Assignment 2

by He Tan and Florian Westphal

Introduction

This assignment builds on what you have learned about Attention Mechanisms and Graph Neural Networks.

The general task is a simplified version of road user trajectory prediction. This task is relevant for example for
self-driving cars, which need to predict where the other traffic participants will be in order to navigate and
avoid collisions.

For this type of prediction, it is important to understand how the different road users are positioned to each
other. For example, one would expect a bicyclist to follow straight the bicycle track unless a pedestrian is standing
in the way. One way to encode these relationships between road users is a graph representation.

Given such a graph representation, as well as certain attributes about each road user, such as their current location,
type, speed and heading, you are supposed to predict their future location.

For this task, you are strongly advised to use this tutorial as starting point:
https://keras.io/examples/graph/gat_node_classification/ (https://keras.io/examples/graph/gat_node_classification/)

Dataset

Unfortunately, there is no publicly available traffic trajectory dataset.
Therefore, we are using a **pedestrian trajectory dataset (https://dil.atr.jp/crest2010_HRI/ATC_dataset/)** recorded in a Japanese mall.
Consequently, there are only pedestrians in the provided dataset.

As in the tutorial, the provided dataset consists of two files for each traffic scene:

- `<scene_id>.edges`
  - two columns containing node IDs
  - `target, source`
  - **Note:** The tutorial models directed edges with source -> target.
    You can either use undirected edges by changing the implementation or adding the missing entries to the edges file,
    e.g., to the line `target, source`, you add the line `source target`. If you want to be more

fancy, you could also try to infer
which other pedestrians the source node can see in their field of view and only add those (this would model that the movement
decisions are based only on the pedestrians in the field of view.)

- `<scene_id>.nodes`
  - seven columns with node properties and target values, which should be predicted
  - `node id, current x, current y, previous x, previous y, future x, future y`
  - the previous x and y represents the location of the pedestrian 1 second ago (you can use those values directly or infer the
    movement direction and some speed estimate yourself)
  - the future x and y represents the target value, i.e., the location where the pedestrian will be in 1 second
  - **Note:** Some pedestrians do not have a future x and y coordinate, so you need to filter those for prediction. However, you can
    still use their current and previous location when predicting the future location of other pedestrians.
- **Dataset (https://ju.instructure.com/courses/10280/files/1620918?wrap=1) (https://ju.instructure.com/courses/10280/files/1620918/download?download_frd=1)**

Assignment Tasks

Task1

Adjust the tutorial implementation to perform the given prediction task and perform a suitable evaluation
on a dedicated test set. For example, you can compute the Euclidean distance between the target point
and the predicted point.

Task 2

Perform hyperparameter tuning of the number of attention heads and try a deeper embedding of the node
features. For the attention heads, just evaluate 2 additional settings. For the embedding of the node features,
instead of the linear transformation of the node states as suggested in the tutorial, try to add one fully connected
layer with ReLU activation and one additional fully connected layer.

Task3

Replace the learned attention mechanism with an attention mechanism based on the Cosine similarity between
node vectors.

Deliverables

- Python script, which implements all three versions of the graph attention network (basic, with changed embedding,
  with changed attention mechanism).
- Short report (1-2 pages) in which you document the results of all three tasks

Quizzes

In order to pass this assignment you also need to **get at least 40% correct on the following quizzes:**

- Q8 - Generative Models
- Q9 - Deep Reinforcement Learning
- Q10 - Self-Supervised Learning
- Q11 - Computer Vision
- Q12 - Neurosymbolic Reasoning