Try again once you are ready

Grade received 50% **Latest Submission Grade** 50% **To pass** 80% or higher

Try again

1. Suppose your training examples are sentences (sequences of words). Which of the following refers to the j^{th} word in the i^{th} training example?

1 / 1 point

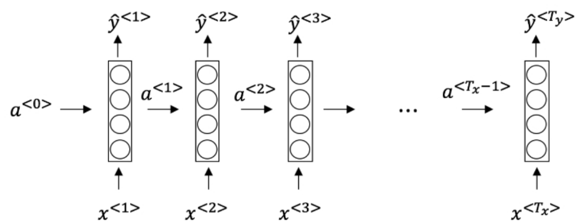
- (selected) $x^{(i)<j>}$
- $x^{<i>(j)}$
- $x^{(j)<i>}$
- $x^{<j>(i)}$

Expand

Correct
We index into the i^{th} row first to get the i^{th} training example (represented by parentheses), then the j^{th} column to get the j^{th} word (represented by the brackets).

2. Consider this RNN:

1 / 1 point



This specific type of architecture is appropriate when:

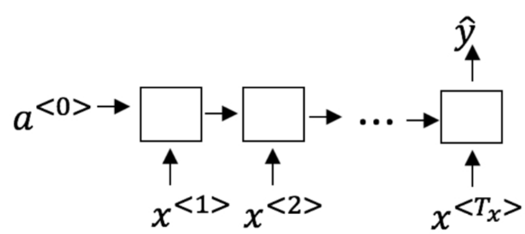
- (selected) $T_x = T_y$
- $T_x < T_y$
- $T_x > T_y$
- $T_x = 1$

Expand

Correct
It is appropriate when every input should have an output.

3. To which of these tasks would you apply a many-to-one RNN architecture? (Check all that apply).

1 / 1 point



- [ ] Speech recognition (input an audio clip and output a transcript)
- [x] Sentiment classification (input a piece of text and output a 0/1 to denote positive or negative sentiment)

  ✓ **Correct**
  Correct!

- [ ] Image classification (input an image and output a label)
- [x] Gender recognition from speech (input an audio clip and output a label indicating the speaker's gender)

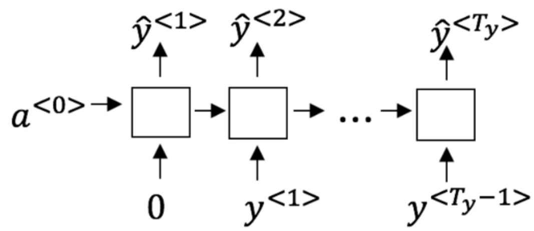
  ✓ **Correct**
  Correct!

Expand

Correct
Great, you got all the right answers.

4. Using this as the training model below, answer the following:

0 / 1 point



True/False: At the t^{th} time step the RNN is estimating $P(y^{<t>})$

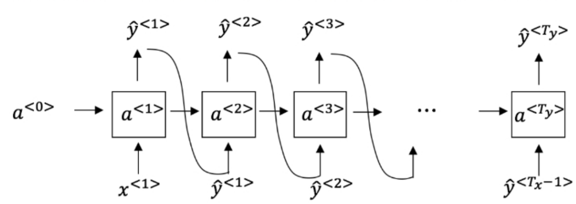
- ◉ True
- ○ False

Expand

Incorrect
In a training model we try to predict the next steps based on the knowledge of all prior steps.

5. You have finished training a language model RNN and are using it to sample random sentences, as follows:

0 / 1 point



What are you doing at each time step t?

- ○ (i) Use the probabilities output by the RNN to pick the highest probability word for that time-step as $\hat{y}^{<t>}$. (ii) Then pass the ground-truth word from the training set to the next time-step.
- ○ (i) Use the probabilities output by the RNN to randomly sample a chosen word for that time-step as $\hat{y}^{<t>}$.(ii) Then pass the ground-truth word from the training set to the next time-step.
- ◉ (i) Use the probabilities output by the RNN to pick the highest probability word for that time-step as $\hat{y}^{<t>}$.(ii) Then pass this selected word to the next time-step.
- ○ (i) Use the probabilities output by the RNN to randomly sample a chosen word for that time-step as $\hat{y}^{<t>}$.(ii) Then pass this selected word to the next time-step.

Expand

Incorrect
The probabilities output by the RNN are not used to pick the highest probability word.

6. You are training an RNN model, and find that your weights and activations are all taking on the value of NaN ("Not a Number"). Which of these is the most likely cause of this problem?

