Prof. Dr. Andreas Hotho, Albin Zehe, Konstantin Kobs
Lehrstuhl für Data Science

13./14.07.2021

8. Assignment in "Machine Learning for Natural Language Processing"

Summer Term 2021

1 General Questions

❓ Something to think about

1. Given two sets of word embeddings E and E', how can you combine these to form a new embedding?

One possibility is similar to what is done in Conceptnet Numberbatch: You could transform E' into a graph by modelling each word as a node and assigning edges the weight of the cosine distance between the two words. Then, you define a new embedding N by minimising the following function:

$$L(N) = \sum_{i=1}^{m} \left[\alpha_i \|n_i - e_i\|^2 + \sum_{(i,j) \in E'} \beta_{ij} \|n_i - n_j\|^2 \right]$$

Here, m is the number of words in the vocabulary, n_i is the embedding of word w_i in N, α_i is a weighting factor for words and β_{ij} is the cosine similarity of w_i and w_j in E'. This will result in an embedding that minimises the distance of each vector n_i to the original vector e_i in E and also minimises the distances between words that are similar in E'.

❓ Something to think about

2. Propose a neural network architecture that would be suited for generating image descriptions, that is, given an image as input, creates a sentence describing the content of the image! Provide reasoning why your architecture is a good choice for this task.

It would be reasonable to use a combination of CNNs and RNNs, for example designing an Encoder-Decoder architecture with a CNN as encoder and an RNN as decoder. The CNN generates a representation of the input image, which is then used as initialisation for the RNN.
This architecture makes sense as CNNs are known to be very good at creating image representations. RNNs are a natural choice for sentence generation, due to their ability to create sequential data.

For example, there is, as usually, a paper by Andrej Karpathy that does just that. See `https://cs.stanford.edu/people/karpathy/deepimagesent/`.

2 Conceptnet Numberbatch

In the lecture, ConceptNet Numberbatch, a set of word embeddings enriched with world knowledge, has been introduced. To optimise the matrix W representing these embeddings, a loss function L has been defined relative to a set of initial embeddings U, $|U| = m$, and a knowledge graph $G = (V, E)$:

$$L(W) = \sum_{i=1}^{m} \left[\alpha_i \|w_i - u_i\|^2 + \sum_{(i,j)\in E} \beta_{i,j} \|w_i - w_j\|^2 \right]$$

Instead of regular gradient descent, ConceptNet Numberbatch uses the following update step to minimise the loss function:

$$W^{k+1} = \text{normalize}\left(\left[\left(SW^k + AW^0 \right)^T (I + A)^{-1} \right]^T \right)$$

Note: There is an error in the original paper which I only noticed after building this assignment. In the paper, the transpose operations are missing, leading to a shape error. The two transpose operations are equivalent to switching the operands of the multiplication. This error probably occurred in the paper because putting the normalisation factor at the end seems more intuitive, but the authors forgot to add the transposes.

1. In the lecture, it was stated that the component $(I + A)^{-1}$ has a "normalising" effect on the formula by preventing words contained in the original embedding from having twice the influence of words not in the original embedding.

   Argue that this is indeed what happens!

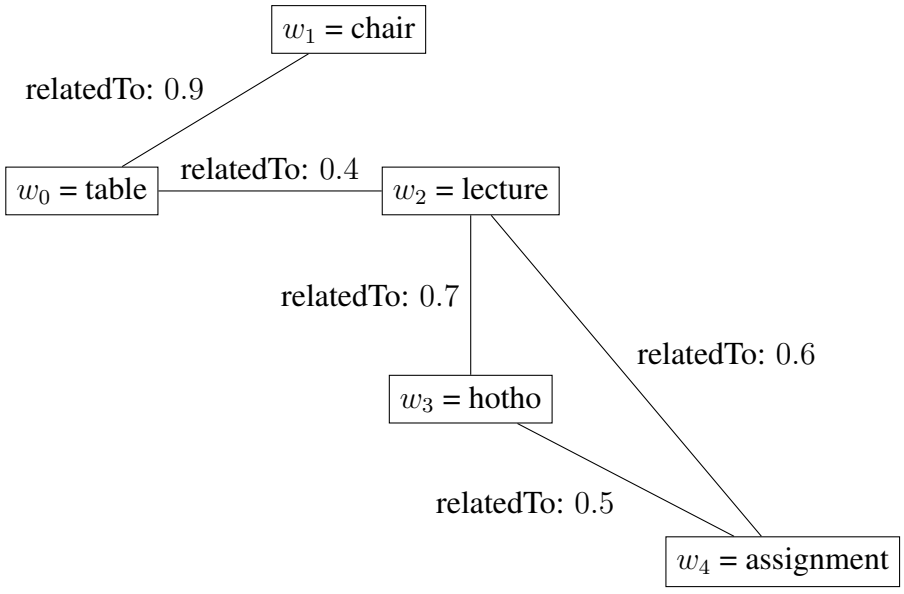
   Some suggested steps:

- What fixed properties does $(I + A)^{-1}$ have?
- Knowing these properties, what happens when the first component gets multiplied with $(I + A)^{-1}$?

