Expert-augmented actor-critic for Vizdoom and Montezuma's Revenge

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

We propose an expert-augmented actor-critic algorithm and test it on two simulated environments with sparse rewards: Vizdoom and Montezuma's Revenge. Compared to previously published approaches, in the case of Montezuma's Revenge our agent achieves the highest average reward and at the same time the training of the agent is more sample efficient. We consistently achieve results around 8000 points and in some experiments our agent solves the first world during evaluation. In the case of Vizdoom, the agent learns to navigate a complicated maze in a scenario which is difficult for model-free algorithms not augmented by expert data.

1 Introduction

Deep reinforcement learning has shown impressive results in simulated environments [24, 23]. Despite this success current approaches tend to not behave well when rewards are sparse. This is limiting real-world applications, notably in robotics, as easily collectible rewards are often binary and sparse. They are obtained only after completing the whole task [3, 35]. Additionally, in such environments, especially outside of the simulation fully random exploration can be prohibitively expensive [22].

Using expert trajectories is one approach for obtaining more efficient exploration strategies. However, standard behavioral (e.g. [28, 5]) achieves weak performance due to compounding errors when drifting away from the supervisor's demonstrations [30]. For example, in [13] authors analyze performance of behavioral cloning on a challenging Atari 2600 game Montezuma's Revenge and show that this method reaches on average only 575 points despite being trained on near-optimal demonstration trajectories that score 30 000 points.

In order to unlock interesting applications of reinforcement learning, sample-efficiency needs to be addressed. Current state-of-the-art deep reinforcement learning models require tens to hundreds of millions of data samples to converge to good policies [12, 23, 38]. This corresponds to years of experience if data is gathered at human-friendly 60 frames per second. One of the approaches to deal with that is to use more advanced optimization for gradient updates, such as one of approximate natural gradient methods [1]. These techniques accelerate gradient descent optimization by changing parameters in the direction that minimizes the loss with respect to small step in the distribution of network output (in our case policy), as opposed to small step in the parameter space metric.

There are a few deep reinforcement learning algorithms that use this curvature information e.g. [33]. From among these we choose Actor-Critic using Kronecker-Factored Trust Region (ACKTR) [38] which we will augment. This approach mixes actor-critic algorithm, deep networks as policy / value function estimators and natural gradient approximation, yielding very sample-efficient training on most of the Atari games.

Submitted to 32nd Conference on Neural Information Processing Systems (NIPS 2018). Do not distribute.

In this work we introduce a simple way to combine ACKTR with expert data in order to guide agent's exploration. In Section 2 we review relevant literature, while in Section 3 we introduce our method. In Section 4 we outline an evaluation of our approach on two standard sparse rewards environments: the notoriously difficult Montezuma's Revenge and Vizdoom My Way Home task. In Section 5 we report results for both of these. In Section 6 we discuss our results. We supply links to supplementary videos with evaluations for Montezuma's Revenge, Montezuma's Revenge with expert state resets and Vizdoom MyWayHome as a mean of qualitative evaluation.

2 Related work

Deep reinforcement learning First strong results in deep reinforcement learning for Atari were presented in [24]. These value-based approaches were soon followed by policy-based methods in [23]. This work introduced asynchronous advantage actor-critic algorithm, which was later extended into ACKTR [38].

Expert data in reinforcement learning. There is very rich literature on imitation learning. A celebrated paper that reviews several positive and negative examples of using these techniques is [30]. The broader idea of combining natural policy gradient algorithm with expert demonstrations has been successfully applied to complex robotics problems in simulation in [29] and to dialogue management in [34].

The publication that is closest to our work is [13]. It modifies Prioritized Dueling Double Deep Q-Networks (PDD DQN) [32, 37] by keeping expert demonstrations in the replay buffer and extending loss function to efficiently utilize demonstrations. They ensure that no action is allowed to have higher Q value than expert's action. L_2 regularization and longer horizon loss are used to better propagate expert information. This work presents the state-for-the-art for Montezuma's Revenge: average reward of 4700 using 200 million frames. This paper uses only 18k expert transitions, compared to our 170k, however scores reached by their expert are similar to scores of our expert (around 32-35k pts). Authors show that performance of behavioral cloning on Montezuma's Revenge is relatively poor, but also give examples of games for which behavioral cloning is effective, such as Video Pinball or Pitfall, where rewards are sparse as well. Among other value-based approaches that utilize expert data authors of [19] extend DQN with a cross-entropy classification of expert action. Our work is different in that we use policy-gradient methods. [19] uses significantly less expert data then we do (around 5k (s, a) expert pairs compared to our $\sim$ 170k pairs), it does not however include evaluation on Montezuma's Revenge or Vizdoom.

