1. Project Description

In modern data mining tasks, graphs are a powerful representation of relationships. For example, in a social network (e.g., Facebook, LinkedIn), the interaction of users can form a graph, where each node represents one user and edges represent social interactions, such as friendships, follows, or interactions on social media platforms. Node classification is a key task that involves predicting labels or categories for nodes within a graph. Thus, this project will focus on the task of node classification in a single graph. This is a competition among different groups. The performance of your submission would determine your grades for this project.

2. Objective

Design and implement a node classification model on the graph (see attached dataset). The objective is to predict the class label for each node based on their features and the relationships they have with other nodes.

3. Dataset Details

There are 2480 nodes in total and each node belongs to one of 7 classes. Each node in the graph is associated with a feature vector and a label (class). The connection between nodes is stored in an adjacent matrix.

The provided files include (see the tutorial “intro_pyg” for more illustration):

1. An adjacency matrix: stored in file adj.npz
   - Each entry in this matrix indicates whether two nodes are connected or not
   - E.g., adj[i, j] = 0 if node i and node j are disconnected, adj[i, j] > 0 otherwise
2. A feature matrix: stored in file features.npy
   - Each row represents the feature vector of a node
   - E.g., features[i] represents the feature vector of node i
3. A list of labels: stored in file labels.npy
   - Include class labels of training labels
4. Data splits (train/test splits): stored in splits.json
   - splits[‘idx_train’]: node index for training nodes
   - splits[‘idx_test’]: node index for testing nodes

#Nodes	#Edges	#Classes	#Features	Train/Test
2480	10100	7	1390	496/1984

4. Model Implementation

Any model for node classification is permitted. For example, one intuitive solution is MLP without using the graph structure (adjacency matrix), but this could be suboptimal (see the tutorial “intro_pyg” for this example).

Note: Though python examples are provided, other programming languages (e.g., MATLAB, C++, JAVA) is also acceptable for this project.

5. Evaluation

Submit a file named {YourTeamName}_submission.txt, which includes the predicted class labels of testing nodes

- Each line in the file corresponds to the predicted label of a node
- In total, there are 1984 nodes, and your file should have 1984 lines

The performance of your prediction will be evaluated based on the accuracy.

- It is the ratio of correctly predicted nodes to all nodes
- The higher, the better

6. Submission Date and Grading (30% in total for this project)

- February 28th (one-page proposal): 5%
- March 7th, 2025 (‘{YourTeamName}_submission.txt’):
- March 21st, 2025 (‘{YourTeamName}_submission.txt’):
- April 4th, 2025 (‘{YourTeamName}_submission.txt’):
- April 16th, 2025 (‘{YourTeamName}_submission.txt’):
- April 30th, 2025 (Final ‘{YourTeamName}_submission.txt’) + report&code: 20% + 5%

Note: Only the last submission (April 30th) is required. But if you want to test your model, you can submit at each submission date.

Proposal (5%): 1 page (using the template: https://www.overleaf.com/latex/templates/aaai-press-latex-template/jymjdgdpdmxp)

- Team members (3~5 students for each group), team name (feel free to choose any team name, which will be used for showing performance ranking on Piazza later)
- Preliminary plan (milestones): what methods are you planning to explore, and why do think these methods are reasonable for this project.

Result submission (20%):

- Each team submits only one '{YourTeamName}_submission.txt' and code
- Make sure that the submitted code can reproduce the results
- Grading (T: total number of teams)

Rank	Grade
1 ~ T / 5	100
T / 5 ~ 2T / 5	90
2T / 5 ~ 3T / 5	80
3T / 5 ~ 4T / 5	70
4T / 5 ~ T	60

Final Report (5%): 2~3 pages (using the template: https://www.overleaf.com/latex/templates/aaai-press-latex-template/jymjdgdpdmxp)

- Methodology and Implementation
- List of team members and their contributions. Specify clearly what is the role of each team member. For example:
    - Student 1: Wrote python scripts for data preprocessing, paper exploration, algorithm implementation, debugging, etc.
    - Student 2: Wrote python scripts for hyperparameter search, final report, etc.

If you have any confusion, feel free to post your question on Piazza or email the instructor (wujun4 at msu dot edu).