✓ **Congratulations! You passed!**

| Grade received **100%** | Latest Submission Grade **100%** | To pass **80%** or higher | Retake the assignment in **24h 59m** | **Go to next item** |

---

1. Suppose you learn a word embedding for a vocabulary of 10000 words. Then the embedding vectors could be 10000 dimensional, so as to capture the full range of variation and meaning in those words.

    ○ True

    ● False

    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.

    ● False

    ○ True

    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!"?

    ● y=1

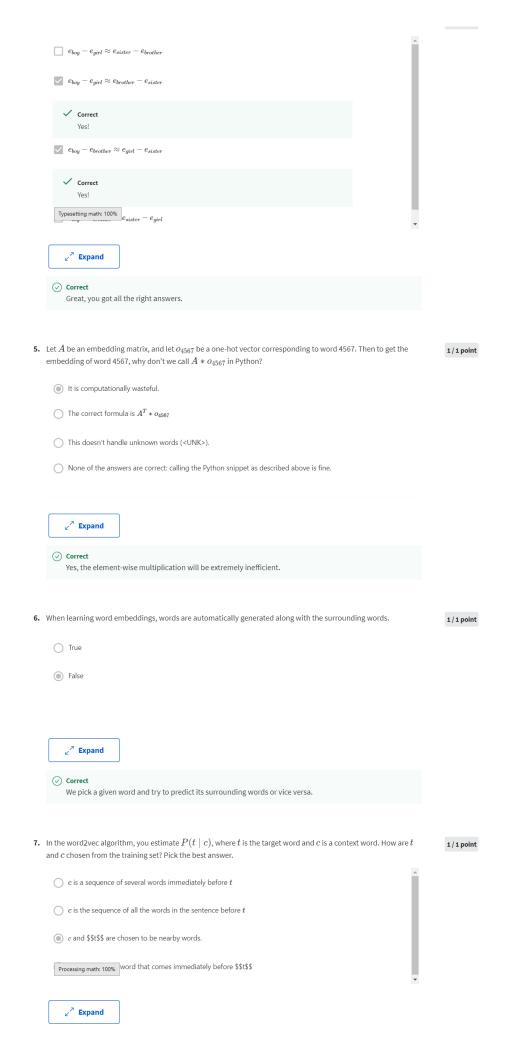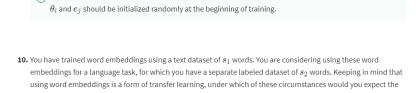   ○ y=0

    [⤢ **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

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

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

✓ **Correct**
Yes!

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

✓ **Correct**
Yes!

Typesetting math: 100% $e_{sister} - e_{girl}$

[Expand]

✓ **Correct**
Great, you got all the right answers.

---

5. Let A be an embedding matrix, and let o_{4567} be a one-hot vector corresponding to word 4567. Then to get the embedding of word 4567, why don't we call $A * o_{4567}$ in Python?

     ⦿ It is computationally wasteful.

     ○ The correct formula is $A^T * o_{4567}$

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

     ○ None of the answers are correct: calling the Python snippet as described above is fine.

1 / 1 point

[Expand]

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

---

6. When learning word embeddings, words are automatically generated along with the surrounding words.

     ○ True

     ⦿ False

1 / 1 point

[Expand]

✓ **Correct**
We pick a given word and try to predict its surrounding words or vice versa.

---

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 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.

     Processing math: 100% word that comes immediately before $$t$$

1 / 1 point

[Expand]

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

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

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

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

☐ θ_t and e_c are both 10000 dimensional vectors.

☑ θ_t and e_c are both 100 dimensional vectors.

> ✓ Correct

☑ θ_t

θ_t and

e_c

e_c are both trained with an optimization algorithm.

⤢ **Expand**

> ⊘ **Correct**
> Great, you got all the right answers.

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

1/1 point

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

True/False: θ_i and e_j should be initialized to 0 at the beginning of training.

○ True

◉ False

⤢ **Expand**

> ⊘ **Correct**
> θ_i and e_j should be initialized randomly at the beginning of training.

10. You have trained word embeddings using a text dataset of s_1 words. You are considering using these word embeddings for a language task, for which you have a separate labeled dataset of s_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

○ $s_1 \ll s_2$

◉ $s_1 \gg s_2$

⤢ **Expand**

> ⊘ **Correct**
> s_1 should transfer to s_2