Authors of [40] combine policy-based approach with expert data. They pre-train policies that are later passed to off-the-shelf actor-critic algorithms. Their framework is theoretically capable of processing non-optimal expert actions, whereas we generally rely on optimality of expert actions to extract good performance. We use expert data during the whole training process, where they use expert data in the pre-training phase only. They work with Atari, however do not report results for Montezuma's Revenge.

Montezuma's Revenge Significant attention has been dedicated specifically to solving Montezuma's Revenge and it is regarded as key testing ground [16]. Efforts have been made to solve this environment by adding natural language instructions [16] (3.5k pts @ 60M frames), extending model with intrinsic curiosity awards [25] (3.7k @ 150M frames) and [4] (avg. 3k @ 100M frames, single runs of 6k @ 100M frames), introducing hierarchy into the model [36] (2.6k pts @ 1000M frames frames) or utilizing expert demonstrations [13] (4.7k @ +200M frames). Numbers in the parenthesis represent best scores in Montezuma's Revenge posted by these approaches. Authors of [20] demonstrate effectiveness of combining hierarchical learning and imitation learning, focusing on first screen of Montezuma's Revenge. They do not post results for later stages of the game. All of these approaches are based on deep reinforcement learning, however expert data is utilized only in [13] and in our approach.

Doom The Vizdoom environment [17] is a popular suite of reinforcement learning tasks based on first-person shooter game Doom. Strong results have been shown in various sub-tasks of this environment spanning from navigation to competitive combat using deep reinforcement learning, both for the sake of learning specific ability e.g. navigation [31, 21, 26], in-game multiplayer combat

[10, 7, 39, 18] or as a way to investigate another aspect of reinforcement learning [2, 14, 11], also transfer learning [8]. These approaches differ from ours in that they do not use expert data.

Similarly as in Montezuma's Revenge, there is a line of work that extends standard reinforcement learning approach with additional elements, such as supervised learning of implicit environment model [10] or curiosity-based intrinsic rewards to increase efficiency of exploration [27]. The latter work is particularily relevant to our work. It is focused on solving the same Vizdoom environment MyWayHome that is used in our experiments presented in Section 5. Importantly, this paper introduces a new variation of the above environment dubbed MyWayHomeVerySparse that makes the reward even more sparse (while making the environment less stochastic) by spawning the agent always in the furthest room (see Figure 1). They show that standard actor-critic approach completely fails in this setting, a result which we replicate in our experiments (see Section 5). Both this work and [15] claim that actor-critic performs well on MyWayHome environment. We confirm these results in Section 5. These papers show that this positive performance can be further improved by intrinsic curiosity or rarity of events reward.

3 Expert-augmented ACKTR

Our approach consists of a modification of the ACKTR algorithm that introduces a new term $g_{\text{expert}(\theta)}$ to the gradient:

$$\nabla_\theta L(\theta) = \underbrace{\text{adv}_i \nabla_\theta \log \pi(a_i|s_i;\theta) + \frac{1}{2}(R_i - V(s_i;\theta))^2/\partial\theta}_{g_{\text{A2C}}(\theta)} + \lambda_{\text{expert}} \underbrace{\text{adv}_i^{\text{expert}} \nabla_\theta \log \pi(a_i^{\text{expert}}|s_i^{\text{expert}};\theta)}_{g_{\text{expert}(\theta)}}$$

Pseudocode for the full framework is visible in listing Algorithm 1. The expert data is sampled from a fixed dataset of strong games. Future discounted rewards are computed for the expert data, by recording the extrinsic rewards collected by the expert. We consider 3 variants for expert advantage estimate: *reward:* $\text{adv}_i^{\text{expert}} = \sum_t \gamma^t r_{i+t}$ *actor-critic:* $\text{adv}_i^{\text{expert}} = \left[\sum_t \gamma^t r_{i+t} - V(s_t)\right]_+$, where $[x]_+ = max(x, 0)$ and *simple:* $\text{adv}_i^{\text{expert}} = 1$. Reward and actor-critic variant are based on policy gradient theorem, while the simple variant is motivated by sparseness of the reward. Sometimes the $\gamma = 0.99$ reward discount factor in Montezuma is close to zero, hindering propagation of information.

