Policy Feedback in Deep Reinforcement Learning to exploit expert knowledge

Federico Espositi and Andrea Bonarini

AI and Robotics Lab
Dipartimento di Elettronica, Informazione e Bioingegneria
Politecnico di Milano
Via Ponzio 34\5 - 20133 Milano MI - Italy
http://airlab.deib.polimi.it/
{federico.espositi,andrea.bonarini}@mail.polimi.it

Abstract. In Deep Reinforcement Learning (DRL), agents learn by sampling transitions from a batch of stored data called Experience Replay. In most DRL algorithms, the Experience Replay is filled by experiences gathered by the learning agent itself. However, agents that are trained completely Off-Policy, based on experiences gathered by behaviors that are completely decoupled from their own, cannot learn to improve their own policies. In general, the more algorithms train agents Off-Policy, the more they become prone to divergence. Many possible sources of the problem have been considered, but their relative importance has never been tested. The main contribution of this research is the proposal of a novel Off-Policy learning framework called *Policy Feedback*, used both as a tool to leverage offline-collected expert experiences, and also as a general framework to better the understanding of the issues behind *Off-Policy Learning*.

"The most crucial problems in Off-Policy learning can be solved with the injection of an On-Policy feedback signal of any magnitude."

Keywords: Machine Learning · Deep Reinforcement Learning · DDPG · Exploitation · Policy Feedback.

1 Introduction

It is a wide known fact that RL algorithms that combine Function Approximation, Bootstrapping and Off-Policy Learning, cannot train an agent successfully. As a consequence of their catastrophic interaction, these three elements have been referred to as the *Deadly Triad*, first mentioned by Sutton and Barto [11] and later studied in more depth in the paper [6], which showed empirically that when all of the three components are present in an RL algorithm, the learning procedure would lead to divergence.
However, many state-of-the art techniques such as DQN [10, 9] and DDPG [8] are considered *Off-Policy* techniques, as they learn from data that is not strictly

constrained to be obtained from the current learning policy, but are able to train agents successfully nonetheless.
However, *Batch Learning* techniques are unsuccessful unless the learning policy is constrained to be related to the one explorative one [5, 7].
It was firstly suggested in [6] that the problem is that the concepts of *On* and *Off* Policy learning are not crisp, and thus, instability is obtained in an increasing fashion as the training procedure leans towards *complete off-policiness* (such as the case for *Batch Learning*).
However, though it seems that the On-Policy experiences are crucial for a successful training, their relative importance in the training process has not yet been characterized.
With this work we intent to propose a novel simple architecture of *Policy Feedback*, that allows to leverage both *on* and *off* policy learning with flexibility. In Policy Feedback, the agent's Experience Replay is divided in two buffers, one containing the expert transitions and the second, called Feedback Replay, contains transitions collected by the learner itself, which provides at training time a feedback signal from the agent's own policy. It can be shown that any amount of this Policy Feedback signal is sufficient to stabilize training and leverage the Off-Policy expert knowledge to the fullest, outperforming algorithms which utilize only experiences gathered from the training agent.
By studying the results obtained with Policy Feedback, we provide a deeper understanding of the issues behind pure Off-Policy learning. In particular, two issues are identified. The first is the classical divergence of the function approximator, while the second is related to the newly introduced *Generalization Over Preference hypothesis*, which states that the agent's policy will be driven to traverse trajectories that are overestimated due to function approximation, and that without any feedback from the agent's actual returns, it is impossible to correct for this harmful bias. Policy Feedback stabilizes the function approximator and leverages the *Generalization Over Preference* to correct for it even with a very small portion of Feedback signal, thereby solving the problems behind Off-Policy Learning.
The aim of this dissertation is to introduce and support the following *Feedback Hypothesis* :

> *"The most crucial problems in Off-Policy learning can be solved with the injection of an On-Policy feedback signal of any magnitude."*

2 Motivations

In the literature, many examples exist showing that even in very simple MDPs, the Deadly Triad would lead to divergence when learning the Q-function.

