Hindsight policy gradients

Paulo Rauber
IDSIA, USI, SUPSI
Manno-Lugano, Switzerland
paulo@idsia.ch

Filipe Mutz*
IFES
Serra, Brazil
filipe.mutz@ifes.edu.br

Jürgen Schmidhuber
IDSIA, USI, SUPSI
NNAISENSE
Manno-Lugano, Switzerland
juergen@idsia.ch

Abstract

Goal-conditional policies allow reinforcement learning agents to pursue specific goals during different episodes. In addition to their potential to generalize desired behavior to unseen goals, such policies may also help in defining options for arbitrary subgoals, enabling higher-level planning. While trying to achieve a specific goal, an agent may also be able to exploit information about the degree to which it has achieved alternative goals. Reinforcement learning agents have only recently been endowed with such capacity for hindsight, which is highly valuable in environments with sparse rewards. In this paper, we show how hindsight can be introduced to likelihood-ratio policy gradient methods, generalizing this capacity to an entire class of highly successful algorithms. Our preliminary experiments suggest that hindsight may increase the sample efficiency of policy gradient methods.

1 Introduction

In a traditional reinforcement learning setting, an agent interacts with an environment in a sequence of episodes, observing states and acting according to a policy that ideally maximizes expected cumulative reward [1]. If the agent is required to achieve a specific *goal* during an episode, then such goal may be encoded as part of the states [2], possibly allowing the generalization of desired behavior to goals that were never encountered before. Recent examples of learning *goal-conditional* behavior include the works of Da Silva et al. [3], Schmidhuber [4], Srivastava et al. [5], Kupcsik et al. [6], Deisenroth et al. [7], Fabisch and Metzen [8], Schaul et al. [9], Zhu et al. [10], Held et al. [11].

An interesting application for goal-conditional policies is hierarchical reinforcement learning [12–30], where they may help in defining *options* for arbitrary subgoals [22]. In that case, instead of planning to perform a sequence of actions, an agent could plan to achieve a sequence of subgoals, abstracting the details involved in lower-level decisions. Recent examples of this approach include the works of Oh et al. [31], Vezhnevets et al. [32], Kulkarni et al. [33].

While trying to achieve a specific goal, an agent may also be able to exploit information about the degree to which it has achieved alternative goals. For example, we may expect a traveller to learn how to arrive at his *eventual* destination, whether or not that was his *intended* destination. This capacity for *hindsight* was introduced by Andrychowicz et al. [34] to off-policy reinforcement learning algorithms that rely on experience replay [35], and has proven to be particularly important in environments with sparse rewards.

In this paper, we show how *importance sampling* can be used to introduce hindsight to likelihood-ratio policy gradient methods [36–39] (Sec. 2), generalizing this idea to a highly successful class of reinforcement learning algorithms that achieve state-of-the-art results in many tasks [40]. Importance sampling has previously been applied to policy gradient methods in order to efficiently reuse information obtained by earlier policies [41]. In contrast, our approach attempts to efficiently learn about

*Work performed while at IDSIA.

Accepted to NIPS 2017 Hierarchical Reinforcement Learning Workshop, Long Beach, CA, USA.

different goals using information obtained by the current policy for a specific goal. Our preliminary experiments suggest that hindsight may indeed increase the sample efficiency of policy gradient methods (Sec. 3).

2 Hindsight policy gradients

Policy gradients. Consider an agent that interacts with its environment in a sequence of episodes, each of which lasts for exactly T time steps. The agent receives a goal g at the beginning of each episode. At each time step t, the agent observes a state s_t, receives a reward $r(s_t, g)$, and chooses an action a_t. In this setting, a policy gradient method may represent a policy by a probability distribution over actions given state and goal. The objective is finding parameters for such a policy that achieve maximum expected return (cumulative reward). For simplicity of notation, we consider finite state, action, and goal spaces. We denote random variables by upper case letters and assignments to these variables by lower case letters.