0 / 1 point

- ◉ Vanishing gradient problem.
- ○ Exploding gradient problem.
- ○ The model used the ReLU activation function to compute g(z), where z is too large.
- ○ The model used the Sigmoid activation function to compute g(z), where z is too large.

Expand

Incorrect
Vanishing and exploding gradients are common problems in training RNNs, but in this case, your weights and activations taking on the value of NaN does not imply that you have a vanishing gradient problem.

7. Suppose you are training an LSTM. You have an 80000 word vocabulary, and are using an LSTM with 800-dimensional activations $a^{<t>}$. What is the dimension of Γ_u at each time step?

1 / 1 point

- (selected) 800
- 80000
- 8
- 100

Expand

Correct
Correct, Γ_u is a vector of dimension equal to the number of hidden units in the LSTM.

8. Sarah proposes to simplify the GRU by always removing the Γu. I.e., setting Γu = 0. Ashely proposes to simplify the GRU by removing the Γr. I. e., setting Γr= 1 always. Which of these models is more likely to work without vanishing gradient problems even when trained on very long input sequences? **0 / 1 point**

GRU

$$\tilde{c}^{<t>} = \tanh(W_c[\Gamma_r * c^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma_u = \sigma(W_u[\, c^{<t-1>}, x^{<t>}] + b_u)$$

$$\Gamma_r = \sigma(W_r[\, c^{<t-1>}, x^{<t>}] + b_r)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$$

$$a^{<t>} = c^{<t>}$$

- Sarah's model (removing Γ_u), because if $\Gamma_r \approx 1$ for a timestep, the gradient can propagate back through that timestep without much decay.
- Sarah's model (removing Γ_u), because if $\Gamma_r \approx 0$ for a timestep, the gradient can propagate back through that timestep without much decay.
- Ashely's model (removing Γ_r), because if $\Gamma_u \approx 0$ for a timestep, the gradient can propagate back through that timestep without much decay.
- (selected) Ashely's model (removing Γ_r), because if $\Gamma_u \approx 1$ for a timestep, the gradient can propagate back through that timestep without much decay.

Expand

Incorrect
No. For the signal to backpropagate without vanishing, we need $c^{<t>}$ to be highly dependent on $c^{<t-1>}$.

9. True/False: Using the equations for the GRU and LSTM below the Update Gate and Forget Gate in the LSTM play a different role to Γu and 1- Γu. **1 / 1 point**

GRU

$$\tilde{c}^{<t>} = \tanh(W_c[\Gamma_r * c^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma_u = \sigma(W_u[\, c^{<t-1>}, x^{<t>}] + b_u)$$

$$\Gamma_r = \sigma(W_r[\, c^{<t-1>}, x^{<t>}] + b_r)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$$

$$a^{<t>} = c^{<t>}$$

LSTM

$$\tilde{c}^{<t>} = \tanh(W_c[a^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma_u = \sigma(W_u[\, a^{<t-1>}, x^{<t>}] + b_u)$$

$$\Gamma_f = \sigma(W_f[\, a^{<t-1>}, x^{<t>}] + b_f)$$

$$\Gamma_o = \sigma(W_o[\, a^{<t-1>}, x^{<t>}] + b_o)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$$

$$a^{<t>} = \Gamma_o * \tanh c^{<t>}$$

- (selected) True
- False

Expand

Correct
Correct! Instead of using Γu to compute 1 - Γu, LSTM uses 2 gates (Γu and Γf) to compute the final value of the hidden state. So, Γf is used instead of 1 - Γu.

10. Your mood is heavily dependent on the current and past few days' weather. You've collected data for the past 365 days on the weather, which you represent as a sequence as $x^{<1>}, \ldots, x^{<365>}$. You've also collected data on your mood, which you represent as $y^{<1>}, \ldots, y^{<365>}$. You'd like to build a model to map from $x \rightarrow y$. Should you use a Unidirectional RNN or Bidirectional RNN for this problem? **0 / 1 point**

- Unidirectional RNN, because the value of $y^{<t>}$ depends only on $x^{<t>}$, and not other days' weather.

- [x] Bidirectional RNN, because this allows the prediction of mood on day t to take into account more information.
- [ ] Unidirectional RNN, because the value of $y^{<t>}$ depends only on $x^{<1>}, \dots, x^{<t>}$, but not on $x^{<1>}, \dots, x^{<365>}$.
- [ ] Bidirectional RNN, because this allows backpropagation to compute more accurate gradients.

Expand

Incorrect

Your mood is contingent on the current and past few days' weather, not on the current, past, AND future days' weather.