Frame Skipping in Deep Reinforcement Learning Experiments on Atari Games

Karan Agarwalla [*] [1] [2] **Rishi Agarwal** [*] [1] [3]

Abstract

Deep reinforcement learning is the rage of the day to learn complicated sequential decision tasks. MDPs (Markov decision processes) are used to model such sequential tasks. Traditionally, the agent senses every state of the MDP and decides which actions to execute at every step. We explore the idea of reduced sensing, wherein the agent senses states at every "d" time steps and repeats the chosen action for the skipped time steps. This is more popularly known as frame-skipping and action repetition. We investigate the role of frame-skip parameter in learning sequential tasks by running experiments on Atari-2600 games and find out if there exists a relation between the frameskip parameter and the discount factor. We present a comparative study of performance of the learning agent across a range of frameskip values and discount factor on two Atari Games- Seaquest and Enduro. Our experiments show that frame-skipping leads to significant improvements in the agent's performance. Although we believed that a higher value of the frame-skip parameter should correspond to a lower value of the discount factor, experiments so far do not support our hypothesis.

1. Introduction

Learning goal-oriented sequential decision tasks is one of the objectives of reinforcement learning. MDPs (Markov decision processes) are used to model sequential tasks. In contrast to supervised learning, where inputs along with labelled outputs are provided, such sequential tasks only provide states, and a reward which indicates the evaluation of the performance of the agent. In addition, state space is infinite (or huge) for many interesting problems. Tabular RL cannot be used in such cases, however, neural networks come to rescue and generalize over the state space by learning relevant features from the states, thus giving rise to deep reinforcement learning.

Reduced sensing is the next step in this journey, wherein the agent does not observe each and every state of the MDP as it takes actions, but only some of the states. More formally, the agent senses states after every "d" steps, henceforth referred to as the frameskip parameter. This concept of reduced sensing is more popularly referred to as "frame-skipping" because of its use in learning to play Atari games, where the agent observes frames as the states. For the "d" steps, where the agent observes no states, the agent is free to perform any ordered set of "d" actions. However, "action repetition" is most popularly used, where the agent repeats the last action for the next "d" steps. The motivation behind this can be explained by a small example - an agent driving a car on the road which decides where to steer every 10 milliseconds will perform reasonably well in comparison to an agent which decides every 1 millisecond. However, the earlier is easier to learn than the latter given the constraints on computational resources and time. Moreover, when trained for an equal number of finite steps, it seems reasonable to assume that the earlier agent will outperform the latter one.

In this paper, we run experiments on Atari-2600 games, implemented by OpenAI gym(Bellemare et al., 2013), to study the impact of frame-skipping and action repetition with different frameskip values, and to find out if there exists any relation between frameskip parameter and the discount factor. We gather data on primarily two Atari games - Seaquest and Enduro. We had started out believing that a higher value of frameskip should correspond to a lower value of discount factor, however our experimental results do not support this. We found that frame skipping indeed improves the performance of the agent significantly for both the games.

2. Related Work

The goal of reinforcement learning is to learn a policy that maximises discounted long term rewards. Q-learning is

[*]Equal contribution [1]Department of Computer Science and Engineering, IIT Bombay, India [2]180050045 [3]180050086. Correspondence to: Karan Agarwalla <karanagarwalla@cse.iitb.ac.in>, Rishi Agarwal <rishiagarwal@cse.iitb.ac.in>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

one of the most popular reinforcement learning algorithms. Deep Q-Networks(Mnih et al., 2013) were introduced in the domain of Atari Games and matched human experts on several games using a single architecture. Recent developments include the use of duelling networks(Wang et al., 2015) that explicitly separate value function and action advantage function. They significantly improved the agent's performance in Atari games. DQN duelling networks forms the underlying architecture of our experimentation.

Ideas based on frame-skipping have been explored in the domain of reinforcement learning. In frame-skipping the agent senses every d^{th} state instead of every state. Recent developments(Kalyanakrishnan et al., 2021) show that frame-skipping does not affect asymptotic consistency in policy evaluation and there exist upper bounds on error due to action repetition in control setting. The original DQN implementation(Mnih et al., 2013) used a frameskip of 4, i.e., chosen action were repeated four times before selecting the next action. This reduced compute time allowing more simulations.

