csoc-ig report

Shambhavi

June 2025

SEQ TO SEQ

1 Introduction

The code involves executing sequence-to-sequence model at three levels:
1.Seq2Seq without Attention
2.Add Attention
3.Transformer
The aim is to convert the Hindi language to the English language

2 Preprocessing

I found there are several null values but the percentage is low, so I dropped them. I put my target language English in y and Hindi in x.

3 Learnings

To start sequence-to-sequence modeling first i tokenize the data. For it I did some research on google and found that NLTK(the Natural Language Toolkit) is better for tokenizing. For building Vocabularies I first tried torchtext but it was not working on my notebook because the version of torchtext isn't playing nicely with installed version of torch (PyTorch). So i used collections. It comes bundled with Python, so you don't need to install anything extra.

I used some unique tokens such as 'pad', 'sos', 'eos' and 'unk'. The pad token is for padding. The sos is for start of sentence. The eos is for end of sentence and unk is for unknown tokens. The encode function converts a list of tokens into a list of integers. Then I create a class of EncoderGRU and DecoderGru to encode and decode the data. The class seq2seq takes encoded value and decoded value and runs a forward propagation. With 32 batch size the RAM was getting over so I changed it to 16.

In the first Part where model is built without Attention,the model does not show the right translation.

In part 2 I used Luong Attention in which softmax function is used. I also created a Attention Decoder, it computes attention weights over all encoder outputs, essentially asking "Which words from the input should I pay attention to right now?". Attention gives the decoder full access to all encoder states—like having a detailed transcript rather than a summary. The seq2seq model is bit different from previous one as it contains attention weights. While training, I came across clip grad norm,a technique that helps prevent the gradients from exploding during backpropagation.

In part 3, I used Hugging Face for translation and pre processed the data. I have fine tuned by defining a a variable trainer.This is the high-level training engine. It takes care of batching, evaluating, saving, logging. I have used GPU T4 x2 for this as without this GPU it was running slow.

The overall code took a lot of time to converge.

4 BLEU SCORE(Bilingual Evaluation Understudy)

The BLEU score for seq2seq model without Attention is 0. The BLEU score for seq2seq model with Attention is 0. The BLEU score for Transformer is 14.58 .

The BLEU score of zero is due to failing to convert Hindi to English accurately. For more number of Epochs it will work better.

5 Summary

Transformers are better than seq to seq because Transformers process entire sequences in parallel, thanks to self-attention, making them much faster to train. seq2seq models struggle to remember earlier tokens in long sequences due to vanishing gradients. Transformers use attention to look at the entire sequence at once, so they capture global context more effectively. BLEU score of Transformer is higher than seq to seq. seq to seq with attention is better than seq to seq without attention because With attention, the decoder can look at all encoder outputs and decide which parts of the input are relevant at each step of the output.