There is More to a Transition: Improving Sample Complexity of Hierarchical Reinforcement Learning Algorithms

Hari Shrawgi Sumanth Modukuri Dhaval Parmar

Abstract

Standard Reinforcement Learning (RL) algorithms do not work well with long-horizon sparse reward tasks. Hierarchical Reinforcement Learning (HRL) provides a way to solve such tasks. Many HRL algorithms proposed in the past have prohibitive sample complexity requirements. In this project, we propose a new technique which can be applied to HRL algorithms to improve their sample complexity. The key observation is that many HRL algorithms suffer from sparse transitions in high-level policy training data. To solve this we can utilize sub-transitions to make the training data denser at the higher temporal scale. The proposed technique can be incorporated into any HRL algorithm that satisfies certain standard requierements proposed by us. We empirically show improvements in the performance of Hierarchical Deep Q-Networks (HDQN) by incorporating the proposed technique.

1. Problem Description

Hierarchical Reinforcement Learning (HRL) holds the power to solve some of the most complicated and challenging tasks (Barto & Mahadevan, 2003; Dayan & Hinton, 1993; Parr & Russell, 1998a; Sutton et al., 1999). HRL specifically is suited for long-horizon tasks with sparse and delayed rewards. The notion of having a hierarchy of policies all working at different temporal scales is one which is quite natural as humans also plan-out and execute their tasks in a similar fashion (Buckner, 2010; Mullally & Maguire, 2014; Wei et al., 2015). Thus, the full potential of HRL is quite high, but it has not been realized as of yet.

This is due to the fact that there is a considerable gap between the current HRL algorithms and the promise of HRL (Nachum et al., 2018). There are multiple parts of the standard HRL algorithm that can be improved upon to get closer to an algorithm that holds up the promise of HRL on real-world tasks (Nachum et al., 2018). We particularly focus on one such aspect to get closer to the ideal HRL algorithm.

We believe that the biggest hurdle in realizing the promise of HRL in the real-world is the inordinate amount of experience required to train these multiple levels of policies (Fruit et al., 2017). Due to this, we cannot perform simple experiments. Further, this leads to a cascading effect that makes improving the other parts of the algorithm also difficult.

Problem: We want to solve the problem of making general HRL algorithms easier to train by reducing the sample complexity requirement. This will lead to better data efficiency and also time-saving in terms of deployment of the model.

2. Literature Review

HRL as a field has been gaining steam recently, with lots of different ideas and techniques being proposed to solve the myriad problems that are preventing us from making HRL work well (Vezhnevets et al., 2017; Bohn et al., 2019; Nachum et al., 2018; Levy et al., 2017; Kulkarni et al., 2016). But HRL is not a new field as it's foundations were laid in the 1990s (Precup & Sutton, 1998; Parr & Russell, 1998b;a; Sutton et al., 1999). In the following sub-sections however, we'll focus on the methods that are close and related to our proposed idea.

2.1. Sample Complexity

HRL algorithms usually rely on experience collected via simulations for their learning. The amount of sample experience that is required for succesful training of a method is called as its sample complexity. Sample complexity required for training affects the real-world applicability of the algorithm heavily. Multiple attempts have been made to curb the sample complexity requirement. We differentiate the attempts into two categories. The first category consists of methods that aim to increase efficiency while using on-policy training methods as in (Li et al., 2019a). The second category is made up of methods using and enhancing off-policy algorithms to increase efficiency (Nachum et al., 2018).

It has been shown recently that off-policy, model-free RL has far lower sample complexity than on-policy algorithms (Barth-Maron et al., 2018; Fujimoto et al., 2018; Haarnoja et al., 2018). Moreover, for the methods in which the high-level policy sets sub-goals for the low-level policy, on-policy methods are less efficient than off-policy methods (Nachum et al., 2018; Gu et al., 2016; Nachum et al., 2017). Furthermore, on-policy methods have also been plagued with issues of instabilty (Gu et al., 2016). But off-policy algorithms are not free of issues. They introduce the non-stationarity problem which arises due to changes in low-level policy which makes an experience with a certain high-level policy unfit for training. This is handled using various methods like off-policy corrections (Nachum et al., 2018) and concurrent sampling (Omidshafiei et al., 2017).

