Recurrent Neural Networks

LATEST SUBMISSION GRADE

100%

1.
Question 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

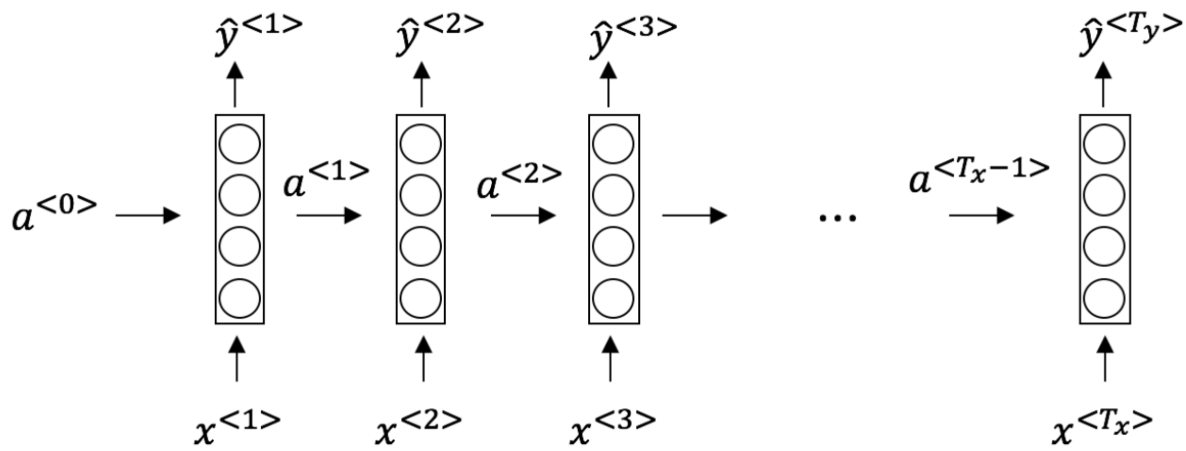
- (selected) **$x^{(i)<j>}$**
- **$x^{<j>(i)}$**
- **$x^{(j)<i>}$**
- **$x^{<i>(j)}$**

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.
Question 2

Consider this RNN:



This specific type of architecture is appropriate when:

1 / 1 point

- [ ] **T_x>T_yT_x >T_y**

- [ ] **T_x<T_yT_x <T_y**

- [x] **T_x=T_yT_x =T_y**

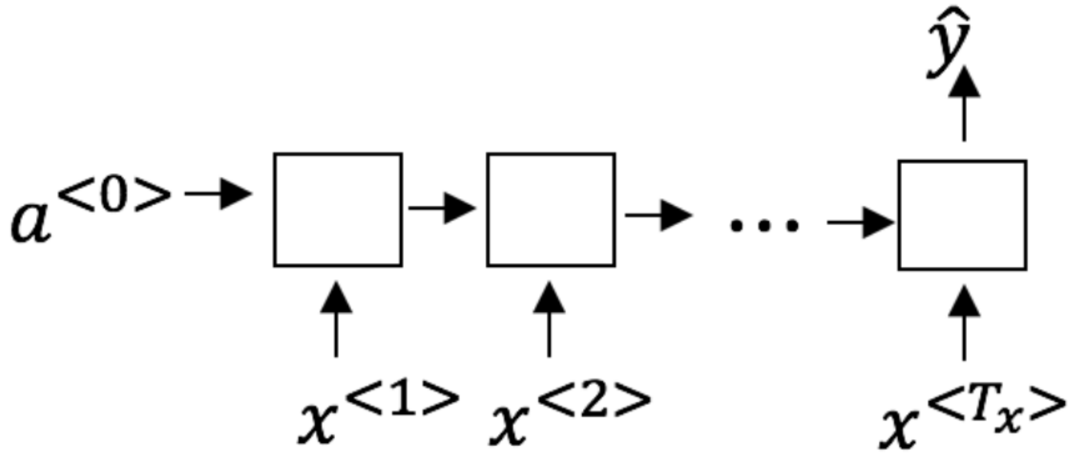
- [ ] **T_x= 1T_x<strong>=1**

Correct

It is appropriate when every input should be matched to an output.

3.
Question 3

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



1 / 1 point

- [ ] Image classification (input an image and output a label)

- [ ] 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

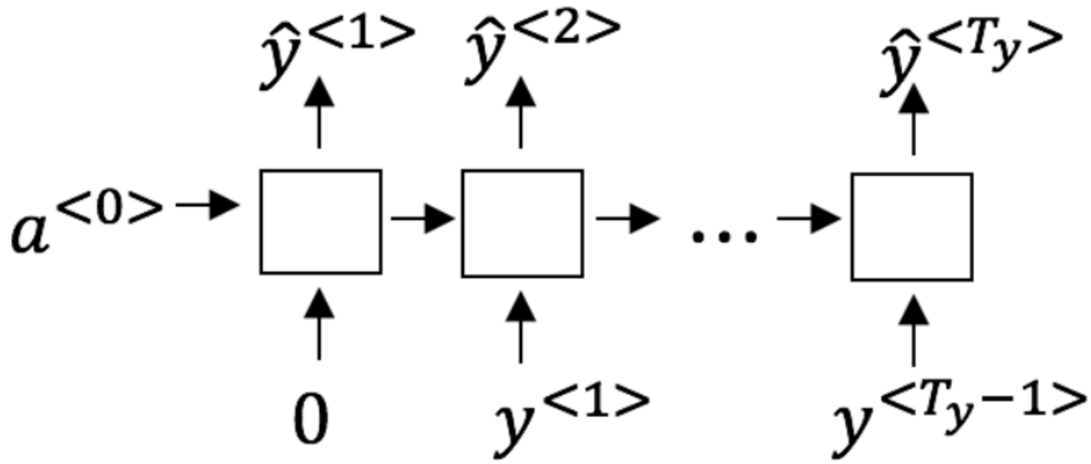
Correct!

