Quiz for RNN & Sequence Models

Thursday, May 9, 2019 12:39 AM

1. Question 1

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

- [x] **x^{(i)<j>}*x*(*i*)<*j*>**
- [ ] **x^{<i>(j)}*x*<*i*>(*j*)**
- [ ] **x^{(j)<i>}*x*(*j*)<*i*>**
- [ ] **x^{<j>(i)}*x*<*j*>(*i*)**

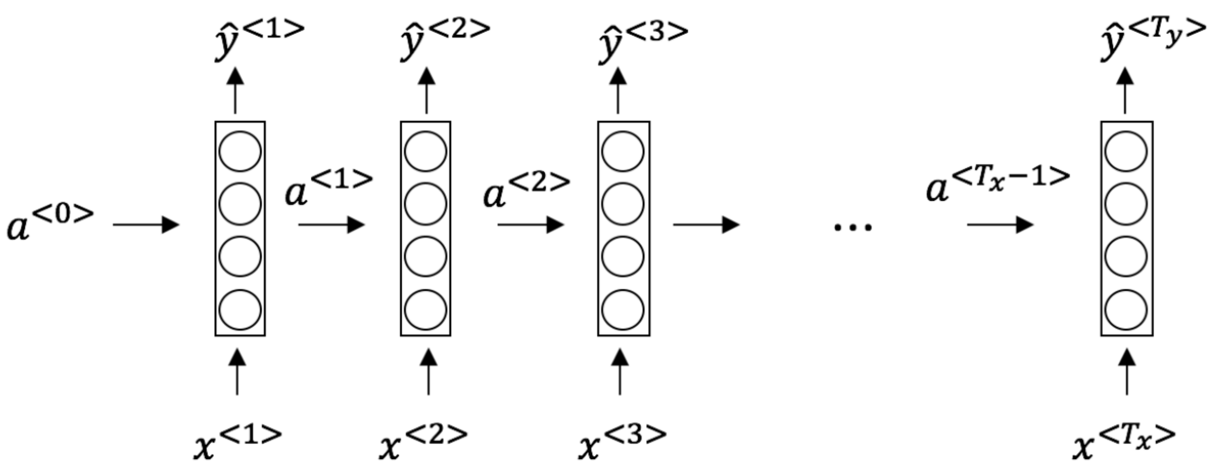
Question 2

1

point

2. Question 2

Consider this RNN:



This specific type of architecture is appropriate when:

- [x] **T_x = T_y*Tx*=*Ty***
- [ ] **T_x < T_y*Tx*<*Ty***
- [ ] **T_x > T_y*Tx*>*Ty***
- [ ] **T_x = 1*Tx*=1**

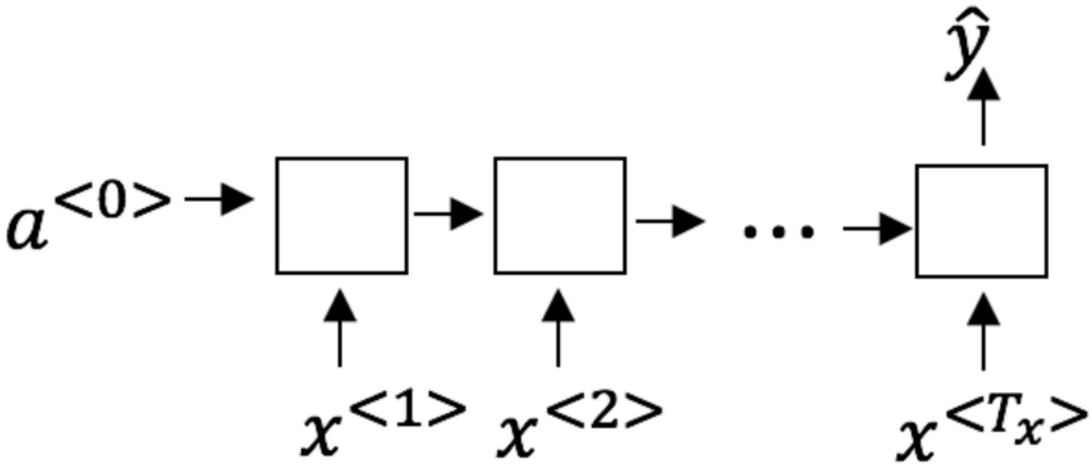
Question 3

1

point

3. Question 3

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



- [ ] 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)
- [ ] 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)

