Congratulations! You passed!

Grade received 90%

Latest Submission Grade 90%

To pass 80% or higher

Go to next item

1. True/False: Suppose you learn a word embedding for a vocabulary of 20000 words. Then the embedding vectors could be 1000 dimensional, so as to capture the full range of variation and meaning in those words. **1 / 1 point**

○

Expand

✓ **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 1000.

2. True/False: t-SNE is a linear transformation that allows us to solve analogies on word vectors. **1 / 1 point**

Expand

Correct

tr-SNE is a non-linear dimensionality reduction technique.

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?)
Having a great time!	1
I'm sad it’s raining.	0
I’m feeling awesome!	1

Even if the word “wonderful” does not appear in your small training set, what label might be reasonably expected for the input text “I feel wonderful!”?

Expand

✓ **Correct**

Yes, word vectors empower your model with an incredible ability to generalize. The vector for “wonderful” would contain a negative/unhappy 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

Expand

Correct

Great, you got all the right answers.

5. True/False: The most computationally efficient formula for Python to get the embedding of word 1021, if C is an embedding matrix, and o_{1021} is a one-hot vector corresponding to word 1021, is $C^T * o_{1021}$.

0 / 1 point

Expand

⊗ **Incorrect**

No, it is computationally wasteful because the element-wise multiplication will be extremely inefficient.

6. When learning word embeddings, we pick a given word and try to predict its surrounding words or vice versa. **1 / 1 point**

Expand

✓ **Correct**

Word embeddings are learned by picking a given word and trying to predict its surrounding words or vice versa.

7. True/False: In the word2vec algorithm, you estimate *P(t / c)*, where *t* is the target word and *c* is a context word. *t* and *c* are chosen from the training set using *c*as the sequence of all the words in the sentence before *t.*

1 / 1 point

⤢ **Expand**

✓ **Correct**

and are chosen from the training set to be nearby words.

8. Suppose you have a 10000 word vocabulary, and are learning 100-dimensional word embeddings. The word2vec model uses the following softmax function: **1 / 1 point**

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

True/False: After training, we should expect θ_t to be very close to e_c when t and c are the same word.

Expand

Correct

To review this concept watch the lecture.

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

1 / 1 point

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

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

Expand

Correct

Great, you got all the right answers.

10. You have trained word embeddings using a text dataset of t_1 words. You are considering using these word embeddings for a language task, for which you have a separate labeled dataset of t_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

Expand

Correct

Transfer embeddings to new tasks with smaller training sets.