How effective are different configurations of a neural network reinforcement learning in playing space invaders?

Introduction

Artificial intelligence (AI) is any computer performing tasks typically requiring a human to operate. These tasks can vary from a wide range of difficulties and goals. Similarly, AI comes in many forms; in this experiment, the focus is on Machine learning. Machine learning is a branch of AI that primarily deals with building models to predict patterns within data. Machine learning, which is a subdomain of artificial intelligence, relies on a multitude of statistical techniques to find patterns and predict outcomes (Portilla, 2023a).

Reinforcement Learning

Reinforcement learning is a subset of machine learning. There are multiple types of machine learning, with reinforcement learning being one of them. Unlike other types of machine learning, reinforcement learning does not rely on large amounts of previously labeled data to make the model (Mnih et al., 2013). Instead, the model, or "agent," learns in a given environment through observations and rewards. The learning process starts with the agent in an "environment" where the agent gets trained. The agent performs an action and then takes in an "observation" and a "reward" value. The observation gives the state of the environment (such as an image of what is happening), and the reward allows the agent to determine how good (or bad) it is doing. This process works in a cycle: the agent performs an action, the agent takes in an observation and reward, and then the agent updates its policy, or way of thinking. With enough repetitions of this cycle, one can hope the agent becomes competent enough to achieve desirable results (Portilla, 2023c).

There are many models to run this reinforcement learning cycle, such as Q-learning. The main idea of Q-learning is that in an environment with a set number of conditions and a set number of actions, theoretically all of the state/action combinations are able to be mapped onto a data table, such as in the game of tic-tac-toe. In each of the cells of a table, there is a Q-Value. The Q-Value is the expected sum of all future rewards, with higher Q-Values meaning an action is more desirable to take. For each state the agent comes across, it could just look up the state in the table and take the action with the best Q value. As the agent experiments with the environment, it adjusts the Q values using its policy. By training the agent, said Q table can be filled out, and the environment could be considered “solved,” with all possible states already having optimal actions that would lead to the highest reward (Portilla 2023c). In the initial training process with the table being empty, the initial actions would have to be chosen purely randomly. As the table gets filled out, the agent starts getting the choice between choosing random actions and choosing actions that it thinks would lead to the best outcome. This problem is called exploration vs exploitation. There are pros and cons for both exploration (randomly chosen action) and exploitation (optimal known action). If an agent solely explored, it would never be able to apply what it is learning and end up with very bad results. On the flip side, if an agent only exploited, it would only use one solution to a problem, without any chance to find a better solution. This balance between exploration and exploitation is determined by a value called “epsilon”, the probability that the agent will take a random action. Epsilon can have a value between1 and 0, with 1 being a 100% chance of the action taken being random, and 0 meaning the actions will always be what the agent decides. By slowly decreasing epsilon throughout the training period, the agent will start to take less random actions as time goes on. This is part of the learning process, with the agent gradually using what it has learned over making random actions. (Portilla, 2023b).

In this experiment, Deep Q-Learning is used for the reinforcement learning agent. Deep Q-Learning, or DQN, is a combination of both Q-Learning and an artificial neural network. Instead of using a table-based method to make decisions, a DQN agent utilizes neural networks to analyze and make decisions about the environment around it.

Artificial Neural Network

An artificial neural network can be thought of as a human brain. In a human brain, there are millions of interconnected cells, called neurons. Each neuron has an input and an output, which are connected to other neurons. Through this connected network, the neurons send signals to each other, and the brain can make decisions. Though this is an oversimplification of how the brain works, it models out basically what a neural network is in terms of machine learning. Instead of cells, the computer uses a series of equations, each of which feed into each other. Through the process of machine learning, the computer adjusts the values within the equations and the connections between the neurons. As the computer is training, it adjusts these parameters so that the network will output desirable numbers, or decisions.

Experiment

Neural Network

The experiment is set up to test 25 different neural network models. There are 5 different shapes, with 5 different numbers of layers. The shapes, arbitrarily named, are as follows: Line, Cone, Inverse Cone, Hourglass, and Inverse Hourglass. Each shape describes the formation of the neurons in the neural network. The line shape is the simplest one, with a constant number of neurons in each layer. The cone has an increasing increment of neurons in each layer, creating a cone shape. Inversely, the inverse cone has a decreasing number of neurons in each layer. The hourglass, much akin to an hourglass, starts out with a wider layer, then it shrinks in the middle, and then the layer widens again. Following this convention, the inverse hourglass starts out with a smaller layer, widens, and then shrinks again. The number of neurons per layer is easier visualized the following table:

	3 Layers			4 Layers				5 Layers					6 Layers						7 Layers						
Line	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128
Cone	8	16	32	8	16	32	64	8	16	32	64	128	8	16	32	64	128	256	8	16	32	64	128	256	512
Inverse Cone	32	16	8	64	32	16	8	128	64	32	16	8	256	128	64	32	16	8	512	256	128	64	32	16	8
HourGlass	64	32	64	64	32	32	64	128	64	32	64	128	128	64	32	32	64	128	256	128	64	32	64	128	256
Inverse Hourglass	32	64	32	32	64	64	32	32	64	128	64	32	32	64	128	128	64	32	32	64	128	256	128	64	32

Table 1: Number of neurons per in each layer of the neural network (left to right).

	3 Layers	4 layers	5 Layers	6 Layers	7 Layers
Line	384	512	640	768	896
Cone	56	120	248	504	1,016
Inverse Cone	56	120	248	504	1,016
HourGlass	160	192	416	448	928
Inverse Hourglass	128	192	320	448	704

Table 2: Total number of neurons per neural network shape

This assortment of neural networks enables the testing of multiple aspects: the shape of a neural network, the number of layers in a neural network, and the number of neurons in a network. Neurons on each layer are a multiple of two for optimization. Each shape was made to have about the same number of neurons as possible, but some shapes end up having more due to their shape.

Environment

The environment in which the agents will be trained on is the Space Invaders environment. In Space Invaders, the player controls a laser cannon, which has 3 actions: move left, move right, and fire. The goal of the game is to gain as many points as possible by firing the laser cannon at descending alien invaders. Each invader destroyed awards the player with points ranging from 5 to 30, increasing depending on the row that they are in. There is an additional seventh type of invader, which occasionally flies across the top of the screen and awards a bonus 200 points. The aliens come in waves, each wave consisting of 6 rows with 6 aliens. Each time a wave of invaders is wiped, a new one will spawn. The player loses a life if either the aliens reach the bottom of the screen or if the player gets hit by one of the bombs dropped by the aliens. The game ends if the player loses all 3 lives.

Agent and Policy

Each of the DQN agents are trained for 1,000,000 steps, or frames. On each step of a Space Invaders game, the agent has 6 possible actions, including doing nothing. The first 100,000 are treated as a warm-up. In this warm-up phase, the epsilon value for the agent will not decrease. This allows for the agent to have some data of how its environment works, without the agent starting its learning process. Every episode the epsilon value is multiplied by 0.99, until it reaches the minimum value of 0.1, from a starting value of 1.

Software and Hardware

To set up the experiment between the different shapes of neural networks, multiple python libraries were used, all of them using python version 3.7. A python library contains modules, each made to complete a smaller sub-task. This makes it much easier to make a reinforcement learning agent, or any other goal. Rather than making a system for a reinforcement learning agent from scratch, instead it is much easier to build off of existing libraries.