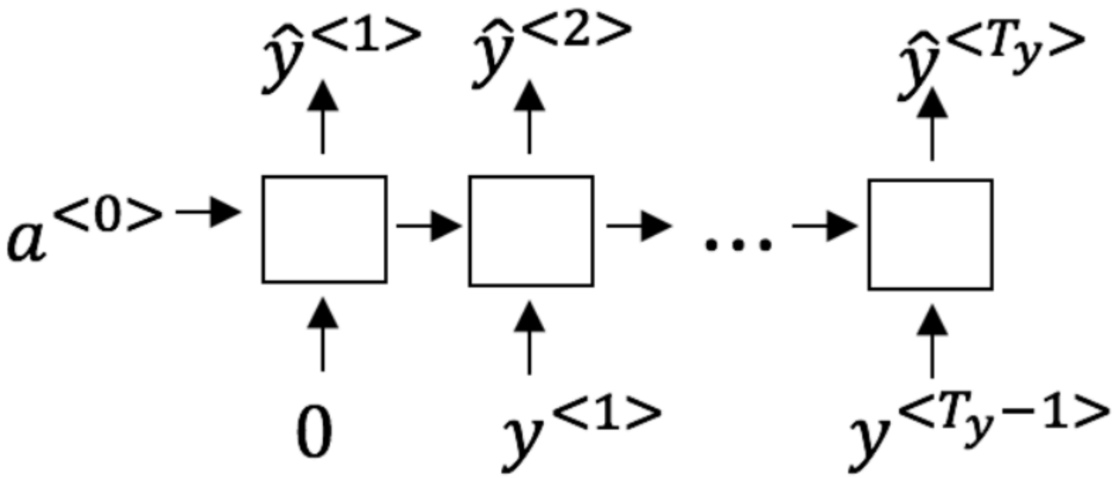
Question 4

1

point

4. Question 4

You are training this RNN language model.



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

- [ ] Estimating $P(y^{<1>},y^{<2>},\ldots,y^{<t-1>})$
- [ ] Estimating P(y^{<t>})*P*(*y*<*t*>)
- [x] Estimating $P(y^{<t>}|y^{<1>},y^{<2>},\ldots,y^{<t-1>})$
- [ ] Estimating $P(y^{<t>}|y^{<1>},y^{<2>},\ldots,y^{<t>})$

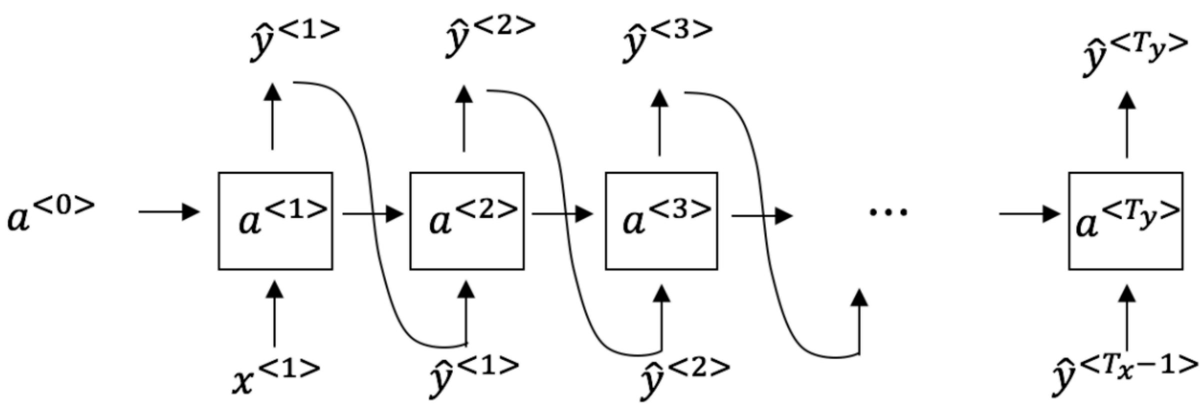
Question 5

1

point

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 *t*t?

- [ ] (i) Use the probabilities output by the RNN to pick the highest probability word for that time-step as $y^{\wedge<t>}$. (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 $y^{\wedge<t>}$. (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 $y^{\wedge<t>}$. (ii) Then pass this selected word to the next time-step.
- [x] (i) Use the probabilities output by the RNN to randomly sample a chosen word for that time-step as $y^{\wedge<t>}$. (ii) Then pass this selected word to the next time-step.

Question 6

1

1

point

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?

