# Topics in Deep Learning 2024: Assignment 1


## Charu Sharma


### Due by: 24th March 2024 (11:59 pm IST)

Instructions: Total marks: 100

1. This assignment is an individual submission, NOT a group activity.
2. Total Marks of the assignment is 100 with duration of two weeks.
3. The submission should be in a pdf file format (preferably in latex). For code submissions, a Jupyter notebook with answers as comments is preferred.
4. Binary evaluation may be conducted for a few subparts of questions wherever applicable.
5. For queries, reach out to TAs via Moodle.
6. Submissions should be done via Moodle only.

---

## Question 1: 25 marks (**Preprocess and Analyze Graph Data)**

1. [5 marks] Given two graph datasets (graph1.edges and graph2.edges), get their statistics (number of nodes, edges, degree (mean, min, max), density and sparsity of graph, etc.) and do the analysis thoroughly (write in a paragraph for each dataset).
2. [3 marks] Perform analysis on properties like finding out cliques (3-clique, 4-clique), node centrality, and clustering coefficients (write in a paragraph for each dataset).
3. [2 marks] Perform more analysis by creating 1-2 new properties of your own.
4. [10 marks] Run link prediction algorithm for the above two graph datasets using both Random Walk algorithms (DeepWalk and Node2Vec). Consider undirected and unweighted graphs for link prediction algorithms. You can use the existing implementations of DeepWalk and Node2Vec.
5. [5 marks] Using the above results, analyze and find out which algorithm will give better representation and why? Mention which algorithm captures what. Explain for each algorithm, why it works or why it doesn't work for both the graphs.

## Question 2: 15 marks (**Traditional ML for Graphs**)

1. [9 marks] We learned three node centralities or importance features: eigenvector centrality, betweenness centrality, and closeness centrality. Think of example tasks or applications where each of these features could be useful. Write the following for each centrality measure:
   a. Explain the problem statement. Introduce the inputs and outputs of the application.
   b. Explain why the chosen centrality measure is more appropriate than the others.
   c. Discuss the kind of classifier that could result in good performance.
2. [6 marks] Consider the idea of color refinement in WL kernels to identify if the two graphs are isomorphic. Show using two pairs of examples (graphs of 6 nodes: one pair isomorphic; another pair is not isomorphic). Think and describe how and when the hashing iterations are terminated.

Note: Avoid giving examples which are available online or mentioned in the class.

## Question 3: 20 marks (**Expressivity in GNNs)**

We studied the difference between the expressiveness of GCN, GraphSAGE and GIN architectures due to their mean, max and sum functions to differentiate between nodes.

1. [10 marks] Provide five examples of pairs of graphs (of at least 3 nodes in each graph) where GCN and GraphSAGE fail to distinguish between the nodes in the graphs in each pair, but GIN is able to distinguish. Show GCN (mean pool), GraphSAGE (max pool) and GIN (sum) operations on these five pairs of graphs. Also run WL test to show that the graphs in each pair are different from each other.
2. [10 marks] Provide five examples of pairs of graphs (of at least 3 nodes in each graph) where GIN fails to distinguish between the nodes in the graphs in each pair. Run WL test to prove that the GIN fails.

Note: Avoid giving examples which are available online or mentioned in the class.

## Question 4: 40 marks (**Implement, evaluate, and understand the impact of GNN variations)**

The goal of this exercise is to encourage a research temper: running experiments, analyzing outputs, drawing inferences, and deciding next steps. You may use the base libraries of pytorch or tensorflow. However, directly using graph libraries for implementing the models (e.g., pytorch- geometric) is **not allowed**. You may use the graph libraries for helping with data loading or other non-model specific parts. Training some of the models below may take some computation time, so get started sooner than later. **Copying from one another is not ok.** Reading papers is encouraged. Acknowledge all your tools and sources.

We recommend experiments be performed on a simple dataset such as Cora/Citeseer. These work on a standard CPU laptop. **Use standard splits** (see https://arxiv.org/abs/1710.10903).

Your output can be a Jupyter notebook (that we can run) that is self-contained, reproducible, includes a thorough analysis, has appropriate comments, and answers to various questions in Markdown / code / plots as appropriate.

1. [2 marks] Explain the characteristics of the dataset. What is the task? What are the number of features, outputs? Where are the features derived from? What are some typical graph properties – node degree?
2. [8 marks] Implement a base class that can support GNNs of different kinds with the 4-step approach (initialization, aggregation, combination, output) that we discussed in class. Build your code such that it is re-usable for some of the questions below.
3. [15 marks] Implement the Graph Convolutional Network (GCN) by inheriting from the base class above. Train and evaluate using the standard protocol on the semi-supervised setup of the citation dataset (see the GCN paper: https://arxiv.org/abs/1609.02907). Understand and document performance across key hyperparameters.

a. Establish an understanding of performance variation due to changing random seeds. Ensure reproducibility if random seeds are the same. Reporting mean + standard-deviation across 3 or 5 runs is common practice. 3 marks
b. Impact of number of layers. Tabulate and explain reasons for the performance variation (if any) across 1, 2, 5 layers. 3 marks
c. Feature dimension. How many options did you try? What is a good value and why? 3 marks
d. Is dropout useful? Why? What dropout probability works best? 3 marks
e. Include loss plots (train and validation). Talk about any observations related to under or overfitting. 3 marks

4. [15 marks] Using the best parameters for GCN, compare them against GraphSAGE, Graph Isomorphism Networks (GIN), and Graph Attention Network (GAT) message passing approaches. You can take existing implementations of the other 3 models.
    a. GraphSAGE: We learned multiple options. Which method works well? Why is it better/worse/equal than GCN? 3 marks
    b. GAT: Why is the method better/worse/equal as compared to GCN? 3 marks
    c. GAT: What are the important hyperparameters for GAT (at least 2)? How does performance change across them? 3 marks
    d. GIN: Are there important hyperparameters for GIN? What are they and how does performance change? 3 marks
    e. Include loss plots for the best variants each of GCN, GraphSAGE, GIN, and GAT. Do we notice any performance difference? Explain it. 3 marks