Data: Parameter vector θ;
Dataset of expert transitions $(s_t^{\text{expert}}, a_t^{\text{expert}}, s_{t+1}^{\text{expert}}, r_t^{\text{expert}})$
initialization;
for $iteration \leftarrow 1$ **to** *max steps* **do**
    for $t \leftarrow 1$ **to** *horizon, e.g.* $T = 20$ **do**
        Perform action a_t according to $\pi_\theta(a|s)$
        Receive reward r_t and new state s_t
    end
    for $t \leftarrow 1$ **to** T **do**
        Compute discounted future reward: $\hat{R}_t = r_t + \gamma r_{t+1} + ... + \gamma^{T-t+1} r_{T-1} + \gamma^{T-t} V_\theta(s_t)$
        Compute advantage: $\text{adv}_t = \hat{R}_t - V_\theta(s_t)$
    end
    Compute A2C loss gradient $g_{A2C} = \nabla_\theta \sum_{t=1}^{T} \left[-\log \pi_\theta(a_t|s_t)\text{adv}_t + \frac{1}{2}(V_\theta - \hat{R}_t)^2\right]$
    Sample mini batch of e.g. $k = 256$ expert data state-action pairs
    Compute chosen expert advantage estimate $\text{adv}_t^{\text{expert}}$ f or each state-action pair.
    Compute expert loss gradient $g_{exp} = \frac{1}{k}\sum_i \text{adv}_i^{\text{expert}} \log \pi_\theta(a_i^{\text{exp}}|s_i^{exp})$
    Update ACKTR inverse Fisher estimate.
    Plug in gradient $g = g_{A2C} + g_{expert}$ into ACKTR Kronecker optimizer.
end

Algorithm 1: Expert-augmented ACTKR

ACKTR estimates approximate of the inverse of Fisher matrix of the current policy in order to approximate natural gradient direction. Our method does not take expert data into account for this estimation procedure. In spite of this, the gradient estimate compute from expert data is still projected into natural gradient direction based on inverse of Fisher matrix estimated only by live interaction with environment.

4 Experimental setup

In order to evaluate our approach we will use two simulated environments. In this section we describe the details of the experimental setup.

Doom The first environment we use is 'DoomMyWayHome-v0' provided as part of the Vizdoom suite [17]. In this scenario, agent is dropped at a random location in a fixed maze of 9 rooms and is given the task of collecting a vest, always placed at the same location, which results in ending an epsisode and getting a reward of 1. Episode terminates with zero reward if agent doesn't collect the vest in 2100 timesteps, which at 35 fps is equivalent to 1 minute of game time. The action space contains four actions that can be combined - move forward, turn left, turn right and no-op. Sample frame from Vizdoom is shown in Figure 2. Additionally to the standard 'MyWayHome' environment, we use a 'very sparse' variant of the environment, introduced by [27], where agent is always spawned in a fixed beginning location - the room that is furthest from the reward.

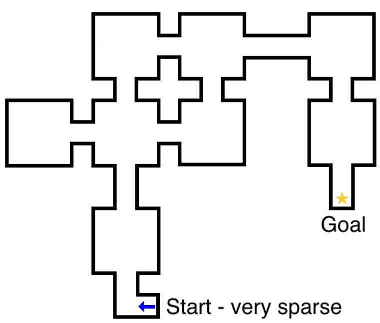


Figure 1: Map of Vizdoom MyWayHome environment.

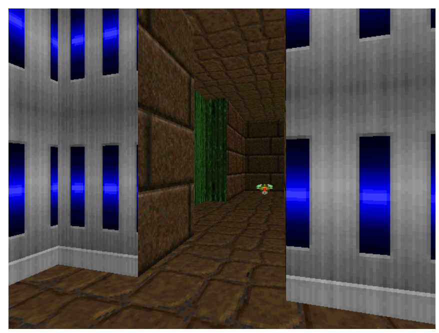


Figure 2: Screenshot from Vizdoom MyWayHome environment.

