Skip to Main Content

Explore

- 1
- thadiboyina v m kishore

1. Sequence Models
2. Week 2
3. Natural Language Processing & Word Embeddings

Previous Next

- **Introduction to Word Embeddings**
  - **Video:** LectureWord Representation

    . Duration: 10 minutes10 min
  - **Video:** LectureUsing Word Embeddings

    . Duration: 9 minutes9 min
  - **Video:** LectureProperties of Word Embeddings

    . Duration: 11 minutes11 min
  - **Video:** LectureEmbedding Matrix

. Duration: 5 minutes5 min

- **Learning Word Embeddings: Word2vec & GloVe**
  - **Video:** LectureLearning Word Embeddings

    . Duration: 10 minutes10 min

  - **Video:** LectureWord2Vec

    . Duration: 12 minutes12 min

  - **Video:** LectureNegative Sampling

    . Duration: 11 minutes11 min

  - **Video:** LectureGloVe Word Vectors

    . Duration: 11 minutes11 min

  - **Reading:** GloVe Word Vectors *CORRECTION*

    . Duration: 1 minute1 min

- **Applications Using Word Embeddings**
  - **Video:** LectureSentiment Classification

    . Duration: 7 minutes7 min

- **Video:** LectureDebiasing Word Embeddings

  . Duration: 11 minutes11 min

- **Lecture Notes (Optional)**
  - **Reading:** Lectures in PDF

    . Duration: 1 minute1 min

- **Quiz**
  - **Quiz:** Natural Language Processing & Word Embeddings

    10 questions

- **Programming Assignments**
  - **Programming Assignment:** Operations on Word Vectors - Debiasing

    . Duration: 3 hours3h

  - **Programming Assignment:** Emojify

    . Duration: 3 hours3h

**QUIZQuiz • 30 MIN30 minutes**

# Natural Language Processing & Word Embeddings

**Submit your assignment**

**DUE DATE** Jun 14, 10:59 AM +04 June 14, 10:59 AM +04
**ATTEMPTS** 3 every 8 hours
Try again
**Receive grade**

**TO PASS** 80% or higher
**Grade**
100%
View Feedback

We keep your highest score
Natural Language Processing & Word Embeddings
Graded Quiz • 30 min
**Due** Jun 14, 10:59 AM +04
**Congratulations! You passed!**
**TO PASS** 80% or higher
Keep Learning
**GRADE**
100%

# Natural Language Processing & Word Embeddings

## LATEST SUBMISSION GRADE

100%

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

**A non-linear dimensionality reduction technique**

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

A supervised learning algorithm for learning word embeddings

An open-source sequence modeling library

**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 $y = 1$.

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

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

**Correct**

**Yes!**

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

**Correct**

Yes!

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

5.
Question 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**

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

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 $P(\text{target} \mid \text{context})$ $P(target|context)$. 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**

False

**True**

**Correct**

7.
Question 7
In the word2vec algorithm, you estimate $P(\text{t} \mid \text{c})$ $P(t|c)$, 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:

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

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

**1 / 1 point**

\theta_t$\theta_t$ and e_c$e_c$ are both 10000 dimensional vectors.

**\theta_t$\theta_t$ and e_c$e_c$ are both trained with an optimization algorithm such as Adam or gradient descent.**

**Correct**

**\theta_t$\theta_t$ and e_c$e_c$ are both 500 dimensional vectors.**

**Correct**

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

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

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

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

**1 / 1 point**

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

**Correct**

**The weighting function helps prevent learning only from extremely common word pairs. It is not necessary that it satisfies this function.**

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

**Correct**

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

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

**Correct**

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**