Another set of experiments(Braylan et al., 2015) demonstrate the need to fine tune the frameskip parameter to achieve optimal performance. Results on several Atari games using DQN and HNEAT have been presented. Seaquest achieved best performance when skipping 180 frames, a gap of 3 seconds between two decisions.

More recently, a policy gradient approach(Sharma et al., 2017) has been proposed to tune the frameskip parameter online. The algorithm, FiGAR(Fine Grained Action Repetition), has shown superior performance on several Atari games. Macro actions(Durugkar et al., 2016), pre-defined action sequences, have also shown significant improvements in learning speed and performance on various Atari games.

3. Problem description

3.1. Markov Decision Processes

An MDP **M** is a five tuple (S, A, T, R, γ). **S** denotes the set of states and **A** denotes the set of actions. **T**(s,a,s') represents the probability of transitioning from state s to s' by taking action a. **R** is the reward function, **R**(s,a,s') denotes the reward obtained by the agent in transitioning from state s to s' on taking action a. γ is called the discount factor.

The agent starts from some state, executes actions and keeps transitioning from one state to another giving a sequence of state, action and reward - $s_0, a_0, r_0, s_1, a_1, r_1 \ldots$. MDPs are of two types - continuous and episodic. In a continuous MDP, the agent endlessly transitions from one state to another, while in an episodic MDP, the agent stops after it reaches some goal state. The metric used to evaluate agents is called the expected long term reward, it is defined as $E[r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots]$. The goal of an agent is to maximize the expected long term reward using the history of transitions available to it.

A policy π denotes a distribution over the action space for each state, π(s,a) denotes the probability with which the agent takes action "a" when in state "s". Such a policy π is Markovian and stationary. Each policy π has an associated value function V^π, such that V^π(s) denotes the expected long term reward obtained by an agent executing policy π starting from state "s". It is known that there exists an optimal policy π^* in the set of deterministic(π(s,a)=1 for an action a and 0 for other actions), Markovian and stationary policies, such that $V^{\pi^*}(s) \geq V^\pi(s)$ for any policy π and state "s".

3.2. Frame-skipping for Atari

In the context of frame-skipping along with action repetition, the agent observes a sequence as follows - $s_0, a_0, r_0 + r_1 + \ldots r_{d-1}, s_d, a_d, r_d + r_{d+1} + r_{d+2} + \ldots r_{2d}, s_{2d}, \ldots$. The goal of the agent is to maximize $E[r_{(0,d)} + \gamma r_{(d,2d)} + \gamma^2 r_{(2d,3d)} + \ldots]$ using the reduced history of transitions available to it, where $r_{(i,j)} = \sum_{k=i}^{j-1} r_k$.

In our case, we model both the games - Seaquest and Enduro as continuous MDPs because there is no goal state in either of the cases. Our problem is to empirically find out the effect of frameskip parameter and discount factor on the agent's performance. To quantify this effect we plot graphs of average reward vs. number of episodes for different values of frameskip parameter and discount factor.

4. Implementation

Our architecture is based on the same low level convolutional structure of DQN(Mnih et al., 2013). Raw Atari frames are 210x64x3 in size making them computationally demanding for convolutions. Hence, we transform them into 84x84x1 gray-scale frames. The transformation function then stacks 4 of these to form input to the DQN Network. The first convolutional layer has 32 8x8 filters with stride 4, the second 64 4x4 filters with stride 2, the third 64 3x3 filters with stride 1 and final convolutional layer consists of 1024 7x7 filters with stride 1. This is then split into two streams of shape (1, 1, 512): value stream and advantage stream. The functions are then obtained by dense layers on these two streams. Using duelling architecture(Wang et al., 2015) enables the network to learn valuable states without having to learn a separate value for each action. Adams Optimizer is used for training.

We use experience replay to train our network. It has several advantages compared to sequential updates. First, the frames can be reused in multiple weight updates allowing for data efficiency. Secondly, correlation between successive

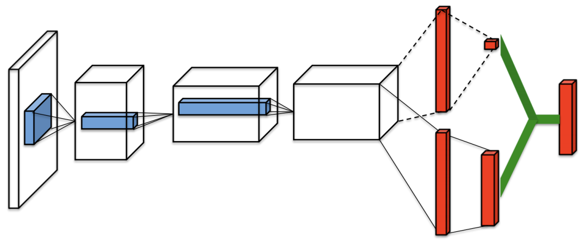


Figure 1. Duelling DQN Network(Wang et al., 2015)

frames are removed, thus reducing variance of updates.