Let $\boldsymbol{\tau} = s_1, a_1, s_2, a_2, \ldots, s_{T-1}, a_{T-1}, s_T$ denote a trajectory. We assume that the probability $p(\boldsymbol{\tau} \mid g, \boldsymbol{\theta})$ of trajectory $\boldsymbol{\tau}$ given goal g and a policy parameterized by $\boldsymbol{\theta}$ is given by

$$p(\boldsymbol{\tau} \mid g, \boldsymbol{\theta}) = p(s_1) \prod_{t=1}^{T-1} p(a_t \mid s_t, g, \boldsymbol{\theta}) p(s_{t+1} \mid s_t, a_t). \tag{1}$$

The expected return $\eta(\boldsymbol{\theta})$ of a policy parameterized by $\boldsymbol{\theta}$ is defined as

$$\eta(\boldsymbol{\theta}) = \sum_g p(g) \sum_{\boldsymbol{\tau}} p(\boldsymbol{\tau} \mid g, \boldsymbol{\theta}) \sum_{t=1}^{T} r(s_t, g). \tag{2}$$

Employing the *likelihood-ratio trick* and noting that $A_t \perp\!\!\!\perp S_{t'} \mid S_t, G, \boldsymbol{\Theta}$ for $t \geq t'$ [39], the gradient $\nabla\eta(\boldsymbol{\theta})$ of the expected return with respect to the policy parameters is given by

$$\nabla\eta(\boldsymbol{\theta}) = \sum_g p(g) \sum_{\boldsymbol{\tau}} p(\boldsymbol{\tau} \mid g, \boldsymbol{\theta}) \sum_{t=1}^{T-1} \nabla \log p(a_t \mid s_t, g, \boldsymbol{\theta}) \sum_{t'=t+1}^{T} r(s_{t'}, g). \tag{3}$$

Therefore, if $\boldsymbol{\tau}$ is a trajectory obtained by following a policy parameterized by $\boldsymbol{\theta}$ to achieve a goal g chosen by the environment, an unbiased estimate $\boldsymbol{\delta}_{\text{PG}}$ of the gradient $\nabla\eta(\boldsymbol{\theta})$ is given by

$$\boldsymbol{\delta}_{\text{PG}} = \sum_{t=1}^{T-1} \nabla \log p(a_t \mid s_t, g, \boldsymbol{\theta}) \sum_{t'=t+1}^{T} r(s_{t'}, g). \tag{4}$$

We will refer to this estimate by the term *policy gradient* (PG), which may be used for gradient ascent.

Hindsight policy gradients. Alternatively, suppose that we are easily able to compute $r(s, g)$ for any state s and goal g, which is equivalent to the assumption made by Andrychowicz et al. [34]. In that case, it is possible to evaluate a trajectory obtained while trying to achieve goal g' for an alternative goal g. For an arbitrary goal g', *importance sampling* [42] allows rewriting the gradient of the expected return as

$$\begin{aligned}\nabla\eta(\boldsymbol{\theta}) &= \sum_g p(g) \sum_{\boldsymbol{\tau}} \frac{p(\boldsymbol{\tau} \mid g', \boldsymbol{\theta})}{p(\boldsymbol{\tau} \mid g', \boldsymbol{\theta})} p(\boldsymbol{\tau} \mid g, \boldsymbol{\theta}) \sum_{t=1}^{T-1} \nabla \log p(a_t \mid s_t, g, \boldsymbol{\theta}) \sum_{t'=t+1}^{T} r(s_{t'}, g) \qquad (5)\\ &= \sum_{\boldsymbol{\tau}} p(\boldsymbol{\tau} \mid g', \boldsymbol{\theta}) \sum_g p(g) \left[\prod_{t=1}^{T-1} \frac{p(a_t \mid s_t, g, \boldsymbol{\theta})}{p(a_t \mid s_t, g', \boldsymbol{\theta})}\right] \sum_{t=1}^{T-1} \nabla \log p(a_t \mid s_t, g, \boldsymbol{\theta}) \sum_{t'=t+1}^{T} r(s_{t'}, g).\end{aligned}$$

