Congratulations! You passed!

Next Item

point

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.

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.

point

What is t-SNE?

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

A supervised learning algorithm for learning word embeddings

Correct

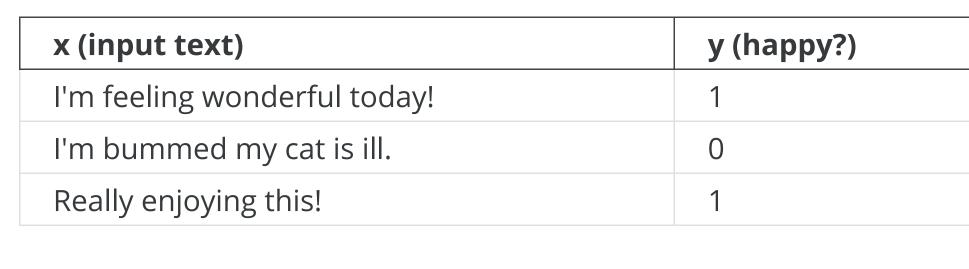
Yes

A non-linear dimensionality reduction technique

An open-source sequence modeling library

point

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.



Suppose you download a pre-trained word embedding which has been trained on a huge

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.

Yes, word vectors empower your model with an incredible ability to generalize.

The vector for "ecstatic would contain a positive/happy connotation which will

True

Correct

False

point

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

probably make your model classified the sentence as a "1".

Correct

Yes!

Yes!

Un-selected is correct

 $e_{boy} - e_{girl} pprox e_{brother} - e_{sister}$

 $e_{boy} - e_{girl} pprox e_{sister} - e_{brother}$

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

 $e_{boy} - e_{brother} pprox e_{sister} - e_{girl}$

Un-selected is correct

point

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? It is computationally wasteful.

Correct

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

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

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

When learning word embeddings, we create an artificial task of estimating

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

 $P(target \mid context)$. It is okay if we do poorly on this artificial prediction task; the more

point

point

point

important by-product of this task is that we learn a useful set of word embeddings. True

Correct

False

context word. How are t and c chosen from the training set? Pick the best answer. c is the sequence of all the words in the sentence before t.

In the word2vec algorithm, you estimate $P(t \mid c)$, where t is the target word and c is a

c and t are chosen to be nearby words.

Correct

c is a sequence of several words immediately before t. c is the one word that comes immediately before t.

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

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

Un-selected is correct

 $heta_t$ and e_c are both 500 dimensional vectors.

Correct

 θ_t and e_c are both 10000 dimensional vectors. **Un-selected is correct**

gradient descent. Correct

 $heta_t$ and e_c are both trained with an optimization algorithm such as Adam or

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

Suppose you have a 10000 word vocabulary, and are learning 500-dimensional word

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

embeddings. The GloVe model minimizes this objective:

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

 θ_i and e_j should be initialized to 0 at the beginning of training. **Un-selected is correct**

 $heta_i$ and e_j should be initialized randomly at the beginning of training.

Correct

Correct

The weighting function helps prevent learning only from extremely common

word pairs. It is not necessary that it satisfies this function.

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

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

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 circumstance would you expect the word embeddings to be helpful? $m_1 >> m_2$

Correct $m_1 \ll m_2$



point