DIFFERENTIABLE PROGRAMMING AND DEEP LEARNING REPRODUCIBILITY CHALLENGE

Anonymous authors
Paper under double-blind review

ABSTRACT

This coursework is a reproducibility test of the ICLR 2021 spotlight paper "Image Augmentation Is All You Need: Regularizing Deep Reinforcement Learning from Pixels". The paper hypothesizes that overfitting is a common problem among RL algorithms, especially those with deep encoders. They show that averaging value predictions under different image augmentations improves sample efficiency. Specific algorithms evaluated include the policy-based method Soft Actor Critic (SAC) in addition to Deep-Q Networks (DQN), a value-based method. This reproducibility study re-implements both algorithms without support of the authors code. The code accompanying this study can be found in the following GitHub repository.

1 PAPER ANALYSIS

Achieving sample efficient reinforcement learning from pixels is a challenge that could open up many applications in the robotics field. The main difficulty is that the objective function for reinforcement learning 1 is not directly differentiable. Instead an approximation of the gradient is achieved via minimization of the temporal difference error. This bootstrapping creates an approximate of 1 resulting in a noisy optimization signal causing instability and slow convergence.

$$q_\pi(s,a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}|S_t = s, A_t = a\right] \tag{1}$$

Image state spaces are challenging due to their high dimensionality, therefore requiring that a state encoder $f(s', v')$ is learnt in tandem to the policy and value functions to reduce state dimensionality. For algorithms such as SAC using the temporal difference error to learn f is inneffective as the optimization signal is noisy and sparse. Instead auxillary losses from unsupervised or self-supervised methods are utilised to learn good state representations Yarats et al. (2021). These representations however do not directly optimize the return expectation so may not be optimal for RL.

I.Kostrikov et.al show in their paper that state encoders produced by unsupervised learning significantly overfit their data. Their rationale for this assumption is derived from their results showing decreased environment returns when utilising larger image encoders. We posit that another factor may be the data collection procedure utilised to build the encoder training set. As should state-space samples be collected with a random policy, samples will not cover the full state space. This will result in states reached by a trained policy being out of the distribution of the encoders training set. Representations produced this way will be of poor quality and will reduce episodic returns.

The key principle behind this paper is to use image augmentations to reduce over-fitting of the actor, critic and encoding networks. They propose two augmentations; the first is that given a critic network Q_θ tasked with predicting the long term expectation of the return, a reduced variance prediction of the temporal difference target can be achieved by averaging the predictions over k image augmentations sampled from $\mathcal{T}$. The second regularization applies the same principle to the computation of the temporal difference loss. Taking the average of the temporal difference loss over M image augmentations of s_t. We assume this version of the loss produces gradients that encourage the encoder to learn state representations that are invariant to the image perturbations from $\mathcal{T}$. This regularizes an overfit encoder and reduces the variance of value predictions. It will result in a more accurate approximation of gradients that would maximise the long term expected return. However, despite achieving improved sample efficiency the method increases the number of forward passes required by the networks from two to $N + M$ times per training step. Furthermore, the time taken to

compute the augmentation can be significant, additionally increasing compute time. One interesting question this poses is the resulting increase in run-time compared to the increase in performance. The questions identified by this analysis are the following:

1. Does DrQ regularization outperform the standard SAC and NDDDQN algorithms in terms of sample efficiency?
2. Do the critic functions learn predictions invariant to transformations from $\mathcal{T}$
3. How does the algorithm affect training speed?

2 DEVELOPMENT

To evaluate the questions outlined above, implementations of the SAC+AE and NDDDQN algorithms will be developed, both with and without the regularization technique the authors coin DrQ Kostrikov et al. (2020). Implementations are developed from scratch using pytorch, pytorch-lightning, the deepmind and atari environment suites and tensorboard for experiment tracking.

