

1/1 points

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.



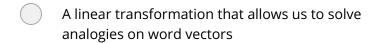
Correct

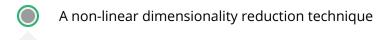


1/1 points

2.

What is t-SNE?





Correct

- A supervised learning algorithm for learning word embeddings
- An open-source sequence modeling library



1/1 points

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



1/1 points

4.

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



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

Correct



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

Un-selected is correct



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

Correct



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

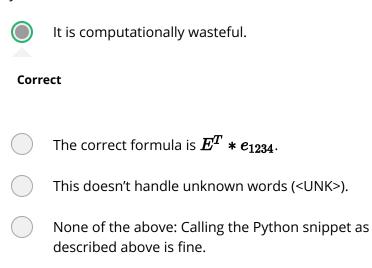
Un-selected is correct



1/1 points

5.

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





1/1 points

6.

When learning word embeddings, we create an artificial task of estimating $P\left(target \mid context\right)$. 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.





1/1 points

7.

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

- $m{c}$ is a sequence of several words immediately before $m{t}$.
- $oldsymbol{c}$ is the one word that comes immediately before $oldsymbol{t}$.
- $m{c}$ is the sequence of all the words in the sentence before $m{t}$.
- igodelimits $oldsymbol{c}$ and $oldsymbol{t}$ are chosen to be nearby words.

Correct



1/1 points

8.

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

$$P\left(t\mid c
ight) = rac{e^{ heta_{t}^{T}e_{c}}}{\sum_{t'=1}^{10000}e^{ heta_{t'}^{T}e_{c}}}$$

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



 $heta_t$ and $extit{e}_c$ are both 500 dimensional vectors.

Correct



Un-selected is correct

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

Correct

After training, we should expect $heta_t$ to be very close to e_c when t and c are the same word.

Un-selected is correct



1/1 points

9.

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

$$\min \ \sum_{i=1}^{10,000} \sum_{j=1}^{10,000} f\left(X_{ij}
ight) \left(heta_i^T e_j + b_i + b_j' - log X_{ij}
ight)^2$$

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

 $heta_i$ and e_j should be initialized to 0 at the beginning of training.

Un-selected is correct

 $heta_i$ and e_j should be initialized randomly at the beginning of training.

Correct



 $oldsymbol{X_{ij}}$ is the number of times word i appears in the context of word j.



Correct



The weighting function f(.) must satisfy f(0) = 0.



Correct

The weighting function helps prevent learning only from extremely common word pairs. It is not necessary that it satisfies this function.



1/1 points

10.

You have trained word embeddings using a text dataset of 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 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



 $m_1 << m_2$

#