Assignment 6: Transformers

CSE 256: Statistical NLP: Spring 202

Part 1 Encoder-Decoder (4 Points)

Q1 How do you think the architecture in Figure1 will perform on long sequences, and why? Consider the amount of information the decoder gets to see about the input sequence (2 points).

The single hidden state at the end of the encoding process will cause information bottleneck. This bottleneck will be likely to make the initial context lost during the encoding process.

Q2 What modification can we make to this architecture to improve its performance on long sequences? Explain how this modification facilitates more accurate translation on long sequences. (2 points)

We add attention to our model, which allows the decoder to have direct access to all the hidden states of the encoder. During each generation, the decoder will calculate the similarity score and build the context vector based on all vector encoder generated for each instances, and concat it to the original vector for the output. This modification improve the accuracy by bypassing the bottleneck issue.

Part 2 Transformers vs RNNs (2 Points)

Q3 Today, Transformers have replaced RNNs in the encoder-decoder architectures. Transformers get rid of the recurrence that is central to RNNs. Instead, they rely completely on attention. In self-attention, each element in the sentence attends to other elements, resulting in context-sensitive representations. Transformers have allowed the NLP community to train on larger datasets than was once possible, what aspect of the Transformer makes this possible in contrast to RNNs? (2 Points)

Transformer model, unlike RNN, is able to process the whole input once, thanks to the attention mechanism. The model is able to calculate any part of the input context and makes parallelization possible, help speed up large dataset processing. RNN, on the contrary, has to process one word at the time.

Part 3 Self-Attention and Variants (6.5 Points)

Q4 Under what conditions might the performance of window attention be at a strong disadvantage in comparison to full self-attention? You can reason using specific examples, or you can provide your argument in more abstract terms. (2 Points)

For window attention, to learn dependencies for a large sequence, we would either have to increase the window size or increase the number of layers, which effectively defeat the purpose of saving memory and process power. Otherwise, we may not be able to obtain good result for tasks that require wider range of tokens.

Q5 Discuss another self-attention pattern (other than full-self attention and window attention). This can be your own novel pattern, we encourage you to think of your own, but you can also consult the literature for help. Explain one strength and one potential limitation of your chosen attention pattern. You may choose to show a visual similar to Figure 2 to illustrate your attention pattern (4.5 Points)

Dilated Sliding Window: instead of window attention, we may use dilation on some window size. For example, we may choose to take even number elements while doing full self-attention. The advantage of this model is that we will be able to cover wider range of the token without changing the existing architecture from window attention. However, skipping tokens will cause lost of information on lower level, which will be propagated later to higher level, and therefore result in unstable or poor model performance.

Part 4 Attention Functions (Bonus: 1.5 Points)

In Part 3, we used dot product attention. Name an alternative attention function, and explain its strengths and weaknesses with respect to dot product attention. (1.5 Points)

Additive Attention: it outperforms dot product attention without scaling for larger values of dimension. At the same time, dot product attention is much faster and more space-efficient in practice, since it mostly done with matrix multiplication, which has been highly optimized in modern computer software.