We maintain two networks for prediction network and target network. The target network is synchronized with the prediction network every C steps. Using two networks has been shown to improve the stability of the learning algorithm. On using a single network, the predicted value and target value both move on updates to parameter θ.

We use double Q-Learning for updates which prevents overestimating value function. It provides better policies(van Hasselt et al., 2015) by averaging over the target and prediction network. The equations change from
$Q_{target}(s,a) = r + \gamma Q(s^{'}, argmaxQ(s^{'}, a^{'}, \theta_t), \theta_t)$ to
$Q_{target}(s,a) = r + \gamma Q(s^{'}, argmaxQ(s^{'}, a^{'}, \theta_t), \theta_{t'})$

Also, error terms are clipped between -1 and 1 using Huber loss function. This significantly improves the stability of the algorithm preventing exploding gradients.

We use ϵ-greedy with ϵ annealed from 1 to 0.1 over 4 million frames, thereafter from 0.1 to 0.01 over next 46 million frames and then fixed at 0.01 for further frames. We trained for a total of 50 million frames with a replay memory that stores the latest 1 million frames. The frameskip parameters are chosen from 4, 8, 16 and 20. We could not simulate frameskip 1 due to compute requirements. The discount factor γ is selected from 0.9 and 0.99.

We perform a extensive evaluation of our process using Arcade Learning Environment(ALE) interface provided by OpenAI Gym(Bellemare et al., 2013). We present results on two Atari Games: Seaquest and Enduro.

Code availability

The algorithm is attached as an appendix[1]. The source code is available at Github.

Hyper-parameters

The hyperparameter values are provided as an appendix [4].

Table 1. Enduro Training Rewards

ENDURO	0.99	0.9
4	716	452
8	490.85	493.75
16	303.9	222.8
20	155.5	135.45

Table 2. Seaquest Training Rewards

SEAQUEST	0.99	0.9
4	5179.5	570
8	3452	4697
16	1819	2912.5
20	1323	1879.5

5. Results

We shall be comparing our results to various baselines presented in [3]. Results for different values of frameskip and discount factor has been presented in [1] and [2]. The values for a particular value of frameskip and discount factor represent the maximum moving average reward over 50 episodes. The models have been trained to 50 million time steps. The best result for Enduro is 716 using frameskip 4 and discount factor 0.99. This is significantly larger than that reported for DQN in (Mnih et al., 2013) due to use of duelling networks. The best result for Seaquest is 5179.5 using frameskip 4 and discount factor 0.99. This is in consonance with the reported values for DQN.

We observe that the best frameskip value for $\gamma = 0.99$ is 4 and for $\gamma = 0.9$ is 8 across both the games. The difference is more noticeable in case of Seaquest than Enduro. In addition, our informal experiments with frame-skip equal to 1 (data not plotted due to time and resource constraints) indicate that frame-skips 4, 8, 16 and 20 perform significantly better in comparison to it. This indicates the significance of frame-skipping.

Interestingly the results do not support our hypothesis - higher frame-skip resembles lower discount factor. We notice that for the smaller discount factor (0.9), higher frame-skip (8) dominates the lower frame-skip (4), whereas for the bigger discount factor (0.99), lower frame-skip (4) outperforms higher frame-skip (8). For our hypothesis to hold, we expect to see exactly the opposite trends. The plots have been presented in the appendix.

6. Conclusion and Discussion

In this paper, we provide empirical evidence via extensive experiments on two Atari games - Seaquest and Enduro, to show that frame-skipping has a significant impact on the

Table 3. Base-Line Comparison(Mnih et al., 2013)

GAME	HUMAN	LINEAR	DQN	OUR METHOD
ENDURO	309.6	129.1	301.8	716
SEAQUEST	20182	664.8	5286	5179.5

agent's performance. We had started the journey with the belief that the frame-skip parameter is related to the discount factor. To gain further insights, we ran experiments with different values of discount factor, the results of which do not support our hypothesis. However, we believe that more experimentation is required to come to a more reasonable conclusion. Running more experiments with different RL algorithms like policy based gradient methods, online reinforcement learning, etc. to gain a firm understanding of the effects of the frame-skip parameter and the discount factor in the control setting is a possible direction for further research. Results from such experiments will instigate theoretical research to understand the impact of frameskip parameter and discount factor more formally.