Montezuma's Revenge The second environment we use is an Atari game Montezuma's Revenge, available as part of Open AI Gym environment collection [6]. Montezuma's Revenge is a 2D platform-based maze. The task of the agent is to navigate the maze, collect valuable items and avoid deadly obstacles. For example, obtaining the first reward requires five marco actions (e.g. "Jump across a gap", "go down a ladder", which corresponds to 3 seconds of well coordinated inputs. Sample screens from the game are presented in Figure 3,

To help generalization, in both expert and live environment data we obscure the part of the screen that shows number of lives left and score.

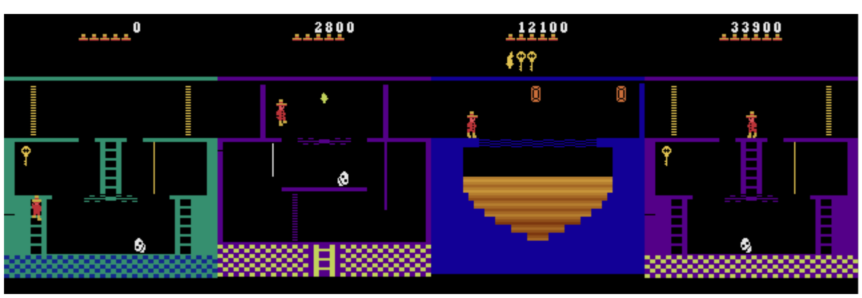


Figure 3: Montezuma's Revenge - sample screens from the game showcasing variety of screens that agent needs to learn to traverse.

Training details We make an effort to follow common conventions set by [24, 23] and in case we diverge we mention it explicitly. Expert and live environment data are processed using the same pipeline. RGB input images are cast to grayscale rescaled to 84 x 84. Four consecutive frames are stacked to enable modeling of temporal dependencies.

We use ACKTR K-Fac optimizer [38], with default learning rate of $\alpha = 0.125$. We use synchronous actor-critic style optimization with 32 environments, each time running the environment 20 steps forward before bootsratping with value function, as opposed to original asynchronous A3C [23], which circumvents the typical problems with programming the asynchronous approach. Expert gradient gets the same weight as standard gradient estimate obtained from live environment interaction. We use expert batch size of 256.

Following common conventions [23], we introduce entropy regularization term with respect to the policy parameters of the form $\beta_{\text{entropy}} \nabla_\theta H(\pi_\theta(s_t))$, where H is entropy and value $\beta_{\text{entropy}} = 0.001$ is chosen for the parameter. We do this for the sake of getting unstuck from local minima. Hyperparameters are kept the same for Montezuma's Revenge and Doom, exceptions are noted in descriptions of the experiment results.

Network architecture We use a small neural net as policy and value estimator, with architecture similar to one used in [24] and widely used in other literature presented in Section 2. Input images are passed through conv. layer with 32 filters size of 8 and stride 4, followed by 64 conv filters of size 4 and stride 2, followed by 64 conv. filters of size 3 and stride 1. These feed into fully connected layer of 512. Non-linearity ReLU is used after each of above layers. This layer is attached to two heads - one with single neuron for value function approximation, the second for with as many neurons as many actions there are, each outputting the logit of action probability. The final policy is stochastic, based on softmax of these logits.

Code The baseline ACKTR models used are publicly available OpenAI Baselines.[9]. We intend to release the code as open source after the review process.

Expert data Expert data has beem collected by the authors playing the game. In both environments the game-play sample obtained is near-optimal.

The amount of data collected is: 128 trajectories, $33\,394$ data points for Vizdoom MyWayHome, 66 trajectories, $19\,404$ data points for Vizdoom MyWayHomeVerySparse, 14 trajectories, $172\,548$ data points for Montezuma's Revenge.

