Note for new idea

Saturday, August 15, 2020 7:27 PM

The way that we update the policy of DDQN algorithm is to sample transition tuples $(s_t, a_t, r_{s_{t+1}}, s_{t+1}, done)$ from the experience replay buffer and minimize the TD-error $|(r_{s_{t+1}} + \gamma argmax_{a_{t+1}} Q_{target}(s_{t+1}, a_{t+1}) - Q(s_t, a_t)|$, where Q_{target} is synched to be Q every n frames to avoid overestimation of Q values.

The downside of this algorithm is the slow propagation of rewards

For example, suppose an agent can only choose 'Left' and 'Right' action, choosing 'Left' can lead it to reach from state s_{x-1} to s_x, ∀ x ∊ I, x ≥ 1, and choosing 'Right' can lead to the termination of the game. The agent scores 1 point at reaching the final state s_n, and 0 elsewhere.

Q(s_{n-1}, a_{left}) can learn the reward at state s_n immediately as the transition tuple of it is drawn from the replay buffer.

However, before Q_{target} is synched with Q, it won't result in meaningful learning for $Q(s_{n-2}, a_{left})$ if $Q_{target}(s_{n-1}, a_{left})$ is out-dated. It has to wait until Q_{target} synched with Q which is happening every n frames.

Likewise, $Q(s_{n-3}, a_{left})$ can learn the new value of $Q(s_{n-2}, a_{left})$ after $Q(s_{n-2}, a_{left})$ learns the new value at Q(s_{n-1}, a_{left}) and Q_{target} is synched with Q, which takes roughly 2 cycles of synchronization of Q_{target}.

This shows that the old states learn the rewards discovered at reaching new states with considerable amount of delays. And this issue can be amplified if we combine DDQN with curiosity-driven approach.

For example, as the agent discovers an unexplored state s_{novel}, it has a very high priority to reach the s_{novel} in the next few episodes and explore starting from state s_{novel}. However, if the curiosity bonus for $Q(s_t, a_t)$ is only calculated based on the novelty at the state s_{t+1}, which is the state that the agent reaches by performing action a_t at state s_t, the likelyhood of reaching s_{novel} could be decreasing becase the novelty at each state before reaching s_{novel} is decaying after they have been visited and the novelty at state s_{novel} can be learned by the states before s_{novel} after a huge number of frames.

Therefore, I am proposing a new way of curiosity bonus. (Adopt the idea of Life Long Curiosity and Episodic Curiosity from NGU DQN)

$$curiosity(s_t) = life\ long\ curiosity\ (s_t)\ \ * episodic\ curiosity\ (s_t)$$

where,

$life\ long\ curiosity\ (s_t)$ =$\min\big(\max\big(1, RND(s_t)\big), 5\big)$

$episodic\ curiosity\ (s_t)$ = (β^{m-t}RND(s_m)).clip(0, 10), where s_m is the state with the highest novelty in the episode after s_t is visited

Hyperparameters	Value
Number of Seeds	8 (Take the average of best 4)
Optimizer (DQN)	RMSprop Optimizer
Learning Rate(DQN)	0.00025
Update Frequency (RND)	Every 4 Frames
Alpha(DQN)	0.95
Optimizer (RND)	Adam Optimizer
Learning Rate (RND)	0.0001
Update Frequency (RND)	Once Every Episode
Batch Size	64
Replay Buffer Size	30000
Replay Buffer Init Size	10000
Discount γ	0.99
Discount β	0.995
Epsilon Start	1
Epsilon End	0.01
Epsilon Decay	25000
Coefficient for Curiosity Bonus	0.001