

ADB PROJECT

SEMANTIC SEARCH



Doaa Magdy Ibrahim - 9202521

Rufaida Mohamed Sayed – 9202550

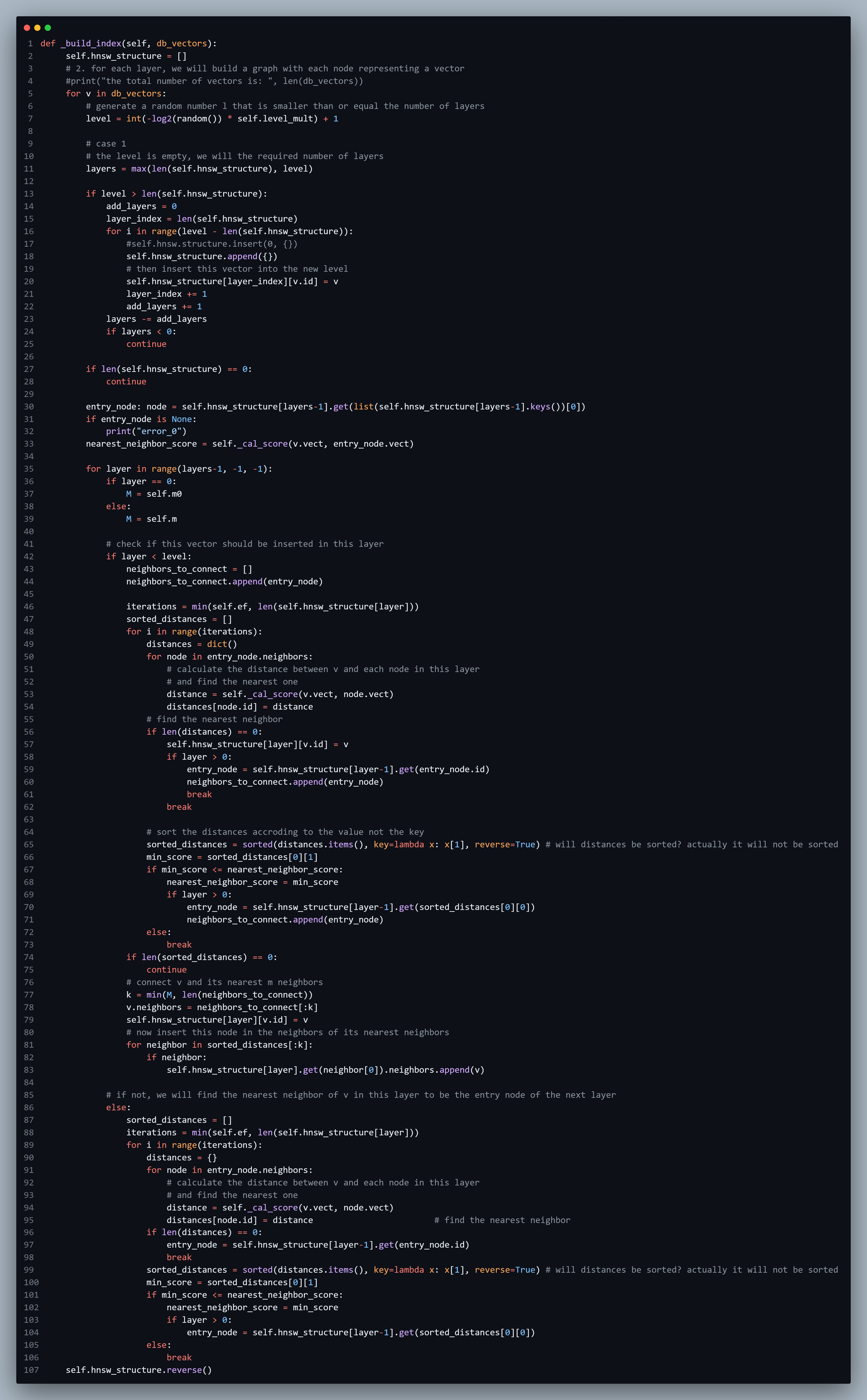
Eng. Abdelrahman Kasseb

Dr. Michael Nawar

Semantic Search

HNSW Algorithm:

* we already implemented an initial version of this indexing but the retrieval implementation is not completed yet.
  + However, we haven’t set it up so that the created index is saved in an external file. In other words, it is not suitable for the size of the database required.
* we found that it is more suitable for
  + multi-dimensional databases.
  + dynamic databases with frequent updates but in our case, it is static.
* its adv.
  + It is very fast because it is implemented as layers of graphs which reduce the number of distance comparisons that are being run.
* Hyper-Parameters
  + m
    - the number of nearest neighbors to be connected to the node.
    - Good value for m lies between 5 & 48
    - Larger values of m are better for high-dimensional data.
    - We may let m = 30.
  + m0
    - the number of nearest neighbors to be connected to the node located in the first (lower) layer.
    - it is always 2\*m = 60.
  + ef\_construction
    - This is the exploration factor.
    - It is the number of neighbors to be explored (i.e., the nubmer of iterations).
    - After we create the list of neighbors through the ef iterations, this list will be truncated to length of m.
    - This parameter controls the speed of building the index.
    - This value is preferred to be 2\*m = 60.
    - It is also recommended to choose the value that results at recall = 0.95-1 during training (we will try it in the code).
  + ef\_search
    - The number of nearest neighbors to explore during search.
    - Usually, it has a default value = 40.
    - I think this value will be determined according to how our code will be tested (i.e., how many nearest vectors should be retrieved for the input query).
    - Cause this number limits the number of retrieved vectors.
  + Level\_mult
    - The number of layers of graphs
    - level\_mult = 1 / log2(m)



What we intend to do next

* Complete the implementation of the HNSW (store the built index in an external file + complete the retrieval function.).
* Implement the LSH indexing and compare it with the HNSW.
* We may implement the inverted file index and combine it with the HNSW so that the performance of HNSW is improved (but need to read more about this part).

References

* [HNSW Indexes with Postgres and pgvector (crunchydata.com)](https://www.crunchydata.com/blog/hnsw-indexes-with-postgres-and-pgvector)
* [Similarity Search, Part 4: Hierarchical Navigable Small World (HNSW) | by Vyacheslav Efimov | Towards Data Science](https://towardsdatascience.com/similarity-search-part-4-hierarchical-navigable-small-world-hnsw-2aad4fe87d37)