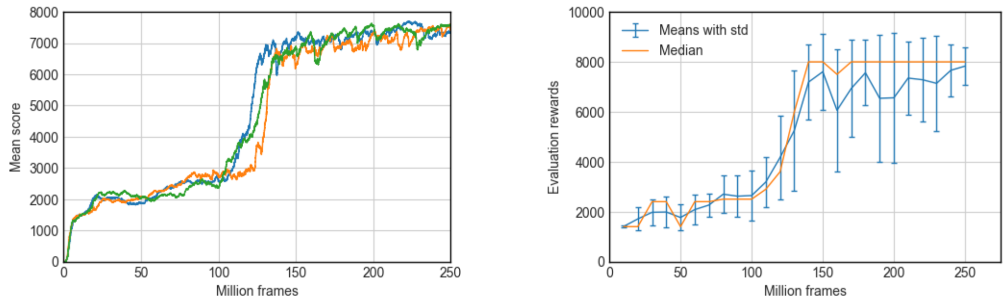
In the case of Montezuma's Revenge, we trim expert trajectories to the first world. Under this assumptions, average score reached by expert trajectory is 24328, with all trajectories reaching more than 30k points.

5 Experiments

5.1 Montezuma's Revenge

We perform an experiment where Expert-augmented ACKTR is used to play Montezuma's Revenge. We choose future discounted reward as expert advantage estimate in order to exclude suboptimal expert actions leading to agent's loss of life. We stop training every hour of wall time in order to run evaluation, results of which are presented in Figure 4b. We repeat the experiment with 3 random seeds to get better estimates of performance statistics.

The agent reaches an average evaluation reward of 6560 points and median 8000 using 200M frames in total. The summary of results is visible in Figures 4a and 4b. Our results are more sample efficient than previous results of previous comparable methods, such as DQfD [13] that reach 4739.6 points in 200M frames. Our agent consistently explores most of the rooms of the first world, as depicted in Figure 5. We link a playlist, where we present videos of a variety of stronger evaluation runs of our algorithm. Qualitatively, it is very often the case the trained actor is not able to get through a particularly tricky central room and therefore is stuck on score of 8000. This obstacle is cleared in the next section at the cost of requiring ability to set the state of the environment.



(a) Training (b) Evaluation

Figure 4: Montezuma's Revenge. Mean and median rewards in evaluation achieved by one parametrization of our method over 3 random seeds.

In comparison, actor-critic methods such as A2C and ACKTR without additional modules promoting exploration do not learn anything useful in this setting. Further, supervised learning of expert data yields average score of 570 in [24] and we report evaluation scores of below 400 in the same setup. Taking into account that the expert training data is near-optimal, this is surprisingly weak result.

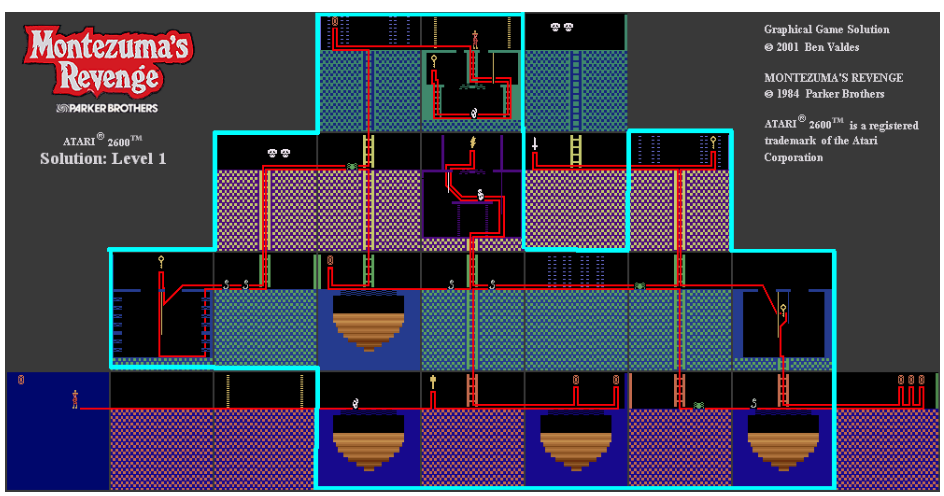
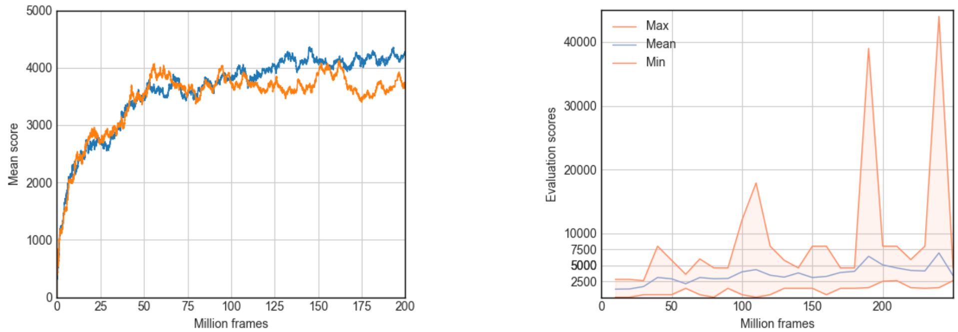


