CSE 493 G1/ 599 G1
Deep Learning
Autumn 2024 Quiz 3

SOLUTIONS

November 8, 2024

Full Name: _____

UW Net ID: _____

Question		Score
True/False	(4 pts)	
Multiple Choice	(8 pts)	
Short Answer	(8 pts)	
Total	(20 pts)	

Welcome to the CSE 493 G1 Quiz 3!

- The exam is 30 min and is **double-sided**.
- No electronic devices are allowed.

I agree to uphold the University of Washington Student Conduct Code during this exam.

Signature: _____ Date: _____

Good luck!

This page is left blank for scratch work only. DO NOT write your answers here.

1 True / False (4 points) - Recommended 4 Minutes

Fill in the circle next to True or False, or fill in neither. Fill it in completely like this: ●. No explanations are required.

Scoring: Correct answer is worth 1 points.

1.1 One-hot vector encodings of words must be the length of the entire vocabulary while learned embeddings are generally much smaller.

◯ True

◯ False

SOLUTION:
True. One-hot encodings create a vector with length equal to the vocabulary size, with a 1 at the word's position and 0s everywhere else. Learned embeddings map words to dense vectors of much smaller dimension (typically 100-300 dimensions), which is more efficient than the full vocabulary size (which could be 50,000+ words).

1.2 One advantage that RNNs have over transformers is that they can process any sequence length while transformers can only take in sequences of a fixed length.

◯ True

◯ False

SOLUTION:
False. Transformers can handle variable-length sequences

1.3 In self-attention, the attention weights for a given position are computed using only the tokens that come before it in the sequence

◯ True

◯ False

SOLUTION:
False. This statement describes causal/masked self-attention, not standard self-attention.

1.4 You are doing general attention. The sequence length of the queries is N and the sequence length of the keys and values is M. The sum of all of the values in the attention map is M.

◯ True

◯ False

SOLUTION:
False. In general attention, the attention weights for each query sum to 1 (due to softmax). With N queries, the sum of all values in the attention map would be N, not M. The sequence length M of keys/values determines the width of the attention map, but doesn't affect the sum of its values.

2 Multiple Choices (8 points) - Recommended 8 Minutes

Fill in the circle next to the letter(s) of your choice (like this: ●). No explanations are required. **Choose ALL options that apply.**

Each question is worth 4 points and the answer may contain one or more options. Selecting all of the correct options and none of the incorrect options will get full credits. For questions with multiple correct options, each incorrect or missing selection gets a 2-point deduction (up to 4 points).

2.1 Which of the following statements about Recurrent Neural Networks (RNNs) are true?

◯ A: The vanishing gradient problem in RNNs occurs because the same weights are repeatedly multiplied during backpropagation through time

◯ B: LSTMs maintain two hidden states (cell state and hidden state) while basic RNNs have only one hidden state

◯ C: When processing a sequence in an RNN, the computation at each timestep must be done sequentially, making RNNs difficult to parallelize

◯ D: LSTMs help mitigate the vanishing gradient problem by creating direct pathways for gradients to flow through their cell state

SOLUTION:

◯ A: True. During backpropagation through time, gradients flow backwards through the same weight matrices repeatedly. If these weights are small, repeated multiplication causes gradients to vanish. If large, they explode.

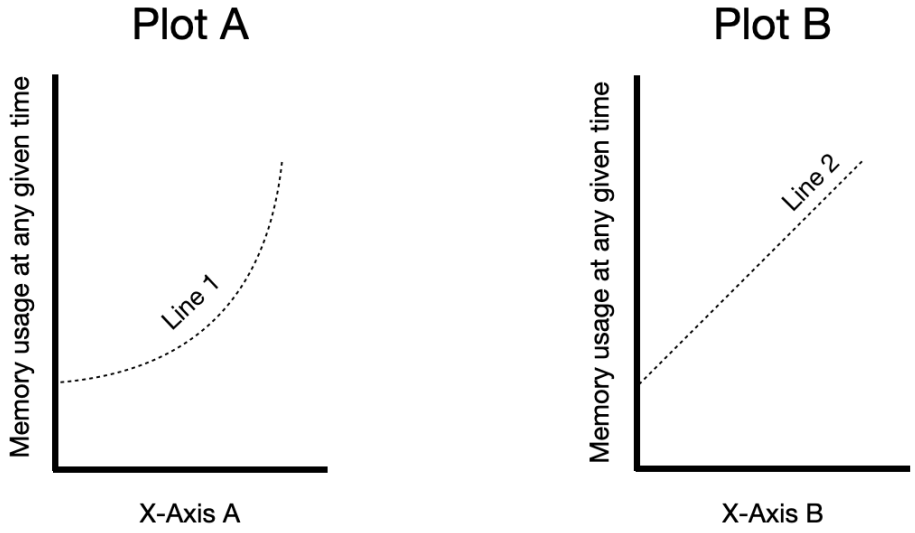
◯ B: True. Basic RNNs have a single hidden state that gets updated at each timestep. LSTMs have both a cell state (long-term memory) and hidden state (short-term memory), with the cell state protected by gates.

