Noname manuscript No.
(will be inserted by the editor)

Neural Networks and their Application in Reinforcement Learning

Reinforcement Learning Seminar – Winter Semester 2018/19

Fabian Otto

Received: date / Accepted: date

Abstract Insert your abstract here. Include keywords, PACS and mathematical subject classification numbers as needed.

Keywords Reinforcement Learning · Neural Networks

1 Introduction

Give introduction with NN hype or similar things. As well as the importance. Define RL and the method in general. Show clear distinction to SL. Maybe include success of recent things in CV and NLP, might also be possible in 3 to transition to more recent approaches in RL. Introduce RL goal and notation, short intro ro MDPs.

2 Definition of Neural Networks

[**?**] and [**?**]. Describe formally how NN are working, how can they be trained, what other methods do we have, etc.

3 History of Neural Networks

(see Sect. **??**). What did lead to the rise and fall of NNs throughout time. Create good transition to recent approaches.

Fabian Otto
Technische Universität Darmstadt, Computer Science Department
E-mail: fabian.otto@stud.tu-darmstadt.de

4 Algorithms

This section provides an overview of NN based RL algorithms including earlier work and its improvements as well as current state-of-the-art. Most algorithms can be either classified as on-policy or off-policy algorithms. On-policy algorithms estimate the value of policy while using it for control whereas in off-policy algorithms, the generating policy, called *behavior policy*, is not necessarily related to the *target policy*, which is improved and evaluated. This allows to use a deterministic target policy, while sampling all actions applying the behavior policy. [15, chapter 5]

4.1 Off-Policy

MDPs One of the earlier NN based off-policy algorithms, named Neural fitted Q Iteration (NFQ), was published by Riedmiller [10] in 2005. NFQ is a model-free algorithm, based on Q-Learning [19] and approximates the action-value function with multilayer-perceptrons. Compared to other approaches at the time, NFQ allows to learn with a relatively good data efficiency. Further, batches from a replay memory [7] are used in order to train the multilayer-perceptron with resilient backpropagation (RPROP) [11]. Thereby, it is possible to dynamically add new samples to the replay memory during learning. Motivated by this idea as well as the success of TD-gammon [16] the arguably biggest improvement in NN based RL – Deep Q-Networks (DQN) – were introduced by Mnih, et al. [9]. They built upon their prior results [8] on the Atari Arcade Environment [1] which already achieved state-of-the-art performance for some of the games. DQNs introduce better scalability to larger data sets by replacing NFQ's RPROP with Stochastic Gradient Descent (SGD) updates. Further, they allow training end-to-end from the raw visual input utilizing convolutional neural networks. Benchmarks of more recent version [9] show an improved performance as well as a better generalization to more Atari games compared to initial work [8]. The performance improvement was mainly achieved by introducing a second target Q-network, which is only periodically updated and thereby reduces sample correlations. However, van Hasselt, et al. [3] show that the current DQN [9] approach is not sufficient to avoid overestimations of action-values under certain conditions. This in itself is not harmful to the policy's performance, but if the overestimation is not uniform and not concentrated at the states of interest, it might affect the policy negatively. To mitigate the risk of overestimation, they propose Double DQN (DDQN). Double Q-Learning in general tries to decouple the selection and evaluation of actions by learning two action-value functions. One determines the greedy policy and the other the value of this policy. In order to reduce the computational cost, van Hasselt, et al. utilize the already existing online Q-network for determining the greedy policy and the target Q-network for estimating its value. Further, this allows them to achieve better performance on most of the Atari games. Decoupling is also done in a more extreme form with Dueling Double DQNs (DDDQN) from Wang, et al. [18]. They introduce two separate function approximators, one for the state-value function and one for the state dependent action advantage function, in order to