2.2. Multi-agent HRL

This section covers some of the literature from the sub-field of HRL which deals with the setting where we have multiple agents. This setting is much heavier in terms of sample complexity and training time requirements (Makar et al., 2001). Thus, there have been many ideas developed for this specific setting to improve sample complexity requirements. One such idea is to restrict the information available to each agent and solve local optimization problems (Guestrin et al., 2002; Peshkin et al., 2001; Schneider et al., 1999). Another idea is to reduce the required parameters for the multi-agent setting in Q-learning (Stone & Veloso, 1999). Most of these ideas, however, are not easily applicable to the single-agent setting.

But there is one recent idea which we feel can be made to work in a single-agent setting. The idea that we'll be focusing on, the sub-transition method, was proposed in (Tang et al., 2018). It deals with the sparsity of transitions at the higher temporal scales. The idea is to utilize the sub-transitions (transitions which happen at a lower temporal scale) for off-policy training of high-level policies. These sub-transitions are normally not included in the training. Using these makes the experience denser at the higher temporal scale. This leads to a reduction in the amount of experience collection required to train the model.

2.3. Sub-goal learning

Sub-goal learning is a specific setting under which sub-goals are set by a high-level policy which are then used by the low-level policy as an input (Li et al., 2019b; Levy et al., 2017; Kulkarni et al., 2016; Vezhnevets et al., 2017; Sutton et al., 1999). Such methods will be our focus for this project as the sub-transition method is quite easily applied to them.

For the purposes of this paper, we specifically target the idea of a high-level policy directing a low-level policy to reach specific goal states. This idea has been proposed way back in 1993 in (Dayan & Hinton, 1993) and again recently in (Kulkarni et al., 2016), (Vezhnevets et al., 2017) and (Nachum et al., 2018). We focus on the method proposed in (Kulkarni et al., 2016), where intrinsic rewards for the low-level policy are provided depending on it's success in achieving the sub-goals set by the high-level policy.

3. Proposed Idea

In this section we first describe the sub-transition method. Next we abstract out the setting in which the sub-transition method can be applied. This abstraction provides an easy criterion to verify whether an HRL algorithm can be used in combination with the method.

3.1. Sub-transition Method

In its most general form, sub-transition method can be stated as "While collecting experience for a higher temporal scale, also add some psuedo-transitions at a lower temporal scale." The basic idea is best explained pictorially as is presented in Figure 1.

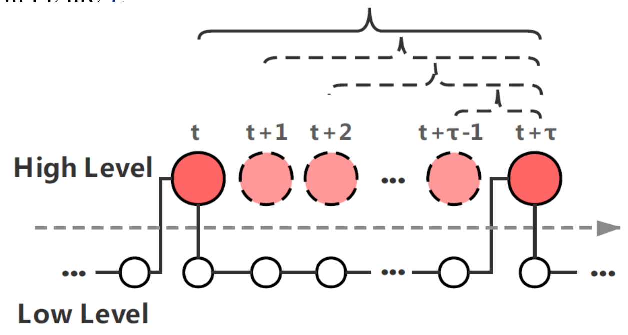

Figure 1. Representation of various sub-transitions that can be captured during one sparse high-level transition (Tang et al., 2018).

Consider the high-level transition at the temporal scale of τ: $\langle o_t, g_t, r_{t:t+\tau-1}, o_{t+\tau}\rangle$. Here o_t & $o_{t+\tau}$ are the observations at time t & $t+\tau$ respectively, g_t is the goal set by the policy at time t and $r_{t:t+\tau}$ is the extrinsic reward (from the environment) collected from time t to $t+\tau$. With the sub-transition method, this experience will be augmented to give: $\{\langle o_{t+k}, g_t, r_{t+k:t+\tau-1}, o_{t+\tau}\rangle\}_{k=0}^{\tau-1}$. This makes the experience denser under the same amount of experience collection. The sub-transition method is a way to improve the efficiency of experience collection.

Why should this work?

- **Theoretical reasoning:** The sub-transition method generates an augmented experience which is denser compared to the standard experience. As a result, the augmented experience will have more visits to different states. This means that more updates are performed on each of the states and thus will lead to more efficient training.
- **Empirical Reasoning:** In (Tang et al., 2018) we have a Hierarchical Independent Learner (h-IL) model which is basically a pair of independent HRL models. The sub-transition method is shown to improve the

performance of this h-IL model. Since the h-IL model consists of independent HRL models, we believe that this method should also work on a single HRL model.