You do not have to provide mathematical proof, just an understandable reasoning!

- I is the identity matrix, that is, $I_{i,i} = 1$ and $\forall i \neq j : I_{i,j} = 0$.
- A is a diagonal matrix with $A_{i,i} = \begin{cases} 1, & w_i \in U \\ 0, \end{cases}$ and $\forall i \neq j : A_{i,j} = 0$
- Therefore, $I+A$ is also a diagonal matrix with $(I+A)_{i,i} = \begin{cases} 2, & w_i \in U \\ 1, \end{cases}$ and $\forall i \neq j : (I + A)_{i,j} = 0$
- Inverting a diagonal matrix is done by inverting each of the elements on its diagonal, thus $(I+A)^{-1}_{i,i} = \begin{cases} 0.5, & w_i \in U \\ 1, \end{cases}$ and $\forall i \neq j : (I+A)_{i,j} = 0$
- Multiplying the first component of the update equation with $(I + A)^{-1}$ therefore is equal to weighting all words contained in the original embedding with 0.5!

2. Assume the following knowledge graph:



And the following set of initial word embeddings U:

$u(\text{chair}) = (0.8, 0.6, 0.7)$, $u(\text{table}) = (0.74, 0.7, 0.6)$, $u(\text{lecture}) = (0.1, 0.5, 0.2)$, $u(\text{assignment}) = (0.2, 0.4, 0.18)$. Use this order of embeddings in your matrices to later comply with the sample solution.

Note that there is no embedding for "hotho".

a) What are the values of U, W^0, A and S? You do not have to normalise matrix S.

$$U = \begin{pmatrix} 0.74 & 0.7 & 0.6 \\ 0.8 & 0.6 & 0.7 \\ 0.1 & 0.5 & 0.2 \\ 0.2 & 0.4 & 0.18 \end{pmatrix}$$

$$W^0 = \begin{pmatrix} 0.74 & 0.7 & 0.6 \\ 0.8 & 0.6 & 0.7 \\ 0.1 & 0.5 & 0.2 \\ 0 & 0 & 0 \\ 0.2 & 0.4 & 0.18 \end{pmatrix}$$

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$S = \begin{pmatrix} 1 & 0.9 & 0.4 & 0 & 0 \\ 0.9 & 1 & 0 & 0 & 0 \\ 0.4 & 0 & 1 & 0.7 & 0.6 \\ 0 & 0 & 0.7 & 1 & 0.5 \\ 0 & 0 & 0.6 & 0.5 & 1 \end{pmatrix}$$

b) Use the update formula to calculate W^1!

$$SW^0 = \begin{pmatrix} 1.5 & 1.44 & 1.31 \\ 1.466 & 1.23 & 1.24 \\ 0.516 & 1.02 & 0.548 \\ 0.17 & 0.55 & 0.23 \\ 0.26 & 0.7 & 0.3 \end{pmatrix}$$

$$AW^0 = W^0$$

$$I + A = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{pmatrix}$$

$$(I + A)^{-1} = \begin{pmatrix} 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0.5 \end{pmatrix}$$

$$
\begin{aligned}
W^1 &= \text{normalize}\left(\left((SW^0 + AW^0)^T (I + A)^{-1}\right)^T\right) \\
&= \text{normalize}\left(\begin{pmatrix}
1.12 & 1.07 & 0.955 \\
1.133 & 0.915 & 0.97 \\
0.308 & 0.76 & 0.374 \\
0.17 & 0.55 & 0.23 \\
0.23 & 0.55 & 0.24
\end{pmatrix}\right) \\
&= \begin{pmatrix}
0.35612083 & 0.34022258 & 0.3036566 \\
0.37541418 & 0.30318091 & 0.3214049 \\
0.21359223 & 0.52704577 & 0.259362 \\
0.17894737 & 0.57894737 & 0.24210526 \\
0.2254902 & 0.53921569 & 0.23529412
\end{pmatrix}
\end{aligned}
$$

c) Calculate $L(W^0)$ and $L(W^1)$ to show that the update step has decreased the loss!

$$
\begin{aligned}
L(W^0) &= 2.3214210229134693 \\
L(W^1) &= 2.125929842206539
\end{aligned}
$$