Figure 5: Rooms explored during a sample evaluation of our agent.

Experiments with expert state resets As the agent always starts the game in the same starting position, the beginning of the game is overrepresented in training. In order to alleviate this, we introduce a mechanism in which environment is restored to a state drawn uniformly from the distribution of states visited in expert trajectories.

This results in general slow-down of training in terms of sample efficiency, however it unlocks occasional very good evaluations. The results of this experiments can be seen in Figures 6a and 6b. The approach presented reliably produces runs where 200M training frames is enough to generate policy that has relatively high variance, but occasionally produces very strong evaluations that solve the first world of the game. Here we present such cherry-picked very strong evaluations.



(a) Training.

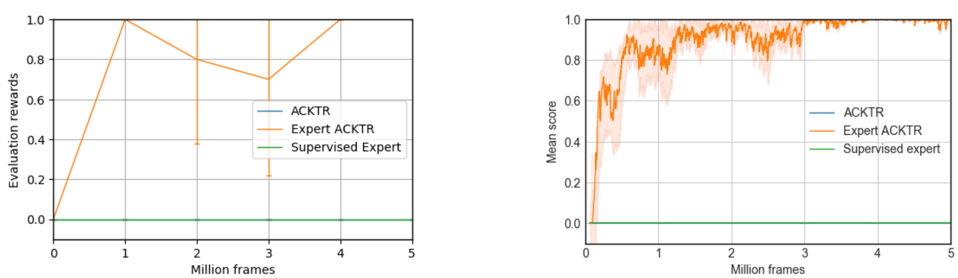
(b) Evaluations

Figure 6: Montezuma's Revenge. Mean and median rewards in evaluation achieved by one parametrization of our method over 2 random seeds.

5.2 Vizdoom

MyWayHome - Very sparse We experiment with a deterministic version of the above MyWayHome environment, introduced by [27], in which the agent is always spawned in the room that is furthest to reward, see Figure 1. Very sparse reward environment version is particularly hard for non-expert augmented methods, as random exploration is extremely unlikely to find any reward. In this case we use simple expert advantage estimate, which is equivalent to supervised "classification" of expert advantage.

Expert-augmented ACKTR is very efficient in this setting, learning perfect performance in 5 million frames. By clicking here you can see agent playing well after only an wall-clock hour of training. Both behavioral cloning of expert transitions and vanilla actor-critic do not manage to learn anything useful in this setup, as can be seen in Figures 7a and 7b.

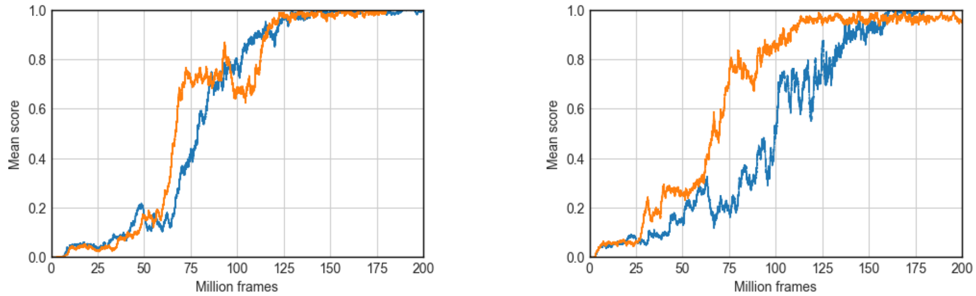


(a) Evaluation. Each bar represents 5 evaluation runs.