represent the action-value[1]. These approximators allow the dueling architecture to learn about the value of a state regardless of the action taken. This is especially important as in many states, the action has no repercussions about the near future. Additional improvements, include gradient clipping and prioritized experience replay (PER) [12]. Combining PER with DDQN [12] showed that sampling rare experiences with a higher probability makes learning with replay memories more efficient [7]. However, sampling non-uniformly from the replay memory introduces a bias, that has to be corrected during the update step.
All previous approaches are using environments, which do not have to deal with delayed and sparse feedback. In real world applications this is often not the case and algorithms still need the ability to learn. One key problem, which arises during training, is insufficient exploration, thereby unstable policies. Using intrinsically motivated agents, exploration can be achieved for the agent itself rather for an external goal. This idea is implemented by Kulkarni, et al. [5] in hierarchical-DQN (h-DQN), which is based on a hierarchy of two DQNs. A top level *meta-controller* is selecting a subgoal in order to optimize the extrinsic reward. The lower level *controller* maximizes an intrinsic reward by solving the subgoal. This allows h-DQN to achieve a significant better performance on the Atari game Montezuma's Revenge.
As the above results show, classical Q-learning [19] is quite successful for discrete action spaces, real world problems however often require continuous action spaces. Lillicrap, et al. [6] adapt the idea of DQN [9] and make it applicable to the continuous action domain. They introduce Deep Deterministic Policy Gradients (DDPG), a model-free Actor-Critic (AC) algorithm with approximate Q-Learning based on deterministic policy gradients (DPG) [13]. DQN's [9] architecture is improved by adding batch normalization layers [4] to deal with different physical units in the observation space. In order to stabilize policies, they adapt the idea of target networks from DQN [9] and use "soft" target updates for AC. Besides continuous action spaces, it is usually hard to collect data from e.g. robots, and therefore dealing with high sample complexity in real world applications. Normalized advantage functions (NAF) [2] approaches this problem with a DQN based algorithm. Similar to DDDQN [18], the Q-network is represented by one state-value function output and one state dependent action advantage function. In order to reduce sample complexity, NAF incorporates *imagination rollouts*. Those synthetic on-policy samples are created by utilizing a mixture of iterative LQG (iLQG) [17] and on-policy trajectories for real world rollouts. Afterwards, the learned model for the states is used to generate replay memory entries. This can be seen as a scalable variant of Dyna-Q [14]. As a drawback, NAF failed to show improvements using iLQG compared to on-policy samples, however, iLQG might be desirable when avoiding damaging actions is critical.

POMDPs

[1] Both estimators share parameters in the networks's convolutional part and are trained together end-to-end.

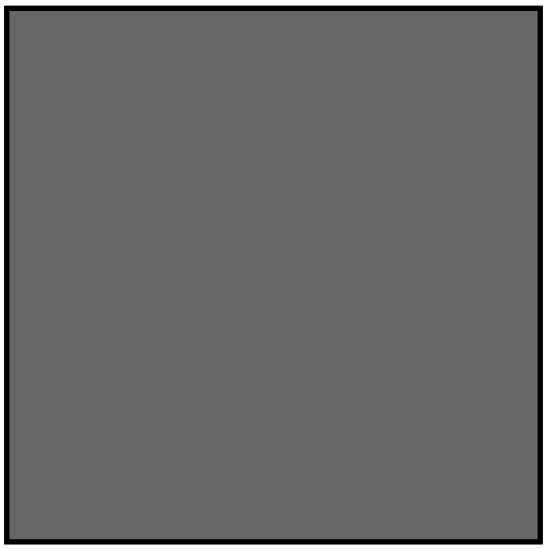

Fig. 1 Please write your figure caption here

Table 1 Please write your table caption here

first	second	third
number	number	number
number	number	number

4.2 On-Policy

4.3 Real World Applications

Talk about DeepMinds AlphaZero, TDGammon, Atari Game Systems, e.g. Minh But also applications outside of Games, maybe seperate this into two different subsections.

Paragraph headings

References

1. Bellemare, M.G., Naddaf, Y., Veness, J., Bowling, M.: The Arcade Learning Environment: An Evaluation Platform for General Agents. Journal of Artificial Intelligence Research **47**, 253–279 (2013). DOI 10.1613/jair.3912. URL http://arxiv.org/abs/1207.4708
2. Gu, S., Lillicrap, T., Sutskever, I., Levine, S.: Continuous Deep Q-Learning with Model-based Acceleration. In: ICML'16 Proceedings of the 33rd International Conference on International Conference onMachine Learning - Volume 48, pp. 2829–2838. JMLR.org, New York, NY, USA (2016). URL http://arxiv.org/abs/1603.00748
3. van Hasselt, H., Guez, A., Silver, D.: Deep Reinforcement Learning with Double Q-learning. In: AAAI Conference on Artificial Intelligence, pp. 2094–2100. Phoenix, Arizona (2016). URL http://arxiv.org/abs/1509.06461