Therefore, if $\boldsymbol{\tau}$ is a trajectory obtained by following a policy parameterized by $\boldsymbol{\theta}$ to achieve an arbitrary goal g', another unbiased estimate $\boldsymbol{\delta}_{\text{HPG}}$ of the gradient $\nabla\eta(\boldsymbol{\theta})$ is given by

$$\boldsymbol{\delta}_{\text{HPG}} = \sum_g p(g) \left[\prod_{t=1}^{T-1} \frac{p(a_t \mid s_t, g, \boldsymbol{\theta})}{p(a_t \mid s_t, g', \boldsymbol{\theta})}\right] \sum_{t=1}^{T-1} \nabla \log p(a_t \mid s_t, g, \boldsymbol{\theta}) \sum_{t'=t+1}^{T} r(s_{t'}, g). \tag{6}$$

We will refer to this estimate by the term *hindsight policy gradient* (HPG), as it evaluates a trajectory on goals other than the intended. Although it would be possible to sample goals to compute an estimate of $\boldsymbol{\delta}_{\text{HPG}}$, in this paper we focus on two variants of this estimate that are particularly simple to compute for the environments that we chose for the experiments presented in Sec. 3.

The first variant is the *uniform hindsight policy gradient* $\boldsymbol{\delta}_{\text{UHPG}}$ given by

$$\boldsymbol{\delta}_{\text{UHPG}} = \sum_{g \in G} \left[\prod_{t=1}^{T-1} \frac{p(a_t \mid s_t, g, \boldsymbol{\theta})}{p(a_t \mid s_t, g', \boldsymbol{\theta})} \right] \sum_{t=1}^{T-1} \nabla \log p(a_t \mid s_t, g, \boldsymbol{\theta}) \sum_{t'=t+1}^{T} r(s_{t'}, g), \tag{7}$$

where $G = \{g \mid \text{exists a } t \text{ such that } r(s_t, g) \neq 0\}$. This variant is named after the fact that $\boldsymbol{\delta}_{\text{UHPG}} \propto \boldsymbol{\delta}_{\text{HPG}}$ when the goal distribution is uniform.

The second variant is the *average hindsight policy gradient* $\boldsymbol{\delta}_{\text{AHPG}}$ given by

$$\boldsymbol{\delta}_{\text{AHPG}} = \sum_{g} q(g) \sum_{t=1}^{T-1} \nabla \log p(a_t \mid s_t, g, \boldsymbol{\theta}) \sum_{t'=t+1}^{T} r(s_{t'}, g), \tag{8}$$

where q is the empirical probability function over *states* in trajectory $\boldsymbol{\tau}$, which comes with the additional assumption that goals represent states. This variant also ignores the *likelihood-ratio*, performing updates as if the trajectory $\boldsymbol{\tau}$ were equally likely given any goal. Perhaps surprisingly, this variant achieves excellent results in our preliminary experiments.

We emphasize that these two variants are not in general unbiased estimates of the hindsight policy gradient, which will be studied further in future work.

3 Experiments

Environments. We performed preliminary experiments in two simple environments, which allowed systematic hyperparameter search and aggregating results across a large number of runs.

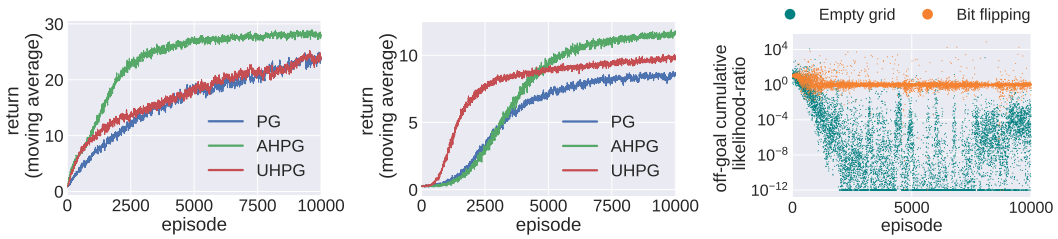
In the *empty grid* environment, the agent starts each episode in the upper left corner of a 6×6 grid, and its goal is to achieve a randomly chosen position in this grid. The actions allow the agent to move in the four cardinal directions or to stay in the same position (which also happens if an action is invalid). A state or goal is represented by a pair of integers between 0 and 5. The reward is zero if the agent is not at the goal, and one otherwise. Each episode lasts for 36 time steps.