The primary library used in the experiment was Tensorflow and Keras. Both of these libraries handle the creation of the neural network and the reinforcement learning agent.

OpenAI Gym library was used to create the environment for the reinforcement learning agent. The gym library utilizes ROMs for atari games, found at atarimania.com.

The program runs on Windows 10, using Visual Studio Code. It uses Jupyter Notebooks, allowing for the code to be run piece by piece. All of this was run on a personal computer with an IntelCore i5-9600K CPU @ 3.70GHz.

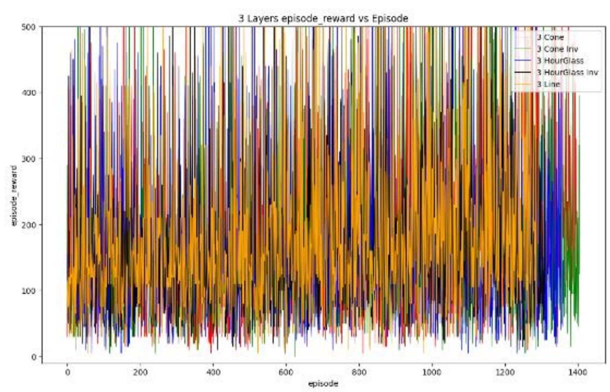
Results

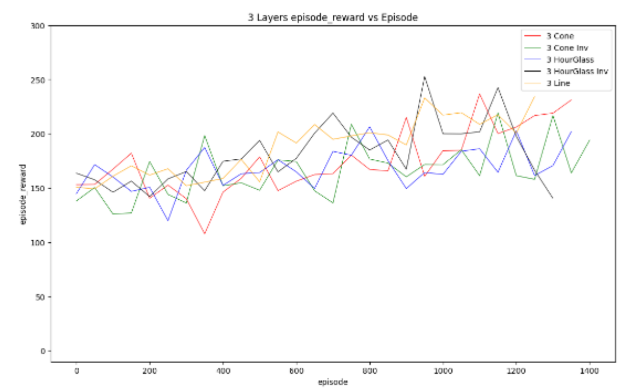
After each episode, the program returns a lot of metrics, which are as follows: loss, mean absolute error, mean_q, mean_eps, episode_reward, nb_episode_steps, nb_steps, episode, and duration. Of these, the ones that really matter are the mean Q and episode reward.

The mean Q, as mentioned in the introduction, is the average expected reward the model will obtain in a with the actions it is taking. This makes it possible to determine how well the model is doing in terms of maximizing future reward potential.

The reward is similar to this; however, it is the actual number of points that the agent obtained. Unlike the mean_q value, this value fluctuates greatly, due to exploration by the agent. If the agent was constantly exploiting, the mean_q value would be similar, if not the same as the reward value. However, the exploration that the agent does isn’t always optimal, often resulting in games where the agent completely fails to gain any points. This in turn creates massive spikes in the original reward graphs, shooting up and down from 0 to 100 every other episode. To mitigate this problem and make the graph more stable and readable, the reward data for each 50th cell is equal to the average reward of the next 50 cells. For example, the reward value for episode 500 would be the average value of the rewards from episodes 500 to 549. This turns the unreadable graph in *Figure 2* to a cleaner and legible version of the data in *Figure 1*.

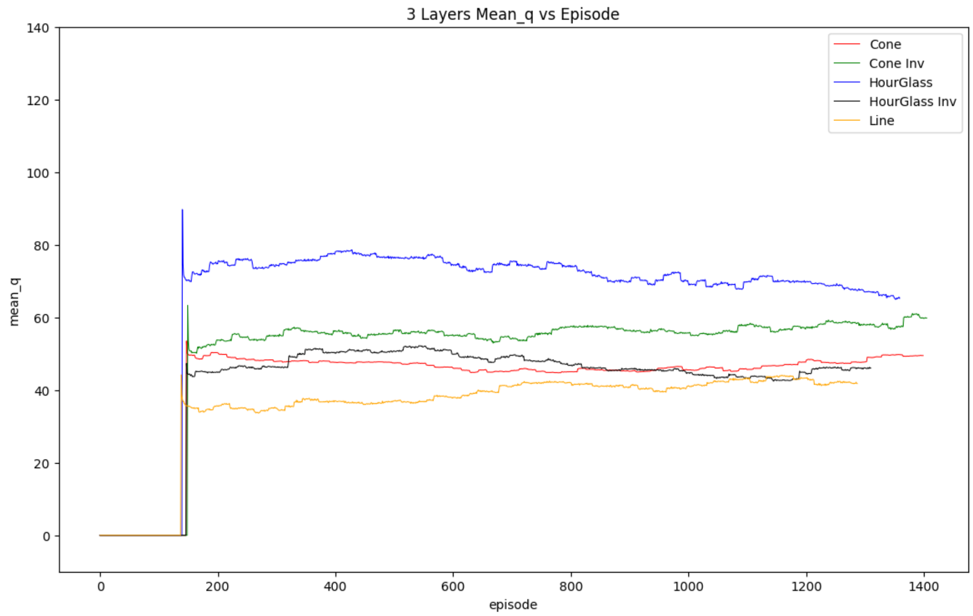
Each of these metrics is measured per episode. The “episode” is each full game of space invaders that the agent plays. Each episode has a varying number of steps, which depends on how long the agent plays. Longer games would naturally lead to more steps. The only limit to how long a game can be is the number of steps that the agent is training for.





3 Layers

Table : Mean Q vs Episode for 3 layers



pisode	Cone L3	ConeInv L	HourG L3	HourGInv	Line L3
0	0	0	0	0	0
50	0	0	0	0	0
100	0	0	0	0	0
150	49.80096	53.67492	70.13706	44.38628	35.4796
200	50.26249	53.72271	74.75035	44.82931	35.57109
250	48.35322	54.62948	75.95137	46.13649	34.6241
300	47.87392	54.90076	74.33812	46.28964	35.02094
350	48.08	56.24974	75.5528	51.21829	37.55039
400	47.65132	56.17435	78.08507	50.30345	37.02182
450	47.64711	54.36033	77.2494	50.98055	36.06189
500	47.41787	56.39069	76.58305	50.77335	36.94671
550	47.32821	55.82466	76.02878	51.91707	36.91693
600	45.98331	55.12111	74.50695	49.90765	37.87582
650	45.51676	54.03781	73.37886	48.54795	39.82614
700	45.32297	54.11669	75.27284	49.59852	41.02008
750	45.16499	55.68828	73.39721	47.81562	41.80854
800	45.02508	57.58176	73.86476	46.81155	42.05348
850	45.55653	57.51816	71.16296	46.209	41.02093
900	45.15554	56.38839	70.11914	45.84248	40.74875
950	46.032	56.75329	70.25867	45.37123	39.73948
1000	45.51752	55.75058	69.34284	44.69021	41.20069
1050	46.03319	55.63816	69.11534	43.45248	42.44404
1100	45.63868	57.86346	69.82653	43.35138	43.22609
1150	46.65488	56.13016	69.75498	42.57866	43.80999
1200	47.93921	57.2314	69.75375	44.42846	43.25565
1250	47.51217	58.47574	68.77866	46.18483	42.14039
1300	47.72071	57.96316	67.03223	46.09607	
1350	49.66063	56.98285	65.36573		
1400		59.68542			

In the trial with 3 layers, all of the layers had a mean_q value within the range of about 40-70, a range of 30 mean_q points. There isn't much variance in the mean_q values, with the ranking of highest to lowest being: Hourglass, Cone Inverse, Cone, Hourglass Inverse, and Line.

