[**Skip to Main Content**](https://www.coursera.org/learn/nlp-sequence-models/exam/nIlU0/natural-language-processing-word-embeddings/view-attempt#main)

[[](https://www.coursera.org/)](https://www.coursera.org/)

Explore

Top of Form



Bottom of Form

* **1**
* Vijay Misra

1. [Sequence Models](https://www.coursera.org/learn/nlp-sequence-models/home/welcome)
2. [Week 2](https://www.coursera.org/learn/nlp-sequence-models/home/week/2)
3. Natural Language Processing & Word Embeddings

[**Previous**](https://www.coursera.org/learn/nlp-sequence-models/supplement/mH69b/lectures-in-pdf)[**Next**](https://www.coursera.org/learn/nlp-sequence-models/programming/qgdaj/operations-on-word-vectors-debiasing)

* **Introduction to Word Embeddings**
* **Learning Word Embeddings: Word2vec & GloVe**
* **Applications Using Word Embeddings**
* **Lecture Notes (Optional)**
* **Quiz**

**[Quiz:](https://www.coursera.org/learn/nlp-sequence-models/exam/nIlU0/natural-language-processing-word-embeddings)**[Natural Language Processing & Word Embeddings](https://www.coursera.org/learn/nlp-sequence-models/exam/nIlU0/natural-language-processing-word-embeddings)

[10 questions](https://www.coursera.org/learn/nlp-sequence-models/exam/nIlU0/natural-language-processing-word-embeddings)

* **Programming Assignments**

**QUIZQuiz • 30 MIN30 minutes**

**Natural Language Processing & Word Embeddings**

**Submit your assignment**

**DUE DATE**Jun 27, 11:59 PM PDTJune 27, 11:59 PM PDT

**ATTEMPTS**3 every 8 hours

Try again

**Receive grade**

**TO PASS**80% or higher

**Grade**

95%

View Feedback

We keep your highest score

Natural Language Processing & Word Embeddings

Graded Quiz • 30 min

**Due** Jun 27, 11:59 PM PDT

**Congratulations! You passed!**

**TO PASS**80% or higher

Keep Learning

**GRADE**

95%

**Natural Language Processing & Word Embeddings**

**LATEST SUBMISSION GRADE**

95%

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.

**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 range between 50 and 400.

2.

Question 2

What is t-SNE?

**1 / 1 point**



An open-source sequence modeling library



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



A supervised learning algorithm for learning word embeddings



A non-linear dimensionality reduction technique

**Correct**

Yes

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.

**1 / 1 point**



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.

Question 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\_{brother} \approx e\_{sister} - e\_{girl}*eboy*​−*ebrother*​≈*esister*​−*egirl*​



e\_{boy} - e\_{brother} \approx e\_{girl} - e\_{sister}*eboy*​−*ebrother*​≈*egirl*​−*esister*​

**Correct**

Yes!



e\_{boy} - e\_{girl} \approx e\_{sister} - e\_{brother}*eboy*​−*egirl*​≈*esister*​−*ebrother*​



e\_{boy} - e\_{girl} \approx e\_{brother} - e\_{sister}*eboy*​−*egirl*​≈*ebrother*​−*esister*​

**Correct**

Yes!

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?

**1 / 1 point**



The correct formula is E^T\* o\_{1234}*ET*∗*o*1234​.



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



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



It is computationally wasteful.

**Correct**

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

6.

Question 6

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

**1 / 1 point**



True



False

**Correct**

7.

Question 7

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

**1 / 1 point**



c*c* is a sequence of several words immediately before t*t*.



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



c*c* is the one word that comes immediately before t*t*.



c*c* and t*t* are chosen to be nearby words.

**Correct**

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 \mid c) = \frac{e^{\theta\_t^T e\_c}}{\sum\_{t’=1}^{10000} e^{\theta\_{t’}^Te\_c}}*P*(*t*∣*c*)=∑*t*’=110000​*eθt*’*T*​*ec*​*eθtT*​*ec*​​

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

**0.75 / 1 point**



\theta\_t*θt*​ and e\_c*ec*​ are both 500 dimensional vectors.

**Correct**



\theta\_t*θt*​ and e\_c*ec*​ are both 10000 dimensional vectors.



After training, we should expect \theta\_t*θt*​ to be very close to e\_c*ec*​ when t*t* and c*c* are the same word.



\theta\_t*θt*​ and e\_c*ec*​ are both trained with an optimization algorithm such as Adam or gradient descent.

You didn’t select all the correct answers

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})^2min∑*i*=110,000​∑*j*=110,000​*f*(*Xij*​)(*θiT*​*ej*​+*bi*​+*bj*​’−*logXij*​)2

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

**0.75 / 1 point**



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



X\_{ij}*Xij*​ is the number of times word j appears in the context of word i.

**Correct**



\theta\_i*θi*​ and e\_j*ej*​ should be initialized randomly at the beginning of training.

**Correct**



\theta\_i*θi*​ and e\_j*ej*​ should be initialized to 0 at the beginning of training.

You didn’t select all the correct answers

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*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?

**1 / 1 point**



m\_1*m*1​ >> m\_2*m*2​



m\_1*m*1​ << m\_2*m*2​

**Correct**