1 Question 1

Let be $X = \{x_1, \ldots, x_m\} \in \mathbb{R}^m$ a set, to simplify the problem, we assume that there is no embedding of the set elements at the beginning (for all $i \in [\![1, m]\!], x_i \in \mathbb{R}$), nor $\tanh$ activation function after the first MLP. Moreover we assume that the hidden dimension (output dimension of the first MLP & input dimension of the second MLP) is equal to the input dimension.

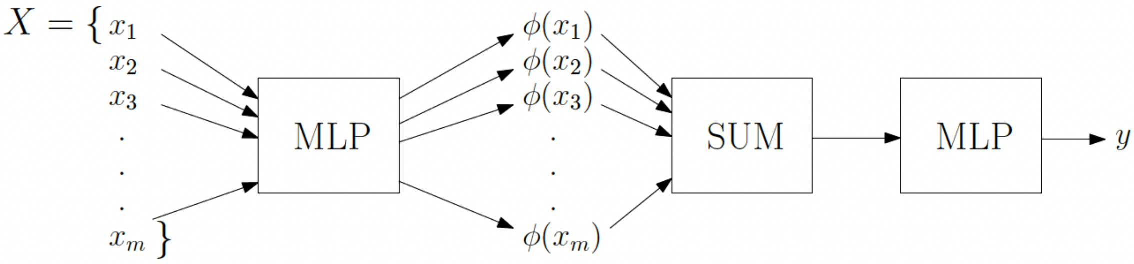

Figure 1: DeepSet model

According the hypothesis, if both MLP functions are the identity function, then the output y will be exactly the sum of the elements of the input multiset. Indeed,

$$y = \rho(\sum_{x \in X} \phi(x)) = \sum_{x \in X} x$$

So that, the first MLP will just be the identity function, the parameters of the weights matrices W and the bias vector b could be like that:

For all $i, j \in [\![1, m]\!], W_{ii}^{(1)} = 1$ and if $i \neq j, W_{ij} = 0$ and for all $j \in [\![1, m]\!], b_j^{(1)} = 0$ (i.e. $W^{(1)} = I_m, b^{(1)} = 0$).

For the second MLP, we can choose that: $W_{11}^{(2)} = 1$ and that $b_1^{(2)} = 0$.

In this case, the output of the DeepSet model will be exactly what expected: the sum of the elements of the input set ($y = W^{(2)} SUM(W^{(1)} X + b^{(1)}) + b^{(2)}$).

However, other weights and bias values or still possible (multiplication factors), this one is the simplest.

2 Question 2

If for the embedding layer, we choose the vector $[1, 0]$, then we obtain,
For the the first set X_1:

$$\begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 1.2 & -0.8 \\ -0.7 & 0.5 \end{pmatrix} = \begin{pmatrix} 1.2 & -0.8 \end{pmatrix}$$

For the the second set X_2:

$$\begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 0.2 & 0.2 \\ -0.3 & 0.1 \end{pmatrix} = \begin{pmatrix} 0.2 & 0.2 \end{pmatrix}$$

Then these two embeddings are passed through the first neural network that we choose as in the last question (with $m = 2$). Then, it is passed through the $tanh$ activation function, therefore we obtain:
For the the first set X_1:

$$\begin{pmatrix} 0.83 & -0.66 \end{pmatrix}$$

For the the second set X_2:

$$\begin{pmatrix} 0.197 & 0.197 \end{pmatrix}$$

Then we use the SUM aggregator function to obtain.
For the the first set X_1:

$$\begin{pmatrix} 0.17 \end{pmatrix}$$

For the the second set X_2:

$$(0.39)$$

Finally, we choose the second linear layer to be the same as in the last question, we obtain the last two values from the last step. Therefore, there exist weight matrices and bias vectors such that the two sets are mapped to different vectors (0.17 and 0.39).

Predicting the Sum of a Multiset of Digits

Here are the test curved that I obtained after the training of both model with 20 epochs:

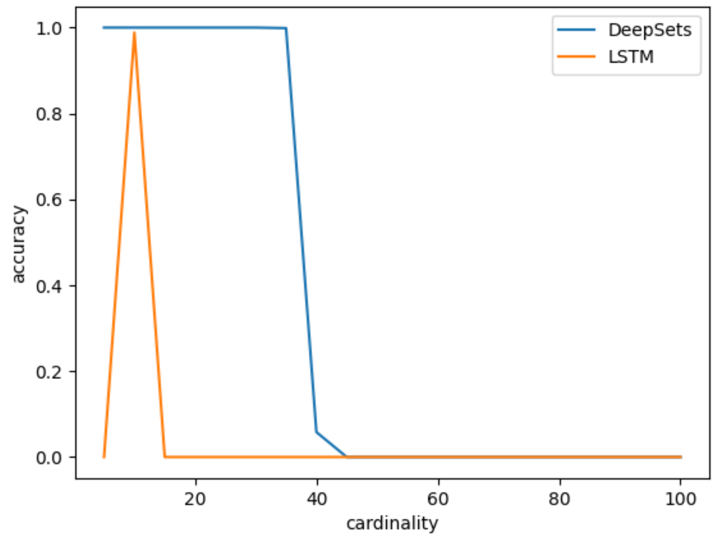

Figure 2: Comparison of DeepSets and LSTM models

We observe that the DeepSets model achieves better accuracy results.

3 Question 3

In a graph classification problem, DeepSets could correspond to a submodule of a graph neural network architecture. Indeed, for a graph classification problem, after some message passing layers, we have an embedding of each graph, then, a readout function computes the sum of the node representations for example: this is exactly what our DeepSets model is doing! Therefore, DeepSets can be used where node embeddings are aggregated into a graph-level representation. Its ability to handle sets in a permutation-invariant way makes it a good fit for graph classification.

4 Question 4

The probability that there are k edges in Erdos-Rényi random graphs with n nodes for a probability p is equal to:

$$p^k(1-p)^{\binom{n}{2}-k}$$

Therefore, the number of edges in a Erdos-Rényi random graphs follows a binomial distribution with parameters $(\binom{n}{2}, p)$

In general for a graph with n nodes, the number of possible edges is the number of pair in the set of nodes which is $\binom{n}{2} = \frac{n(n-1)}{2}$. Therefore, the expected number of edges will be $p\binom{n}{2} = \mathbb{E}(\mathcal{B}(\binom{n}{2}, p))$.

For the variance of number of edges, we have:

$$\mathbb{V}(\mathcal{B}(\binom{n}{2}, p))) = \binom{n}{2}p(1-p)$$

We assume that there will be $n = 15$ nodes in Erdos-Rényi random graphs.

- For p=0.2,

$$\text{Expected number of edges: } 0.2 * \binom{15}{2} = 21$$

$$\text{Variance of number of edges: } \binom{15}{2} * 0.2 * 0.8 = 16.8$$

- For p=0.4,

$$\text{Expected number of edges: } 0.4 * \binom{15}{2} = 42$$

$$\text{Variance of number of edges: } \binom{15}{2} * 0.4 * 0.6 = 25.2$$

Example of graph generation with Variational Graph Autoencoder