ASSIGNMENT 1 – Q3

REPORT

This report describes how discriminative subgraphs are identified and the reasoning behind the proposed approach, based on the implemented code.

The overall idea is to reduce the size of the candidate set for each query by carefully selecting a small set of informative subgraphs during index construction. The method does not rely on any additional pruning after candidate generation; instead, the candidate set is directly obtained using the selected subgraphs.

First, frequent subgraphs are mined from the database graphs using a frequent subgraph mining step. An external tool such as gSpan is used when available to obtain a large and diverse collection of frequent graph fragments. If the external tool is unavailable, simple fallback patterns such as single-node and single-edge subgraphs are generated. This step focuses only on coverage and diversity, not on discriminative power.

From the mined frequent subgraphs, a subset of discriminative subgraphs is selected. Each candidate subgraph is evaluated based on how often it appears across the database graphs. Subgraphs that appear in almost all graphs or in very few graphs are discarded, as they do not help in distinguishing between graphs. Preference is given to subgraphs that occur in a moderate number of graphs and capture meaningful structural information, such as multiple nodes and edges. This selection ensures that the chosen subgraphs have stronger filtering ability.

The selected discriminative subgraphs are then used as binary features. Each database graph and query graph is converted into a feature vector that records whether a particular discriminative subgraph is present. During querying, the candidate set is generated by selecting all database graphs that contain all the discriminative subgraphs present in the query. This is implemented by intersecting the sets of graphs corresponding to each active feature in the query vector.

By identifying discriminative subgraphs during index construction and using them directly for feature-based matching, the approach efficiently narrows down the candidate set for each query, satisfying the objective of graph indexing in this assignment.

References

References : [1] X. Yan and J. Han, “gSpan: Graph-Based Substructure Pattern Mining,” in Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM), Maebashi City, Japan, 2002, pp. 721 724. Available: https://sites.cs.ucsb.edu/\~xyan/papers/gSpan.pdf

[2] M. Kuramochi and G. Karypis, “Frequent Subgraph Discovery,” in Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM), San Jose, CA, USA, 2001, pp. 313–320.

Available: https://ieeexplore.ieee.org/document/1316833

[3] X. Yan, “gSpan Software Package,”

Available: https://sites.cs.ucsb.edu/\~xyan/software/gSpan.htm

[4] “Frequent Subgraph Mining (FSG) – Lecture Notes,” University of Texas at Arlington.

[5] OpenAI. ChatGPT: Large Language Model for Conversational AI. Used for assistance with code syntax and debugging, and summarizing explanations of algorithmic behavior.

Available at: https://chat.openai.com

[6] All LLM generated code has been mentioned in the scripts, starting with the comment “#### LLM-assisted code (ChatGPT); all logic and correctness verified by the authors.” and ending with the comment design “####”.