2.1 Tsitsiklis and Van Roy's Counterexample

Consider the MDP shown in Figure 1a, where all immediate rewards are equal to zero. The true Value function V is therefore 0 for each state [12].

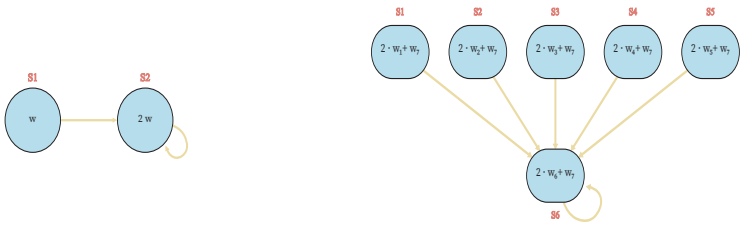
For this MDP, a linear function approximator $\hat{V}(s) = \phi_s^T \cdot W$ is used for the Value function, where ϕ_s is the unique vector of features associated to each state s, and W is the vector of learnable parameters that defines $\hat{V}$.
At least one solution W^* exists, for which the function approximator represents the true Value function correctly, namely W^*=0.
$\hat{V}$ is learned by feeding to the solver a sequence of transitions, and for each transition received at timestep k a single update step is made in the form :

$$W_{k+1} = W_k + \alpha \cdot \rho_k (r_k + \gamma \cdot \hat{V}(s_{k+1}) - \hat{V}(s_k)) \cdot \nabla_W \hat{V}(s_k) \tag{1}$$

When learning On-Policy, the solver would be fed mostly the $s_2 \to s_2$ transition to perform updates of Equation 1. In that case, $W \to 0$ and the learning is consistent.
However for this example, Off-Policy learning is performed, by sampling only the same single transition $s_1 \to s_2$, which is not compatible with the agent's state-visitation distribution. It is shown that in this case, it can happen that $W \to \infty$.

2.2 Baird's Counterexample

The MDP is composed of 6 states, with transition dynamics illustrated in Figure 1b [1]. Immediate reward for each transition is zero, and therefore, also for this system $V_{s_i} = 0 \ \forall \ i = 1, \ldots, 6$.
The value function is approximated by a linear combination of the state features with the parameter vector $W = (w_1, \ldots, w_7)^T$, so that $\hat{V}(s) = \hat{V}(s|W) = \phi_s^T \cdot W$.
The features of each state are shown in Figure 1b.
If the learning is On-Policy, the solver will be fed mostly with transition $s_7 \to s_7$ and the learning will converge.
However, Baird noticed that if the transitions are sampled Off-Policy, and, in particular, if they are all sampled with uniform distribution, the learning will diverge.



(a) Tsitsiklis and Van Roy's Counterexample

(b) Baird's Counterexample

Fig. 1: Counterexamples For The Deadly Triad

2.3 The Deadly Triad

Value function approximation shares parameters over different states, and so, an update to the value for a state may cause undesirable changes to the value of other states. However when learning On-Policy, the agent experiences the true future consequences of past decisions. Referring to Tsitsiklis and Van Roy's Counterexample, whenever the $s_1 \rightarrow s_2$ transition is experienced and W is increased, the agent will then immediately experience a sequence of $s_2 \rightarrow s_2$ transitions, and it will gradually understand that no reward is ever going to be obtained, and $W \rightarrow 0$.
Instead, when learning Off-Policy with Bootstrapping, there is no way of grounding the current estimate over the true long-term consequences of the immediate choices.
These considerations are similar to the ones in [7], in which researchers state that with pure *Off-Policy* learning, the policy is trained for actions that are "suggested" by the learning policy, but which are not present in the training set, and for which the Q values that are learned are therefore inaccurate and in general prone to divergence.

