9.19: Computational Psycholinguistics, Pset 2
due 4 October 2023

20 September 2023

Note: this release of the pset is *partial.* We will add at least one more problem to it after we have finished in-class presentation of the material that the problem covers.

1 Distance, similarity, and analogies in word embeddings

This problem set involves manipulating and using **word embeddings**: representations of the semantics of words as high-dimensional vectors. We will be using off-the-shelf semantic vectors derived in previous work (Pennington et al., 2014), called GloVe vectors. You should download one of the data zip files from https://nlp.stanford.edu/projects/glove/. We recommend `glove.6B.zip`, but you may use any of the vector files provided on the website. Note that working with larger vector files will make processing times in your code slower. Unzipping the file `glove.6B.zip`, you will see a number of `.txt` files. For the exercises below, we recommend using the 300-dimensional vectors in the file `glove.6B.300d.txt`. (Note that all words the GloVe word vector files are converted to lower-case, so you will have to do the same in this exercise.)

The Colab notebook contains Python code for downloaing the GloVe vectors and reading them into a dictionary data structure. We call the resulting dictionary `e` (for embedding), so calling `e['car']` returns an array representing the semantics of the word *car*, and so on.

1. One of the main functions of semantic vectors is to represent similarity relations among words. For example, *frog* and *toad* are very similar in meaning, while *frog* and *yesterday* are very dissimilar.

   Write a function to compute the **cosine similarity** between two vectors. Cosine similarity is a score between -1 and 1 indicating similarity, where 1 is maximal similarity and -1 is minimal similarity. Cosine similarity between two vectors $\mathbf{A}$ and $\mathbf{B}$ is defined as:

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2}\sqrt{\sum\limits_{i=1}^{n} B_i^2}}.$$

Hint: You will probably find it faster and more convenient to use the function `numpy.dot` from the `numpy` package rather than manually implementing all the summations above!

(a) Verify that your implementation of cosine similarity is correct by checking that it is **symmetrical**: it should be the case that `similarity(x,y) == similarity(y,x)` for all `x` and `y`. Demonstrate that this is the case with a few examples.

(b) As sanity checks, verify that the following similarity relations are true in the GloVe vectors given your implementation of cosine similarity. Report the similarity relations for these examples.

  i. *car* is closer to *truck* than to *person*
  ii. *Mars* is closer to *Venus* than to *goes*
  iii. *warm* is closer to *cool* than to *yesterday*
  iv. *red* is closer to *blue* than to *fast*
  v. Come up with two more examples that demonstrate correct similarity relations.
  vi. Come up with two examples where cosine similarity in the semantic vectors does not align with your intuitions about word similarity.

(c) For the examples where cosine similarity does not match your intuitions, what do you think went wrong?

(d) **Extra credit**: Try a different distance metric, such as Euclidean distance. Does it result in qualitatively different patterns on your test suite?

2. Write a function to perform the **analogy task**: Given words w_1, w_2, and w_3, find a word x such that $w_1 : w_2 :: w_3 : x$. For example, for the analogy problem *France*:*Paris* :: *England*:x, the answer should be *London*. To solve analogies using semantic vectors, letting $e(w)$ indicate the embedding for a word w, calculate a vector $y = e(w_2) - e(w_1) + e(w_3)$ and find the word whose vector is closest to y.

   (a) Explain why the analogy-solving method described above makes sense.

   (b) Write a function to calculate y. The output should be a semantic vector.

   (c) Write a function to find the nearest words to y in terms of cosine similarity, and output the top 5.

   (d) Report the top 5 results for the following analogies, and describe whether you think they are sensible and why:

i. France : Paris :: England : x
ii. man : woman :: king : x
iii. tall : taller :: warm : x
iv. tall : short :: warm : x
v. Come up with 4 more analogies, 2 of which work in your opinion, and 2 of which do not work.

(e) Did you notice any patterns or generalizations while exploring possible analogies? For the ones that went wrong, why do you think they went wrong?