The reward values vary wildly, due to the epsilon randomness as mentioned previously. By episode 1250, there was a range of about 70 points, from 234 to 158. The shape with the highest reward was the Line, with 234.3 points. After that, the next highest is the Cone, with 216.8 points total. Following it is Hourglass Inverse, with 166.4 points and Hourglass, with

Figure : Reward vs Episode for 3 layers

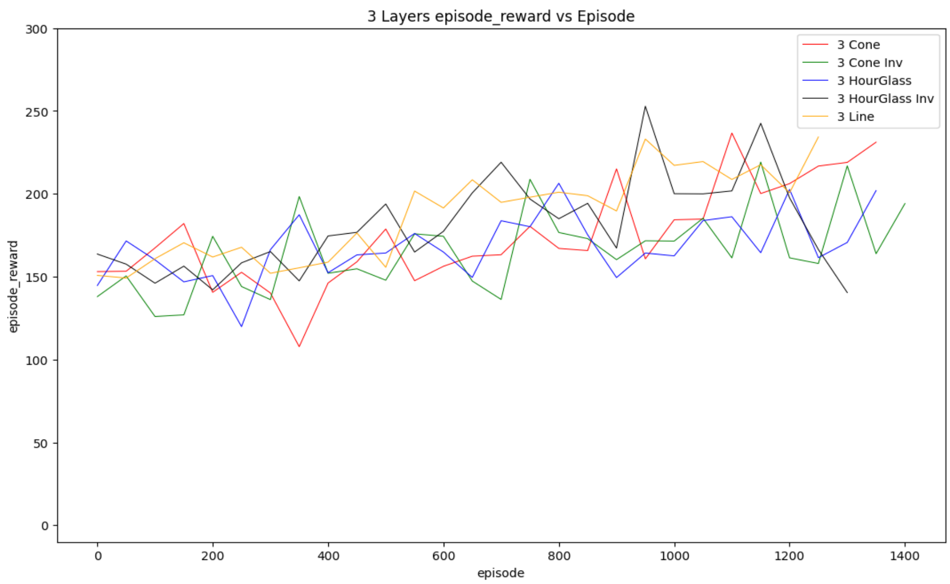


Table : Reward vs Episode for 3 layers

episode	3 Cone	3 Cone Inv	3 HourGla	3 HourGla	3 Line
0	153.1	138	144.7	163.7	150.8
50	153.3	150.5	171.6	157.6	149.2
100	167.4	126	160.2	146.1	160.9
150	182.1	127	146.9	156.4	170.5
200	140.6	174.4	150.7	142.1	161.9
250	152.7	144.1	119.9	158.4	167.8
300	140.2	136.2	166.3	165.2	152.1
350	107.8	198.4	187.4	147.5	155.4
400	146.2	152.1	152.4	174.6	158.8
450	159	154.8	163.2	176.8	176.4
500	178.8	147.9	164.3	193.9	155.7
550	147.6	175.9	176.1	164.9	201.7
600	156.3	174.4	164.9	177.4	191.5
650	162.4	147.4	149.6	200.6	208.5
700	163.3	136.3	183.8	219.1	194.9
750	180.4	208.8	180.2	196.9	198
800	167.1	176.7	206.4	185	201
850	165.8	173	175.3	194.3	198.9
900	215.1	160.3	149.5	167.3	189.7
950	160.8	171.7	164.3	252.9	233.1
1000	184.4	171.5	162.6	200.1	217.2
1050	184.8	185.3	183.9	200	219.5
1100	236.7	161.4	186.2	201.8	208.7
1150	200.2	219.2	164.5	242.6	217.6
1200	206.2	161.4	202.7	197.6	200.7
1250	216.8	158	161.5	166.4	234.3421
1300	219	216.9	170.7	140.4545	
1350	231.2	163.9	202		
1400		194.1667			

161.5 points. Cone

Inverse is in last place with 158 points. The rankings for points were: Line, Cone, Hourglass Inverse, Hourglass, and Cone Inverse.

4 Layers

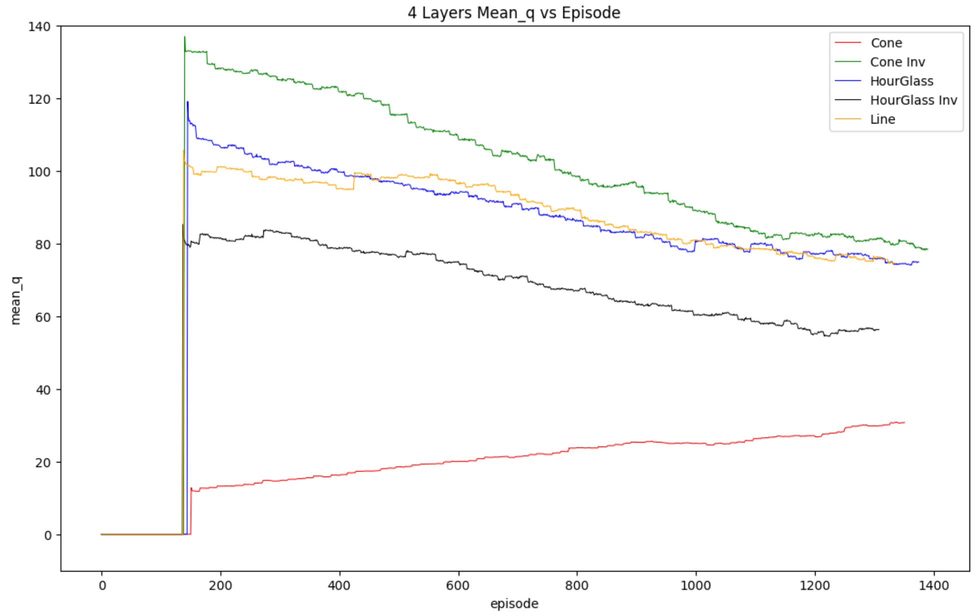


Table : Mean Q vs Episode for 4 layers

episode	Cone L4	ConeInv L	HourG L4	HourGInv	Line L4
0	0	0	0	0	0
50	0	0	0	0	0
100	0	0	0	0	0
150	0	132.8967	112.7839	79.3189	100.9918
200	13.27792	127.9435	106.5162	81.59983	101.0178
250	13.67948	127.3267	104.7838	81.94883	99.76888
300	14.67626	124.9718	102.3568	83.05201	97.7493
350	15.56244	123.9103	100.8359	81.28409	96.73761
400	16.23098	121.6693	99.66191	78.9544	95.11283
450	17.42373	119.7251	98.37873	77.5714	98.90981
500	18.57258	115.3805	96.45022	76.09059	98.85583
550	19.30232	111.3351	94.53222	77.10629	98.17405
600	20.07859	109.384	93.84384	75.07297	96.36163
650	21.20098	105.7822	91.4998	71.44739	96.47832
700	21.36285	105.056	90.87684	70.50069	93.59208
750	21.96216	102.9692	88.02268	67.97479	90.41776
800	23.84226	98.50131	86.46761	66.90276	89.15114
850	24.12611	95.47411	83.44232	65.80533	85.26221
900	25.35414	95.69492	81.46557	63.23215	83.57449
950	25.10349	93.79899	80.49031	62.70362	82.52144
1000	25.03151	89.12784	80.62026	60.38181	81.03654
1050	24.84282	85.82601	80.07104	60.90616	79.45175
1100	26.32839	83.50413	80.00714	58.4	78.54123
1150	26.95342	81.26017	77.39605	58.64081	77.03039
1200	27.00879	82.78471	77.24337	55.23132	75.83159
1250	28.80532	81.91178	76.42851	56.24861	76.86696
1300	29.76252	81.1354	75.84217	55.98946	76.00033
1350	30.69794	80.78181	74.40258		
1400					

