RL Project Environment: Expando

Fabian Beringer

Concept

- n player, turn based strategy game
- players alternately place pieces on 2D grid by moving cursor
- two types of pieces: City, Farm
  - Farm: generate reward each turn if adjacent to city, **but** increase population
  - City: grant additional “room” for player
- player receives happiness penalty if population > room
- game ends after a specified number of turns, player with highest reward wins

State and Action representation

State: M * N * K + S dimensional vectors, with:

- M, N: grid size
- K: dimension of one-hot encoding of a piece
- S=4: x, y cursor coordinates, room and population score of a player -> continuous

Action:

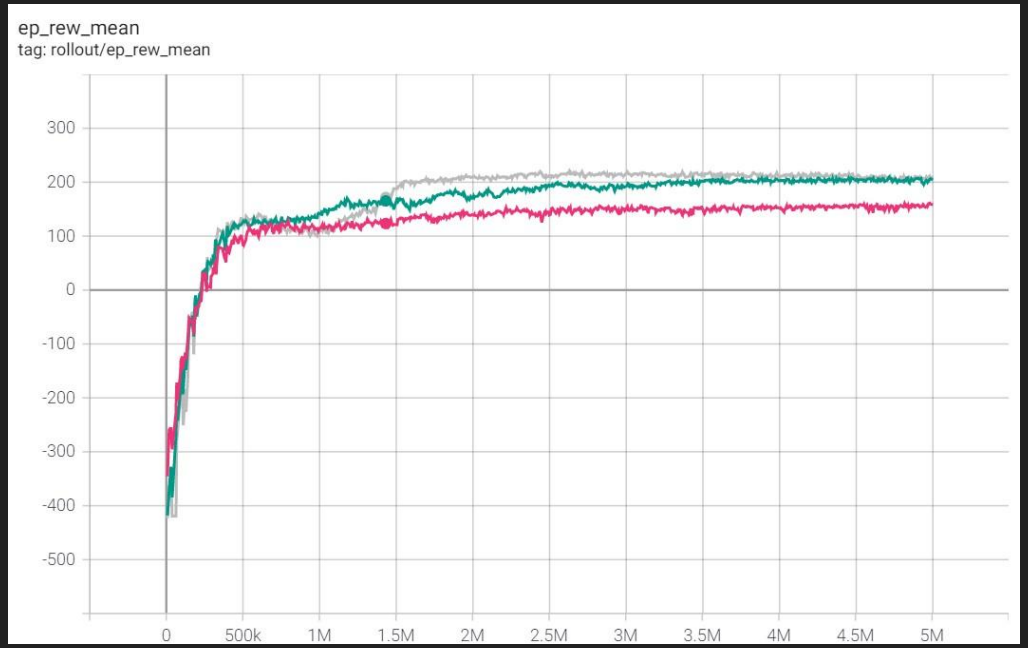
- Multidiscrete: (cursor_direction, piece_type)
- cursor_direction ∈ [UP, DOWN, LEFT, RIGHT, NO-OP]
- piece_type ∈ [Farm, City]
- Or: discrete integer encoding of cartesian product

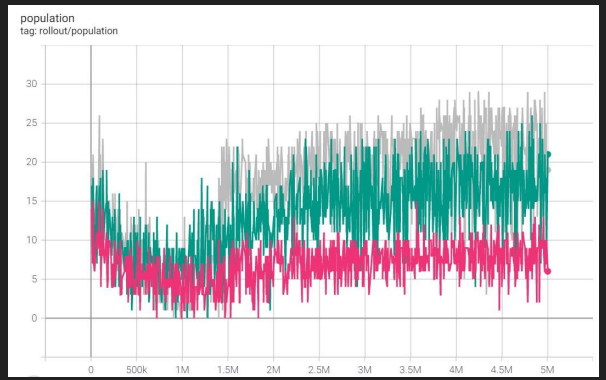
Implementation

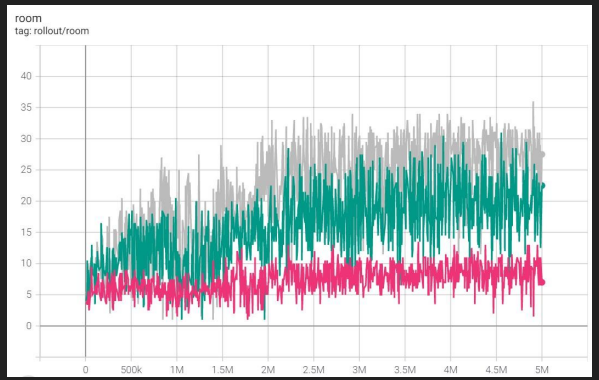
- environment configuration can be stored and loaded from yaml files (including random seed)
- amount of reward etc. fully customizable via configs
- can be extended with new piece types
- user can define all policies used in the environment and get observations from their perspective

libraries used:

- python + numpy, pyglet for visualisation
- stable-baselines3 library for the DQN implementation







Stable Baselines DQN trained against random policy on different grid sizes:

Pink: (8, 8)
Green: (15, 20)
Grey: (20, 30)

All DQN hyperparameters (and the environment setting) can be found in the config file in the git repo. Everything was set to the defaults, except batch size and learning rate were selected beforehand through a small grid search with:
lr: 1e-3, 1e-4, 1e-5
bs: 64, 128, 512

Random policy vs DQN trained against random

Links

https://github.com/yolomeus/rl_project

https://github.com/DLR-RM/stable-baselines3

https://github.com/facebookresearch/hydra