Assignment-3 Report

Advaith Malladi

2021114005, advaith.malladi@research.iiit.ac.in

Theory Questions

Question 1:

Self-attention is a crucial component in transformer-based models, and it's used to capture dependencies in sequences. It allows the model to weigh the importance of different elements in the input sequence when making predictions. Self-attention works by assigning a weight to each element in the sequence based on its relevance to the current element being processed. This enables the model to consider how each element depends on others, capturing long-range and short-range dependencies effectively. By doing so, self-attention facilitates the modeling of complex relationships within the data, making it a powerful tool for various natural language processing tasks.

Question 2:

Transformers use both word embeddings and positional encodings because they don't inherently have a sense of order or position in the input data. Unlike RNNs or LSTMs, transformers process sequences in parallel, which means they can't naturally distinguish the order of words in a sentence. To address this, positional encodings are added to the word embeddings. These positional encodings are numerical values added to each word's embedding vector to convey their position within the sequence. They are typically designed to be sinusoidal functions of different frequencies to provide unique positional information.

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{model}})$$

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{model}})$$

Hyper Parameter Tuning:

- All hyperparameters model variants were trained for 5 epochs
- encoding_dimensions: 100 heads: 2 batch_size: 20
  - train loss: 2.3
  - val loss: 3.08
- encoding_dimensions: 200 heads: 2 batch size: 20
  - train loss: 1.6
  - val loss: 2.1
- encoding_dimensions: 300 heads: 3 batch size: 20
  - train loss: 0.89
  - val loss: 1.4
- encoding_dimensions: 512 heads: 4 batch size: 3 (as model is big)
  - train loss: 4.7
  - val loss: 5.2

About Batch Size: Batch Size was not something I could experiment too much with as Batch Size for a given model size is determined the capacity and VRAM of my GPU (RTX-3080 ti)

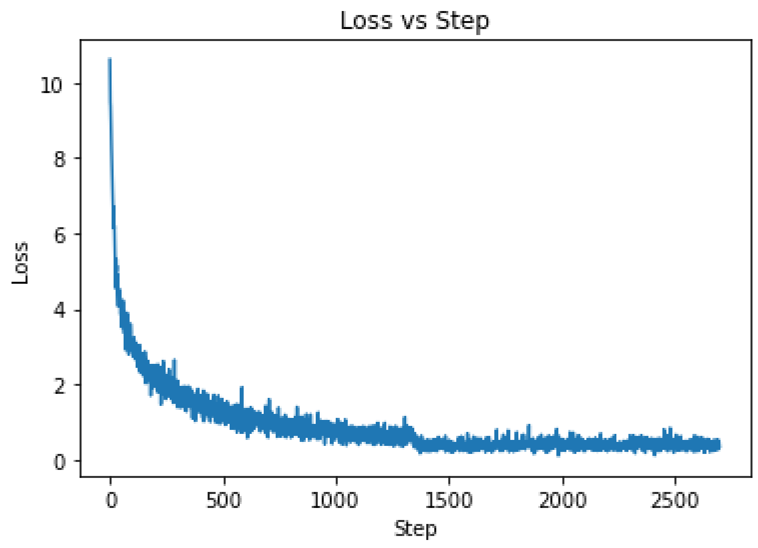
Best Hyper Parameters:

- encoding_dimensions: 300 heads: 3 batch size: 20

- train loss: 0.89
- val loss: 1.4

Using these hyperparameters, the final model training was done and BLEU scores were generated

Here is the Loss vs Step graph for this model



Bleu Scores:

Dev: 0.708686061998218 (887)

Test: 0.701904847588353 (1305)

Train: 0.80283830801331867

Analysis:

- The nature of this task seems to be moderately complex as if we increase the dimensions to 512, the model is having a same time learning. The model does not hit minima at 100 or 200 dimensions also, as mentioned above.
- So, as we observed that 300 dim and 3 heads seems to be the perfect architecture, we can say that the nature of the task is moderately complex.

- After looking at the translations generated by the model, I observed that if the model is able to generate the first word correctly after the <sos> token, the model ends up generating the entire sequence correctly.
- If the model fails to generate the first token correctly, the entire sequence goes wrong.
- The model seems to learn very well on the training data but does give an equally effective performance of the dev and test splits.
- The model performs well on both dev and test splits but it is not up to the train standard. So we can assume that the model slightly overfits.
- The amount of sentences in the train set (30000) is not sufficient for the model to perform english to french translation efficiently in general but the model understands the patterns in the train set.
- As it is giving an almost similar performance in dev and test, we can say that nature of the sentences in dev and test is extremely similar to those in train set as suggested by the bleu scores.