In the 4 layer trials, there was more variance between the mean_q values, with the cone at the bottom. By episode 1250, the cone only had a mean_q value a little under 29. This is really small, in comparison to the other shapes, ranging from 56 to 81. Overall, there is a range of 50, including the outlier, and a range of 30, when disregarding the cone shape. Past about episode 375, the Hourglass and Line follow similar trends, within 10 mean_q of each other. Towards the end, the Line shape had about 0.2 more mean_q than the Hourglass, a 76.4 versus a 76.8. The ranking of the shapes, from highest to lowest is: Cone Inverse, Line, Hourglass, Hourglass Inverse, and Cone.

The final reward values are quite spread out. At episode 1250, the highest reward value is 242.2 points by Hourglass Inverse. The lowest reward value is 167 points, by Cone Inverse. This

Figure : Reward vs Episode for 4 layers

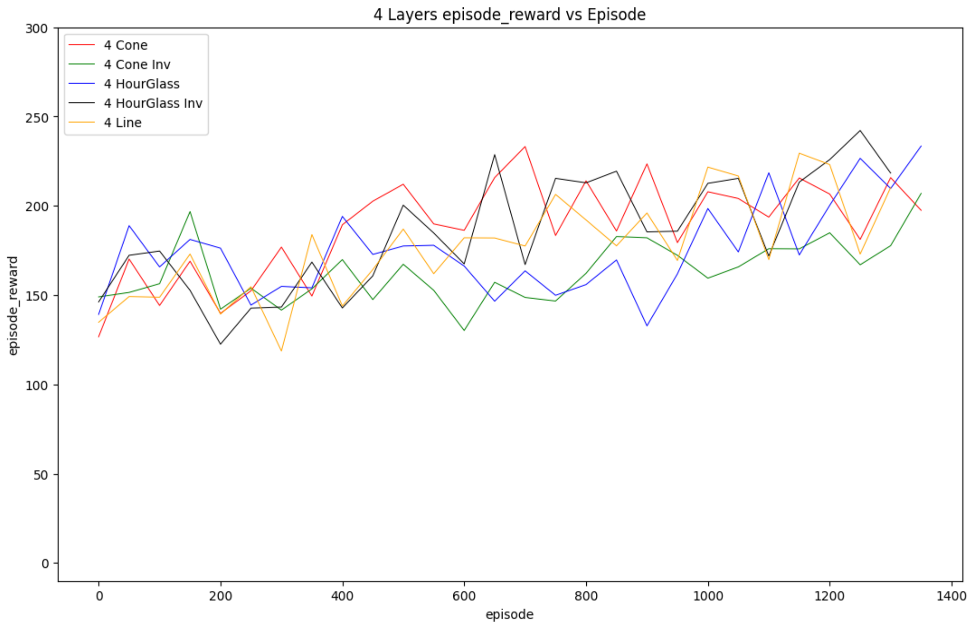


Table : Reward vs Episode for 4 layers

episode	4 Cone	4 Cone Inv	4 HourGla	4 HourGla	4 Line
0	126.7	149	139.2	146.2	134.8
50	170.2	151.5	188.9	172.3	149.2
100	144.2	156.4	165.8	174.7	148.8
150	169	196.8	181.2	152.7	173
200	139.8	142.1	176.3	122.5	139.4
250	152.4	153.9	144.4	142.7	154.7
300	176.9	141.6	154.9	143.3	118.7
350	149.5	153.5	154.1	168.5	183.9
400	189.4	169.9	194.1	142.8	143.9
450	202.6	147.5	172.8	160.8	164.3
500	212.1	167.3	177.5	200.4	187
550	189.9	152.7	177.9	184.8	162
600	186.3	130.2	166.3	167.5	182.1
650	215.8	157.2	146.6	228.7	182
700	233.2	148.7	163.6	167.1	177.5
750	183.4	146.7	149.9	215.4	206.4
800	214	162.2	155.9	212.9	192.1
850	185.9	182.8	169.7	219.4	177.6
900	223.5	182.1	132.8	185.4	196
950	179.4	172.3	161.9	185.8	169.4
1000	207.9	159.5	198.5	212.6	221.7
1050	204.1	165.8	174.2	215.4	216.7
1100	193.7	176	218.5	171.9	169.9
1150	215.6	175.9	172.5	213.3	229.5
1200	206.6	184.9	200.3	225.9	223.1
1250	181.2	167	226.6	242.2	173
1300	215.8	177.7	209.7	218.3333	210.2857
1350	197.5	206.9512	233.4615		
1400					

results in a range of 75 points between all the shapes. The final ranking for the shapes was: Hourglass Inverse, Hourglass, Cone, Line, and lastly Cone Inverse.

