Review 2

Bilal Akliai

13 June 2024

I implemented the different algorithms of the research paper in Python.

What I have for the moment : In terms of results, it is not really efficient. For around 35,000 molecules, my algorithm is slow (about 5 minutes), and gets a result of around 60%. I think there's still a lot of optimization I can do.

The choices I've made : I represented the HNSW structure as a list of layers. Each layer is represented by a dictionary. In each dictionary, the key is a compound and the value is a set of its neighbors at that level.

Example :

Layer 0:

$$\begin{aligned}&\text{compound1} : \{\text{compound2}, \text{compound3}\}\\&\text{compound2} : \{\text{compound1}, \text{compound3}\}\\&\text{compound3} : \{\text{compound1}, \text{compound2}, \text{compound4}\}\\&\text{compound4} : \{\text{compound3}\}\end{aligned}$$

Layer 1:

$$\begin{aligned}&\text{compound1} : \{\text{compound3}\}\\&\text{compound3} : \{\text{compound1}, \text{compound4}\}\\&\text{compound4} : \{\text{compound3}\}\end{aligned}$$

Layer 2 (top layer):

$$\begin{aligned}&\text{compound3} : \{\text{compound4}\}\\&\text{compound4} : \{\text{compound3}\}\end{aligned}$$

I also tried to optimize search_layer because it is an important algorithm.

My questions :

They recommend using 1/log(M) for mL (normalization factor for level generation), however I have the impression that this results in one or two layers at most. So, I think I have to update M (because the meaning of M is "number of established connections"). But, why this value is initially given to us ?

Concerning the "select_neighbors_heuristic" algorithm, I don't see how to make good use of the second factor "keepPrunedConnections".