Your grade: 100%

Your latest: **100%** • Your highest: **100%**

To pass you need at least 80%. We keep your highest score.

Next item →

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

● $x^{(i)<j>}$

○ $x^{<i>(j)}$

○ $x^{(j)<i>}$

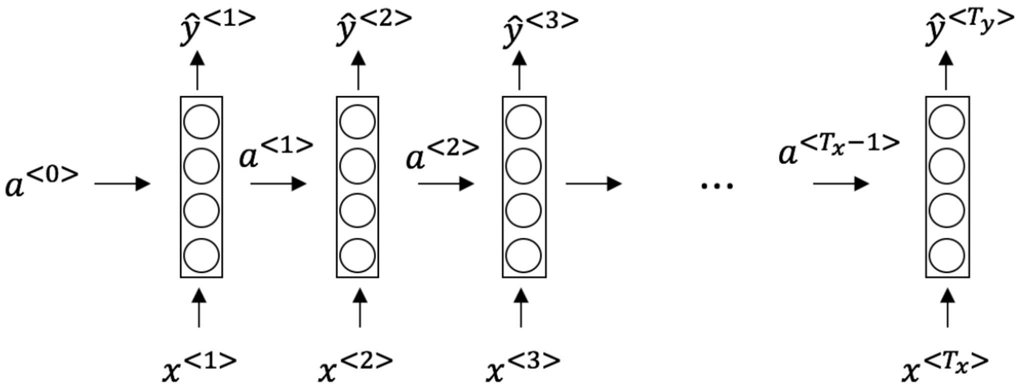
○ $x^{<j>(i)}$

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



True/False: This specific type of architecture is appropriate when Tx>Ty

● False

○ True

✓ **Correct**
Correct! This type of architecture is for applications where the input and output sequence length is the same.

3. Select the two tasks combination that could be addressed by a many-to-one RNN model architecture from the following: **1 / 1 point**

○ **Task 1:** Gender recognition from audio. **Task 2:** Image classification.

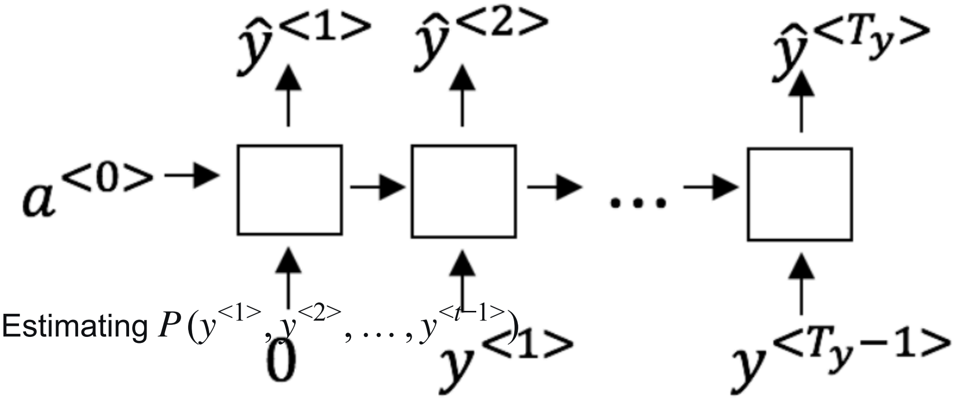
○ **Task 1:** Speech recognition. **Task 2:** Gender recognition from audio.

● **Task 1:** Gender recognition from audio. **Task 2:** Movie review (positive/negative) classification.

○ **Task 1:** Image classification. **Task 2:** Sentiment classification.

✓ **Correct**
Gender recognition from audio and movie review classification are two examples of many-to-one RNN architecture

4. You are training this RNN language model. **1 / 1 point**



○ Estimating $P(y^{<1>}, y^{<2>}, \dots, y^{<t-1>})$

○ Estimating $P(y^{<t>})$

At the t^{th} time step, what is the RNN doing?

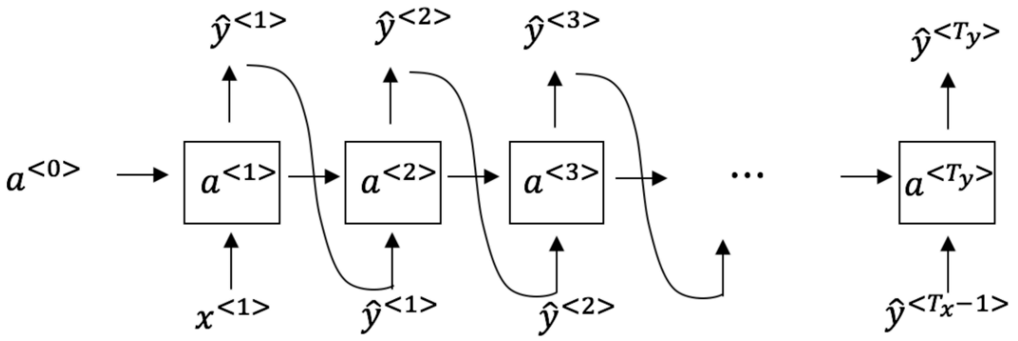
● Estimating $P(y^{<t>} \mid y^{<1>}, y^{<2>}, \dots, y^{<t-1>})$

○ Estimating $P(y^{<t>} \mid y^{<1>}, y^{<2>}, \dots, y^{<t>})$

Correct
Yes, in a language model we try to predict the next step 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:

1 / 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.

