Homework 2 for CS378 NLP semantics: Embeddings and word meaning

Due: October 15, 2024 at 1:59 p.m.

In this homework, you will work with embeddings computed by neural models, both word type embeddings and word token embeddings.

This homework comes with the following files:

- cs378nlpsem_hw2.pdf: this file.
- data files: problem2_data.txt, problem2_ratings_as.csv, problem2_ratings_cw.csv, problem3_data.txt

For your solution, please submit:
- code files **problem1.py, problem2.py, problem3.py**
- one text file with text comments, as requested in the individual problems. This should be PDF, ASCII, or Word, named **homework2.**

If any of these instructions do not make sense to you, please get in touch with the instructor right away.

You are welcome to use the code from the jupyter notebooks available on Canvas under files/ Class materials. Notebooks are available both as .ipynb files and as PDFs. For this homework assignment, you are allowed to use generative AI as a coding assistant. If you do, please remember that AI coding assistants can only provide a first (and likely buggy) draft. You need to understand the code in depth, and improve it from the AI-provided draft into the final code. If you use generative AI as a coding assistant, you need to add, to your text response document, a section where you reflect on your experience with the AI coding assistant: How good was the code? What bugs did you need to fix? What facets of coding was it particularly good at, and where was it bad?

A perfect solution to this homework will receive **100 points**.

Problem 1: Using gensim's GLoVE vectors, and visualization (10 pts.)

In this problem, you will experiment with visualization of word type vectors.

As demonstrated in notebook *nn_type_embeddings*, use gensim to load the space glove-wiki-gigaword-300.

Code for visualization is given in the notebook *nn_type_embeddings*, function pca_visualize_neighborhood(). Make a function tsne_visualize_neighborhood() that uses t-sne (https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html) instead of PCA.

Use your code to visualize the 20 nearest neighbors, in this GLoVE space, of each of the following words:
- charge
- load
- show

(Note: This GLoVE space does not list part of speech tags.)
You can use the Python gensim package for computing nearest neighbors as demonstrated in the notebook *nn_type_embeddings.*

In your text response file homework2, write one paragraph about what you see in the visualization: What senses of the target words show up? Anything noteworthy about the graphical grouping of neighbors? Submit your code in a file called problem1.py.

Problem 2: Interpretable dimensions in embedding space (45 pts.)

In this problem, you will experiment with interpretable dimensions in embedding space. Please again use gensim, and the word type vectors in the glove-wiki-gigaword-300 space that you used for problem 1.

PROBLEM 2A.
For the first part of the problem, you will work with cognitive features from the data of Grand et al 2022. You will work with:
- the category **animals**, and property **size**
- the category **clothing**, and property **wealth**

The file problem2_data.txt contains the seed adjectives for size and wealth., and the animal and clothing words.
The file problem2_ratings_as.csv contains average human ratings for size of animals.
The file problem2_ratings_cw.csv contains average human ratings for clothing by wealth.

For both category = animals, property = size and category = clothing, property = wealth,
- Compute a dimension for the property using seed adjectives and GLoVE vectors, computing the dimension as the average over all difference vectors across a positive and a negative adjective.
- Project each category word onto the dimension.
- Measure performance using correlation as Pearson's r (pearsonr in scipy.stats.stats)

Report the performance for both animals/size and clothing/wealth

PROBLEM 2B.
For the second part of the problem, look again at **Grand** data, **animals by size and clothing by wealth,** and test out **different seed adjectives**.

For "size", come up with three large-size and three small-size adjectives that are different from those used in 2a. Then project all animal words on that dimension, and again measure correlation. Report the correlation results in the text file.

For “wealth”, come up with three high-wealth and three low-wealth adjectives that are different from those used in 2a. Then project all clothing words on that dimension, and again measure correlation. Report the correlation results in the text file.

For both 2a and 2b, report text responses in the file homework2, and submit your code in a file called problem2.py.

Problem 3: Clustering word token vectors (45 pts.)

In this problem, you will work with token embeddings. Please use the huggingface package with BERT base uncased. You are welcome to use all the code in the demo notebook nn_token_embeddings.

The file problem3_data.txt has 200 passages from the spoken-word portion of COCA, the Corpus of Contemporary American English. All these sentences include the word “charge”.

PROBLEM 3A.

For each of the 200 passages, obtain a token embedding of the target word “charge” from layer 7 of BERT base.

Then use clustering to obtain text clusters. You can use k-means clustering from sklearn, or experiment with another clustering technique from that same Python package. Fix the random seed, so you can re-obtain the same clusters when you re-run your code. Use 5 clusters.

Manually inspect a few passages from each of the clusters: Can you see what the clusters seem to be doing? Do passages in the same cluster tend to have the same word sense of “charge”? Write about this in the text response file.

Visualize the token embeddings as clusters; Use the visualization from the notebook nn_token_embeddings.ipynb to show each token of “charge” as a point in 2-dim space. (If you like, you can change it from using PCA to using t-sne like in problem 1.)
To show the clusters, adapt the code such that it uses a separate color for each cluster. Hint: You can give the ax.scatter() function a list of cluster labels for each datapoint to be used as colors.

PROBLEM 3B.

Use BERT base to obtain replacements for the word “charge” in each of the 200 passages in problem3_data.txt. To do this, replace the occurrence of “charge” by “[MASK]”. Obtain the top 20 substitutes for each token. Again, see the notebook nn_token_embeddings.

Then again use the same clustering technique as in 3a to group the 200 passages into 5 clusters, but this time cluster tokens by their masked replacements. To do this, map each token to a vector of 1s and 0s, with one dimension for each substitute word that appeared for any of the 200 passages. There should be a 1 if the substitute was suggested for the token, and a 0 else. For example, say you have 3 passages only, with replacements [command, lead], [accused, tasked], [command, accused], then your overall list of replacements would be [command, lead, accused, tasked], and the first passage would have a vector [1, 1, 0, 0], the second passage has [0, 0, 1, 1] and the third passage has [1, 0, 1, 0].

Now inspect your five clusters again by looking at the passages (you don't need to visualize): How do the clusters compare to the clusters from problem 3a? Comment on this in the text file. It may be useful to inspect the masked substitutes along with the sentences.

For problems 3a and 3b, please record your code in problem3.py, and your text answers in the file homework2.