To evaluate DrQ with SAC from pixels, a 4 layer convolutional autoencoder (AE) with an architecture from Yarats et al. (2021) is trained on 200k samples of the state and trained for 300 epochs. The encoder of this AE is subsequently used to transform each image observation of the environment state into a 50 dimensional vector. Each observation is then concatenated with observations of the last 8 transitions in order to capture the dynamics of the environment and form a Partially Observable MDP (POMDP). The SAC algorithm is implemented with 4 critic networks utilising the clipped double-Q trick Fujimoto et al. (2018) and polyak weight averaging between target and value critics Hasselt et al. (2016). A single MLP is implemented for the policy to predict the parameters of a unimodal Gaussian from which actions are sampled Haarnoja et al. (2018). To provide a baseline for comparison to DrQ regularised SAC from pixels, SAC is implemented directly with state observations as well as observations extracted from images with the encoder (SAC+AE).

The first stage of implementing DrQ Kostrikov et al. (2020) in DQN Mnih et al. (2015) was creating a baseline of the DQN algorithm. The paper however does not use the original DQN paper, but rather an improved version (NDDDQN) with parameters tuned for the Atari 100k Benchmark. This version of DQN uses 3 main modifications: Double DQN Van Hasselt et al. (2016), Duelling DQN Wang et al. (2016) and N-Step DQN Sutton & Barto (1998). DrQ only required a small change from the NDDDQN algorithm; performing the image augmentation after sampling a minibatch from the Experience Replay. No change was required in the update rule as was done in SAC, as the paper used [K=1,M=1] for DQN.

3 RESULTS

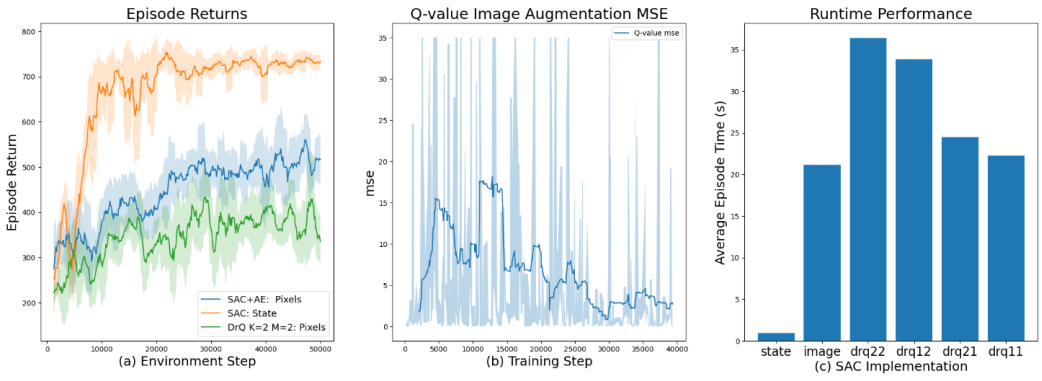

Figure 1: These figures are drawn from data collected on the deepmind cartpole environment with SAC. (a) Shows episode returns for SAC variants over 50k interactions. (b) shows the mse error between q-values predicted under different image augmentations. In this graph DrQ is implemented with $k = 2, M = 2$. (c) shows the average episode runtime of SAC variants where "drq21" represents DrQ implementation on pixels with $K = 2, M = 1$

Game	NDDDQN (Ours)	DrQ (Ours)	NDDDQN (Theirs)	DrQ (Theirs)
Alien	519.7	591.7	558.1	702.5
Amidar	69.2	68.3	63.7	100.2
Assault	464.5	515.1	589.5	490.3
Asterix	360.1	488.7	341.9	577.9
BankHeist	27.0	2.2	74.0	205.3
BattleZone	1280.2	2067.3	4760.8	6240.0
Boxing	4.0	7.0	-1.8	5.1
Breakout	9.4	3.7	7.3	14.3
ChopperCommand	955.1	552.1	624.4	870.1
CrazyClimber	55409.4	44611.4	5430.6	20072.2
DemonAttack	154.6	127.9	403.5	1086.0
Freeway	13.2	4.5	3.7	20.0
Frostbite	287.9	348.5	202.9	889.9
Gopher	724.5	235.3	320.8	678.0
Hero	4236.2	2696.4	2200.1	4083.7
Jamesbond	76.3	42.3	133.2	330.3
Kangeroo	42.2	76.6	448.6	1282.6
Krull	3015.5	462.8	2999.0	4163.0
KungFuMaster	9538.4	0.0	2020.9	7649.0
MsPacman	1025.6	1050.2	872.0	1015.9
Pong	-14.9	-20.9	-19.4	-17.1
PrivateEye	-138.4	-74.6	351.3	-50.4
Qbert	518.4	366.3	627.5	769.1
RoadRunner	7333.6	2362.5	1491.9	8296.3
Seaquest	245.7	199.2	240.1	299.4
UpNDown	1267.2	650.7	2901.7	3134.8
Median Human Normalised Score	0.119	0.078	0.094	0.270