✓ **Correct**

6. True/False: If you are training an RNN model, and find that your weights and activations are all taking on the value of NaN (“Not a Number”) then you have a vanishing gradient problem. **1 / 1 point**

● False

○ True

✓ **Correct**
Vanishing and exploding gradients are common problems in training RNNs, but in this case, your weights and activations taking on the value of NaN implies you have an exploding gradient problem.

7. Suppose you are training an LSTM. You have a 10000 word vocabulary, and are using an LSTM with 100-dimensional activations $a^{<t>}$. What is the dimension of Γ_u at each time step? **1 / 1 point**

○ 1

● 100

○ 300

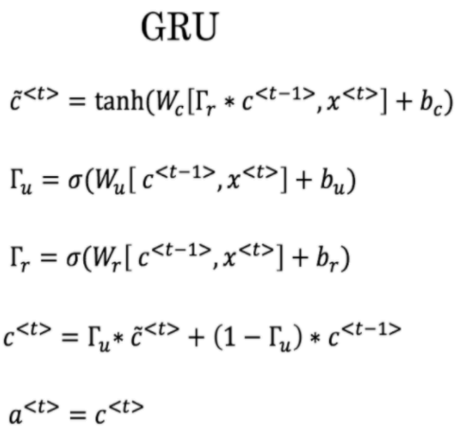
○ 10000

Correct

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

8. Here are the update equations for the GRU.

1 / 1 point



GRU

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

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

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

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

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

Alice proposes to simplify the GRU by always removing the Γ_u. I.e., setting Γ_u = 0. Betty proposes to simplify the GRU by removing the Γ_r. I. e., setting Γ_r = 1 always. Which of these models is more likely to work without vanishing gradient problems even when trained on very long input sequences?

○ Alice'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.

○ Alice'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.

● Betty'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.

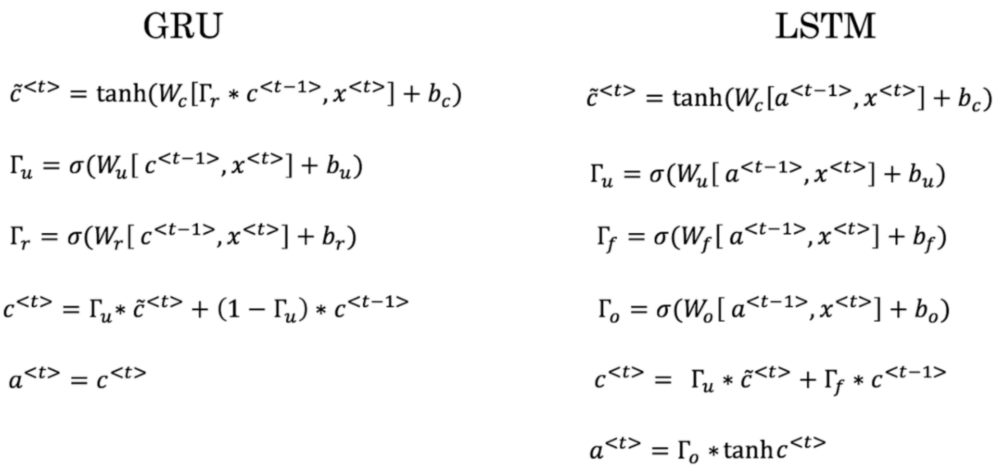
○ Betty's model (removing Γ_r), because if $\Gamma_u \approx 1$ for a timestep, the gradient can propagate back through that timestep without much decay.

Correct

Yes. For the signal to backpropagate without vanishing, we need $c^{<t>}$ to be highly dependent on $c^{<t-1>}$.

9. Here are the equations for the GRU and the LSTM:

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

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

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

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

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

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

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

From these, we can see that the Update Gate and Forget Gate in the LSTM play a role similar to _______ and ______ in the GRU. What should go in the blanks?

● Γ_u and $1 - \Gamma_u$

○ Γ_u and Γ_r

○ $1 - \Gamma_u$ and Γ_u

○ Γ_r and Γ_u

Correct
Yes, correct!

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*→*y*. Should you use a Unidirectional RNN or Bidirectional RNN for this problem?

● Unidirectional RNN, because the value of $y^{<t>}$ depends only on $x^{<1>}, \ldots, x^{<t>}$, but not on $x^{<1>}, \ldots, x^{<365>}$.

○ 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 $x^{<t>}$, and not other days' weather.

○ Bidirectional RNN, because this allows backpropagation to compute more accurate gradients.

Correct