5 Layers

Figure : Mean Q vs Episode for 5 layers

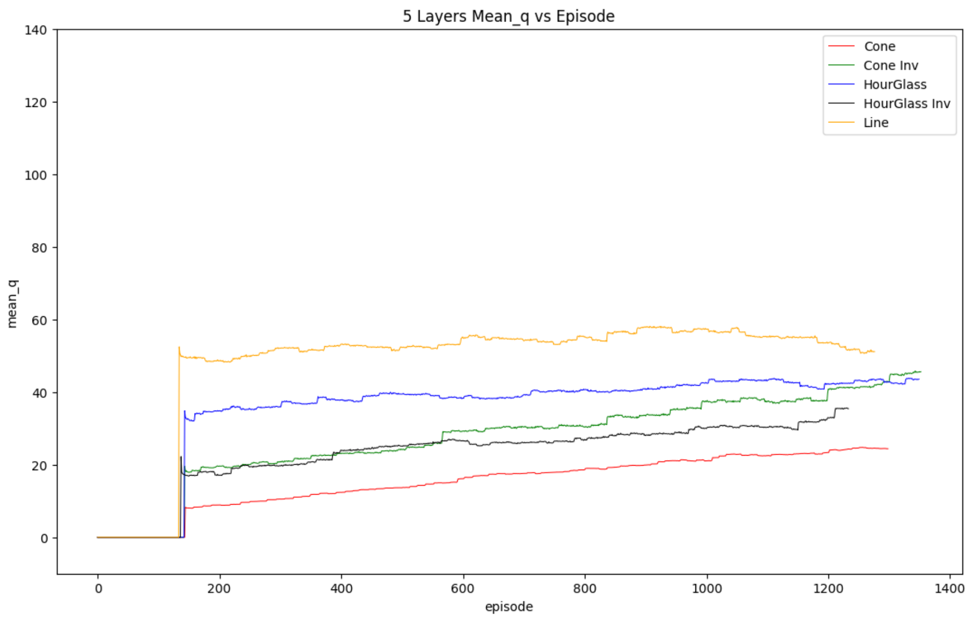


Table : Mean Q vs Episode for 5 layers

episode	Cone L5	ConeInv L	HourG L5	HourGInv	Line L5
0	0	0	0	0	0
50	0	0	0	0	0
100	0	0	0	0	0
150	8.056392	17.95869	32.4201	16.84772	49.49234
200	8.904922	19.63029	34.83655	17.07463	48.57996
250	9.76433	20.04585	35.32563	19.46817	49.75383
300	10.5097	20.70029	35.83996	19.71388	52.09201
350	11.75959	21.75148	36.85542	20.70778	51.57311
400	12.40356	23.02654	37.78649	23.99329	53.20378
450	13.21706	23.25316	39.06893	24.5667	52.27029
500	13.64286	24.10117	39.42125	25.1303	52.32247
550	14.65682	25.36806	39.04459	25.80097	51.85539
600	16.13572	29.2593	38.33887	26.64475	55.11264
650	17.38945	30.02003	38.27948	26.11775	54.96644
700	17.58602	30.32359	39.26839	26.02247	54.09015
750	17.90531	30.03906	40.348	26.49818	53.17264
800	18.62104	30.45956	40.64107	26.95988	55.2564
850	19.26873	33.6167	40.33277	28.08489	56.67418
900	19.88463	33.39932	40.68533	28.00877	57.96232
950	21.10257	35.22448	41.73456	28.88183	56.40161
1000	21.06585	37.36953	42.50061	29.93981	57.26405
1050	22.81383	37.34842	42.94269	30.29056	57.78848
1100	22.46211	37.31393	43.17355	30.34855	55.28828
1150	22.99005	37.87744	42.54039	29.52994	54.94064
1200	23.33786	40.81104	42.31749	32.93101	53.58232
1250	24.58736	41.18224	43.20077		51.80332
1300		42.8642	42.70116		
1350		45.62401			
1400					

The 5 layer trials had more distinct and clear separation between the shapes. By episode 1250, the mean_q values ranged from 24 to 50, a range of 25 between the mean_q values. However, the Hourglass Inverse never made it to 1250 episodes, possibly due to longer games in general. The Cone Inverse started around the same as the Hourglass Inverse, but then climbed to the level of the Hourglass. Every other shape had a much slower increase in mean_q. The final rankings of the shapes are (from highest to lowest): Cone, Cone Inverse, Hourglass, Hourglass Inverse, and Line.

The final reward values are quite spread out. The only shape that didn't make it to episode 1250 was Hourglass Inverse. This isn't necessarily a bad thing; it just means that it had longer games that took up more frames. Interestingly, by episode 1200, the highest reward value is 266.9 points by Hourglass Inverse. The lowest reward value is 168.9 points, by Cone Inverse.

Figure : Reward vs Episode for 5 layers

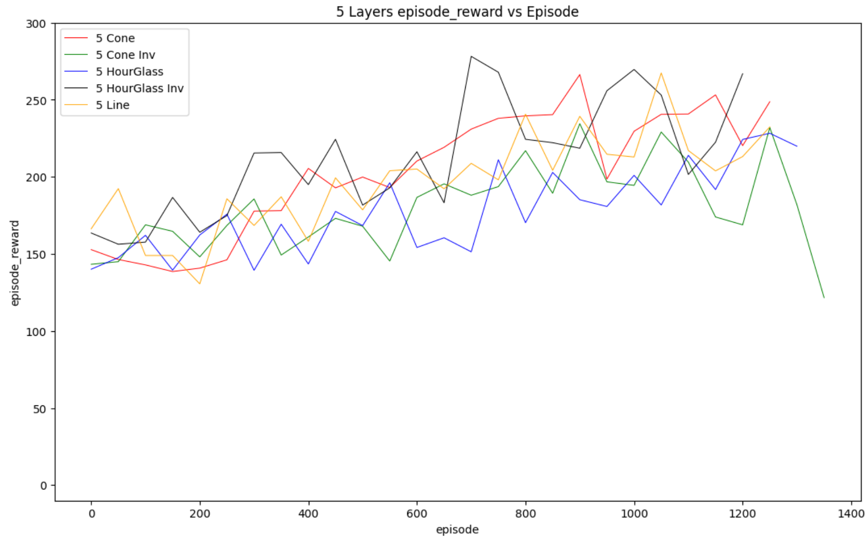
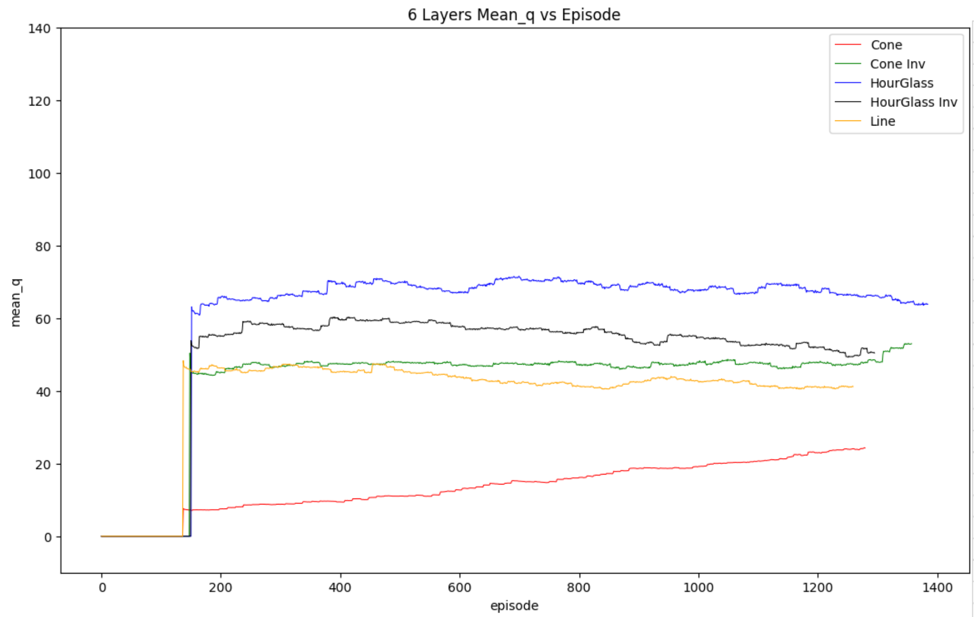


Table : Reward vs Episode for 6 layers

episode	5 Cone	5 Cone Inv	5 HourGlass	5 HourGlass	5 Line
0	152.8	143.3	140.1	163.6	166.3
50	146.4	145	147.6	156.3	192.4
100	142.9	168.9	162.1	157.7	149
150	138.6	164.7	139.5	186.7	149
200	140.8	148.1	162.3	164.1	130.6
250	146.2	168.5	175.8	174.9	185.8
300	177.8	185.7	139.4	215.5	168.5
350	178.1	149.3	169.3	215.8	187.1
400	205.5	161	143.5	195	158.2
450	192.9	173.1	177.6	224.4	199.4
500	199.9	168	168.4	181.7	178.6
550	193.1	145.4	196.2	192.9	204
600	210.3	186.7	154.2	216.3	205.1
650	219.2	195.5	160.5	183.2	192.3
700	231	188.1	151.4	278.2	208.8
750	238	193.7	211.1	267.9	198.1
800	239.6	217	170.3	224.4	240.7
850	240.4	189.4	202.9	222.2	204.3
900	266.4	234.5	185.2	218.6	239.3
950	198.6	196.8	180.8	255.9	214.7
1000	229.6	194.5	201	269.7	212.9
1050	240.6	229.1	181.8	253.1	267.4
1100	240.8	209.6	214	201.6	217.1
1150	253.2	174	191.8	222.5	203.9
1200	220.3	168.9	224.3	266.911765	213.2
1250	248.77551	232	228.3		232.407407
1300		181.9	219.9		
1350		121.666667			
1400					

This results in a range of 98 points between all the shapes. The final ranking for the shapes was: Hourglass Inverse, Hourglass, Cone, Line, and lastly Cone Inverse.

