



**Congratulations! You passed!**

TO PASS 80% or higher

Keep Learning

GRADE  
**100%**

## Natural Language Processing & Word Embeddings

LATEST SUBMISSION GRADE

100%

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.

1 / 1 point

- 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.

2. What is t-SNE?

1 / 1 point

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

**Correct**

Yes

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.

1 / 1 point

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

False

✓ **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 classify the sentence as a "1".

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

1 / 1 point

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

✓ **Correct**

Yes!

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

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

✓ **Correct**

Yes!

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

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?

1 / 1 point

- It is computationally wasteful.
- 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.

✓ **Correct**

Yes, the element-wise multiplication will be extremely inefficient.

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

1 / 1 point

- True
- False

✓ **Correct**

7. In the word2vec algorithm, you estimate  $P(t | c)$ , 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.

1 / 1 point

$c$  is the sequence of all the words in the sentence before  $t$ .

$c$  is the sequence of all the words in the sentence after  $t$ .

- $c$  is a sequence of several words immediately before  $t$ .
- $c$  and  $t$  are chosen to be nearby words.
- $c$  is the one word that comes immediately before  $t$ .

✓ Correct

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

1 / 1 point

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

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.
- $\theta_t$  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.

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

1 / 1 point

$$\min \sum_{i=1}^{10,000} \sum_{j=1}^{10,000} f(X_{ij})(\theta_i^T e_j + b_i + b_j - \log X_{ij})^2$$

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

✓ Correct

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

✓ 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.

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 circumstances would you expect the word embeddings to be helpful?

- $m_1 \gg m_2$
- $m_1 \ll m_2$

✓ Correct