2.4 *Off-Policy* Learning in the Literature

Most state-of-the art algorithms are not *On-Policy*, but are still successful in practice. In DQN[10, 9] and DDPG[8], low magnitude randomness is injected into the learner's policy to promote exploration. As a result, the experiences are not gathered completely On-Policy, but the behavioural (exploratory) policy is closely related to the learner's. Moreover, the training data are collected into the Experience Replay which thus also contains transitions related to older versions of the agent's policy, different from the current one.
The key point is that the concepts of "On-Policy" and "Off-Policy" are not

binary. In [6] it was argued that the more algorithms train agents Off-Policy, meaning, the more the explorative policy is different from the training policy, the more they become prone to divergence.
To test this, Prioritized Experience Replays were used, and the training was made progressively more Off-Policy by changing the Importance Sampling weights. As the corrective weight's effect faded, the training turned towards Off-Policy, and learning diverged more frequently.

The use of a fixed batch of expert data for training is the limit case of complete Off-Policy Learning and, as could be expected, policies cannot be learned in this setting, even if the transitions were sampled from an expert policy, as shown in [4, 7].
Current state-of-the-art techniques for *Batch Learning* address this limitation by constraining the learned policy to be similar to the one seen used to collect experiences [5, 7].

2.5 Feedback For Counterexamples

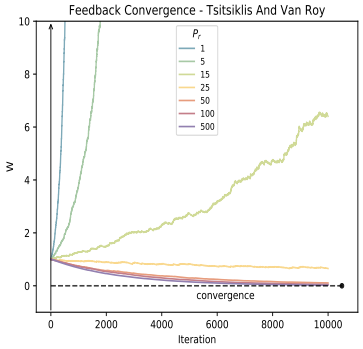
We propose a framework to explicitly combine contributions from *On-Policy* and *Off-Policy* experiences, the *Policy Feedback* architecture. The first intuition about *Policy Feedback* came when studying Tsitsiklis and Van Roy's and Baird's counterexamples.
For both systems, the On-Policy distribution is concentrated on the recursive transition($s_2 \rightarrow s_2$ and $s_7 \rightarrow s_7$ respectively), which is the stable transition that allows the procedure to convergence. This is because by following the true system dynamics, it will be impossible for the training procedure to continue overestimating artificially the amount of future rewards the agent would supposedly receive.
However, Off-Policy samplings increases the proportion of the other transitions which lead to artificial overestimations, causing divergence.

The observation that gave rise to *Policy Feedback* is the following:
for both examples, we can consider two policies, the agent's policy which follows the MDP's dynamics and that always leads to stable learning, and the behavioral policy which visits all transitions with the off-policy sampling distribution. We can model these two policies with two different Experience Replays, in which transitions are stored to meet the proportions enforced by the corresponding policy. If transitions are sampled from these batches with uniform probability, the distribution of sampled transitions will meet the corresponding visitation distribution. The buffer of the behavioral policy, $\mathcal{B}_b$, contains a single copy of each transition, while the buffer of the agent's policy, $\mathcal{B}_a$, contains a single transition, the recursive one. When sampling with uniform distribution from this batch, only the recursive transition will be sampled, over and over, as prescribed by the agent's policy.
We introduce a proportionality coefficient P_r to formalize both Experience Replays with a single formulation $\mathcal{B}_{P_r}$. P_r represents the quantity of recursive

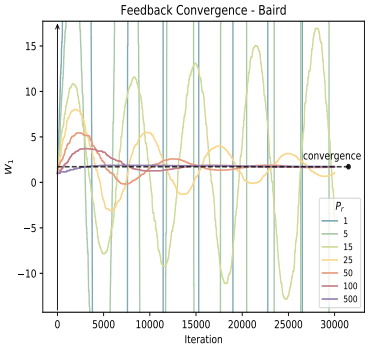
transitions present in the batch, for each copy of the other transitions. The two limit cases are the following :