Acknowledgements

We modified the open source implementation of DQN at https://github.com/fg91/Deep-Q-Learning. We are grateful to Professor Shivaram Kalyanakrishnan, IIT Bombay for his continuous guidance and support. We are thankful to him for offering us the necessary computational resources.

References

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, jun 2013.

Braylan, A., Hollenbeck, M., Meyerson, E., and Miikkulainen, R. Frame skip is a powerful parameter for learning to play atari. In *AAAI Workshop: Learning for General Competency in Video Games*, 2015. URL http://aaai.org/ocs/index.php/WS/AAAIW15/paper/view/10156.

Durugkar, I. P., Rosenbaum, C., Dernbach, S., and Mahadevan, S. Deep reinforcement learning with macro-actions. *CoRR*, abs/1606.04615, 2016. URL http://arxiv.org/abs/1606.04615.

Kalyanakrishnan, S., Aravindan, S., Bagdawat, V., Bhatt, V., Goka, H., Gupta, A., Krishna, K., and Piratla, V. An analysis of frame-skipping in reinforcement learning. *CoRR*, abs/2102.03718, 2021. URL https://arxiv.org/abs/2102.03718.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. A. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013. URL http://arxiv.org/abs/1312.5602.

Sharma, S., Lakshminarayanan, A. S., and Ravindran, B. Learning to repeat: Fine grained action repetition for deep reinforcement learning. *CoRR*, abs/1702.06054, 2017. URL http://arxiv.org/abs/1702.06054.

van Hasselt, H., Guez, A., and Silver, D. Deep reinforcement learning with double q-learning. *CoRR*, abs/1509.06461, 2015. URL http://arxiv.org/abs/1509.06461.

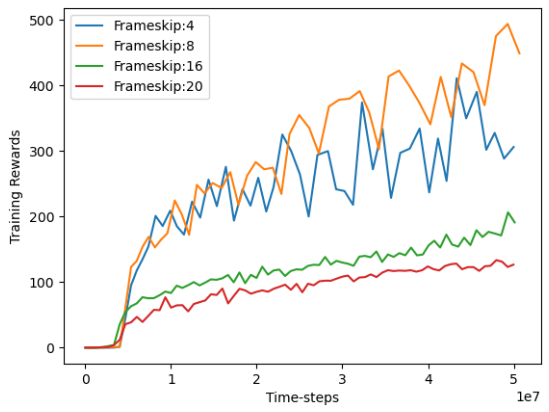
Wang, Z., de Freitas, N., and Lanctot, M. Dueling network architectures for deep reinforcement learning. *CoRR*, abs/1511.06581, 2015. URL http://arxiv.org/abs/1511.06581.

Table 4. List of hyperparameters and their values

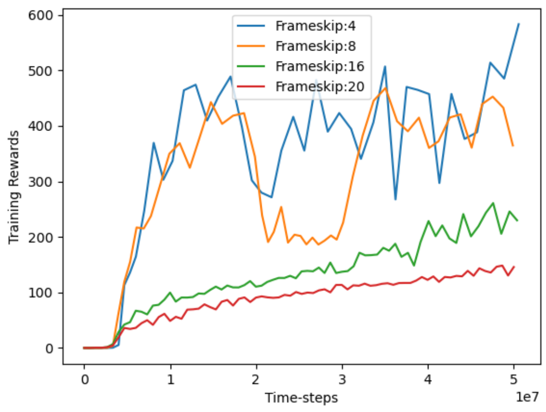
Hyperparameter	Value	Description
Batch Size	32	Number of training states over which each SGD update is computed
Replay Memory Size	1000000	Number of transitions stored in replay memory
Replay Memory Start Size	200000	Number of transitions before an agent start learning
Training Steps	50000000	Number of frames seen by the agent
Agent History Length	4	The number of frames stacked together to create a state
No-Op Steps	10	Number of 'do-nothing' actions at the beginning of an episode
Episode Length	72000	Maximum number of frames in an episode; equivalent to 20 minutes of game play at 60 frames per second
Discount Factor	0.9/0.99	Discount Factor γ used in Q-Learning Update
Learning Rate	0.00025	The learning rate used by Adam's Optimizer
Frameskip	4/8/16/20	Number of times each action selected by agent is repeated
Target Network Update Frequency	5000*max(Frameskip, 8)	The frequency with which the target network is updated with the main network
Update Frequency	max(Frameskip, 16)	Number of frames before an update is performed on the main network
Initial Exploration	1	Initial value of ϵ, exploration probability, in ϵ-greedy action selection
Median Exploration	0.1	Value of ϵ after being annealed over 4 million frames
Final Exploration	0.01	Final value of ϵ after being annealed over 46 million frames

