QUEST 5: Understanding Self-Attention

⚠️ This QUEST requires you to capture a few flags (🚩). If you don't know what that is or how to do it, review the instructions in QUEST 1.

For this QUEST, you will implement a simplified version of the LSTMs for Machine Reading paper (the paper that introduced self-attention) while combining it with a tiny feature from the Attention is All you Need paper (the paper that used both self- and cross-attentions to introduce the Transformer). To finish this QUEST successfully, you are NOT expected to implement your own attention mechanism. In fact, you are strongly encouraged to use PyTorch's attention layer implementation.

🚩 To [capture this flag], look up the relevant nn.Module subclass PyTorch API documentation that can be used for self-attention documentation, and submit the URL for that class documentation.

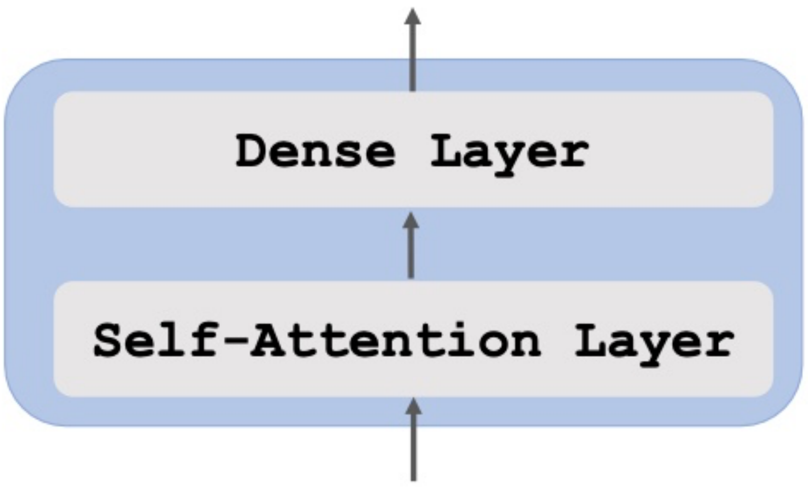
1. To make things easy, I recommend you start with your LSTM implementation from QUEST1 including the starter code got for that.
2. Modify your code to get rid of all the LM pre-training and implement an LSTM-based text classifier trained on the clickbait detection dataset.

🚩 Download and review the dataset using Pandas or your favorite tool, and [capture this flag] by entering the percentage of clickbait headlines present in the data.

🚩 Submit the F1 score you got for your LSTM headline classifier for this problem to [capture this flag].

🚩 To [capture this flag], submit your Colab notebook URL or repo URL for the headline classification. Your notebook/repo URL should clearly point to this work without any additional things. Don't make life difficult for your graders 😀

3. Next, you will implement an encoder block from the transformer model. It's just a self-attention layer and a dense layer. Pick the sizes of the attention layer and the dense layer that makes the most sense here, but your self-attention layer should be able to attend every state of the LSTM (output) you built earlier.



4. Now, connect this encoder block to your LSTM and retrain to produce an LSTM Headline Classifier with Attention.

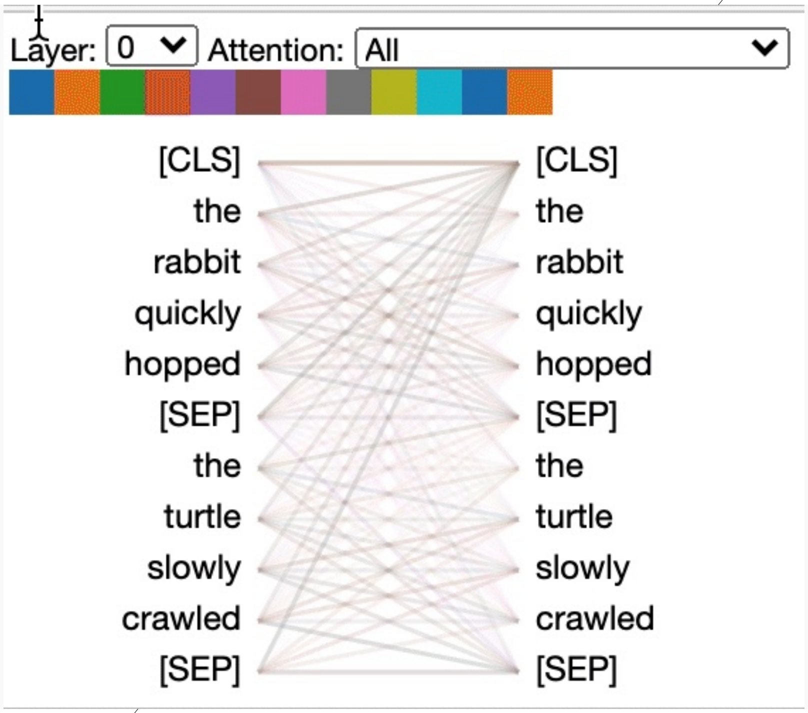
🚩 Submit the F1 score you got for your LSTM headline classifier with attention to [capture this flag].

🚩 To [capture this flag], submit your Colab notebook URL or repo URL for the headline classification. Again, your notebook/repo URL should clearly point to this work without any additional things.

Reflect on the results.

This officially ends this QUEST. While we don't plan to grade any of the following or offer bonuses, you can practice your modeling skills by trying out the following:

1. Adding a residual connection in your encoder block and seeing it if improves performance.
2. Initializing the LSTM in step 4 with the pre-trained weights from the LSTM in step 2 and observing what happens.
3. Visualizing the attention layer using Bertviz. Instead of sentence pair, it will be between the classification heads and the input tokens.


Layer: 0
Attention: All
[CLS]
the
rabbit
quickly
hopped
[SEP]
the
turtle
slowly
crawled
[SEP]
[CLS]
the
rabbit
quickly
hopped
[SEP]
the
turtle
slowly
crawled
[SEP]