REPORT TITLE HERE

Connor Wallis (22506057) & Reilly Evans (23615971)

**Export This As A PDF Before Submitting**

Introduction

One of the primary purposes of computers over their history has been to automate tasks that humans normally perform. Many different methods of creating agents for this purpose have been designed, from simple rule based programs to complex machine learning algorithms. In this report we will construct two agents for the task of playing levels from Super Mario Bros: one rule-based agent implemented by hand, and one Proximal Policy Optimisation (PPO) agent trained using stable baselines. We will compare the performance of these two agents …

-discussion of Mario

Our agents are Hand Implemented Rule based agent and PPO from Stable Baselines

When we make changes to existing code (Laurens, pytorch tutorial) document it!

‘poetry run nes\_py --rom super-mario-bros.nes --mode human’ for human controlled Mario

Analysis

-explain how RuleBased and PPO were constructed/trained

-compare reward value from gym

-perform other relevant experiments

Analyze and contrast the performance of the chosen AI methods.

• Discuss their respective strengths, weaknesses, and suitability for playing Super Mario Bros.

* Rule based is easy to understand; parameters can be tweaked easily to make it jump shorter, etc.,
* Rule based cannot adapt on its own; new palletes like underground or new enemies like Lakitus require adding them to the enemy recognition program, coding new actions to avoid them, etc.
* Experiments could include testing speed of level completion (be it time, frames or actions) or the amount of memory used; many of the listed examples don’t really work with a rule based agent that doesn’t learn

Performance Metrics

-points metric

-progression metric

You will notice that gym-super-mario-bros reward function assumes the objective of the game is to move as far right as possible. You are encouraged to come up with other performance and evaluation metrics for your agents. Novel and interesting metrics that you come up with will be rewarded.

* Points as a metric (collect coins/powerups gives lives and increases survivability; death penalty prevents infinite lives trick from causing problems
* Progression as a metric (get as far into the game as possible in terms of levels, or alternatively beat 8-4 from 1-1 as fast as possible; rewards finding the Warp Zones)
* Not Dying as a metric (get as far as possible without dying; rewards careful playing and prevents using the game’s checkpoints to cheese things
* Random Stages as a metric (use gym-super-mario-bros’s random stages function to try stages and measure number of stages out of say 10 completed)

Visualisation/Debugging

-rule based visualisation (terminal printing, Lauren’s code, freezing the game)

-PPO printing

Includewhatvisualizationtechniquesyouusedtogaininsightsintotheagent’sdecision- making process.

Include what debugging/profiling tools you utilised to optimize the algorithms and enhance performance.

* Besides what was already in Lauren’s code, printing to the terminal when decisions are made helps identify actions
* Freezing the game when a decision is made (via spamming the terminal with 250,000 messages) helps to identify exactly what constitutes a scenario where said decision is made
* B
* C

References

* Gym-super-mario-bros
* Lauren Gee’s mario-locate-objects code
* APA 7