COL761 Assignment 3

Rules

- Submission deadline is 21 November 2023.
- Submission will be made on the GitHub repository of your team.
- Please check details about evaluation hardware/software mentioned at the end of this document very carefully.
- This assignment contains two questions, the second of which is competitive.
- Do not copy.

Q1 Uniformly Distributed Points in High-Dimensional Spaces

[20 points]

This problem is about the behaviour of a uniform distribution of points in high-dimensional spaces. Generate a dataset of 1 million random points in d-dimensional space (d varying as 1, 2, 4, 8, 16, 32, and 64). Assume that the points are uniformly distributed over $[0, 1]$ in each dimension and that the dimensions are independent. Use precision of 5 decimal places. Choose 100 query points at random from the dataset. Examine the the farthest and the nearest data point from each query. Compute the distances using L_1, L_2, and L_∞. Plot the average ratio of farthest and the nearest distances versus d for the three distance measures. Make sure to not include the query point itself in the nearest data point computation. Explain the results. Submit the code and report. Use python.

Q2 Graph Classification and Regression Using Graph Neural Networks

[60 points]

Task Overview:
This assignment aims to engage you in performing graph property prediction using Graph Neural Networks (GNN) for two primary tasks: classification and regression. You will be provided with dataset files along with an evaluator script for assessment. Additionally, an optional node and edge feature encoder will be made available for use. This assignment is designed to be competitive, encouraging you to explore and innovate in their approaches. The primary tasks involve predicting graph properties through classification and regression using GNN models.

Task 1: Classification

- Using the provided dataset, you are required to predict a binary label associated with graphs. The choice of model architecture, as well as the training paradigm, is left to the you. The evaluation metric for this task will be the Receiver Operating Characteristic Area Under the Curve (ROC-AUC). Alongside model development, you should plot training and validation learning curves, visualize graphs using NetworkX, and investigate graphs where the model demonstrates notable misclassifications or poor performance.

Task 2: Regression

- In this task, you will predict a continuous numerical value associated with the graphs in the dataset. Similar to the classification task, you have the flexibility to choose their model architecture and training approach. The evaluation metric for this task will be the Root Mean Squared Error (RMSE). Similarly, you are expected to plot learning curves and visualize graphs to identify areas where the model exhibits significant errors in prediction.

Additional Requirements:

- **Baseline Models:** You should implement simple baseline models such as random predictions, logistic regression, and linear regression to benchmark and quantify the improvements achieved using GNNs.
- **Visualization:** Utilize NetworkX library to visualize graphs in the dataset and visually represent misclassifications or significant errors made by the model.
- **Analysis:** You are expected to provide a comprehensive analysis of their chosen model's performance and how it compares to baseline models. You may plot learning curves, gradient-norm curves, etc.

- **Implementation Details:** While the node and edge feature encoder is optional, you should document their decision regarding its utilization, highlighting its impact on model performance.

Evaluation Metrics:

1. **Classification Task:** ROC-AUC
2. **Regression Task:** RMSE

Submission Requirements:

- On moodle submit an *install.sh* script that will clone the GitHub repository/branch/folder of your submission. We are providing an example submission datastructure that it should generate: it should generate the "MyTeam" directory (both the directory and its sub-directory "A3"), in the directory where install.sh script is run. It should not contain any GitHub personal access token (PAT) as that interferes with our automatic evaluation script. PAT of COL761 account will be added to environment's password store, you don't need to worry about it.
- In "MyTeam/A3", there should be two files "interface1.sh" and "interface2.sh", for Q1 and Q2, respectively. Since Q1 takes no inputs, the interface1.sh script also takes no input. The format of interface2.sh is defined in the script provided in the example directory structure provided.
- Python scripts containing the implemented models, visualization, analysis, and results.
- The report documenting the approach, reasoning behind model choices, visualizations, analysis, and comparisons with baseline models.

The assignment promotes exploration, innovation, and thorough analysis in the field of graph property prediction using GNNs. Encourage you to utilize the provided resources and explore beyond them to improve their model performance.

Environment specifications: Your code will be evaluated on HPC. A python environment containing PyTorch Geometric 2, PyTorch <= 2 (most probably 1.13 or 1.11), scikit-learn, NumPy, NetworkX 3.0, pandas, and matplotlib will be activated. Please ensure that you adhere to these specifications. Refer to the documentation of these libraries, especially of how to create a Dataset object using PyTorch Geometric and PyTorch Geometric Data objects (which will be what is provided to you in the dataset file).

Competitive Marking Scheme: The marking will consist of the following steps.

- Compute z-scores for your metric. If your class' metric distribution is $M = \{m_1, m_2, \dots\}$, and your achieved metric is m_y, then the z-score is $z = (m_y - \text{mean}(M))/\,\text{std}(M)$.
- Map to marks using a sigmoid like curve. We find the following as best.

$$\text{marks}(\%) = \begin{cases} \widehat{\sigma}(az) & z \geq 0 \\ \widehat{\sigma}(bz) & z \leq 0 \end{cases}$$

  where $\widehat{\sigma}(z) = \frac{d}{d+e^{-z}}$, $d = 4, a = 1.4, b = 0.8$.
- For RMSE (regression), where lower is better, the distribution will be flipped along the y-axis by taking $z = -(m_y - \text{mean}(M))/\,\text{std}(M)$. Rest of the steps will remain same.