Week 2: NLP and Word Embeddings

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.

   Ans: False

2. What is t-SNE?

   Ans: 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.

I'm feeling wonderful today	1
I am 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 $y = 1$.

   Ans: True

4. Which of these equations do you think should hold for a good word embedding?

   Ans:

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

5. Let $\mathbf{E}$ 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 $E * o1234$ in Python?

   Ans: It is computationally wasteful.

6. When learning word embeddings, we create an artificial task of estimating $P(target|context)$. 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.

   Ans: True

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?

   Ans: They are chosen to be nearby words.

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

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

   Ans:

   - θ_t, e_c are 500-dimensional vectors.
   - θ_t, e_c are both trained with an optimization algorithm such as ADAM

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

$$min \sum_{i=1}^{10000} \sum_{i=1}^{10000} f(x_i) \left[\theta_i^T e_j + b_i + b'_j - log(X_{ij})\right]^2$$

   Ans:

   - θ_t, e_j should be initialized randomly at beginning
   - $X_{i,j}$ is the number of times j appears in context of i
   - Weighting function f must satisfy $f(0) = 0$

10. You have trained word embedding using a text dataset of $m1$ words. You are considering using these for a language task, for which you have a separate labeled dataset of $m2$ words. Keeping in mind that using word embedding is a form of transfer learning, under which of these circumstance would you expect the word embeddings to be helpful?

    Ans: $m1 >> m2$