Table 1: Results evaluating various algorithms in 26 Atari games with 100k training steps and 125k evaluation steps. All results are averaged over 5 different seeds. Furthermore, the bottom of the table shows the median human normalised scores for each algorithm over all 26 games.

4 Discussion and Conclusion

Graph (a) from figure 1 shows the results answering question 1 for SAC. It is contrary to the work of Kostrikov et al. (2020), exhibiting reduced episodic returns. This could be for three reasons. The first is difference is in the way the encoder was trained. In Yarats et al. (2021) it is shown that for SAC+AE alternating training phases between AE reconstruction loss and RL Temporal difference loss can improve expected returns, particularly when the alternation is frequent. For DrQ regularised SAC+AE the paper does not specify which type of encoder training regime is performed, however their code does suggest it is pretrained. For this reason we have only pretrained the AE for both SAC+AE and DrQ SAC+AE implementations. Additionally the encoder used in our work did not overfit the state samples achieving a mse of 13.71 on the training set and 13.81 on the test set, which may have hidden the benefits of regularization. For question 2, graph (b) of figure 1 shows that as training progressed the Q-values predicted under different image augmentations converged meaning the networks were learning to become invariant to image augmentations validating our hypothesis. To evaluate question 3, graph (c) is shown. Here the value of M had the largest runtime impact. It is due to the use of the clipped double-Q trick Fujimoto et al. (2018) which required two critic encoder forward passes per image augmentation during prediction of the temporal difference loss.

Two more potential reasons for the disparity between ours and the authors results for SAC is firstly that we were unable to run the agents to the number of environment interactions completed in the paper. This was due compute constrains as the DrQ SAC+AE implementation with $K = 2, M = 2$ requires 8 separate neural networks with 19 forward passes and 4 backward passes per training step. Another reason for the disparity is that the authors published code shows a slightly different method to that described the paper with some states being passed to the encoder without augmentation. Our method followed exactly that of the papers with all states being augmented.

Table 1 shows the results for NDDDQN with and without the DrQ regularization, answering our first research question for DQN. The results presented in this table strongly show a reduced performance,

both against the baseline NDDDQN, and the results reported by the DrQ paper. The reason for this is unclear, as the paper's appendix did a good job of extensively stating the parameters used. Whilst RL algorithms tend to be very high in variance, it is unlikely the results are due to random chance as 26 games were tested, with 5 random seeds of every game. Our reproduced DrQ algorithm did perform exceptionally well in some games, namely CrazyClimber, however also performed very poorly in many. Some possible explanations for this include that the network wasn't overfitting to begin with, hence DrQ was unnecessary, or conversely DrQ prevented the network from overfitting, but agent's did not see benefit from a more general policy.

In summary we were not able to reproduce the findings of the paper but have hypothesised a number of reasons why that may be. Provided more compute and time the evaluation of these hypothesis may narrow down on the fundamental reasons for this disparity.

REFERENCES

Scott Fujimoto, Herke Van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. volume 4, 2018.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. volume 5, 2018.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. 2016. doi: 10.1609/aaai.v30i1.10295.

Ilya Kostrikov, Denis Yarats, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *arXiv preprint arXiv:2004.13649*, 2020.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Richard S Sutton and Andrew G Barto. Reinforcement learning: an introduction mit press. *Cambridge, MA*, 22447, 1998.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pp. 1995–2003. PMLR, 2016.

Denis Yarats, Amy Zhang, Ilya Kostrikov, Brandon Amos, Joelle Pineau, and Rob Fergus. Improving sample efficiency in model-free reinforcement learning from images. volume 12A, 2021. doi: 10.1609/aaai.v35i12.17276.