- P_r=1 , $\mathcal{B}_{P_r} = \mathcal{B}_b$. All transitions are present in the same proportion (Off-Policy, uniform sampling).
- $P_r \to \infty$, $\mathcal{B}_{P_r} \to \mathcal{B}_a$. The probability of sampling a transition that is not the recursive one is infinitesimal (On-Policy sampling)

However, training does not have to be limited to these two cases.
By adjusting $P_r \in (1, \infty)$, we can adjust the sampling distribution.
Choosing $P_r > 1$, we are injecting into the Off-Policy batch a certain amount of transitions sampled On-Policy. As a consequence, the overall batch distribution moves away from the behavioral's and closer to the agent's.
Does the training process need $P_r \to \infty$ to be stable? It does not.
Indeed, for the systems in both the counterexamples, there exists a P_{r_min} so that if $P_r > P_{r\_min}$, the learning stabilizes and is able to converge to the correct solution.
Different values of P_r were tested for both the examples. The learning trends are shown in Figure 2.



(a) Policy Feedback For Tsitsiklis and Van Roy's Counterexample



(b) Policy Feedback For Baird's Counterexample

Fig. 2: Policy Feedback For Counterexamples For Different Feedback Intensity P_r
The proportion of On-Policy signal P_r is increased. After a threshold value, system becomes asymptotically stable.
In Figure 2b, only training behaviour of the first parameter W_1 is shown, as it is representative of all parameters W_i , $i = 1, \dots 7$

The overall takeaway of this section is that there exists a minimum proportion of transitions sampled On-Policy which stabilizes the learning, even if the remaining experiences are gathered completely Off-Policy.
This *feedback* signal of the agent's policy, if large enough, is sufficient to avoid divergence, even in the presence of other destabilizing distributions.

If in general this idea could be leveraged to contain the issues that arise when learning Off-Policy, while at the same time making the most out of expert transitions gathered offline, this could lead to an increase in performance. This is the core idea of *Policy Feedback*.

3 Design

The *Policy Feedback* method is based upon the standard DDPG algorithm [8], modified by combining Batch Learning with the bio-inspired experience replay introduced in [13].
In *Policy Feedback*, the Experience Replay is composed of two separate buffers.

- The *Expert Memory* $\mathcal{M}_{\mathcal{E}}$, the fixed batch containing the expert transitions collected offline.
- The *Feedback Memory* $\mathcal{M}_{\mathcal{F}}$, a classical FIFO Experience Replay which is progressively filled with the most recent experiences gathered by the learner.

As the expert already provides highly fruitful experiences, there is no need for the agent to perform noisy exploration. In fact, the purpose of the agent's experience is not exploration itself, but an evaluation of the current policy's performance, which allows for online correction.
When building mini-batches for training, a single parameter, P_{on_p} -probability of On-Policy-, controls the proportion of transitions belonging to either $Memory$ Replay. P_{on_p} controls the magnitude of the equivalent feedback signal (and the amount of on-policiness of the algorithm).
The two limit cases are the following :

- With P_{on_p}=1 , *Policy Feedback* corresponds to classical DDPG with no exploration noise.
- With P_{on_p}=0 , *Policy Feedback* corresponds to full Off-policy Batch Learning from the expert batch.

4 Experiments And Results

All the agent hyperparameters follow those of the DDPG paper [DDPG].
Experiments were repeated for different values of P_{on_p} using the same Expert Replays $\mathcal{M}_{\mathcal{E}}$, for two environments of the gym library : *"LunarLanderContinuous-v2"* and *"Swimmer-v2"* [2]. Results are shown in Figure 3.