◯ C: True. RNNs inherently process sequences one step at a time, with each step depending on the previous hidden state. Unlike transformers which can process all tokens in parallel, RNNs must compute steps sequentially.

◯ D: True. The LSTM cell state acts as a ”memory highway” with additive updates controlled by gates, allowing gradients to flow back through time with less vanishing/exploding. The forget gate helps control what information persists in this pathway.

2.2 We are shown two plots that compare memory usage patterns, but we're missing some key information about what they represent. Here's what we know:

- We have two plots (A and B) of memory usage
- One plot shows memory usage vs sequence length
- One plot shows memory usage vs size of hidden state
- We don't know which plot represents which relationship
- On each plot, we show either a transformer model and/or an RNN model. Each plot could show the same model or each show a different model.



◯ A: X-axis A could only be sequence length

◯ B: There is not enough information to know if X-axis A is sequence length or size of hidden state

◯ C: Line 1 could only be the transformer based model.

◯ D: There is not enough information to know if line 1 is the RNN or Transformer

◯ E: Line 2 could only be the RNN based model.

◯ F: There is not enough information to know if line 2 is the RNN or Transformer

SOLUTION:

◯ A: True. Plot A shows a curve that grows quadratically (due to self-attention), which can only represent transformer's memory usage vs sequence length relationship. The quadratic growth (n^2) is due to the attention matrix size.

◯ B: False. We can determine that X-axis A must be sequence length based on the quadratic growth pattern characteristic of transformer's attention mechanism.

◯ C: True. Line 1 shows quadratic growth which can only represent a transformer's memory usage vs sequence length. RNNs have linear memory growth with sequence length.

◯ D: False. The quadratic growth pattern in Line 1 is uniquely characteristic of transformer models due to their attention mechanism.

◯ E: False. Line 2 shows linear growth, which could be either:

RNN's memory vs sequence length (linear)
Either model's memory vs hidden state size (linear)

◯ F: True. Line 2's linear pattern could represent:

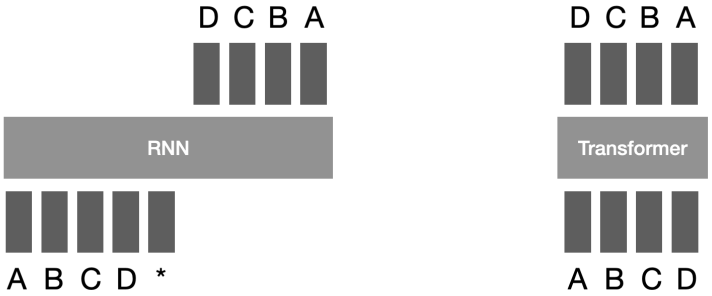
RNN with sequence length
RNN with hidden state size
Transformer with hidden state size
So we cannot definitively determine which model Line 2 represents.

