Data Science Initiative
BROWN

Deep Q-Learning for Atari River Raid

Yash Mehta, Yuetian Li, Yingfei Hong, Nange Li

Introduction

We implemented Deep Q-Learning Networks (DQN) to play Atari 2000 River Raid with two strategies: fixed Q-targets and replay memory mechanism to handle the oscillation in training due to shifting Q target values and ensure efficient training. We also introduced custom reward function. Our goal is to train the agent to perform 'much' better than the random model.

Methodology

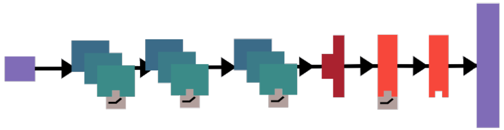
1. Preprocessing: converted RGB game frames to grayscale, resized the frame to 84×84 pixels; apply k-skipping by concatenating k=4 consecutive frames for the model to predict one action, and repeat this action k times to get four observations
2. Custom modifications in reward functions: We added a large negative death reward to train agent to avoid death. Rescaled and normalized the rewards so gradient calculations were more consistent and allowed stable training.
3. Fixed Q target value at every time step allows for stable learning:

$$Q(s,a,w) = r(s,a) + \gamma \max_a \widehat{Q}(s',a,w')$$

where s is the current state, s' is the next state after taking the action a, r is the reward for taking the action a, γ represents the discounted factor, Q is the Q network, Q hat is the target network and every T steps: w' = w.
4. Replay Memory mechanism uses past experiences to update the Q network in addition to the update in each time step. Given a fixed number of past experiences, we will randomly sample a small batch of experiences from the buffer. For each experience, the error will be:

$$error = r + \gamma \max_a \widehat{Q}(s',a) - Q(s,a)$$

Model Architecture



	Channels	Filter	Stride	Activation	#parameters
Convolutional layer 1	32	12	4	ReLU	18464
Convolutional layer 2	64	6	2	ReLU	73792
Convolutional layer 3	64	4	1	ReLU	65600
	Hidden size				
Dense layer1	640			ReLU	656000
Dense layer2 (output)	18 (Number of actions)			None	11538

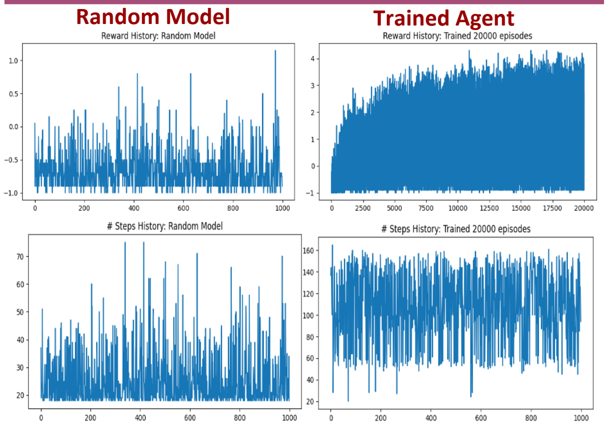
Discussion and Outlook

Overall, the current project is successful in training the agent to learn control policies based on high-dimensional sensory input through DQN and the agent outperforms the random state greatly.

Future directions:

1. The agent failed to distinguish the fuel depots from enemies, but shoot all to receive points. To improve, we can extract information representing the power bar and add a term to the reward function.
2. Tune k value in k-skipping method. A larger k provides the model with more information to predict an action, but as the action is repeated k times, it might not be the optimal strategy.
3. Try dual DQN to reduce the instability of training.

Results



	Random Model	Trained Agent
Average rewards	-0.660	0.932
last 50 episodes' reward	-0.615	1.094
last 50 episode' #steps	26.6	105.6