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

SOLUTIONS

Feb 16, 2024

Full Name: ____

UW Net ID: ____

Question	Score
True/False (5 pts)	
Multiple Choice (8 pts)	
Short Answer (9 pts)	
Total (22 pts)	

Welcome to the CSE 493 G1 Quiz 3!

- The exam is 20 min and is **double-sided**.
- No electronic devices are allowed.

I understand and agree to uphold the University of Washington Honor Code during this exam.

Signature: ______________________________ Date: __________________

Good luck!

Mean: 14.35
Median: 15.0
Stdev: 4.27

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

1 True / False (5 points) - Recommended 5 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 2 points. To discourage guessing, incorrect answers are worth -1 points. Leaving a question blank will give 0 points.

1.1 Skip-gram algorithm makes use of global co-occurrence statistics.

○ True

○ False

SOLUTION:
False, skip-gram learns word embeddings by predicting the context words given a target word within a local context window.

1.2 Word embeddings generated by Skip-gram model are sparse representations of words in a high-dimensional space.

○ True

○ False

SOLUTION:
False. Word embeddings generated by Skip-gram model are not sparse; they are dense representations in a lower-dimensional space.

1.3 RNNs with attention mechanisms enable the model to focus on specific parts of the input sequence while making predictions, improving its ability to capture long-range dependencies.

○ True

○ False

SOLUTION:
True. RNNs with attention mechanisms dynamically allocate attention to different parts of the input sequence, allowing the model to focus on relevant information during computation.

1.4 Transformers are inherently more memory-efficient than RNNs when processing long sequences due to their ability to capture long-range dependencies in parallel.

○ True

○ False

SOLUTION:
False. While transformers use parallel computation (attention scores for all inputs can be done in parallel), they still require a lot of memory to store attention matrices, making them less memory-efficient for very long sequences compared to RNNs.

1.5 Transformers are primarily used for sequence-to-sequence tasks and are not a good choice for tasks involving non-sequential data.

○ True

○ False

SOLUTION:
False. While transformers excel in sequence-to-sequence tasks, they are also suitable for a wide range of tasks, including image classification, image captioning, speech recognition, etc.

2 Multiple Choices (8 points) - Recommended 6 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 Select all statements that are true about recurrent neural networks.

○ A: Training recurrent neural networks can be affected by the exploding gradient problem.

○ B: Gradient clipping might help if your RNN is troubled by vanishing gradients.

○ C: Unlike standard feedforward networks, recurrent neural networks can learn from sequences of variable length.

○ D: Unlike traditional RNNs, LSTMs do not suffer from the exploding gradient problem.

○ E: None of the above.

SOLUTION:
A and C. (B) gradient clipping helps with exploding gradients not vanishing ones, (D)While LSTMs are designed to mitigate the vanishing gradient problem, they can still experience the exploding gradient problem.

2.2 Which of the following statements are true about batch normalization?

○ A: Batch Normalization is helpful in regularizing neural networks and reducing overfitting.

○ B: BatchNorm parameters are recalculated based on batch statistics during inference.

○ C: BatchNorm helps mitigate the vanishing gradient problem during training by maintaining more stable activations throughout the network.

○ D: Batch normalization mitigates the effects of poor weight initialization and allows the network to initialize our weights to smaller values close to zero.

○ E: BatchNorm requires the computation of mean and standard deviation across the entire dataset before training can begin.

SOLUTION:
A, C, D. (B) BatchNorm parameters learned during the training phase are utilized to normalize activations during inference. (E) BatchNorm computes the mean and standard deviation separately for each mini-batch during training, not across the entire dataset.

Grading Note: Many students were confused as to why A was correct. Refer to Section 3.4 of the BatchNorm paper and this explanation from Ian Goodfellow. Ranjay further elaborated on our discussion board that:

> Batchnorm does multiple things:
>
> - Just like Dropout, it adds random noise into your network. In Dropout, we sample the randomness by setting neurons to zero. In Batchnorm, we sample the randomness indirectly by randomly sampling a minibatch of random inputs.

- If our neural networks were small to moderate, i.e. they had a moderate number of parameters, the optimal amount of dropout or batchnorm would reduce the training accuracy, reducing overfitting. In return, we would produce simpler models that generalize better in validation.
- Since our neural networks are overparameterized with thousands, millions, and even billions of parameters, batchnorm's ability to prevent overfitting is reduced, resulting in no changes in training accuracy or even better training accuracy in some cases because the model doesn't find a local minimum. In return, we still may see the same or sometimes higher validation accuracy because our models are more able to adapt to new images.
- Batchnorm also makes our models more resilient to bad initialization by prevent "dead neurons".
- Batchnorm also normalizes the gradients flowing back, preventing sharp changes to the weights, allowing us to increase the learning rate during training.