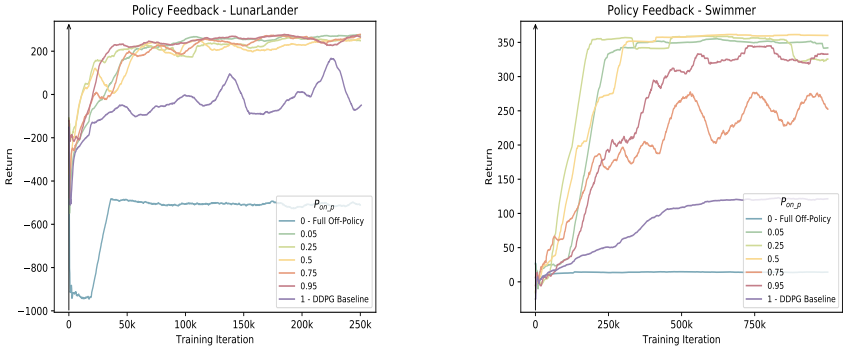


Fig. 3: Policy Feedback For Different Values Of Feedback Intensity P_{on_p}

In the full-Off-Policy case, with P_{on_p}=0, the agent is completely unable to improve its policy.
Instead, with any tested value of $P_{on\_p} > 0$ the learning was successful, and actually led to a considerable increase of the speed of learning with respect to the to the DDPG baseline, presenting similar features in terms of learning curves and obtained scores.
Notice that for *Swimmer*, the higher values of P_{on_p}, corresponding to a more On-Policy learning, lead to slower trends. Indeed, this environment is structurally more sensible to local minima, and by providing datapoints that are more explorative, it could be hard to exploit the expert transitions to escape them.

5 Discussion

In Figure 4, training losses of the neural Q functions are shown for the corresponding experimental results in Figure 3.

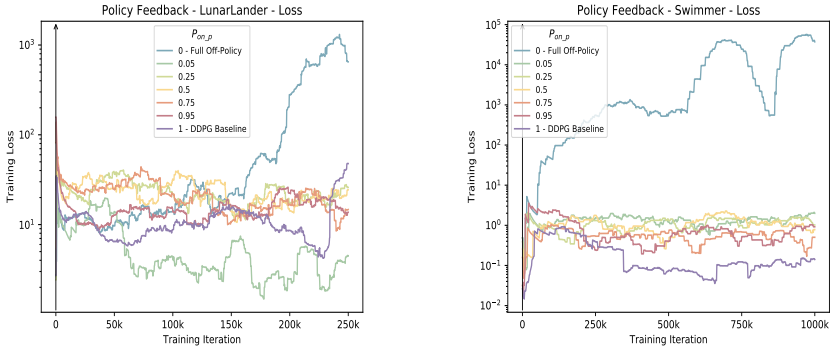


Fig. 4: Q-function Training Loss In Policy Feedback

It is clear from results in Figure 3 that the On-Policy feedback signal is crucial for the training process to be successful, and is in general enough to support and increase performance of the baseline DDPG procedure.

The Generalization Over Preference Hypothesis For the *LunarLander* case, results show that the low performance obtained when learning completely off-Policy seems to be decorrelated from Q-instability. We therefore assume that Q-divergence is not the only factor at play during Off-Policy learning, and propose to find another. In this study, the *Generalization Over Preference Hypothesis* is presented :

> " An issue with complete Off-Policy Learning with Function Approximation and Bootstrapping is the generalization of the estimated Q-function over regions of the state-action space that are not explored by the *Behavioral Policy*, but to which the training agent would be led to as a result of training. "

Indeed, the Q function is learned from the transitions on the Experience Replay, and the policy is updated to choose actions which maximize the learned Q. We call these actions the *preference*. Due to function approximation, Q generalizes and may be artificially high in some regions, leading to *preferences* that are actually low-performing.
We argument that *generalization over preference* is a crucial cause of low-performance of off-policy learning, and that it is only a subset of the *generalization* problem. Indeed, to solve the latter, the Q function would need to be learned accurately across the entire state-action space [4, 3], while the *generalization over preference* can be addressed by only adjusting the Q values over the current *preference*. This is exactly what is done in *Policy Feedback*, which always feeds training with experiences collected by the learner, and which are thus related precisely to the *preference*, which are therefore adjusted. As such, *Policy Feedback* provides an empirical confirmation of the *generalization over preference hypothesis*