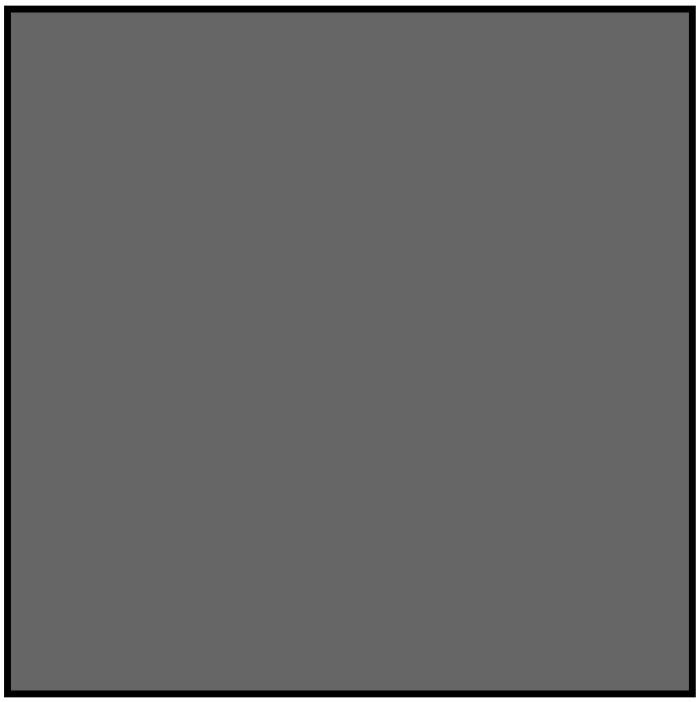

Fig. 2 Please write your figure caption here

4. Ioffe, S., Szegedy, C.: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In: International Conference on Machine Learning (ICML), ICML'15, pp. 448–456. JMLR.org (2015). URL http://dl.acm.org/citation.cfm?id=3045118.3045167
5. Kulkarni, T.D., Narasimhan, K., Saeedi, A., Tenenbaum, J.: Hierarchical Deep Reinforcement Learning: Integrating Temporal Abstraction and Intrinsic Motivation. In: D.D. Lee, M. Sugiyama, U.V. Luxburg, I. Guyon, R. Garnett (eds.) Advances in Neural Information Processing Systems 29, pp. 3675–3683. Curran Associates, Inc. (2016). URL http://papers.nips.cc/paper/6233-hierarchical-deep-reinforcement-learning-integrating-temporal-abstraction-and-intrinsic-motivation.pdf
6. Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D.: Continuous control with deep reinforcement learning. In: International Conference on Learning Representations (ICLR) 2016. London, UK (2016). URL http://arxiv.org/abs/1509.02971
7. Lin, L.J.: Self-improving reactive agents based on reinforcement learning, planning and teaching. Machine Learning **8**(3), 293–321 (1992). DOI 10.1007/BF00992699. URL https://doi.org/10.1007/BF00992699
8. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.: Playing Atari with Deep Reinforcement Learning. NIPS Deep Learning Workshop 2013 (2013). URL http://arxiv.org/abs/1312.5602
9. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., Hassabis, D.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529–533 (2015). DOI 10.1038/nature14236. URL http://www.nature.com/articles/nature14236
10. Riedmiller, M.: Neural Fitted Q Iteration - First Experiences with a Data Efficient Neural Reinforcement Learning Method. In: J. Gama, R. Camacho, P.B. Brazdil, A.M. Jorge, L. Torgo (eds.) Machine Learning: ECML 2005, pp. 317–328. Springer Berlin Heidelberg, Berlin, Heidelberg (2005)

11. Riedmiller, M., Braun, H.: A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm. In: IEEE International Conference on Neural Networks (IJCNN), pp. 586–591 (1993)
12. Schaul, T., Quan, J., Antonoglou, I., Silver, D.: Prioritized Experience Replay. In: Interantional Conference for Learning Representations (ICLR) (2015). URL http://arxiv.org/abs/1511.05952
13. Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., Riedmiller, M.: Deterministic Policy Gradient Algorithms. In: International Conference on Machine Learning (ICML), ICML'14, pp. I–387—-I–395. JMLR.org, Beijing, China (2014). URL http://dl.acm.org/citation.cfm?id=3044805.3044850
14. Sutton, R.S.: Integrated architectures for learning, planning, reacting based on approxmiate dynmaic programming. In: International Conference for Machine Learning (ICML)2, pp. 216–224 (1990)
15. Sutton, R.S., Barto, A.G.: Reinforcement learning: An introduction, second edi edn. MIT Press, Cambridge, MA (2018). URL http://incompleteideas.net/book/the-book.html
16. Tesauro, G.: TD-Gammon, a Self-Teaching Backgammon Program, Achieves Master-Level Play. Applications of Neural Networks **6**(2), 215–219 (1994). DOI 10.1162/neco.1994.6.2.215. URL https://doi.org/10.1162/neco.1994.6.2.215
17. Todorov, E., Li, W.: A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems. In: American Control Conference, pp. 300–306 vol. 1 (2005). DOI 10.1109/ACC.2005.1469949
18. Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M., De Freitas, N.: Dueling Network Architectures for Deep Reinforcement Learning. In: Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48, ICML'16, pp. 1995–2003. JMLR.org (2016). URL http://dl.acm.org/citation.cfm?id=3045390.3045601
19. Watkins, C.J.C.H.: Learning from delayed rewards. Ph.D. thesis, King's College, Cambridge (1989)