Reinforcement Learning with Ms. Pacman: Moving Towards a Generalized Pathfinding Strategy

Laurel Bingham and Nicholas S. Flann

***Abstract*—Despite the emergence of Google's Deepmind AI, the Atari game, Ms. Pacman has remained one of the poorest performing games within the deep learning community. This is theorized to be due to the random nature of the ghosts, which forces the AI to abstract and learn strategy, rather than memorize a consistent path or pattern. There is a need to develop a deep learning agent capable of maximizing both the area explored by the agent, and safety of the agent by avoiding threats. Solving Ms. Pacman takes significant steps towards this goal, providing an environment which can be abstracted to real world pathfinding challenges. This paper applies two reinforcement learning strategies to this problem, namely the Deep Q-Network, much like the original Deepmind project, and the newer Double DQN model (DDQN). Steps are taken to find the smallest possible input size such that the model can learn the features of the dataset, while improving computation time. While the original Deepmind papers showed little to no improvement between the DQN and existing strategies, they did not attempt to tailor the hyper parameters or architecture of the network to the game Ms. Pac-Man. This paper attempts to do so by forcing the agent to explore out of local maxima by tuning both the minimum randomness and reward structure of the agent.**

***Key Words*—DQN, DDQN, Atari, Ms. PacMan**

I. Introduction

The goal of the game Ms. Pac-Man is to navigate a maze, where the player must attempt to eat all of the on-screen dots while avoiding the attacks of the four ghosts to advance to the next level. Most importantly, unlike many of the Atari games of this era, the movement of the ghosts is stochastic. According to the documentation of Ms. Pac-Man, "The ghosts do not move in 'scatter' and 'chase' cycles as they did in the original game; Blinky and Pinky will move randomly, while Inky and Sue will head for their "scatter" corners only during the first behavior mode of a round. From there, while they will change direction occasionally, they will remain in constant attack." [1] It is this randomness in movement that creates the challenge in Ms.Pac-Man that is not present in many of the other Atari games.

In 2013, a team at Deepmind published a paper on reinforcement learning within the Arcade Learning Environment (ALE), where they applied their deep learning approach, which they called a Deep Q-Network (DQN). This algorithm, tested on the Atari games Beam Rider, Breakout, Enduro, Pong, Q*bert, Seaquest, and Space Invaders, showing that the DQN approach outperformed traditional approaches on six out of the seven games, and outperformed human experts on three of them[2]. By 2015, Deepmind published a second paper in an expanded study, where Ms. Pac-Man was among the 49 games trained. While the DQN outperformed existing models on 46 of the 49 games trained, as well as outperforming human performance in 29 of the games, Ms. Pac-Man was among the 3 games which showed no improvement [3].

Notably, however, the goals of the deepmind studies were to show the general applicability of the DQN. The same hyper parameters were used for all models[3]. While the initial results of training a DQN on the game of Ms. Pac-Man showed no improvement, there was no attempt to tune the hyperparameters or architecture of the learning algorithm to the game Ms. Pacman specifically. That tuning is what this paper goes on to explore.

II. Methods

This study seeks to attempt to improve the performance of DQN and DDQN Ms. Pac-Man to beat the fully random strategy in two main ways. First, by tuning the minimum randomness of an agent's actions. The baseline 1% random action of an agent is compared against the tuned 5% random

978-1-4673-8988-4/17/$31.00

action chance. This is expected to force the agent to continue exploring and learning past the initial strategy of always choosing left to grab the first few pellets, then waiting against that wall for the rest of the game. If the agent is more consistently forced to turn up and down, it is hypothesized that eventually it should learn to continue exploring the map after gathering the reward from those initial pellets.

The other proposed solution to the local maxima problem is to alter the reward structure of the agent. Table 1 below shows the initial reward structure of the system:

TABLE I: Original Reward Structure

Action	**Reward**
Consumes Pellet	10
Consumes Power Pellet	50
Eats First Ghost	200
Eats Second Consecutive Ghost	400
Eats Third Consecutive Ghost	800
Eats Forth Consecutive Ghost	1600

In this original structure, rewards are given for eating pellets and consuming ghosts. Any no-op actions or movements in spaces without pellets are ignored by the reward system. Because the agent sees significant reward drops if it ever doubles back, or makes an attempt to move up or down in a space that does not allow it in it's initial moves, it tunes itself to ignore all moves except the one that initially gives it rewards. In the newly proposed reward structure, the no-op and double back moves are less harshly penalized. 90% of the reward given for eating a pellet is given in the case where the agent would receive nothing. The new reward system is shown implemented in the table below:

TABLE II: Adjusted Reward Structure

Action	**Reward**
Consumes Pellet	10
Consumes Power Pellet	50
Eats First Ghost	200
Eats Second Consecutive Ghost	400
Eats Third Consecutive Ghost	800
Eats Forth Consecutive Ghost	1600
Would Gain 0 Reward	9

With far less penalty for choosing an initially inefficient action, it was hypothesized that the agent would learn to explore the board- discovering strategies that allow it to gather many more pellets.
In total, six DQN and DDQN nets are tested. Per parameter adjusted, two nets are trained. The average score across 10 runs after training between the pair of nets is recorded for analysis. Of the trained nets, there is one DQN with 1% random movement and the original reward structure, a DQN with 5% random movement and the original reward structure, and a DQN with the altered reward structure and 5% random movement. Similarly, DDQNs of the same three structures are implemented. Finally, a random agent is included as a baseline to measure performance against.

Each net is trained for 2000 epochs. It should be recognized that this number of epochs is orders of magnitude lower than the Deepmind paper's training attempts, however 2000 was the largest number this study was capable of consistently testing. In initial runs, 10,000 epoch training cycles were tested, but little to no improvement was shown despite the extreme increases in training time. Other notable features include the usage of the Adam optimizer, and a learning rate of .003, which remained consistent across all runs.

III. Results

The performance of each model can be seen below.

TABLE III: Average Model Performance

Architecture	**Min Randomness**	**Reward**	**Average Score**
DQN	0.01	Default	130
DQN	0.05	Default	150
DQN	0.05	Adjusted	**190**
DDQN	0.01	Default	200
DDQN	0.05	Default	190
DDQN	0.05	Adjusted	**230**
Random Play	-	-	**180**

A. DQN Performance

The 1% random DQN with the original reward structure demonstrates the initial problem the nets encounter. A strategy worse than random movement was consistently found, where the agent ran left until it was killed by ghosts. Occasionally the random movement would push the ghost to move enough that it gathered additional points, but otherwise the agent would find itself trapped in a

corner quickly.

The 5% random agent found a similarly poor strategy. It exhibited more 'uncertain' appearing movements, where it would double back or wiggle around to the left and right, but it too moved into one of the corners, where it would wait for death after gathering only a few additional pellets.

Finally, of the DQNs, the agent with 5% minimum randomness and rewards performed the best. It began exploring the vertical paths in addition to running into the corners. While it still heavily favored lateral movement, and frequently backtracked, it was the first agent that showed a consistent tendency to explore the map to try to gather additional points after the agent shifted from explorative to exploitative strategy. However, this agent still just barely outperformed the random benchmark, showing that there was still much room to improve.

B. DDQN Performance

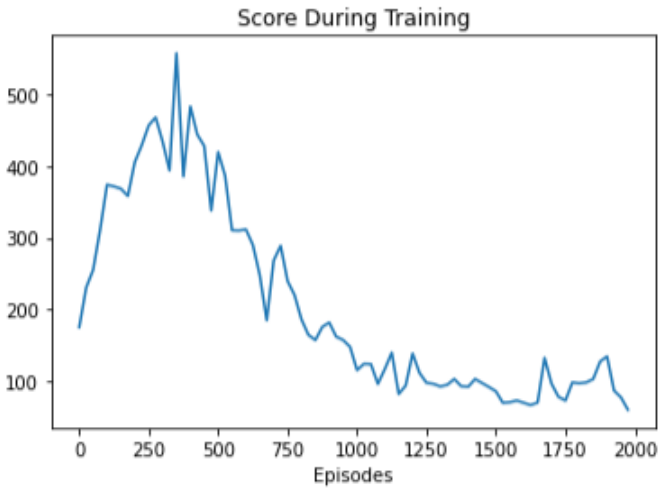
As expected, the deeper DDQN models consistently outperformed both the DQN models and the random benchmark, albeit by relatively small margins. However, while the points appear to only be a small amount larger, the change in how the points were gathered was the most notable feature of the DDQN agents. All agents showed some explorative tendencies.

The baseline DDQN still showed a very notable tendency to hide in the corner. However, the agent would occasionally attempt to move down once it reached the corner, gathering some extra points from the movement. This alone gave it consistently more points than the random movement model. The increased randomness DDQN with the base reward structure showed a similar strategy, though it performed on average slightly worse. It had more runs where it wiggled around in the starting area, though it also had more runs where it reached the very bottom corner of the map, where it could occasionally consume a power pellet. It had several outlier runs where it reached 700+ points by consuming a power pellet, and then catching one or two ghosts before the power-up ran out. Finally, the adjusted reward structure agent with the boosted randomness showed the best results, consistently scoring over 200 points, with an average of 230. This agent showed the most vaired movement. It explored vertically as well as horizontally. While it still showed some tendencies to wiggle in place or backtrack repeatedly, it also took a different path following each death in the game's cycle- the first of the agents to consistently show this behavior. In addition to pathing towards the power pellets in the corner, it also explored some of the maze's less direct routes, gathering the additional points along the way.