In the *bit flipping* environment, the agent starts each episode in the same state ($\mathbf{0}$) represented by 6 bits, and its goal is to achieve a randomly chosen state. The actions allow the agent to flip each of the bits individually, or to remain in the same state. The reward is zero if the agent is not at the goal state, and one otherwise. Each episode lasts for 16 time steps. Andrychowicz et al. [34] use a similar environment to evaluate their hindsight approach.

These environments have two important characteristics in common. Firstly, the optimal behavior corresponds to reaching the goal state as fast as possible and staying there. Secondly, the UHPG/AHPG can be computed using only the states (goals) observed during a trajectory.

Policy optimization. The policy is represented by a feedforward neural network with a single hyperbolic tangent hidden layer and a *softmax* output layer. Parameters are updated using Adam [43]. The UHPG is scaled by the inverse of the cumulative likelihood-ratio across goals in G (or a large constant if the denominator is close to zero) to avoid large discrepancies between step sizes.

Hyperparameter search. Each technique was evaluated in 21 runs lasting 10000 episodes for every combination of environment, hidden layer size in $\{8, 16, 32, 64\}$, and learning rate in $\{10^{-1}, 5 \times 10^{-2}, 10^{-2}, 5 \times 10^{-3}, 10^{-3}, 5 \times 10^{-4}, 10^{-4}, 5 \times 10^{-5}\}$. The average return is computed for each individual run, and its average (*average performance*) is used to select the best combination for each technique after the corresponding standard deviation is subtracted. This criterion discourages hyperparameter combinations with large fluctuations in performance [40]. For the empty grid, all techniques settle on a hidden layer of size 8 and a learning rate of 5×10^{-3}, except for the AHPG with a learning rate of 10^{-2}. For the bit flipping environment, learning rate is 5×10^{-3} for PG and UHPG, 10^{-3} for AHPG; hidden layer size is 32 for PG and AHPG, and 8 for UHPG.



(a) Empty grid results (b) Bit flipping results (c) Off-goal cumulative LR

Figure 1: (a) Empty grid results. PG: 16.07 ± 2.31, **AHPG:** 23.51 ± 3.78, UHPG: 17.06 ± 2.15. (b) Bit flipping results. PG: 5.76 ± 1.49, AHPG: 7.34 ± 1.38, **UHPG:** 7.77 ± 2.83. (c) Cumulative likelihood-ratio across goals other than the intended (UHPG, single run, values clipped at 10^{-12}).

Analysis. The results of 51 additional runs using the hyperparameters described above are summarized in Figures 1a and 1b, which also include the average performance for each technique. The learning curves represent the average *smoothed returns* across runs. These smoothed returns are obtained using a moving average across episodes with a window of size 10.

These results indicate that the AHPG achieves better sample efficiency than the PG, specially in the empty grid. Besides substituting the distribution over goals by the empirical distribution of states in a trajectory, this HPG variant ignores the fact that a trajectory obtained while trying to achieve goal g' may be unlikely when trying to achieve goal g. Although this could potentially disrupt the behavior of the policy on trajectories that are more likely to occur when trying to achieve g, this risk appears to be outweighed by the benefits of hindsight in the environments that we considered.

The UHPG achieves comparable performance to the PG in the empty grid. This is explained by the fact that the likelihood-ratio becomes very small for goals other than the intended as the policy begins to exhibit distinct behaviors for distinct goals (see Fig. 1c). If the likelihood-ratio were always zero for goals other than the intended, the UHPG would become equivalent to the PG (*cf.* Eqs. 4 and 7).

