Soël MEGDOUD
soel.megdoud@ens-paris-saclay.fr

Lab session # LAB7
ALTEGRAD 2024

12/05/24

Question 1

In the DeepSets model, each input element x_i is first mapped through an embedding layer to a latent space of dimensionality h_1:

$$E(x_i) = \text{Embedding}(x_i).$$

The embedded elements are then transformed by a per-element MLP with weights W_1 and biases b_1, followed by a tanh activation function:

$$\phi(x_i) = \tanh(W_1 E(x_i) + b_1).$$

To ensure that the network computes the sum, we require $\phi(x_i)$ to be proportional to x_i. This is achieved by setting W_1 and b_1 such that the tanh activation operates in its approximately linear region, avoiding saturation:

$$\phi(x_i) \approx a x_i,$$

where a is a small constant scaling factor.
The outputs $\phi(x_i)$ are then aggregated using a sum:

$$s = \sum_i \phi(x_i).$$

The aggregated sum s is passed through a final MLP layer with weights W_2 and bias b_2 to produce the output $\hat{y}$:

$$\hat{y} = W_2 s + b_2.$$

To obtain $\hat{y} = \sum_i x_i$, we adjust W_2 and b_2 to compensate for the scaling factor a:

$$W_2 = \frac{1}{a}, \quad b_2 = 0.$$

By learning these parameters, the DeepSets architecture effectively computes the sum of the multiset elements, satisfying the task objective.

Question 2

Let's define ϕ and ρ as:
1. **Element-wise transformation:**

$$\phi(x) = \text{ReLU}(Wx + b),$$

where $W = I$ (the identity matrix), $b = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, and ReLU is the rectified linear unit function applied element-wise.
- For X_1:

$$\phi\left(\begin{bmatrix} 1.2 \\ -0.7 \end{bmatrix}\right) = \text{ReLU}\left(\begin{bmatrix} 1.2 \\ -0.7 \end{bmatrix}\right) = \begin{bmatrix} 1.2 \\ 0 \end{bmatrix},$$
$$\phi\left(\begin{bmatrix} -0.8 \\ 0.5 \end{bmatrix}\right) = \text{ReLU}\left(\begin{bmatrix} -0.8 \\ 0.5 \end{bmatrix}\right) = \begin{bmatrix} 0 \\ 0.5 \end{bmatrix},$$
$$f(X_1) = \begin{bmatrix} 1.2 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0.5 \end{bmatrix} = \begin{bmatrix} 1.2 \\ 0.5 \end{bmatrix}.$$

- For X_2:

$$\phi\left(\begin{bmatrix}0.2\\-0.3\end{bmatrix}\right) = \text{ReLU}\left(\begin{bmatrix}0.2\\-0.3\end{bmatrix}\right) = \begin{bmatrix}0.2\\0\end{bmatrix},$$
$$\phi\left(\begin{bmatrix}0.2\\0.1\end{bmatrix}\right) = \text{ReLU}\left(\begin{bmatrix}0.2\\0.1\end{bmatrix}\right) = \begin{bmatrix}0.2\\0.1\end{bmatrix},$$
$$f(X_2) = \begin{bmatrix}0.2\\0\end{bmatrix} + \begin{bmatrix}0.2\\0.1\end{bmatrix} = \begin{bmatrix}0.4\\0.1\end{bmatrix}.$$

Since:

$$f(X_1) = \begin{bmatrix}1.2\\0.5\end{bmatrix} \neq \begin{bmatrix}0.4\\0.1\end{bmatrix} = f(X_2),$$

the embeddings of X_1 and X_2 are different.
By choosing appropriate weight matrices and bias vectors (here, $W = I$ and $b = \mathbf{0}$), we have shown that there exists a DeepSets model that maps X_1 and X_2 to different vectors.

Question 3

DeepSets can be part of a Graph Neural Network (GNN) for graph classification. In this context, graphs are treated as sets of node features, and DeepSets can aggregate these features into a single graph-level representation. By summing transformed node embeddings and applying another transformation, it ensures permutation invariance, which is crucial since the order of nodes in a graph doesn't matter. This makes DeepSets a natural fit for the pooling stage of GNNs, providing a flexible and learnable way to summarize graph information.