Algorithm 1 Deep Q-Learning Network

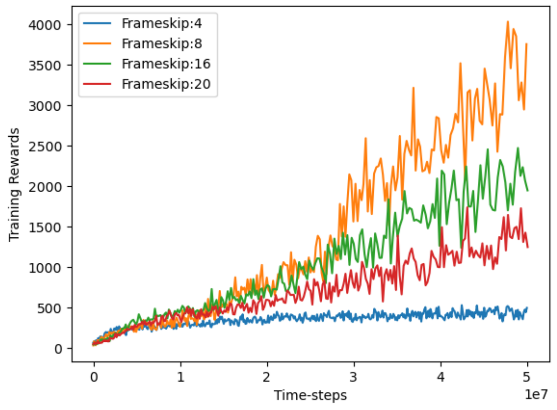
```
Initialise replay memory $D$ to capacity N
Initialise action value function(main network) $Q$ with random weights $\theta$
Initialise target action value function(target network) $Q^{'}$ with weights $\theta^{-} = \theta$
Initialise $timestep = 0$, $trainsteps$, $episodelength$, $updatefrequency$, $frameskip$
repeat
  Initialise $episodesteps = 0$
  Initialize sequence $s_1 = x_1$ and preprocessed sequence $\phi_1 = \phi(s_1)$
  repeat
    With probability $\epsilon$ select a random action $a_t$ otherwise $a_t$ = $argmax_a Q(\phi_t, a, \theta)$
    Execute action $a_t$ in emulator and observe reward $r_t$ and image $x_{t+1}$ repeating $frameskip$ times
    Set $s_{t+1} = s_t, a_t, x_{t+1}$ and $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in $D$
    if $timestep$ mod $updatefrequency == 0$ then
      Sample random minibatch of transition $(\phi_j, a_j, r_j, \phi_{j+1})$ from D
      Define $a^{max}(\phi_j, \theta) = argmax_{a'} Q(\phi_j, a^{'}, \theta)$
      $y_j = \begin{cases} r_j, & \text{if episode terminates at step j+1} \\ r_j + \gamma Q^{'}(\phi_{j+1}, a^{max}(\phi_{j+1}, \theta), \theta^{-}), & \text{otherwise} \end{cases}$
    end if
    Do a gradient descent step with on $(y_j - Q(\phi_j, a_j, \theta))^2$ with respect to network parameters $\theta$
    Every C steps reset $Q^{'} = Q$
    $episodesteps = episodesteps + frameskip$
  until $episodesteps \leq episodelength$
  $timestep = timestep + episodesteps$
until $timestep \leq trainsteps$
```



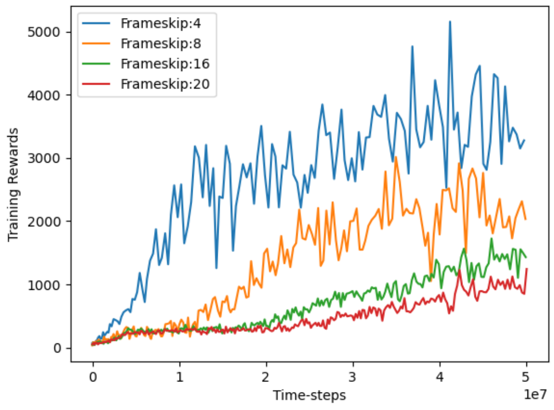
(a) Enduro with $\gamma = 0.9$



(b) Enduro with $\gamma = 0.99$



(c) Seaquest with $\gamma = 0.9$



(d) Seaquest with $\gamma = 0.99$

Figure 2. Plot of training rewards vs time-steps. The x-axis corresponds to the number of time steps for which the agent is trained and the y-axis corresponds to the moving average of the reward obtained by the agent. The two plots above correspond to the Atari game- Enduro, while the ones below correspond to the Atari game - Seaquest. The plots to the left are for the discount factor being equal to 0.9 and the ones to the right correspond to the discount factor being equal to 0.99. Each plot shows four lines which denote the four different values of the frame-skip parameter. The colours blue, yellow, green and red correspond to the frame-skip values 4, 8, 16 and 20 respectively.