(b) Training

Figure 7: Vizdoom MyWayHomeVerySparse. Mean and median evaluation rewards and training rewards for 2 runs with different random seeds.

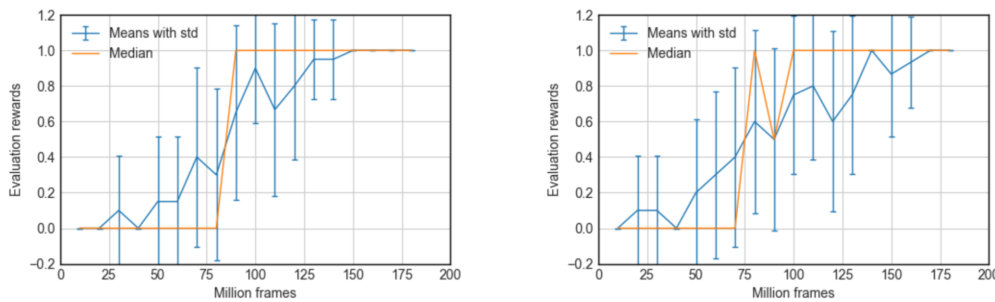
MyWayHome - Standard We compare standard ACKTR with Expert-augmented ACKTR on standard MyWayHome which has significantly denser rewards. In this case both methods obtain similar results, expert data helping only slightly. ACKTR deals with this environment relatively well, as actor is spawned uniformly over the whole space, sometimes getting spawned close to the goal, helping gradual build-up of the optimal policy. As shown in Figures 9b and 9a, expert data helps only slightly, reaching perfect evaluation in 150M frames as opposed to 175 frames needed by method without expert data.



(a) With expert data.

(b) Without expert data.

Figure 8: Vizdoom MyWayHome. Average training enviroment rewards averaged over 2 random seeds.



(a) With expert data

(b) Without expert data

Figure 9: Vizdoom MyWayHome. Mean and median evaluation, averaged over 2 random seeds. Each bar represents 5 evaluation runs.

6 Conclusions and future work

Based on experimental results from the simulated environment we hypothesize that the presented algorithm is a practical method of getting good performance in cases when multiple interactions with environment is possible and good quality expert data exists. It could be particularly useful in settings when neither supervised learning from expert data nor random exploration yield good results, such as in Montezuma's Revenge.

We leave the following extensions to future work:

1. Running Expert-augmented ACKTR on the rest of classic Atari environments, in order to obtain comparison with the state of the art.

2. Checking dependence of strategy discovered on quality of expert data. How do performance measures look as a function of of expert data quality?

3. Experiments in simulated robotics, where are sparse rewards are common - e.g. object manipulation, navigation. Comparisons of the presented method with similar previously published approaches such as [29].

Furthermore, for the sake of simplicity we did not use importance sampling of expert actions (which in principle should be used as expert data is off-policy). Implementing this might be challenging as it requires estimating distribution of the expert policy only from trajectories. On the other hand algorithm using importance sampling would be consistent with the theory behind ACKTR and thus potenially more efficient.

References

[1] Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

[2] Artemij Amiranashvili, Alexey Dosovitskiy, Vladlen Koltun, and Thomas Brox. Td or not td: Analyzing the role of temporal differencing in deep reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2018.

[3] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Advances in Neural Information Processing Systems*, pages 5048–5058, 2017.

[4] Marc G. Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Rémi Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 1471–1479, 2016.

[5] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. *CoRR*, abs/1604.07316, 2016.

[6] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

[7] Devendra Singh Chaplot and Guillaume Lample. Arnold: An autonomous agent to play FPS games. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 5085–5086, 2017.

[8] Devendra Singh Chaplot, Guillaume Lample, Kanthashree Mysore Sathyendra, and Ruslan Salakhutdinov. Transfer deep reinforcement learning in 3d environments: An empirical study. In *NIPS Deep Reinforcemente Leaning Workshop*, 2016.

[9] Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, and Yuhuai Wu. Openai baselines. `https://github.com/openai/baselines`, 2017.

[10] Alexey Dosovitskiy and Vladlen Koltun. Learning to act by predicting the future. *CoRR*, abs/1611.01779, 2016.

[11] Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl$ˆ2$: Fast reinforcement learning via slow reinforcement learning. *CoRR*, abs/1611.02779, 2016.

