## Congratulations! You passed!

Next Item

2 / 2 points

1. Let's recall how we treated words as one-hot sparse vectors in BOW and dense embeddings in neural networks:

~100k columns

| | good | movie | very | a | did | like |
|---|---|---|---|---|---|---|
| very → | 0 | 0 | **1** | 0 | 0 | 0 |

~300 columns

| | $x_1$ | $x_2$ | $x_3$ |
|---|---|---|---|
| very → | **0.7** | **0.4** | **0.5** |

Choose correct statements below.

- [ ] For **both** word representations we can take a **weighted sum** of vectors corresponding to tokens of any text to obtain good features for this text for further usage in linear model. The **weight** for any token can be an IDF value for that token.

**Correct**
Yes, this is true. For BOW we effectively get bag of TF-IDF values, where TF is a binary variable. Don't forget to normalize these features row-wise!

- [ ] You can replace **word2vec** embeddings with any **random** vectors to get a good features descriptor as a **sum** of vectors corresponding to all text tokens.

**Un-selected is correct**

- [ ] For **both** word representations we can take a **sum** of vectors corresponding to tokens of any text to obtain good features for this text for further usage in linear model.

**Correct**
Yes, this is true. Don't forget to normalize these features row-wise!

- [ ] Linear model on top of a **sum** of neural representations can work faster than on top of BOW.

**Correct**
This is true! We only need to train 300 parameters here. Don't forget to normalize these features row-wise!

2 / 2 points

2. Let's recall 1D convolutions for words:

Word embeddings

| | | |
|---|---|---|
| cat | **0.7** | **0.4** |
| sitting | **0.2** | **-0.1** |
| there | **-0.5** | **0.4** |
| or | **-0.1** | **0.8** |
| here | **-0.5** | **0.3** |

What is the result of 1D convolution + maximum pooling over time for the following kernel **without padding**?

| | |
|---|---|
| 1 | 0 |
| 0 | 1 |

0.6

**Correct Response**
That's it!

1 / 1 points

3. Let's recall 1D convolutions for characters. Choose correct statements.

- [ ] 1D convolutions for characters consume one-hot encoded vectors for characters.

**Correct**
That's right, they are not that long, so this is okay.

- [ ]

One 1D convolutional layer for spotting character 3-grams is enough for solving a practical task.





**Un-selected is correct**

- [ ] 1D convolutions work better than BOW for huge datasets.

**Correct**

This is true.