

# Congratulations! You passed!

Next Item





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



False

#### Correct

The dimension of word vectors is usually smaller than the size of the vocabulary. Most common sizes for word vectors ranges between 50 and 400.





What is t-SNE?





## Correct

Yes



An	open-source	sequence	modeling	library
	open source	sequence		y

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.



True

#### Correct

Yes, word vectors empower your model with an incredible ability to generalize. The vector for "ecstatic would contain a positive/happy connotation which will probably make your model classified the sentence as a "1".





1/1 point

4

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

$$e_{boy} - e_{qirl} \approx e_{brother} - e_{sister}$$

## Correct

Yes!

$$e_{boy} - e_{qirl} pprox e_{sister} - e_{brother}$$

Un-selected is correct

$lacksquare e_{boy} - e_{brother} pprox e_{girl} - e_{sister}$
Correct Yes!
$e_{boy} - e_{brother} pprox e_{sister} - e_{girl}$
Un-selected is correct
5. Let $E$ be an embedding matrix, and let $o_{1234}$ be a one-hot vector corresponding to word 1234. Then to get the embedding of word 1234, why don't we call $E*o_{1234}$ in Python?
It is computationally wasteful.
Correct Yes, the element-wise multiplication will be extremely inefficient.
$ \bigcirc  \text{ The correct formula is } E^T*o_{1234}. $
This doesn't handle unknown words ( <unk>).</unk>

None of the above: calling the Python snippet as described above is fine.

 $P(target \mid context)$ . 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.

When learning word embeddings, we create an artificial task of estimating

True

False

Correct



7

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

- c is the one word that comes immediately before t.
- c is the sequence of all the words in the sentence before t.
- $\bigcirc$  c and t are chosen to be nearby words.

#### Correct

c is a sequence of several words immediately before t.



1/1 point

8.

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

$$P(t \mid c) = \frac{e^{\theta_t^T e_c}}{\sum_{t'=1}^{10000} e^{\theta_t^T e_c}}$$

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

 $\theta_t$  and  $e_c$  are both 500 dimensional vectors.

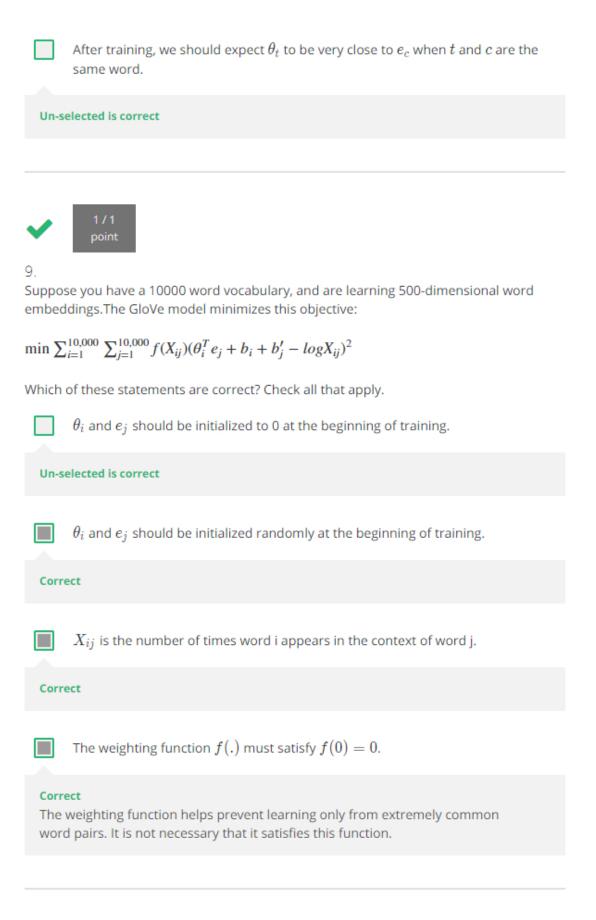
### Correct

 $\theta_t$  and  $e_c$  are both 10000 dimensional vectors.

# Un-selected is correct

 $\theta_t$  and  $e_c$  are both trained with an optimization algorithm such as Adam or gradient descent.

# Correct





10.

considering daning these trond embeddings for dilanguage tash, for timelifud have d

separate labeled dataset of  $m_2$  words. 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?



 $m_1 >> m_2$ 

Correct



 $m_1 \ll m_2$ 





