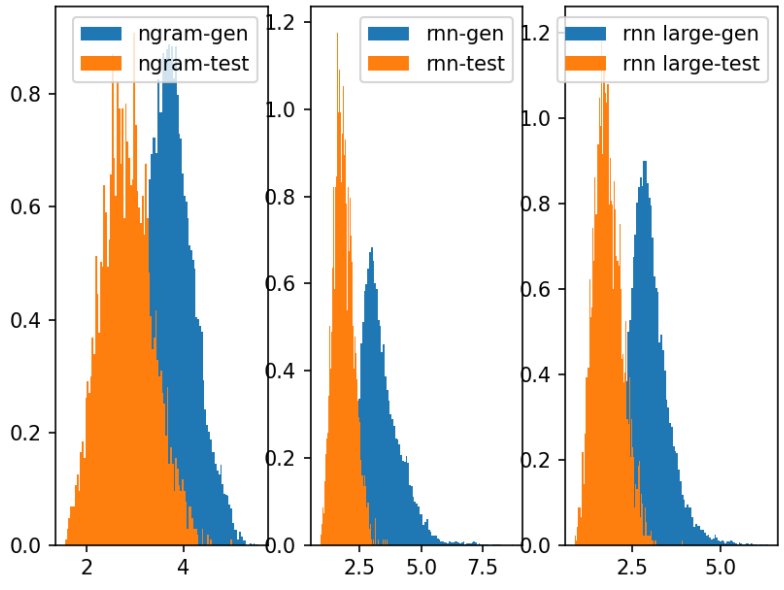

Figure 2: Comparison between perplexity of 'test' and 'gen' data