Lab 3: Advanced Deep Learning

Vijay Venkatesh Murugan
M1 Data AI - IP Paris
vijay.murugan@ip-paris.fr

Question 1.

What is the expected number of degrees of a node in Erdős–Rényi random graphs without self-loops, i.e., edges from a node to itself, $n = 15$ nodes, and edge probabilities $p = 0.1$ and $p = 0.4$?

Solution

Given:

- The number of nodes is $n = 15$,
- The edge probabilities are $p = 0.1$ and $p = 0.4$,
- There are no self-loops (i.e., no edges from a node to itself).

The expected degree of a node in an Erdős–Rényi random graph $G(n, p)$ is given by:

$$\mathbb{E}[k] = (n - 1) \cdot p$$

For $p = 0.1$:

$$\mathbb{E}[k] = (15 - 1) \cdot 0.1 = 14 \cdot 0.1 = 1.4$$

For $p = 0.4$:

$$\mathbb{E}[k] = (15 - 1) \cdot 0.4 = 14 \cdot 0.4 = 5.6$$

- For $G(15, 0.1)$, the expected degree of a node is $\mathbb{E}[k] = 1.4$.
- For $G(15, 0.4)$, the expected degree of a node is $\mathbb{E}[k] = 5.6$.

Question 2

Why Fully Connected Layers are Not Commonly Used as Readout Functions

Answer

Fully connected layers depend on the input size and the ordering of nodes. Graphs, however, differ in size and structure, making such layers unsuitable for aggregating node features into a fixed-size graph representation. Instead, permutation-invariant operations like the *sum* or *mean* are preferred for their robustness and generality.

Considering the scenario in Fig 1 with G_1, G_2, and G_3. Each graph has a different number of nodes (n_1, n_2, n_3), resulting in varying feature matrices X. If we used fully connected layers as a readout function:

- The input to the fully connected layer would need to be of a fixed size, requiring padding or truncation, which could distort the representation of the graphs.
- Fully connected layers are not permutation-invariant. The ordering of the nodes in the input would affect the output, making them sensitive to how nodes are indexed, which contradicts the intrinsic invariance of graphs.
- Summation or mean operations, on the other hand, aggregate features across all nodes in a graph in a permutation-invariant way, ensuring that the representation is consistent regardless of node order or graph size.

The sum or mean operations are preferred as they are:

- Permutation-invariant, preserving the graph's symmetry.
- Independent of graph size and structure.
- Computationally efficient and straightforward to implement.

article amsmath graphicx amsfonts

Question 3

Assume that for a set of three graphs we receive the following matrix Z:

$$Z = \begin{bmatrix} 0.21 & -0.95 & 0.40 \\ -0.40 & 0.12 & 0.68 \\ 0.89 & 0.34 & 1.31 \\ 0.68 & 0.08 & -0.50 \\ 0.62 & 0.18 & -0.10 \\ 0.89 & 0.34 & 1.31 \\ 0.81 & -0.20 & 1.29 \\ 0.89 & 0.34 & 1.31 \\ 0.61 & -0.14 & -0.31 \end{bmatrix}$$

where rows 1, 2, and 3 correspond to nodes in the first graph G_1, rows 4, 5, 6, and 7 correspond to nodes in G_2, and the remaining two rows correspond to nodes in G_3.

Compute the representations of the three graphs (i.e., z_{G_1}, z_{G_2}, and z_{G_3}) for each one of the following three readout functions:

1. **Sum**
2. **Mean**
3. **Max**

Which of these functions is able to distinguish these graphs best?

Answer

We are given the node feature matrix Z for three graphs G_1, G_2, and G_3, where:

- Rows 1, 2, and 3 correspond to nodes in G_1,
- Rows 4, 5, 6, and 7 correspond to nodes in G_2,
- Rows 8 and 9 correspond to nodes in G_3.

The matrix Z is:

$$Z = \begin{bmatrix} 0.21 & -0.95 & 0.40 \\ -0.40 & 0.12 & 0.68 \\ 0.89 & 0.34 & 1.31 \\ 0.68 & 0.08 & -0.50 \\ 0.62 & 0.18 & -0.10 \\ 0.89 & 0.34 & 1.31 \\ 0.81 & -0.20 & 1.29 \\ 0.89 & 0.34 & 1.31 \\ 0.61 & -0.14 & -0.31 \end{bmatrix}$$

We compute the representations z_{G_1}, z_{G_2}, and z_{G_3} using the following readout functions:

(i) Sum

For each graph, we compute the sum of the rows corresponding to its nodes:

$$z_{G_1} = \sum_{i=1}^{3} Z[i,:] = \begin{bmatrix} 0.21 & -0.95 & 0.40 \end{bmatrix} + \begin{bmatrix} -0.40 & 0.12 & 0.68 \end{bmatrix} + \begin{bmatrix} 0.89 & 0.34 & 1.31 \end{bmatrix}$$

$$z_{G_1} = \begin{bmatrix} 0.70 & -0.49 & 2.39 \end{bmatrix}$$

$$z_{G_2} = \sum_{i=4}^{7} Z[i,:] = \begin{bmatrix} 0.68 & 0.08 & -0.50 \end{bmatrix} + \begin{bmatrix} 0.62 & 0.18 & -0.10 \end{bmatrix} + \begin{bmatrix} 0.89 & 0.34 & 1.31 \end{bmatrix} + \begin{bmatrix} 0.81 & -0.20 & 1.29 \end{bmatrix}$$

$$z_{G_2} = \begin{bmatrix} 3.00 & 0.40 & 2.00 \end{bmatrix}$$

$$z_{G_3} = \sum_{i=8}^{9} Z[i,:] = \begin{bmatrix} 0.89 & 0.34 & 1.31 \end{bmatrix} + \begin{bmatrix} 0.61 & -0.14 & -0.31 \end{bmatrix}$$

$$z_{G_3} = \begin{bmatrix} 1.50 & 0.20 & 1.00 \end{bmatrix}$$

(ii) Mean

For each graph, we compute the mean of the rows corresponding to its nodes:

$$z_{G_1} = \frac{1}{3} \sum_{i=1}^{3} Z[i,:] = \frac{1}{3} \begin{bmatrix} 0.70 & -0.49 & 2.39 \end{bmatrix}$$

$$z_{G_1} = \begin{bmatrix} 0.233 & -0.163 & 0.797 \end{bmatrix}$$

$$z_{G_2} = \frac{1}{4} \sum_{i=4}^{7} Z[i,:] = \frac{1}{4} \begin{bmatrix} 3.00 & 0.40 & 2.00 \end{bmatrix}$$

$$z_{G_2} = \begin{bmatrix} 0.750 & 0.100 & 0.500 \end{bmatrix}$$

$$z_{G_3} = \frac{1}{2} \sum_{i=8}^{9} Z[i,:] = \frac{1}{2} \begin{bmatrix} 1.50 & 0.20 & 1.00 \end{bmatrix}$$

$$z_{G_3} = \begin{bmatrix} 0.750 & 0.100 & 0.500 \end{bmatrix}$$

(iii) Max

For each graph, we compute the element-wise maximum of the rows corresponding to its nodes:

$$z_{G_1} = \max\{Z[1,:], Z[2,:], Z[3,:]\} = \max\left\{ \begin{bmatrix} 0.21 & -0.95 & 0.40 \end{bmatrix}, \begin{bmatrix} -0.40 & 0.12 & 0.68 \end{bmatrix}, \begin{bmatrix} 0.89 & 0.34 & 1.31 \end{bmatrix} \right\}$$

$$z_{G_1} = \begin{bmatrix} 0.89 & 0.34 & 1.31 \end{bmatrix}$$

$$z_{G_2} = \max\{Z[4,:], Z[5,:], Z[6,:], Z[7,:]\}$$

$$z_{G_2} = \max\left\{ \begin{bmatrix} 0.68 & 0.08 & -0.50 \end{bmatrix}, \begin{bmatrix} 0.62 & 0.18 & -0.10 \end{bmatrix}, \begin{bmatrix} 0.89 & 0.34 & 1.31 \end{bmatrix}, \begin{bmatrix} 0.81 & -0.20 & 1.29 \end{bmatrix} \right\}$$

$$z_{G_2} = \begin{bmatrix} 0.89 & 0.34 & 1.31 \end{bmatrix}$$

$$z_{G_3} = \max\{Z[8,:], Z[9,:]\} = \max\left\{ \begin{bmatrix} 0.89 & 0.34 & 1.31 \end{bmatrix}, \begin{bmatrix} 0.61 & -0.14 & -0.31 \end{bmatrix} \right\}$$

$$z_{G_3} = \begin{bmatrix} 0.89 & 0.34 & 1.31 \end{bmatrix}$$

Among the three readout functions:

- **Sum** can distinguish the graphs best because it accounts for the total information (e.g., graph size and structure).
- **Mean** loses information about the graph size as it normalizes by the number of nodes.
- **Max** may lose information because it only considers the largest values and ignores other nodes' contributions.

Question 4

Give an example of two non-isomorphic graphs G_1 and G_2 (different from the ones shown in Figure 2) which can never be distinguished by the GNN model which uses the sum operator both for neighborhood aggregation and as the model's readout function.

Answer

Consider the following two graphs:

- G_1: A cycle graph with 4 nodes connected in a loop.
- G_2: A star graph with 4 leaves, where one central node is connected to 4 outer nodes.

These two graphs are non-isomorphic because they have different structures: G_1 forms a closed loop, while G_2 has a single central hub with connected leaves. However, a GNN that uses the sum operator for both neighborhood aggregation and readout cannot distinguish between them. This is because the sum operator simply aggregates the degrees of the nodes and ignores important structural information, such as loops or central hubs. As a result, both graphs would produce identical vector representations, even though they are structurally different.