- [ ] Vanishing gradient problem.
- [x] Exploding gradient problem.
- [ ] 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.

Question 7

1

point

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>}$a^{<t>}$. What is the dimension of Γ_u at each time step?

- [ ] 1
- [x] 100
- [ ] 300
- [ ] 10000

Question 8

1

point

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 Γ_u. I.e., setting Γ_u = 1. 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.
- [x] 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.

Question 9

1

point

9. Question 9

Here are the equations for the GRU and the LSTM:

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

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

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

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

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

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

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

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 the blanks?

- [x] Γ_u and $1-\Gamma_u$
- [ ] Γ_u and Γ_r
- [ ] $1-\Gamma_u$ and Γ_u
- [ ] Γ_r and Γ_u

Question 10

1

point

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

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

1. Question 1

Suppose you learn a word embedding for a vocabulary of 10000 words. Then the embedding vectors should be 10000 dimensional, so as to capture the full range of variation and meaning in those words.

- [ ] True
- [x] False

Question 2

1

point

2. Question 2

What is t-SNE?

- [ ] A linear transformation that allows us to solve analogies on word vectors
- [x] A non-linear dimensionality reduction technique
- [ ] A supervised learning algorithm for learning word embeddings
- [ ] An open-source sequence modeling library

Question 3

1

point

3. Question 3

Suppose you download a pre-trained word embedding which has been trained on a huge corpus of text. You then use this word embedding to train an RNN for a language task of recognizing if someone is happy from a short snippet of text, using a small training set.

x (input text)	**y (happy?)**
I'm feeling wonderful today!	1
I'm bummed my cat is ill.	0
Really enjoying this!	1

Then even if the word “ecstatic” does not appear in your small training set, your RNN might reasonably be expected to recognize “I’m ecstatic” as deserving a label y = 1*y*=1.

- [x] True
- [ ] False

Question 4

1

point

4. Question 4

Which of these equations do you think should hold for a good word embedding? (Check all that apply)

- [x] e_{boy} - e_{girl} \approx e_{brother} - e_{sister}*eboy−egirl≈ebrother−esister*
- [ ] e_{boy} - e_{girl} \approx e_{sister} - e_{brother}*eboy−egirl≈esister−ebrother*
- [x] e_{boy} - e_{brother} \approx e_{girl} - e_{sister}*eboy−ebrother≈egirl−esister*
- [ ] e_{boy} - e_{brother} \approx e_{sister} - e_{girl}*eboy−ebrother≈esister−egirl*

Question 5

1

point

5. Question 5

Let E*E* be an embedding matrix, and let o_{1234}*o*1234 be a one-hot vector corresponding to word 1234. Then to get the embedding of word 1234, why don’t we call E * o_{1234} *E*∗*o*1234 in Python?

- [x] It is computationally wasteful.
- [ ] The correct formula is E^T* o_{1234}E^T∗o1234

- [ ] The correct formula is E^T * o_{1234}E^T*o_{1234}.
- [ ] This doesn’t handle unknown words (<UNK>).
- [ ] None of the above: calling the Python snippet as described above is fine.

Question 6

1

point

6. Question 6

When learning word embeddings, we create an artificial task of estimating $P(target|context)$. It is okay if we do poorly on this artificial prediction task; the more important by-product of this task is that we learn a useful set of word embeddings.

- [x] True
- [ ] False

Question 7

1

point

7. Question 7

In the word2vec algorithm, you estimate $P(t|c)$, where tt is the target word and cc is a context word. How are tt and cc chosen from the training set? Pick the best answer.

- [ ] cc is the one word that comes immediately before tt.
- [ ] cc is a sequence of several words immediately before tt.
- [x] cc and tt are chosen to be nearby words.
- [ ] cc is the sequence of all the words in the sentence before tt.

Question 8

1

point

8. Question 8

Suppose you have a 10000 word vocabulary, and are learning 500-dimensional word embeddings. The word2vec model uses the following softmax function:

$P(t|c)=e\theta Ttec$

$\sum 10000t\prime=1e\theta Tt\prime ec$

Which of these statements are correct? Check all that apply.

- [x] \theta_tϑt and e_cec are both 500 dimensional vectors.
- [ ] \theta_tϑt and e_cec are both 10000 dimensional vectors.
- [x] \theta_tϑt and e_cec are both trained with an optimization algorithm such as Adam or

- [x] \theta_tϑ*t* and e_c*ec* are both trained with an optimization algorithm such as Adam or gradient descent.
- [ ] After training, we should expect \theta_tϑ*t* to be very close to e_c*ec* when t*t* and c*c* are the same word.