6 Layers



ean Q vs Episode for 6 layers

episode	Cone L6	ConeInv L	HourG L6	HourGInv	Line L6
0	0	0	0	0	0
50	0	0	0	0	0
100	0	0	0	0	0
150	7.24504	45.33889	0	53.8129	45.34111
200	7.512554	45.17693	65.88946	55.15736	46.25269
250	8.584582	47.35928	64.96603	58.94073	45.55804
300	8.768366	46.70022	66.0514	58.30571	46.69989
350	9.429469	47.99236	67.24793	57.08979	46.41361
400	9.439787	47.48636	69.4995	59.57943	45.37047
450	10.63745	47.22135	69.85202	59.4293	44.96562
500	11.06904	47.86468	69.77532	58.56676	45.73747
550	11.00384	47.85215	68.14716	59.30718	44.99841
600	12.69535	47.07211	67.01381	57.70129	43.8848
650	14.09992	46.76255	69.1371	57.33026	42.52969
700	15.1383	47.28965	71.40614	56.9417	42.22117
750	14.95023	46.96228	70.60328	56.5116	41.84553
800	16.17094	47.06876	69.53366	56.01641	41.30788
850	17.24008	47.1759	68.29411	56.1747	40.51455
900	18.68126	46.37615	69.2699	53.2937	42.39082
950	18.66667	47.37786	68.45356	55.33424	43.39823
1000	19.16295	47.93846	67.77895	54.65373	42.47778
1050	20.20064	48.38036	68.02104	53.48896	42.61387
1100	20.68533	47.5535	68.16944	52.55783	41.2691
1150	21.71957	46.26229	68.89005	52.42575	40.75182
1200	22.94984	47.89613	67.33531	51.80691	40.94419
1250	23.96666	47.48042	66.33021	49.39271	41.24804
1300		48.04267	66.22807		
1350		52.82664	64.27374		
1400					

The 6 layer trials again had more variance in the mean_q values. The lowest, Cone, had a mean_q value of 24, and the highest, Hourglass, had a mean_q value of 66. This results in a range of 40 between all of the shapes. Excluding the outlier, cone, there is a range of 25 between all of the shapes. The ranking of the shapes is as follows: Hourglass, Cone Inverse, Hourglass Inverse, Line, and Cone.

For the 6 layers, the reward values are more consistent with each other. Most of the shapes stopped by episode 1250. By then, the highest reward value is 251 points by Line. The lowest reward value is 192 points, by Cone. This results in a range of 59 points between all the

Figure : Reward vs Episode for 6 layers

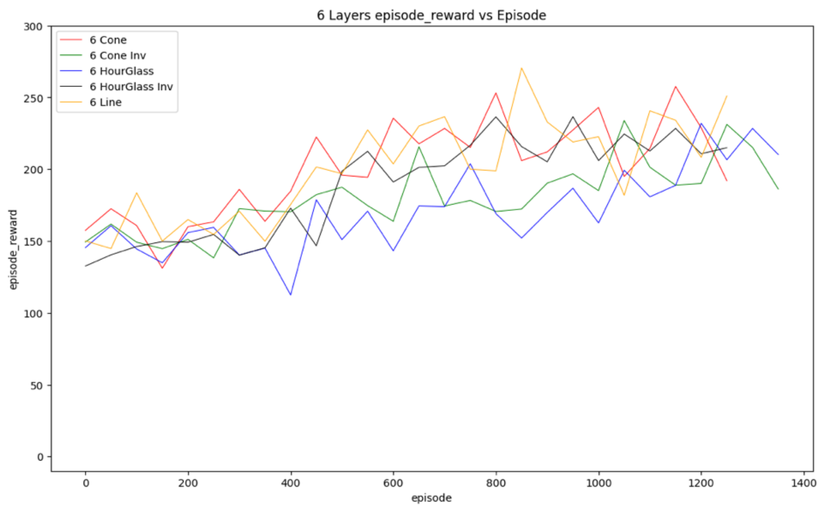


Table : Reward vs Episode for 6 layers

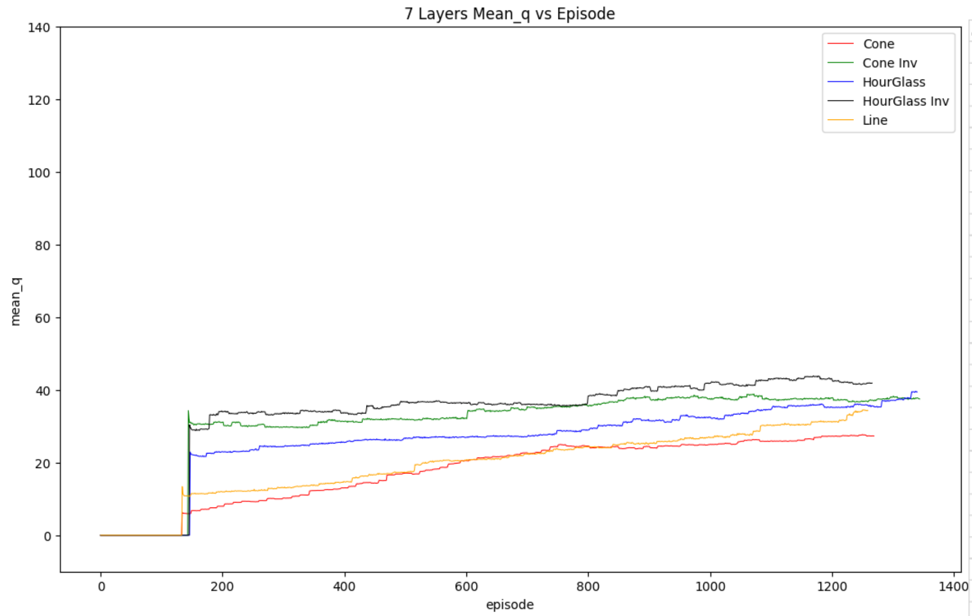
episode	6 Cone	6 Cone Inv	6 HourGlass	6 HourGlass	6 Line
0	157.4	149.3	145.4	132.6	150.1
50	172.5	161.8	160.7	140.4	144.8
100	160.8	149.2	144.5	146.2	183.6
150	131.1	144.7	134.9	149.6	150
200	160	151.2	155.9	149.2	165
250	163.4	138.3	159.5	154.6	154.6
300	186	172.6	140.3	140.2	170.8
350	163.8	170.9	145.1	145.3	149.9
400	184.7	170.4	112.5	172.9	175.5
450	222.5	182.3	178.8	146.7	201.6
500	195.9	187.5	151	198.6	196.9
550	194.4	174.7	170.8	212.5	227.4
600	235.6	163.8	143.2	191.1	203.7
650	217.7	215.7	174.5	201.3	230.1
700	228.4	174.4	174	202.4	236.6
750	215.2	178.3	203.8	216.7	200
800	253.2	170.6	169.1	236.5	198.9
850	206	172.3	152.1	215.8	270.5
900	212.1	190.3	169.9	205.2	232.9
950	227.3	196.8	186.8	236.6	219
1000	243.1	185.1	162.6	206	222.7
1050	195	233.9	199.3	224.5	181.9
1100	214.5	201.4	180.8	212.7	240.7
1150	257.6	188.9	188.8	228.4	234.1
1200	228.9	190.1	232	210.8	208.3
1250	192	231.2	206.6	215	251
1300		215.2	228.4		
1350		186.25	210.285714		
1400					

shapes. The final ranking for the shapes was: Line, Cone Inverse, Hourglass Inverse, Hourglass, and finally Cone.

