Practical Session 3

Graph Neural Networks

Graphs in Machine Learning
MVA Master Program – ENS Paris-Saclay

michal.valko@inria.fr
daniele.calandriello@inria.fr
omar.darwiche-domingues@inria.fr
pierre.perrault@inria.fr

About this Session

Complete the exercises and submit a written report.

1 Neural Relational Inference

This practical session is based on the paper *Neural Relational Inference for Interacting Systems* by Kipf et al., 2018.

We will use material provided by Marc Lelarge and Timothée Lacroix:

https://github.com/timlacroix/nri_practical_session

1.1 Motivation and Problem Formulation

A wide range of dynamical systems can be seen as a group of interacting components. For example, we can think of a set of 2-dimensional particles coupled by springs. Assume we are given only a set of trajectories of such an interacting dynamical system. How can we learn its dynamical model in an unsupervised way?

Key Information

Formal Setup:
We are given as input trajectories of N objects, each trajectory has length T.

- Each object i, for $i = 1, \ldots, N$, is represented by vertex v_i
- Let $\mathbf{x}_i^t$ be the feature vector of object i at time t (e.g., position and velocity) with dimension D
- Let $\mathbf{x}^t = \{\mathbf{x}_1^t, \ldots, \mathbf{x}_N^t\}$ be features of all N objects at time t
- Let $\mathbf{x}_i = (\mathbf{x}_i^1, \ldots, \mathbf{x}_i^T)$ be trajectory of object i
- Input data: 3D array $\mathbf{x}$ of shape $N \times T \times D$, denoted by $\mathbf{x} = (\mathbf{x}^1, \ldots, \mathbf{x}^T)$

The element $\mathbf{x}_{i,t,d}$ is the d-th component of the feature vector of object i at time t.

We assume the dynamics can be modeled by a graph neural network (GNN) given an unknown graph $\mathbf{z}$ where $\mathbf{z}_{i,j}$ represents the discrete edge type between objects v_i and v_j.

1.2 Learning Objectives

In this context, we want to learn simultaneously:

1. **Edge type estimation**: The edge types $\mathbf{z}_{i,j}$
2. **Future state prediction**: A model that, for any time t, takes $\mathbf{x}^t$ as input and predicts $\mathbf{x}^{t+1}$ as output

1.3 Model Architecture

The Neural Relational Inference (NRI) model consists of:

Key Information

NRI Components:

- **Encoder:** Uses trajectories $\mathbf{x} = (\mathbf{x}^1, \ldots, \mathbf{x}^T)$ to infer pairwise interaction vectors $\mathbf{z}_{i,j} \in \mathbb{R}^K$ for i, j in $\{1, \ldots, N\}$, where K is the number of *edge types*
- **Decoder:** Takes $\mathbf{x}^t$ and $\mathbf{z} = \{\mathbf{z}_{i,j}\}_{i,j}$ as input to infer $\mathbf{x}^{t+1}$

Both the encoder and the decoder are implemented using graph neural networks. For more details, read Section 3 of the paper here.

2 Exercises

Complete the code in the following notebook:

`https://github.com/timlacroix/nri_practical_session/blob/master/NRI_student.ipynb`

and answer the questions below in your report.

Note: For the report, no code submission is required. The Github repository contains a `solutions` folder, which you are allowed to use to complete the notebook.

Questions

1. Explain what are the edge types $\mathbf{z}_{i,j}$.
2. In the NRI model, explain how the encoder and the decoder work.
3. Explain the LSTM baseline used for joint trajectory prediction. Why is it important to have a "burn-in" phase?
4. Consider the training of the LSTM baseline. Notice that the negative log-likelihood is lower after the burn-in than before. Why is this surprising? Why is this happening?
5. Consider the problem of trajectory prediction. What are the advantages of the NRI model with respect to the LSTM baseline?
6. Consider the training of the NRI model. What do you notice about the edge accuracy during training? Why is this surprising?
7. What do you expect to happen with the NRI model when there is no interaction between the objects?