## **StoryWall Task**

This week you will experiment with reinforcement learning techniques and how parameters affect the behaviour of the trained agent. Your task is to train your computer to play the Atari 2600 game Gopher (you can see the full manual [here](https://atariage.com/manual_html_page.php?SoftwareLabelID=217) and can try playing it [here](https://www.retrogames.cz/play_094-Atari2600.php)) using a Deep Q Network.

I strongly recommend playing the game so you can get a feel for how it works, but here’s a brief summary. In the game you are a farmer trying to protect your carrots from a gopher who is digging holes on your farm. You can stop the gopher by filling up the holes it digs, or waiting until it reaches the surface and hitting it with your shovel. Filling up holes will give you 20 points, and hitting the gopher will give you 100 points. The game ends once the gopher has eaten all of your carrots.

The following [GitHub repository](https://github.com/tmadfouni/ACTL3143-Storywall) contains the code that you’ll need for this week and explains how it all works. For those interested, the main libraries used for this task are [Gymnasium](https://gymnasium.farama.org/), which provides a standardised interface for interacting with reinforcement learning environments and creates the Atari environment, and [Stable Baselines 3](https://stable-baselines3.readthedocs.io/en/master/), which provides a few reinforcement learning algorithm implementations.

As mentioned before, you’re trying to see how parameters affect the behaviour of the trained agent. More specifically for this task, by simply changing the discount rate (gamma value) in setup.py for DQN, you should be able to train agents that play the game differently. To train the agent once you’ve set a certain discount rate, you just need to run the training.py file. This will take a while, it took me around 8 hours on my GPU.

While it is training, the algorithm will regularly run evaluations on the current model to see how it is performing, so you can track the progress of its performance. Once the training is completed, or at least once an evaluation has been run, you can run evaluation.py, which will perform a manual evaluation and track the reward logs (which rewards were received by the agent and when), the mean reward, and episode length over the evaluation. It will also generate a recording of your agent playing the game. This is a fairly brief explanation of how the code works, if you want more detail, see the README in the GitHub.

Once you’ve run this for a couple of distinct discount rates, you can then investigate these evaluation outputs to see what behaviour each of your agents has and whether you’ve managed to create significantly different behaviours or performances. You’ll then write up a short response describing how you changed the parameters, what the different behaviours of your agents were, and how you determined those behaviours were different.

One last note, it would be great to see everyone running the code and seeing how the agent trains. However, the code does take quite a while to run. With that in mind, the GitHub already has a few different pre-trained agents inside it that you can run the evaluations on if you choose to. These are stored in the Results/DQN/Gopher directory.

Enjoy 😊

P.S. the code is written so that you can experiment with different algorithms and different Atari games. You can alter all of this in the setup.py file. For a full list of the games you can train on see <https://ale.farama.org/environments/>