6 Conclusions

In this paper, the *Policy Feedback* framework has been introduced, with he intent to more deeply understand the issues behind *Off-Policy* learning and characterize the relative importance of *On-Policy* samples coming from the agent's own policy. The architecture is based on the DDPG algorithm, but combines experiences coming from both the agent's on policy and an offline expert though the use of two separate memory buffers, from which the relative number of samples can be controlled by a single parameter, thus controlling the equivalent feedback magnitude used during trianing.
Results show that while the absence of such a signal leads to unstable and unfeasible training, the injection of any amount of feedback from the agent's own policy is able to stabilize learning, allowing the agent to leverage the expert experiences to the fullest, increasing the performance of the purely *On-policy* DDPG.
Policy Feedback leverages the *Generalization Over Preference* issue to its advantage, since it uses the transitions that the agent is directly led to by the training process and immediately corrects for the inconsistencies.

Moreover, it was shown that any amount of feedback from the agent's policy distribution was able to lead the training Q loss towards convergence, while the Full-Off-Policy case, compatibly with the studies on the Deadly Triad, would still diverge.
As a result, *Policy Feedback* is able to solve both the problems of Off-Policy learning, and leverage the available expert transitions to the fullest.
These results support the *Feedback Hypothesis* introduced at the beginning.

References

[1] L. Baird. "Residual Algorithms: Reinforcement Learning with Function Approximation". In: *In Proceedings of the Twelfth International Conference on Machine Learning*. Morgan Kaufmann, 1995, pp. 30–37.

[2] G. Brockman et al. "OpenAI Gym". In: *CoRR* abs/1606.01540 (2016). arXiv: 1606.01540. URL: http://arxiv.org/abs/1606.01540.

[3] T. de Bruin et al. "Improved Deep Reinforcement Learning for Robotics Through Distribution-Based Experience Retention". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2016), pp. 3947–3952.

[4] T. de Bruin et al. "The Importance of Experience Replay Database Composition in Deep Reinforcement Learning". In: Jan. 2015.

[5] S. Fujimoto, D. Meger, and D. Precup. "Off-Policy Deep Reinforcement Learning without Exploration". In: *CoRR* abs/1812.02900 (2018). arXiv: 1812.02900. URL: http://arxiv.org/abs/1812.02900.

[6] H. van Hasselt et al. "Deep Reinforcement Learning and the Deadly Triad". In: *CoRR* abs/1812.02648 (2018). arXiv: 1812.02648. URL: http://arxiv.org/abs/1812.02648.

[7] A. Kumar et al. *Stabilizing Off-Policy Q-Learning via Bootstrapping Error Reduction*. 2019. arXiv: 1906.00949 [cs.LG].

[8] T. P. Lillicrap et al. *Continuous Control With Deep Reinforcement Learning*. 2015. arXiv: 1509.02971 [cs.LG].

[9] V. Mnih et al. "Human-Level Control Through Deep Reinforcement Learning". In: *Nature* 518 (Feb. 2015), pp. 529–33. DOI: 10.1038/nature14236.

[10] V. Mnih et al. *Playing Atari with Deep Reinforcement Learning*. 2013. arXiv: 1312.5602 [cs.LG].

[11] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. Second. The MIT Press, 2018. URL: http://incompleteideas.net/book/the-book-2nd.html.

[12] J. N. Tsitsiklis and B. Van Roy. "An Analysis of Temporal-Difference Learning with Function Approximation". In: *IEEE Transactions on Automatic Control* 42.5 (May 1997), pp. 674–690. ISSN: 2334-3303. DOI: 10.1109/9.580874.

[13] Z. Zhang et al. "Asynchronous Episodic Deep Deterministic Policy Gradient: Towards Continuous Control in Computationally Complex Environments". In: *CoRR* abs/1903.00827 (2019). arXiv: 1903.00827. URL: http://arxiv.org/abs/1903.00827.