The off-goal likelihood-ratio no longer vanishes in the bit flipping environment (see Fig. 1c), likely due to shorter episodes. In this environment, the UHPG also appears to outperform the AHPG during early training. More experiments are required to investigate this behavior.

4 Conclusion

We introduced techniques that enable optimizing goal-conditional policies using hindsight. In this context, hindsight refers to the capacity to exploit information about the degree to which an arbitrary goal has been achieved while another goal was intended. Prior to this work, hindsight has been limited to off-policy reinforcement learning algorithms that rely on experience replay [34]. Our preliminary experiments suggest that hindsight may increase the sample efficiency of policy gradient methods. In future work, we will further study the properties of the proposed techniques, include improvements commonly found in policy gradient methods (such as baselines), and perform comprehensive experiments on more challenging environments.

Acknowledgments

We would like to thank Sjoerd van Steenkiste and Klaus Greff for their valuable feedback. This research was supported by the Swiss National Science Foundation (grant 200021_165675/1) and CAPES (Filipe Mutz, PSDE, 88881.133206/2016-01). We are also grateful to NVIDIA Corporation for donating a *DGX-1* machine and to IBM for donating a *Minsky* machine.

References

[1] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. Bradford Book, 1998. ISBN 9780262193986.

[2] J. Schmidhuber and R. Huber. Learning to generate artificial fovea trajectories for target detection. *International Journal of Neural Systems*, 2(1 & 2):135–141, 1991. (Based on TR FKI-128-90, TUM, 1990).

[3] B. C. Da Silva, G. Konidaris, and A. G. Barto. Learning parameterized skills. In *Proceedings of International Conference of Machine Learning*, 2012.

[4] J. Schmidhuber. POWERPLAY: Training an increasingly general problem solver by continually searching for the simplest still unsolvable problem. *Frontiers in Psychology*, 2013. (Based on arXiv:1112.5309v1 [cs.AI], 2011).

[5] R. K. Srivastava, B. R. Steunebrink, and J. Schmidhuber. First experiments with PowerPlay. *Neural Networks*, 41(0):130 – 136, 2013. ISSN 0893-6080. Special Issue on Autonomous Learning.

[6] A. G. Kupcsik, M. P. Deisenroth, J. Peters, and G. Neumann. Data-efficient generalization of robot skills with contextual policy search. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence, AAAI 2013*, pages 1401–1407, 2013.

[7] M. P. Deisenroth, P. Englert, J. Peters, and D. Fox. Multi-task policy search for robotics. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 3876–3881. IEEE, 2014.

[8] A. Fabisch and J. H. Metzen. Active contextual policy search. *The Journal of Machine Learning Research*, 15(1):3371–3399, 2014.

[9] T. Schaul, D. Horgan, K. Gregor, and D. Silver. Universal value function approximators. In *Proceedings of the International Conference on Machine Learning*, pages 1312–1320, 2015.

[10] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 3357–3364. IEEE, 2017.

[11] D. Held, X. Geng, C. Florensa, and P. Abbeel. Automatic goal generation for reinforcement learning agents. *arXiv preprint arXiv:1705.06366*, 2017.

[12] J. Schmidhuber. Learning to generate sub-goals for action sequences. In *Artificial Neural Networks*, pages 967–972. Elsevier Science Publishers B.V., North-Holland, 1991.

[13] M. B. Ring. Incremental development of complex behaviors through automatic construction of sensory-motor hierarchies. In *Machine Learning: Proceedings of the Eighth International Workshop*, pages 343–347. Morgan Kaufmann, 1991.

[14] J. Tenenberg, J. Karlsson, and S. Whitehead. Learning via task decomposition. In *From Animals to Animats 2: Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, pages 337–343. MIT Press, 1993.

[15] P. Dayan and G. Hinton. Feudal reinforcement learning. In *Advances in Neural Information Processing Systems (NIPS) 5*, pages 271–278. Morgan Kaufmann, 1993.

[16] J. Schmidhuber. On learning how to learn learning strategies. Technical Report FKI-198-94, Fakultät für Informatik, Technische Universität München, 1994.