Question 9

1

point

9. Question 9

Suppose you have a 10000 word vocabulary, and are learning 500-dimensional word embeddings.The GloVe model minimizes this objective:

$$\min \sum_{i=1}^{10,000} \sum_{j=1}^{10,000} f(X_{ij})(\theta_i^T e_j + b_i + b'_j - \log X_{ij})^2$$

Which of these statements are correct? Check all that apply.

- [ ] \theta_iϑ*i* and e_j*ej* should be initialized to 0 at the beginning of training.
- [x] \theta_iϑ*i* and e_j*ej* should be initialized randomly at the beginning of training.
- [x] X_{ij}*Xij* is the number of times word i appears in the context of word j.
- [x] The weighting function f(.)*f*(.) must satisfy f(0) = 0*f*(0)=0.

Question 10

1

point

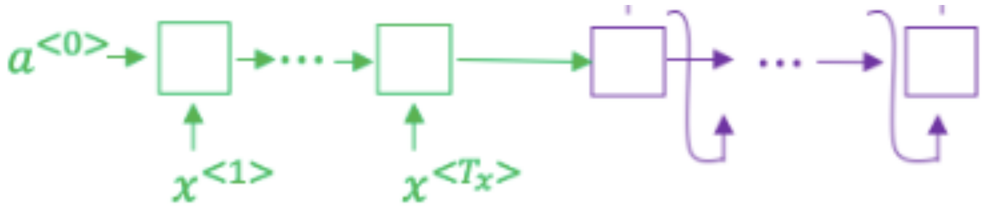
10. Question 10

You have trained word embeddings using a text dataset of m_1*m*1 words. You are considering using these word embeddings for a language task, for which you have a separate labeled dataset of m_2*m*2words. Keeping in mind that using word embeddings is a form of transfer learning, under which of these circumstance would you expect the word embeddings to be helpful?

- [x] m_1*m*1 >> m_2*m*2
- [ ] m_1*m*1 << m_2*m*2

1. Question 1

Consider using this encoder-decoder model for machine translation.



This model is a "conditional language model" in the sense that the encoder portion (shown in green) is modeling the probability of the input sentence *xx*.

- [ ] True
- [x] False

Question 2

1

point

2. Question 2

In beam search, if you increase the beam width B*B*, which of the following would you expect to be true? Check all that apply.

- [x] Beam search will run more slowly.
- [x] Beam search will use up more memory.
- [x] Beam search will generally find better solutions (i.e. do a better job maximizing $P(y|x)$)
- [ ] Beam search will converge after fewer steps.

Question 3

1

point

3. Question 3

In machine translation, if we carry out beam search without using sentence normalization, the algorithm will tend to output overly short translations.

- [x] True
- [ ] False

Question 4

1

point

4. Question 4

Suspose you are building a speech recognition system, which uses an RNN model to map from audio clip *xx* to a text transcript *yy*. Your algorithm uses beam search to try to find the value of *yy*that maximizes $P(y|x)$.

On a dev set example, given an input audio clip, your algorithm outputs the transcript $y^\wedge$= “I’m building an A Eye system in Silly con Valley.”, whereas a human gives a much superior transcript y^* =$y*$= “I’m building an AI system in Silicon Valley.”

According to your model,

$$P(y^\wedge|x)=1.09*10-7$$

$$P(y*|x)=7.21*10-8$$

Would you expect increasing the beam width B to help correct this example?

- [x] No, because $P(y*|x) \leq P(y^\wedge|x)$ indicates the error should be attributed to the RNN rather than to the search algorithm.
- [ ] No, because $P(y*|x) \leq P(y^\wedge|x)$ indicates the error should be attributed to the search algorithm rather than to the RNN.
- [ ] Yes, because $P(y*|x) \leq P(y^\wedge|x)$ indicates the error should be attributed to the RNN rather than to the search algorithm.
- [ ] Yes, because $P(y*|x) \leq P(y^\wedge|x)$ indicates the error should be attributed to the search algorithm rather than to the RNN.

Question 5

1

point

5. Question 5

Continuing the example from Q4, suppose you work on your algorithm for a few more weeks, and now find that for the vast majority of examples on which your algorithm makes a mistake, $P(y*|x) > P(y^\wedge|x)$. This suggest you should focus your attention on improving the search algorithm.