3 Short Answers (8 points) - Recommended 8 Minutes

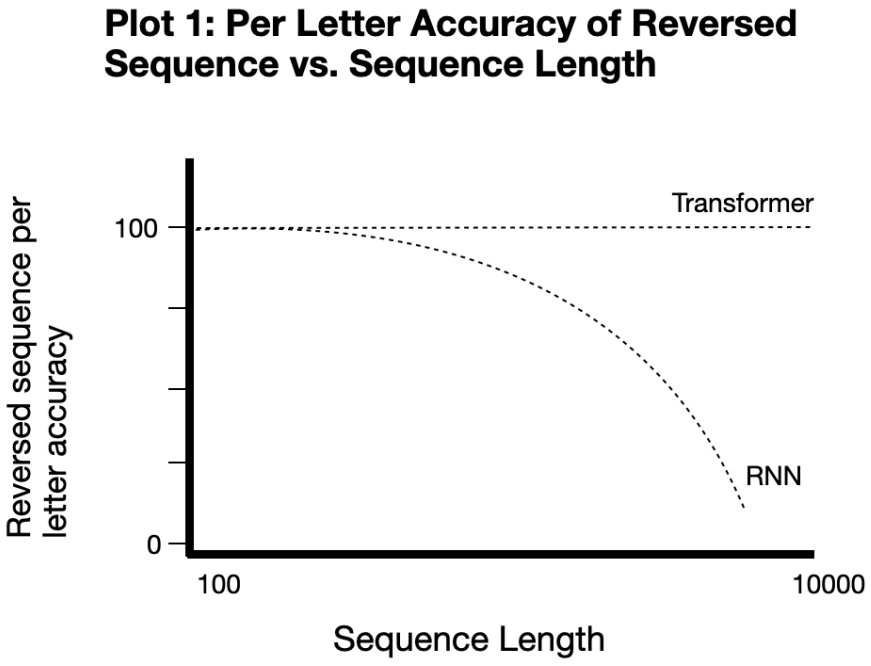
Please make sure to write your answer only in the provided space.

3.1 RNN vs Transformer

You are attempting to build a model to complete the task of reversing a sequence of letters. You are considering two options, a transformer based model and an RNN. For the RNN, you input the sequence in the first x steps, then a token indicating the end of the sequence, and then on next x steps the model outputs the reversed sequence. For the transformer model, you input the sequence of length x with positional embeddings. Then it outputs a sequence of length x, corresponding to the reversed input.



You train both models on a variety of sequences. You monitor the gradients and find there are no vanishing or exploding gradients. You evaluate the models using per letter accuracy (so what percent of the letters the model get correct, rather than what percent of sequences the model gets entirely correct). When you evaluate the models on sequences of different length, you find the following results:



3.1.1 Accuracy (3 points)

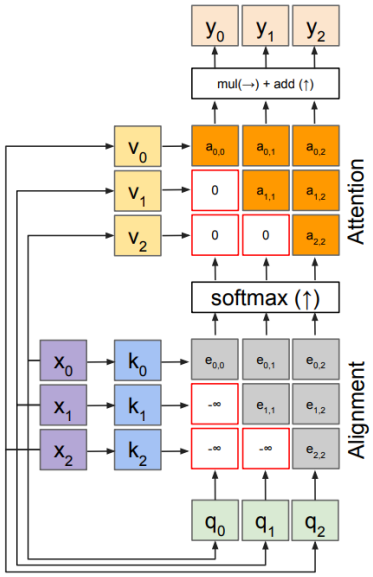
Explain the trends you see in each line in Plot 1.

3.1.2 Smaller RNN (2 points)

You create another RNN with a hidden dimension that is half the size of your first RNN. Plot the accuracy of reversed sequence vs sequence length of this smaller RNN on Plot 1. Label this line "RNN 2".

3.1.3 Masked Attention Transformer (3 points)

You create another transformer. You now use the below masked self-attention, identical to what we discussed in lecture. Plot the accuracy of reversed sequence vs sequence length of this masked transformer on Plot 1. Label this line "masked".



SOLUTION:

1. Accuracy trends explanation:
   - Transformer maintains ∼100% accuracy across all sequence lengths because:
     - Each position can attend to all input positions simultaneously via self-attention
     - No information bottleneck or compression issues
   - RNN performance degrades with longer sequences because:
     - Must compress entire input sequence into fixed-size hidden state
     - Information from early tokens gets compressed/lost when processing long sequences
     - Must accurately store both forward sequence and generate reverse sequence through same hidden state
2. RNN with halved hidden dimension (RNN 2):
   - Shows same pattern as original RNN but:
     - Drops off much earlier
     - This is because smaller hidden state means less capacity to store sequence information
     - Information compression becomes problematic at shorter sequences
3. Masked transformer achieves 50% accuracy because:
   - For first half of output positions: 0% accuracy
     - Can't see future tokens needed for correct reversal
   - For second half of output positions: 100% accuracy
     - Can see all necessary tokens for correct placement
   - Average = (0% + 100%)/2 = 50% accuracy
   - This is constant across all sequence lengths since it's a fundamental limitation of causal masking
   - Example: For input "ABCD":
     - First half (outputting "DC"): Can't see future → wrong
     - Second half (outputting "BA"): Can see all tokens → correct
     - Result: 2/4 = 50% per-letter accuracy

Plot 1: Per Letter Accuracy of Reversed Sequence vs. Sequence Length