Question 4

The expected value of E, $\mathbb{E}[E]$, is given by:

$$\mathbb{E}[E] = \binom{n}{2} \cdot p = \frac{n(n-1)}{2} \cdot p$$

The variance of E, $\text{Var}(E)$, is given by:

$$\text{Var}(E) = \binom{n}{2} \cdot p \cdot (1-p)$$

For $p = 0.2$:

$$\mathbb{E}[E] = 105 \cdot 0.2 = 21$$
$$\text{Var}(E) = 105 \cdot 0.2 \cdot (1 - 0.2) = 105 \cdot 0.2 \cdot 0.8 = 16.8$$

For $p = 0.4$:

$$\mathbb{E}[E] = 105 \cdot 0.4 = 42$$
$$\text{Var}(E) = 105 \cdot 0.4 \cdot (1 - 0.4) = 105 \cdot 0.4 \cdot 0.6 = 25.2$$

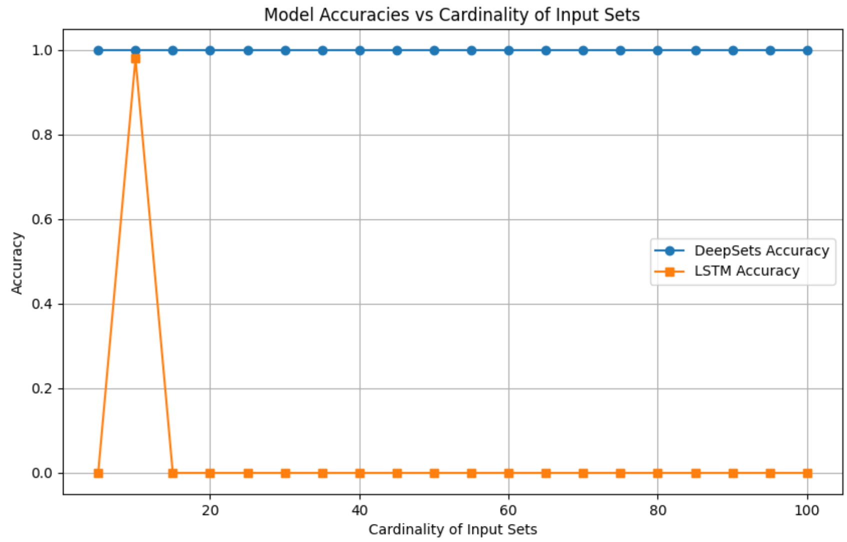

Figure 1: Model Accuracies vs Cardinality of Input Sets

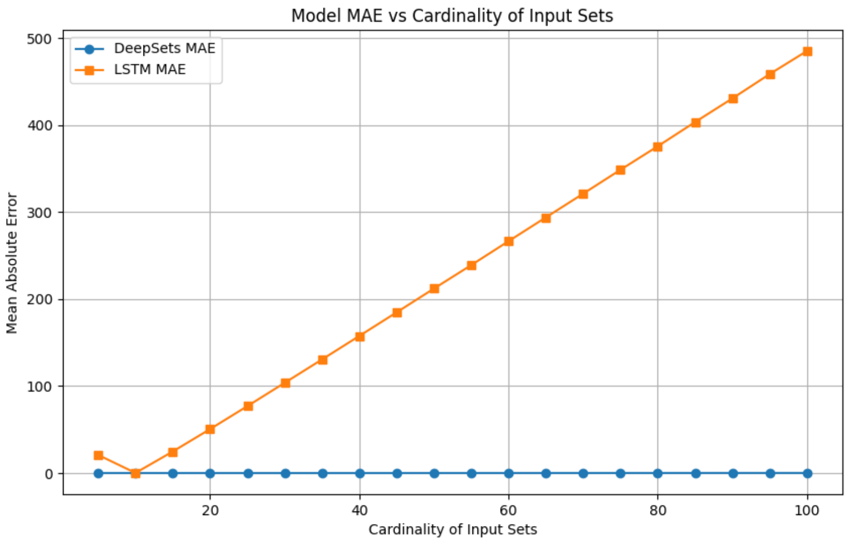

Figure 2: Model MAE vs Cardinality of Input Sets

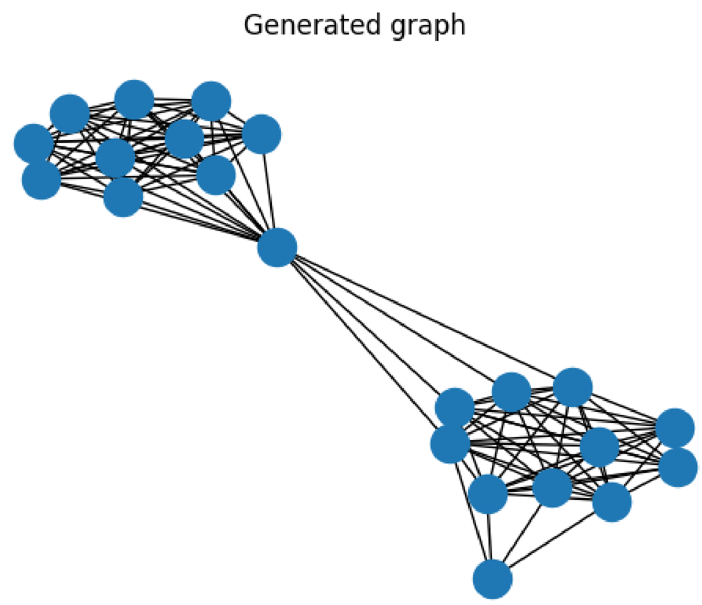

Figure 3: Graph generated