- [x] True.
- [ ] False.

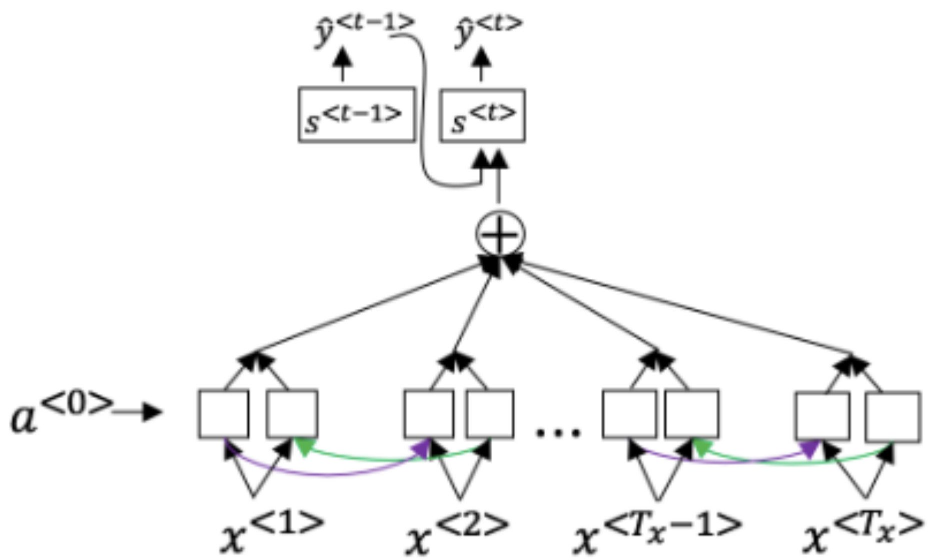
Question 6

1

point

6. Question 6

Consider the attention model for machine translation.



Further, here is the formula for $\alpha^{<t,t'>}$.

$$\alpha^{<t,t'>} = \frac{\exp(e^{<t,t'>})}{\sum_{t'=1}^{T_x} \exp(e^{<t,t'>})}$$

Which of the following statements about $\alpha^{<t,t'>}$ are true? Check all that apply.

- [x] We expect $\alpha^{<t,t'>}$ to be generally larger for values of $a^{<t'>}$ that are highly relevant to the value the network should output for y^{<t>}$y^{<t>}$. (Note the indices in the superscripts.)
- [ ] We expect $\alpha^{<t,t'>}$ to be generally larger for values of a^{<t>}$a^{<t>}$ that are highly relevant to the value the network should output for $y^{<t'>}$. (Note the indices in the superscripts.)
- [ ] $\sum_t \alpha^{<t,t'>} = 1$ (Note the summation is over tt.)
- [x] $\sum_{t'} \alpha^{<t,t'>} = 1$ (Note the summation is over t'.)

Question 7

1

point

7. Question 7

The network learns where to "pay attention" by learning the values $e^{<t,t'>}$, which are computed using a small neural network:

computed using a small neural network:

We can't replace s^{<t-1>}*s*<*t*−1> with s^{<t>}*s*<*t*> as an input to this neural network. This is because s^{<t>}*s*<*t*>depends on α<*t*,*t*′> which in turn depends on e<*t*,*t*′>; so at the time we need to evalute this network, we haven’t computed s^{<t>}*s*<*t*> yet.

- [x] True
- [ ] False

Question 8

1

point

8. Question 8

Compared to the encoder-decoder model shown in Question 1 of this quiz (which does not use an attention mechanism), we expect the attention model to have the greatest advantage when:

- [x] The input sequence length T_x*Tx* is large.
- [ ] The input sequence length T_x*Tx* is small.

Question 9

1

point

9. Question 9

Under the CTC model, identical repeated characters not separated by the “blank” character (_) are collapsed. Under the CTC model, what does the following string collapse to?

__c_oo_o_kk___b_ooooo__oo__kkk

- [ ] cokbok
- [x] cookbook
- [ ] cook book
- [ ] coookkbooooooookkk

Question 10

1

point

10. Question 10

In trigger word detection, x^{<t>}*x*<*t*> is:

- [x] Features of the audio (such as spectrogram features) at time t*t*.
- [ ] The t*t*-th input word, represented as either a one-hot vector or a word embedding.
- [ ] Whether the trigger word is being said at time t*t*.
- [ ] Whether someone has just finished saying the trigger word at time t*t*.