1 Question 1

It is easily arguable that LSTMs are not permutation invariant. Indeed, we can define that permutation invariance is the property of a model to remain the same even when the order of the inputs changes. However, in LSTMs (as shown in Fig 1), the order of the input will affect the output. The LSTM will then not produce the same result when the input is ordered differently. Indeed, one capacity of the LSTM with its cell states and hidden states is to remember information during time. So when the input is different the output will be also different.

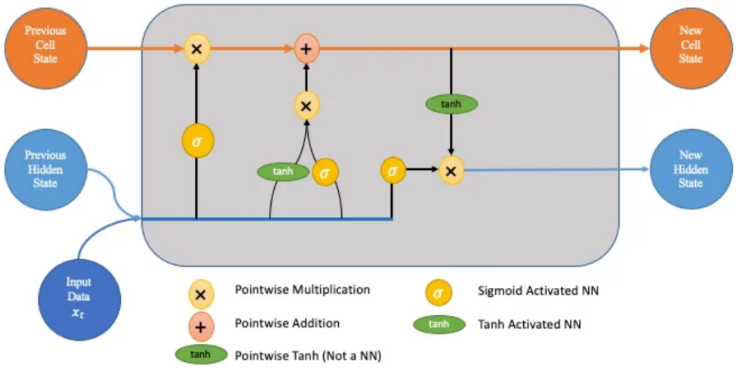

Figure 1: Architecture of an LSTM. Adapted from [2].

Therefore, as sets are unordered items, LSTMS are not suitable for this use-case.

2 Question 2

We can describe the architectural differences like that; the GNN consists of two message passign layers followed by a sum or mean readout function and finally a fully connected layer. In the message passing layers, nodes of the graph do rely on each others as the adjacency matrix is updated accordingly. The relationships between nodes are therefore crucial.

On the other hand, DeepSets objective is to handle set to learn a representation that will be **invariant** to the permutation of instances in the set. Deepsets does not consider relationships between elements in the set (on the contrary of GNNs).

There are no differences between a set and a graph without edges. Indeed, graph without edges is only a set of nodes with different features and they could be handled both by a GNN or a DeepSet architecture.

3 Question 3

3.1 Edge Probability Matrices for Homophilic and Heterophilic Graphs

First, the graphical representation of homophilic vs heterophilic graphs can be visualized in Fig 2.

- **Homophilic Graph:** In a homophilic graph, nodes within the same block should be more likely connected compared to nodes in other blocks. For a stochastic block model with $r = 2$, a suitable edge probability matrix P can be:

$$P_{\text{homophilic}} = \begin{pmatrix} 0.85 & 0.1 \\ 0.1 & 0.85 \end{pmatrix}$$

  Here, the diagonal elements will represent a high probability of an edge within the same block, while the off-diagonal elements will represent a lower probability of an edge between blocks.

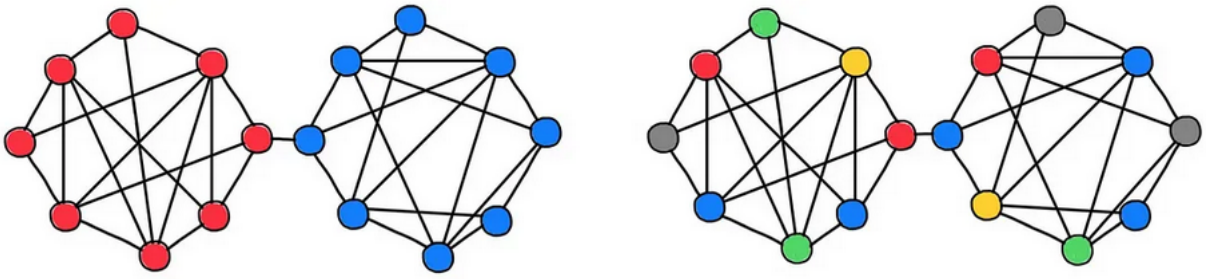

Figure 2: Homophilic graphs on the left and Heterophilic graphs (few edges are present between nodes in different communities) on the right. Adapted from [1].

- **Heterophilic Graph:** In a heterophilic graph, it is the contrary than homophilic graph, nodes from different blocks are more likely to be connected. Therefore the diagonal values should be low while the off-diagonal values should be high. It can be:

$$P_{\text{heterophilic}} = \begin{pmatrix} 0.1 & 0.85 \\ 0.85 & 0.1 \end{pmatrix}$$

3.2 Expected Number of Edges

We have a stochastic block model with $n = 20$ and $r = 4$, it means that there are four blocks of 5 nodes. First, each block has 5 nodes, so there are $5 \times 5 = 25$ possible edges between any two distinct blocks. Furthermore there are 6 pairs of distinct blocks possible ($\binom{4}{2} = 6$).

Then, the probability that these edges are created is 0.05. Therefore the total expected numberof inter-block edges is:

$$\text{nbedges} = 25 \times 6 \times 0.05 = 7.5 \tag{1}$$

4 Question 4

A weighted graphs is a graph where the entries of the adjacency matrix are not limited to values of 0 or 1. Indeed, the binary cross-entropy loss is suitable for unweighted graphs where adjacency matrix values only indicate the absence or presence of an edge.

For **weighted graphs**, where the edge weights can vary continuously ,the **Mean Squared Error (MSE)** loss, which measures the average of the squares of the differences between the predicted and actual values is a nice choice as it handles continuous values.
The MSE loss for a weighted graph's adjacency matrix can be formulated as:

$$L_{\text{MSE}} = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} (A_{ij} - \hat{A}_{ij})^2$$

Here, A_{ij} is the actual weight of the edge in the adjacency matrix, and $\hat{A}_{ij}$ is the predicted weight from the model. This loss function will penalize the model based if the predicted weights are far away from the actual weights. It will lead the model to learn a more accurate reconstruction of the weighted graph. The Mean absolute error can also be a good loss for this assignment.

References

[1] Michael Bronstein. Graph neural networks beyond weisfeiler-lehman and vanilla message passing. In *Medium — Towards Data Science*, 2022.

[2] Rian Dolphin. Lstm networks — a detailed explanation. In *Medium — Towards Data Science*, 2020.