Figure : Mean Q vs Episode for 7 layers

Table : Mean Q vs Episode for 7 layers

7 Layers



episode	Cone L7	ConeInv L	HourG L7	HourGInv	Line L7
0	0	0	0	0	0
50	0	0	0	0	0
100	0	0	0	0	0
150	6.724948	30.65914	22.18342	29.03689	11.4323
200	7.978594	31.07061	22.98269	33.94058	11.88394
250	9.227761	30.05942	23.47495	33.36958	12.29691
300	10.22549	29.76909	24.31047	33.54129	13.08587
350	12.25601	30.31083	25.17122	34.19765	13.88494
400	13.03151	31.26368	25.49889	33.65302	14.70331
450	14.39542	31.77541	26.32633	34.83183	16.63853
500	17.06823	31.82943	26.58233	36.6499	17.26861
550	18.18613	32.05547	27.17889	36.82598	20.34611
600	20.49776	32.16796	26.93047	36.46839	20.70177
650	21.56469	34.26051	27.34772	35.97501	21.0794
700	22.67471	35.33319	27.13374	35.92899	22.27185
750	24.36212	35.43221	28.83728	35.81604	23.76649
800	24.53803	35.4772	28.9626	38.28857	24.33365
850	24.09765	37.19291	30.2432	39.84471	24.82411
900	24.4236	37.95245	31.46073	40.81954	25.19816
950	24.46247	37.96158	31.8324	40.94156	26.25446
1000	24.97275	37.6224	32.48808	41.82786	26.82028
1050	25.8132	37.90705	33.37862	41.11379	27.7835
1100	25.93267	37.37449	34.64328	43.0329	30.23779
1150	26.13472	37.51183	35.58726	42.99851	30.42658
1200	27.26282	37.72671	35.26533	42.98187	31.10318
1250	27.70599	36.77588	35.87446	41.55604	34.42745
1300		38.05954	37.13112		
1350					
1400					

For the 7 layer trials, there was a much closer range between the shapes. The lowest, Cone, has a mean_q value of 27, and the highest, Hourglass Inverse has a mean_q value of 41. This results in a range of about 14. The final ranking of the shapes are as follows: Hourglass Inverse, Cone Inverse, Hourglass, Line, and Cone. It is notable that towards the end The Cone Inverse and Hourglass had about the same Q values, 38 and 37 respectively.

For the 7 layers, the reward values are all close to each other. Similar to the 6 layer shapes, most shapes stopped by episode 1250. By then, the highest reward value is 255.8 points by Hourglass. The lowest reward value is 204.7 points, by Cone. This results in a range of 51

Figure : Reward vs Episode for 7 layers

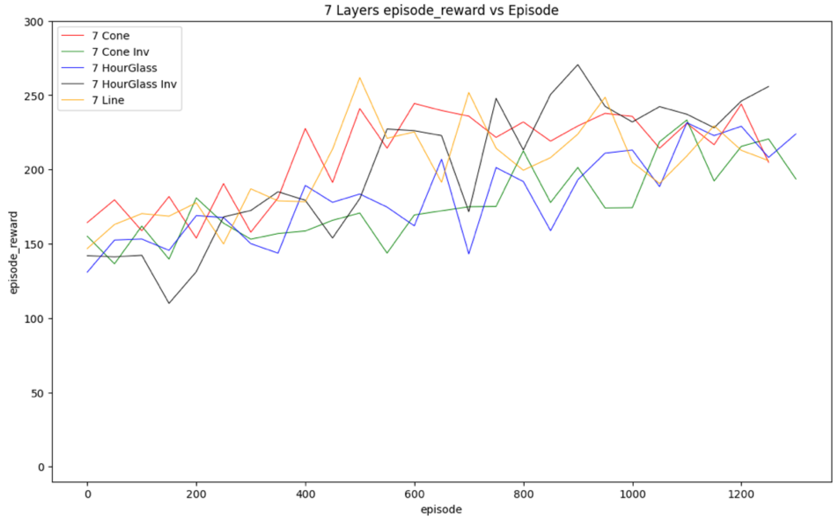


Table : Reward vs Episode for 7 layers

episode	7 Cone	7 Cone Inv	7 HourGlass	7 HourGlass	7 Line
0	164.3	155.1	130.9	142	146.8
50	179.6	136.5	152.5	141.2	163
100	158.8	161.9	153.2	142.3	170.3
150	181.8	139.7	145.6	109.9	168.6
200	153.8	180.9	169	131.2	177.6
250	190.5	163.9	167.7	168.1	149.9
300	157.8	153.1	150.1	172.4	187
350	180.9	156.9	143.7	185	178.9
400	227.6	158.6	189.3	179.3	178.3
450	191.3	165.8	177.9	153.9	213.7
500	241	170.7	183.5	180.4	261.8
550	214.3	143.7	174.7	227.2	221
600	244.4	169.4	162.1	226.1	225.3
650	239.7	172.2	206.9	222.8	191.4
700	235.9	174.9	143.2	171.6	251.8
750	221.6	175.1	201.3	247.9	214.3
800	232	212.5	191.9	213.2	199.5
850	219.1	177.8	158.8	250.5	208
900	229.3	201.4	193.1	270.6	223.8
950	237.8	174.1	211	242.5	248.6
1000	235.7	174.3	213.1	232	204.9
1050	214.3	218.7	188.5	242.3	190.6
1100	231.1	233.3	231.5	237.2	209.1
1150	216.7	192.3	222.8	228.1	229.4
1200	244	215.6	229.1	246.1	212.9
1250	204.75	220.6	208.1	255.882353	206
1300		193.666667	223.902439		
1350					
1400					

points between all the shapes. The final ranking for the shapes was: Hourglass Inverse, Cone Inverse, Hourglass, Line, and lastly Cone.

Discussion

Based on these results, the effectiveness of each of the shapes can be analyzed.

Ranking	3 Layer	4 Layer	5 Layer	6 Layer	7 Layer
1	HourGlass	Cone Inv	Cone	HourGlass	HourGlass Inv
2	Cone Inv	Line	Cone Inv	Cone Inv	Cone Inv
3	Cone	HourGlass	HourGlass	HourGlass Inv	HourGlass
4	HourGlass Inv	HourGlass Inv	Hourglass Inv	Line	Line
5	Line	Cone	Line	Cone	Cone

Table : Q Value rankings of each shape by number of layers

The most straight forward analysis would be of the placements that each shape placed in terms of Q values.

Rankings	Cone	Cone Inv	HourGlass	HourGlass Inv	Line	First
1	1	1	2	1	0	HourGlass
2	0	4	0	0	1	Cone Inv
3	1	0	3	1	0	HourGlass
4	0	0	0	3	2	HourGlass Inv
5	3	0	0	0	2	Cone

Looking at how many times each shape placed, it can be possible to determine what shape consistently ranks above the others. If a ranking consistently has one shape, it can be determined that the shape is really ranked that place against the other shapes. Alternatively, if a rank has multiple shapes evenly, that rank is likely to be up to chance.