[17] G. Weiss. Hierarchical chunking in classifier systems. In *Proceedings of the 12th National Conference on Artificial Intelligence*, volume 2, pages 1335–1340. AAAI Press/The MIT Press, 1994.

[18] J. R. Olsson. Inductive functional programming using incremental program transformation. *Artificial Intelligence*, 74(1):55–83, 1995.

[19] F. J. Gomez and R. Miikkulainen. Incremental evolution of complex general behavior. *Adaptive Behavior*, 5:317–342, 1997.

[20] D. Precup, R. S. Sutton, and S. Singh. Multi-time models for temporally abstract planning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1050–1056. Morgan Kaufmann, 1998.

[21] T. G. Dietterich. Hierarchical reinforcement learning with the MAXQ value function decomposition. *J. Artif. Intell. Res. (JAIR)*, 13:227–303, 2000.

[22] R. S. Sutton, D. Precup, and S. Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

[23] I. Menache, S. Mannor, and N. Shimkin. Q-Cut – dynamic discovery of sub-goals in reinforcement learning. In *Proc. ECML'02*, pages 295–306, 2002.

[24] M. Ghavamzadeh and S. Mahadevan. Hierarchical policy gradient algorithms. In *Proceedings of the Twentieth Conference on Machine Learning (ICML-2003)*, pages 226–233, 2003.

[25] A. G. Barto and S. Mahadevan. Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems*, 13(4):341–379, 2003.

[26] B. Bakker and J. Schmidhuber. Hierarchical reinforcement learning based on subgoal discovery and subpolicy specialization. In *Proc. 8th Conference on Intelligent Autonomous Systems IAS-8*, pages 438–445, Amsterdam, NL, 2004. IOS Press.

[27] J. Schmidhuber. Optimal ordered problem solver. *Machine Learning*, 54:211–254, 2004.

[28] S. Whiteson, N. Kohl, R. Miikkulainen, and P. Stone. Evolving keepaway soccer players through task decomposition. *Machine Learning*, 59(1):5–30, May 2005.

[29] S. Singh, A. G. Barto, and N. Chentanez. Intrinsically motivated reinforcement learning. In *Advances in Neural Information Processing Systems 17 (NIPS)*. MIT Press, Cambridge, MA, 2005.

[30] M. Ring, T. Schaul, and J. Schmidhuber. The two-dimensional organization of behavior. In *Proceedings of the First Joint Conference on Development Learning and on Epigenetic Robotics ICDL-EPIROB*, Frankfurt, August 2011.

[31] J. Oh, S. Singh, H. Lee, and P. Kohli. Zero-shot task generalization with multi-task deep reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2661–2670, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[32] A. S. Vezhnevets, S. Osindero, T. Schaul, N. Heess, M. Jaderberg, D. Silver, and K. Kavukcuoglu. FeUdal networks for hierarchical reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3540–3549, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[33] T. D. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Advances in Neural Information Processing Systems*, pages 3675–3683, 2016.

[34] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba. Hindsight experience replay. *arXiv preprint arXiv:1707.01495*, 2017.

[35] L. Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning*, 8(3/4):69–97, 1992.

[36] R. J. Williams. Reinforcement-learning in connectionist networks: A mathematical analysis. Technical Report 8605, Institute for Cognitive Science, University of California, San Diego, 1986.

[37] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[38] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems (NIPS) 12*, pages 1057–1063, 1999.

[39] J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural networks*, 21 (4):682–697, 2008.

[40] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel. Benchmarking deep reinforcement learning for continuous control. In *Proceedings of International Conference on Machine Learning*, pages 1329–1338, 2016.

[41] T. Jie and P. Abbeel. On a connection between importance sampling and the likelihood ratio policy gradient. In *Advances in Neural Information Processing Systems*, pages 1000–1008, 2010.

[42] C. M. Bishop. *Pattern Recognition and Machine Learning*. Information science and statistics. Springer, 2013. ISBN 9788132209065.

[43] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2014.