[12] Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Daniel Horgan, Bilal Piot, Mohammad Gheshlaghi Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. *CoRR*, abs/1710.02298, 2017.

[13] Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Gabriel Dulac-Arnold, Ian Osband, John Agapiou, Joel Z. Leibo, and Audrunas Gruslys. Deep q-learning from demonstrations, 2017.

[14] Peter H. Jin, Sergey Levine, and Kurt Keutzer. Regret minimization for partially observable deep reinforcement learning. *CoRR*, abs/1710.11424, 2017.

[15] Niels Justesen and Sebastian Risi. Automated curriculum learning by rewarding temporally rare events. *CoRR*, abs/1803.07131, 2018.

[16] Russell Kaplan, Christopher Sauer, and Alexander Sosa. Beating atari with natural language guided reinforcement learning. *CoRR*, abs/1704.05539, 2017.

[17] Michał Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski. Vizdoom: A doom-based ai research platform for visual reinforcement learning. In *Computational Intelligence and Games (CIG), 2016 IEEE Conference on*, pages 1–8. IEEE, 2016.

[18] Tejas D. Kulkarni, Ardavan Saeedi, Simanta Gautam, and Samuel J. Gershman. Deep successor reinforcement learning. *CoRR*, abs/1606.02396, 2016.

[19] Aravind S. Lakshminarayanan, Sherjil Ozair, and Yoshua Bengio. Reinforcement learning with few expert demonstrations, 2016.

[20] Hoang M. Le, Nan Jiang, Alekh Agarwal, Miroslav Dudík, Yisong Yue, and Hal Daumé III. Hierarchical imitation and reinforcement learning, 2018.

[21] Lisa Lee, Emilio Parisotto, Devendra Singh Chaplot, and Ruslan Salakhutdinov. Lstm iteration networks: An exploration of differentiable path finding. 2018.

[22] Sergey Levine, Peter Pastor, Alex Krizhevsky, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *CoRR*, abs/1603.02199, 2016.

[23] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937, 2016.

[24] 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, 2015.

[25] Georg Ostrovski, Marc G. Bellemare, Aäron van den Oord, and Rémi Munos. Count-based exploration with neural density models. *CoRR*, abs/1703.01310, 2017.

[26] Emilio Parisotto and Ruslan Salakhutdinov. Neural map: Structured memory for deep reinforcement learning. *CoRR*, abs/1702.08360, 2017.

[27] Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. *CoRR*, abs/1705.05363, 2017.

[28] Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In *Advances in neural information processing systems*, pages 305–313, 1989.

[29] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*, 2017.

[30] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. pages 627–635, 2011.

[31] Nikolay Savinov, Alexey Dosovitskiy, and Vladlen Koltun. Semi-parametric topological memory for navigation. *CoRR*, abs/1803.00653, 2018.

[32] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *CoRR*, abs/1511.05952, 2015.

[33] John Schulman, Sergey Levine, Pieter Abbeel, Michael I. Jordan, and Philipp Moritz. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 1889–1897, 2015.

[34] Pei-Hao Su, Pawel Budzianowski, Stefan Ultes, Milica Gasic, and Steve J. Young. Sample-efficient actor-critic reinforcement learning with supervised data for dialogue management. *CoRR*, abs/1707.00130, 2017.

[35] Matej Večerík, Todd Hester, Jonathan Scholz, Fumin Wang, Olivier Pietquin, Bilal Piot, Nicolas Heess, Thomas Rothörl, Thomas Lampe, and Martin Riedmiller. Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards. *arXiv preprint arXiv:1707.08817*, 2017.

[36] Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, and Koray Kavukcuoglu. Feudal networks for hierarchical reinforcement learning. *arXiv preprint arXiv:1703.01161*, 2017.

[37] 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*, pages 1995–2003, 2016.

[38] Yuhuai Wu, Elman Mansimov, Shun Liao, Roger Grosse, and Jimmy Ba. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation, 2017.

[39] Yuxin Wu and Yuandong Tian. Training agent for first-person shooter game with actor-critic curriculum learning. 2017.

[40] Xiaoqin Zhang and Huimin Ma. Pretraining deep actor-critic reinforcement learning algorithms with expert demonstrations, 2018.