In the number one spot, each shape took it at least once, except for line. This would lead to the conclusion that first place is mostly up to chance, and can't lead to any meaningful result.

For second place, there is a more conclusive result. Cone Inverse placed in second place 4 times. The only other shape in rank 2 is Line. This would mean that Cone Inverse is second overall.

For third rank the majority shape is Hourglass, placing 3 times. The other two shapes that placed in 3rd are Cone and Hourglass Inverse. Though not as certain of a placement as the 2nd place, Hourglass can be placed as 3rd rank overall.

For rank 4, Hourglass Inverse placed 3 times and line placed twice. This would place Hourglass as rank 4. Rank 5 is similar, with Cone placing 3 times and Line placing twice.

The only shape that hasn't consistently placed in a rank is Line. On average, it was in multiple ranks, scattered across the board. This would point to the success of Line being more based on the randomness of exploration, rather than actually being a more effective shape. Due to this, Line would most appropriately be placed in last place.

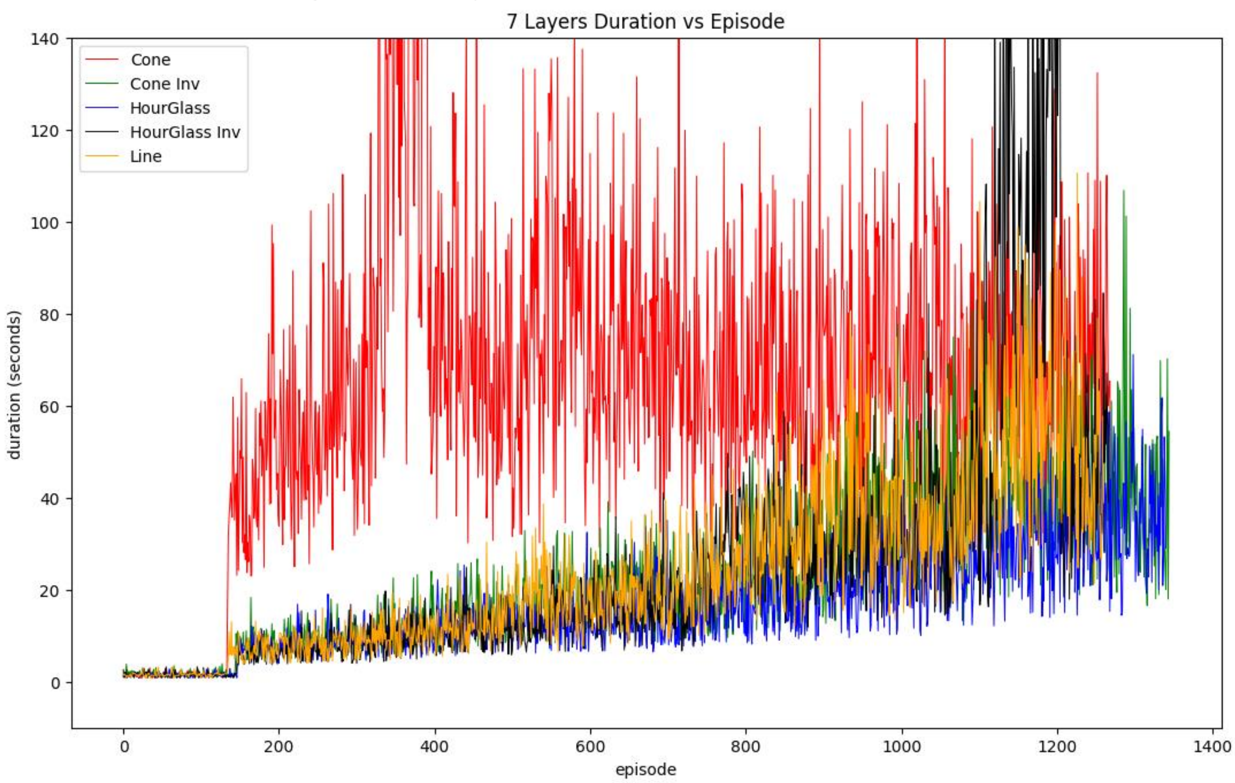
From this, the rankings for the shapes can be in the following descending order: Cone Inverse, Hourglass, Hourglass Inverse, Cone, and lastly, Line.

Limitations and Improvements

One of the biggest flaws in this experiment was the number of episodes that the agents played of space invaders. Though 1,000,000 frames might sound like a lot, for a reinforcement learning agent it simply isn't enough for it to play at a proficient level. Even though the game seems relatively simple for a human to play, the agent requires tens of millions of episodes to begin to recognize the gameplay patterns and play decently. At minimum, it would require about 10,000,000 frames or even 25,000,000 to 50,000,000 frames to really get it great at the game. The solution to not having enough episodes is, obviously, to tell the agent to run for a couple million more episodes. However, there are two problems with this: processing power and time.

Training the reinforcement learning agent is a very CPU demanding task. It needs to run both the agent and the game. Training one agent for 1,000,000 episodes can take from 10 hours upwards of around 24 hours. Multiplying this number by at least 10 for 10,000,000 episodes results in a very unreasonable length for testing one model, let alone 25. The increase isn’t linear either. The less random actions the agent takes (as epsilon gets lower and lower), the more thinking the computer needs to calculate, and the longer the agent takes. This means that a game from episode 900,000 with 0.2 epsilon would take significantly longer than a game from episode 50,000, where all the actions are random. As demonstrated in figure 3, the first couple hundred episodes take a significantly shorter amount of time before it increases exponentially. So, the increase in time depends on the epsilon value of the model, meaning that as epsilon decreases, the training time increases. Note that the outlier for cone is due to other processes running on the computer and taking up processing power.

This problem with time can be solved with more processing power, with either multiple computers (which would divide the total training time significantly, as multiple computers can train multiple models simultaneously) or with faster CPUs (which would train each model faster). This is easiest achieved with cloud computing, which rents processing power by the hour. However, the cost to rent computing power for 25 models, for a model with 1,000,000 frames, can be at least $250 with AWS. Adding another couple million would increase the time, and cost, to train the models significantly, possibly with the lower end CPUs being about $300 for 2 weeks, and the higher end CPUs going up to possibly $500. In the end, the biggest limitation to this experiment was budget deficiency.



References

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). *Playing atari with deep reinforcement learning*. https://arxiv.org/abs/1312.5602

Portilla, J. (2023, April 1). *Practical AI with python and reinforcement learning video 25 supervised learning process* [Lecture transcript]. Udemy. Retrieved September 7, 2023, from https://www.udemy.com/course/practical-ai-with-python-and-reinforcement-learning/learn/lecture/27288080#overview

Portilla, J. (2023, April 1). *Practical AI with python and reinforcement learning video 95 q-learning theory - part four - programmatic Q updates* [Speech transcript]. Udemy. Retrieved August 28, 2023, from https://www.udemy.com/course/practical-ai-with-python-and-reinforcement-learning/learn/lecture/27122258#overview

Portilla, J. (2023, April 1). *Practical AI with python and reinforcement learning video 92 q-learning theory - part 1 - table intution* [Lecture transcript]. Udemy. Retrieved August 28, 2023, from https://www.udemy.com/course/practical-ai-with-python-and-reinforcement-learning/learn/lecture/27122252#overview