Finally, it is worth mentioning that overfitting is not defined as "fitting the training data perfectly". In cases where your validation data is a subset of your training data, fitting the training data perfectly would produce the best classifier. Overfitting occurs when your model fits the training data in return for lower validation performance. Since batchnorm doesn't reduce validation performance in your experiments, I would say that it is overfitting.

3 Short Answers (9 points) - Recommended 9 Minutes

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

Consider the task of sentiment classification in movie reviews using two distinct neural network architectures depicted in Figure 1. “Architecture 1” relies on LSTM, while “Architecture 2” leverages self-attention mechanism **(without using positional encoding or masking)**. Both models are trained to classify movie reviews as either **positive (1)** or **negative (0)**. Both models have similar embedding layers, resulting in identical word representations. These representations undergo further processing, with the resulting tokens passed through sigmoid layers to predict the likelihood of a review being “positive”. Subsequently, these probabilities are rounded to binary decisions (0 or 1) to make one prediction after each new word of the sentence. The final sentiment classification is determined by aggregating all predictions through majority voting. For instance, the LSTM-based model has 5 predictions consisting of four “0”s and one “1”. Therefore the final label is negative (0).

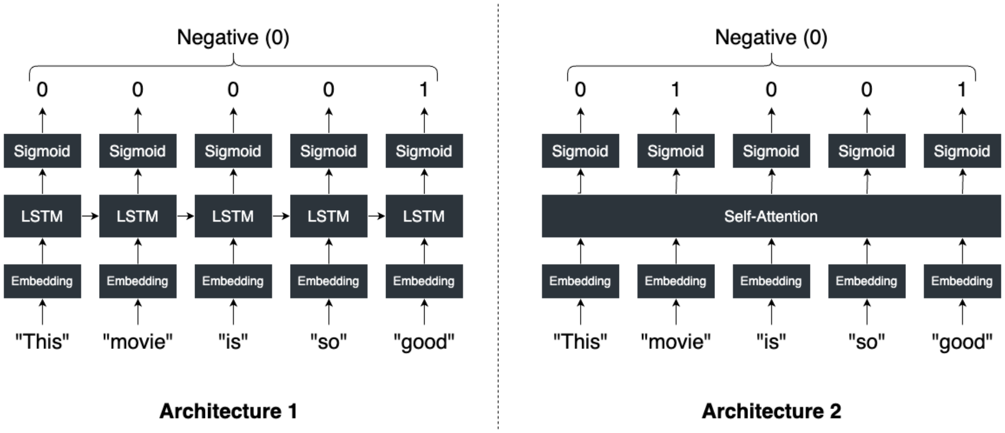

Figure 1: The 2 architectures used for sentiment classification.

Given this setup, consider the following scenarios:

1. The LSTM-based architecture labels the review *“This movie is so good”* as negative due to the output (00001) containing four “0”s and one “1” as shown in Figure 1 (left). Predict how this model will classify the review *“This movie is so bad”*, justifying your answer with appropriate reasoning.

   (a) Positive (1)

   (b) Negative (0)

   (c) Insufficient information

SOLUTION:
(B) Negative. The two reviews have the first four tokens in common *"This movie is so"*. As the LSTM outputs only depend on previous tokens, it will output "0"s again for this review. Therefore, the majority vote will be "0" (negative).

2. Architecture 2, based on self-attention, also categorizes the same review, *"This movie is so good"* as negative because the output (01001) consists of three "0"s and two "1"s as shown in Figure 1 (right). Predict the sentiment classification for the review *"This movie is so bad"* using this model, justifying your answer with appropriate reasoning.

   (a) Positive (1)

   (b) Negative (0)

   (c) Insufficient information

SOLUTION:
(C) Insufficient information. In the attention layer (without masking), each token attends over all other tokens. Therefore, the outputs depend not just on the previous tokens but also the next ones. Since the last token changed from "good" to "bad", the output for all previous tokens (*"This movie is so"*) might change as well. Therefore, it's not possible to predict the final output.

3. The self-attention architecture classifies the following reviews as both positive: *"The movie was not good; it was bad."* and *"The movie was not bad; it was good."* Analyze the underlying reason for this mis-classification and propose a potential solution.

SOLUTION:
Attention is permutation-invariant, and the architecture does not include positional embeddings. Since the reviews have the same tokens and only the permutation has changed, both of the models output the exact same scores, and therefore they are both classified similarly. Adding positional encodings will fix this issue.