Asmt 2: Similarity Search

Turn in through GradeScope by 1:00pm, then come to class:
Wednesday, September 17
100 points

Overview

In this assignment you will experiment with nearest neighbor searching algorithms, and different distances. We will promarily use GloVe word embeddings as the main example.

Download the glove.2024.wikigiga.100d.zip data set from the GloVe page `https://nlp.stanford.edu/projects/glove/`

As usual, it is recommended that you use LaTeX for this assignment. If you do not, you may lose points if your assignment is difficult to read or hard to follow. Find a sample form in this directory: `http://www.cs.utah.edu/~jeffp/teaching/latex/`

Click here for an example template specifically created for this assignment.

1 Investigating GloVe (30 points)

Each line of the downloaded GloVe file contains a *token*, not necessarily an English language word. The tokens are listed in the file in the order of frequency in the data set.

A: (5 points) Report the first 10 English language words in the data set (exclude tokens which are not words).

B: (5 points) These words are not exactly how they appeared mostly commonly in the text corpus. Note one thing that has been changed.

C: (10 points) Compute the **cosine distance** between all pairs of the corresponding 100-dimensional vectors associated with these first 10 words. Report this in a 10×10 all-pairs distance matrix.
(You should confirm the matrix is symmetric, the distances are in range $[0, 2]$*, and that* 0 *distances only appear on the diagonal.)*

D: (10 points) Compute the **angular distance** between all pairs of these first 10 words. For each of the $\binom{10}{2} = 45$ unique distances **report them** in sorted order from smallest to largest.
Which pairs of words are closest, and which pair are the furthest?
(You should compare this sorted order is the same as it would be with the cosine distance.)

2 Nearest Neighbor Search (45 points)

Next we will perform nearest neighbor search on this data set.

A: (10 points) Load the first $m = 100{,}000$ tokens into a vector database (e.g., FAISS[1], DiskANN, KGraph[2], ...) Report which software you used, and how long this took, and if you can how much space it requires. If there were configuration parameters you set, report these as well.

[1] FAISS needs the Conda python package, after that it should be not too bad to install.
[2] KGraph may be easier to get started, but is less optimized.

B: (15 points) Find the token `data`, and its associated vector representation. Use your vector database to report the $k = 10$ tokens that are closest to `data` in Euclidean distance. Report how long this took in seconds.

C: (10 points) Check how well this worked by again using the vector database to find the $k' = 30$ closest tokens in Euclidean distance to the `data` vector. Then explicitly calculate the Euclidean distance of the vectors of these k' tokens and check if the closest $k = 10$ out of those $k' = 30$ are the same as in step **B**. Report these distances.

D: (10 points) Now do a brute force calculation. Compute the Euclidean distance between the vector for `data` and *every* other vector for the top $m = 100{,}000$ tokens.
Report the smallest 10 distances and tokens. See if this matches those found in steps **B** and **C**.
Report approximately how long this took in seconds.

3 Angular Hashed Approximation and Random Vectors (25 points)

We will now approximate the Angular Similarity using random vectors. We will use a normalized version of the angular similarity defined for two vectors $a, b \in \mathbb{R}^d$ so:

$$\mathsf{s}_{\text{ang}}(a, b) = 1 - \frac{1}{\pi} \arccos(\langle \bar{a}, \bar{b} \rangle)$$

If a, b are not unit vectors (e.g., in $\mathbb{S}^{d-1}$), then this implicitly converts them to $\bar{a} = a/\|a\|_2$ and $\bar{b} = b/\|b\|_2$ as part of the computation. The similarity $\mathsf{s}_{\text{ang}}(a, b)$ reports a value between 0 and 1, with, as usual, 1 meaning most similar. **Hint:** Compute L2 norm via `np.linalg.norm( )`.

A: (5 points) For the vectors for words `data` and `mining`. Compute and report the angular similarity between them.

B: (5 points) Compute a random unit vector, uniformly chosen from $\mathbb{S}^{99}$.
Report its angular similarity with the vectors for `data` and with `mining`.

C: (5 points) Compute some $t = 10$ random unit vectors $u_1, u_2, \ldots, u_t$, and for each take the dot-product with the vectors for `data` and `mining`.
Use these 10 pair of dot-product values (you only need to compare their signs) to estimate the angular similarity between the vectors for `data` and `mining`. Report this value. *(Hint: It should be a multiple of* 0.1*)*

D: (10 points) Next repeat this experiment with larger and larger values of t until you have estimated the true angular similarity within a value of 0.01. Report how large a value of t that you needed to get this much accuracy. This is a random process, so you may want to repeat it a few times with new randomness (e.g., resetting your random seed) to make sure your estimate is not too far off from the expected value.

4 Bonus (2 points)

Find the two vectors in the first $m = 100{,}000$ tokens of the GloVe dataset which have the smallest cosine distance to each other. Report the pair, and their cosine distance. Explain how you did it.