IV. Discussion

While the first challenge of training a Ms. Pac-Man agent was addressed- by finding an architecture and reward structure which forced exploration and strategy development past the first local maxima was faced, there are still significant hurdles to overcome to create an agent that can beat human play. The main issue found was that regardless of the reward structure or minimum randomness implemented on the agent- once the initial exploration period was over, the agent stopped learning new strategy quickly. Below shows the initial agent this problem was discovered on, the DQN with 1% randomness and base reward structure.

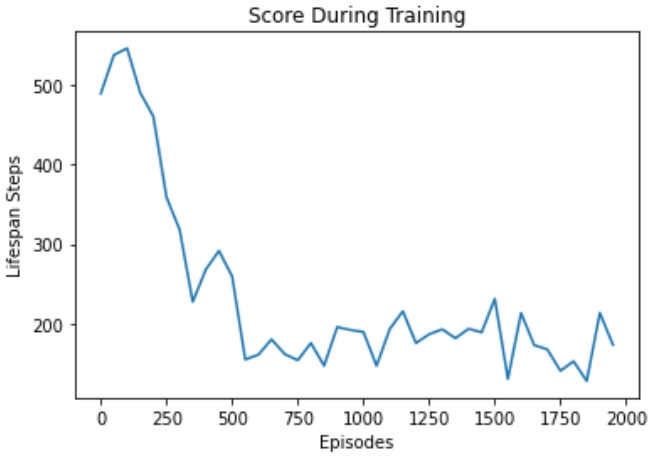
Fig. 1: Score Over Time from Initial DQN



Going off of the graph, the agent found it's optimal strategy while it was still choosing a random move approximately 30% of the time.

At that point, it was scoring at average over 500 points. As it learned more, it's score dropped until it hit it's average 130 points. While the 500 point score is much better than the models reported here after training, a model with 20-40% random movement is not considered a good agent by the authors of this study. A successful agent should be using the information around it consistently to make decisions, rather than relying heavily on chance. The goal then, of the study was to attempt to overcome this massive drop in score by the time the agent was functioning on it's own strategy. However, this was not accomplished. Even the best performing agent from the DDQN showed a similar pattern, which can be seen below in Figure 2. In this case, the agent learned the 500 point strategy faster, and more quickly appears to forget it in favor of its own strategy.

Fig. 2: Score Over Time from Best DDQN Model



However, in both cases, by the time the minimum randomness is reached (approximately 550 epochs at 5 % percent randomness, or around 950 for 1% randomness), the agent reaches an apparently flat-lined score that no longer improves over time. The initial DQN, at least, showed no score improvement in this even when trained for an additional 8000 epochs. It is possible, however, that if the DDQN were trained for that length of time the score could improve. That was not tested in this study. In either case, the Deepmind papers trained for millions of iterations. It could be possible that both architectures could show score improvement if extended out for that amount of training. Unfortunately, it is beyond the scope of this study to test that theory.

V. Conclusions

Overall, it was found that increasing the randomness of an agent, did increase the agent's average score. The more impactful effect, however, was to change the reward structure of the agent. The reward structure was where the most obvious penalties for explorative strategies appeared. By changing this, the agent became more free to explore side paths, double back, and eventually learn to explore different routes on each life.

This study was successful in beating the fully random benchmark strategy with strategies learned by the agents. While these learned strategies were still less successful than the high percentage random strategies seen during the training period, the improvements shown with adjusted hyperparameters show promising leads for future work. While the DQN and DDQN approaches have not yet shown greater performance than human play, with additional training time, they may have the potential to reach those benchmarks.

References

[1] "Ms. Pac-Man (game)," Pac-Man Wiki. https://pacman.fandom.com/wiki/Ms.Pac-Man(game)
[2] Mnih, V., Kavukcuoglu, K., Silver, D. et al. "Playing Atari with Deep Reinforcement Learning." ArXiv abs/1312.5602 (2013): n. pag.
[3] Mnih, V., Kavukcuoglu, K., Silver, D. et al. Human-level control through deep reinforcement learning. Nature 518, 529–533 (2015). https://doi.org/10.1038/nature14236
[4] L. Griswold, "Deep split Q-learning and ms. Pacman," Medium, 11-May-2021. [Online]. Available: https://towardsdatascience.com/deep-split-q-learning-and-ms-pacman-5749791d55c8.