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

Correct

Correct!

4.
Question 4
You are training this RNN language model.



At the t^{th} time step, what is the RNN doing? Choose the best answer.

1 / 1 point

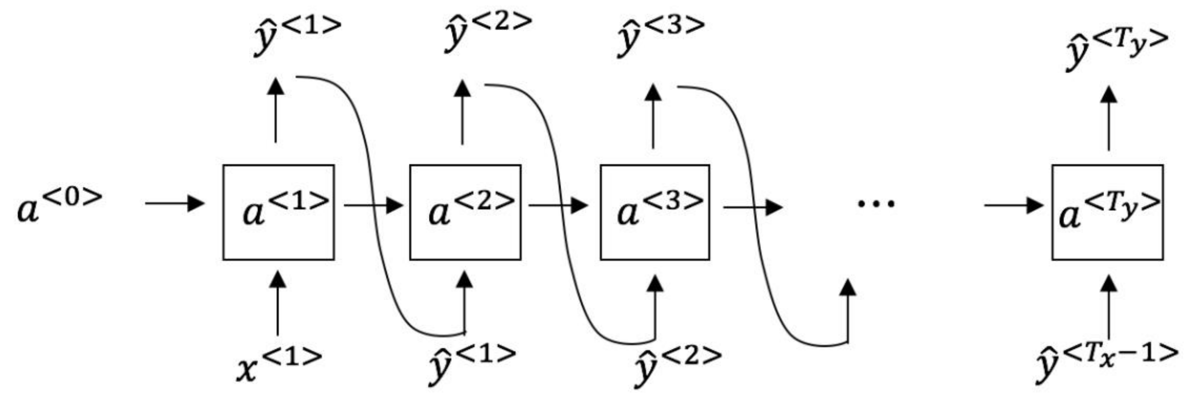
- (selected) Estimating $P(y^{<t>} \mid y^{<1>}, y^{<2>}, \ldots, y^{<t-1>})$
- Estimating $P(y^{<t>} \mid y^{<1>}, y^{<2>}, \ldots, y^{<t>})$
- Estimating $P(y^{<t>})$
- Estimating $P(y^{<1>}, y^{<2>}, \ldots, y^{<t-1>})$

Correct

Yes, in a language model we try to predict the next step based on the knowledge of all prior steps.

5.
Question 5

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



What are you doing at each time step tt?

1 / 1 point

(•)

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

Correct

6.
Question 6

You are training an RNN 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?

1 / 1 point

( )

ReLU activation function g(.) used to compute g(z), where z is too large.

( )

Sigmoid activation function g(.) used to compute g(z), where z is too large.

Vanishing gradient problem.

Exploding gradient problem.

Correct

7.

Question 7

Suppose you are training a 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

10000

100

300

Correct

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

8.

Question 8

Here’re the update equations for the GRU.

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

1 / 1 point

(●) Betty's model (removing \Gamma_rΓ_r), because if \Gamma_u \approx 0$\Gamma_u \approx 0$ for a timestep, the gradient can propagate back through that timestep without much decay.

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

( ) Alice's model (removing \Gamma_uΓ_u), because if \Gamma_r \approx 1$\Gamma_r \approx 1$ for a timestep, the gradient can propagate back through that timestep without much decay.

( ) Alice's model (removing \Gamma_uΓ_u), because if \Gamma_r \approx 0$\Gamma_r \approx 0$ 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>}$c^{<t>}$ to be highly dependent on c^{<t-1>}$c^{<t-1>}$.

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

GRU	LSTM
$\tilde{c}^{<t>} = \tanh(W_c[\Gamma_r * c^{<t-1>}, x^{<t>}] + b_c)$	$\tilde{c}^{<t>} = \tanh(W_c[a^{<t-1>}, x^{<t>}] + b_c)$
$\Gamma_u = \sigma(W_u[\, c^{<t-1>}, x^{<t>}] + b_u)$	$\Gamma_u = \sigma(W_u[\, a^{<t-1>}, x^{<t>}] + b_u)$
$\Gamma_r = \sigma(W_r[\, c^{<t-1>}, x^{<t>}] + b_r)$	$\Gamma_f = \sigma(W_f[\, a^{<t-1>}, x^{<t>}] + b_f)$
$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$	$\Gamma_o = \sigma(W_o[\, a^{<t-1>}, x^{<t>}] + b_o)$
$a^{<t>} = c^{<t>}$	$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$
	$a^{<t>} = \Gamma_o * c^{<t>}$

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?

1 / 1 point

(•) \Gamma_u Γ_u and 1-\Gamma_u $1-\Gamma_u$

( ) \Gamma_r Γ_r and \Gamma_u Γ_u

( ) 1-\Gamma_u $1-\Gamma_u$ and \Gamma_u Γ_u

( ) \Gamma_u Γ_u and \Gamma_r Γ_r

Correct

Yes, correct!

10.
Question 10

You have a pet dog whose 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>}, …, x^{<365>} $x^{<1>},\dots,x^{<365>}$. You've also collected data on your dog's mood, which you represent as y^{<1>}, …, y^{<365>} $y^{<1>},\dots,y^{<365>}$. You'd like to build a model to map from x \rightarrow y $x \rightarrow y$. Should you use a Unidirectional RNN or Bidirectional RNN for this problem?

1 / 1 point

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

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

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

Bidirectional RNN, because this allows the prediction of mood on day t to take into account more information.

Correct

Yes!