

# Natural Language Processing & Word Embeddings

**10/10 points (100.00%)**

Quiz, 10 questions

 **Congratulations! You passed!**[Next Item](#)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.



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.

1 / 1  
points

2.

What is t-SNE?



A linear transformation that allows us to solve analogies on word vectors



A non-linear dimensionality reduction technique

**Correct**

Yes



A supervised learning algorithm for learning word embeddings

☐ An open-source sequence modeling library

## Natural Language Processing & Word Embeddings

10/10 points (100.00%)

Quiz, 10 questions



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**

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".



False



1 / 1  
points

4.

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





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



## Natural Language Processing & Word Embeddings

10/10 points (100.00%)

Quiz, 10 questions

Yes!



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



Un-selected is correct



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



Correct

Yes!



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



Un-selected is correct



1 / 1  
points

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.



The correct formula is  $E^T * o_{1234}$ .



This doesn't handle unknown words (<UNK>).



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

# Natural Language Processing & Word Embeddings

**10/10 points (100.00%)**

Quiz, 10 questions

6.

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



True

**Correct**

False

1 / 1  
points

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 a sequence of several words immediately before  $t$ . $c$  is the sequence of all the words in the sentence before  $t$ . $c$  and  $t$  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:

## Natural Language Processing & Word Embeddings

10/10 points (100.00%)

Quiz, 10 questions

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

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

☐ After training, we should expect  $\theta_t$  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:

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

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

☐  $\theta_i$  and  $e_j$  should be initialized to 0 at the beginning of training.



Un-selected is correct

☐  $\theta_i$  and  $e_j$  should be initialized randomly at the beginning of training.

## Natural Language Processing & Word Embeddings

10/10 points (100.00%)

Quiz, 10 questions

Correct

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

Correct

☐ The weighting function  $f(\cdot)$  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?

☒  $m_1 \gg m_2$

Correct

☐  $m_1 \ll m_2$



# Natural Language Processing & Word Embeddings

**10/10 points (100.00%)**

Quiz, 10 questions