CS685 Quiz 1: *Neural language models*
Released 2/20, due 2/27 on Gradescope (please upload a PDF!)
Please answer both questions in 2-4 sentences each. Make sure to also fill out the AI disclosure!

1. **Explain what the "bottleneck" of a recurrent neural network is and how attention provides a way to get around this bottleneck.**

Answer: The "bottleneck" of the recurrent neural network is its last hidden state which is a single fixed length context vector. In a RNN, you force to encode all of the useful information about the prefix in a single vector which is its last hidden state to predict the next word. As the data grows in size, this encoding process becomes increasingly complex and performance of the model will worsen in case of long sequences and very deep recurrent models because:
1. It is not reasonable especially when you have very long , complex prefix sequences.
2. It's hard to imagine encoding all the important semantic and syntactic information about that prefix in one vector. As Ray Mooney said in 2014 - "You can't represent the meaning of a sentence in a BLEEPING vector" . Also, RNNS cannot be parallelized around timestep because one hidden layer depends on the previous layer.

To overcome this bottleneck, an attention-based model uses all hidden states (far away ones as well) which are generated during the encoding phase. The encoder feeds all hidden states to the decoder, where the decoder will compute the correct encoder hidden state to utilize at each step by using a scoring process (using attention weights at each time step). The context vector generated by this process is then concatenated with the decoder's hidden state for that time step and given as input to a feed-forward neural network which generates the final output. Thus, an attention-based model captures the essential information leaving out the irrelevant ones as a way to get around the above mentioned bottleneck.

2. **You are given two language models trained on Wikipedia. One is an unsmoothed 5-gram model (i.e., prefixes are four tokens long), while the other is a fixed-window neural language model with an identical prefix size. Which model's estimate of the conditional probability distribution P(*w* | "*chalkboards flap their wings*") is likely to be more reasonable and why?**

Answer: The fixed-window neural language model's estimate of the conditional probability distribution P(*w* | "*chalkboards flap their wings*") is likely more reasonable because it considers the order of the prefix words. Moreover using a fixed window, it captures the context and relationships of the word aptly in order to predict a word. (Although using a small window size can be unfruitful and using a large window size may be complex and time consuming) Thus it can work well for long term dependencies.

The 5-gram model, on the other hand, does not consider the order of the prefix words. For example, in the 5-gram model, the conditional probability distribution estimated for "chalkboards flap their wings" and "wings flap their chalkboards" will be the same. It cannot capture the context and relationship of words aptly. Moreover, if a word is not present in the training phase (prefix), such words will be incorrectly predicted for a 5-gram model.

Hence, the fixed-window neural language model would be more expressive as well as intuitive and thus, would provide a more reasonable estimate of the conditional probability distribution P(*w* | "*chalkboards flap their wings*").

AI Disclosure

AI1: Did you use any AI assistance to complete this homework? If so, please also specify what AI you used.
Your answer here

No

(only complete the below questions if you answered yes above)

AI2: If you used a large language model to assist you, please paste *all* of the prompts that you used below. Add a separate bullet for each prompt, and specify which problem is associated with which prompt.

- *Your response here*

AI3: (*Free response)* For each problem for which you used assistance, describe your overall experience with the AI. How helpful was it? Did it just directly give you a good answer, or did you have to edit it? Was its output ever obviously wrong or irrelevant? Did you use it to get the answer or check your own answer?

- *Your response here*