3.2. Setting for the use of sub-transition method

Following are the core requirements [1] that any HRL algorithm must satisfy to make it suitable for the use of sub-transition method:

1. **Multiple temporal layers**: The algorithm must have multiple layers with a clear distinction between the temporal scales. This might seem like a trivial requirement as any HRL algorithm is based on this idea, but clear distinction between the temporal scales may not always be present.
2. **Experience Replay and Off-policy training**: Usage of experience replay complements the use of sub-transitions. In general, On-policy methods are never used with experience replay as we would be required to generate new actions again for every sample experience.
3. **Sub-goal learning**: This allows for easier handling of the experience replay buffer modifications required for additions of the extra tuples.

This small list of requirements is easily satisfied by a large number of currently used HRL algorithms (Levy et al., 2017; Nachum et al., 2018; Rafati & Noelle, 2019; Jendele et al., 2020). One of these algorithms, HDQN method, was the focus of our experiments. In the next section we present the results of applying the sub-transition method to HDQN.

4. Experimental Results

Setup: We consider a Markov decision process (MDP) with long-horizon sparse rewards. The MDP is illustrated in Figure 2.

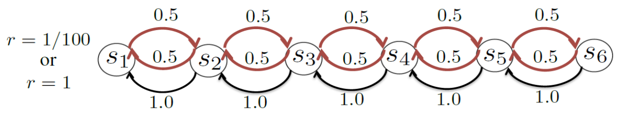

Figure 2. An MDP where the reward at the terminal state s_1 depends on whether s_6 is visited (r = 1) or not (r = 1/100). The agent always starts in state s_2. (Kulkarni et al., 2016).

Results: We compare the performance of vanilla HDQN on this setup with the modified HDQN algorithm with sub-transition method. All the ϵ parameters are annealed from 1 to 0.1 using exponential decay. We tried out 10 independent runs of both the models, where one run consists of 100,000 steps. The evolution of averaged reward over these 10 runs is plotted in Figure 3. One of the individual runs is shown in Figure 4 as this presents a clearer comparison between the two which is not visible in Figure 3.

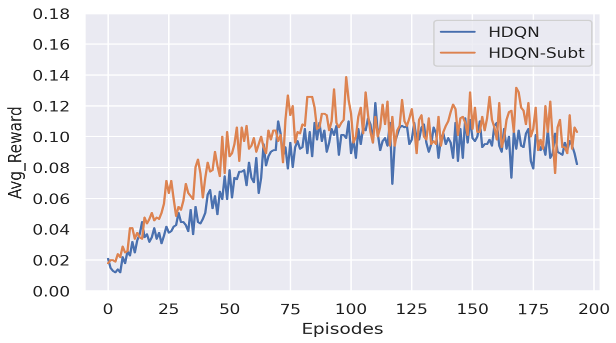

Figure 3. Averaged performance over 10 runs (coarsened).

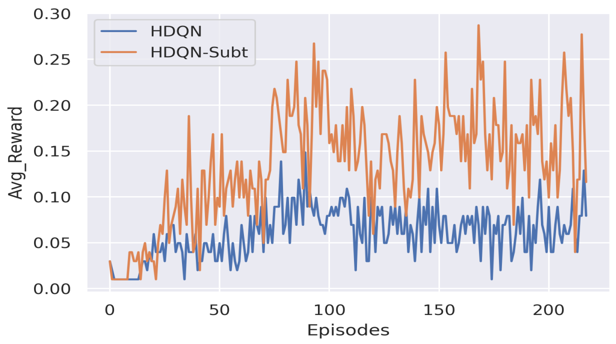

Figure 4. Performance on a sample run (coarsened).

As is clear by these results, the sub-transition method improves the performance of HDQN method:

- The peaks are reached earlier, i.e. over fewer episodes, when using the sub-transition method. This points to an increase in learning efficiency.
- The averaged rewards are higher at almost all time steps. This shows that the improvement persists throughout the training and is not just an initial boost to learning.

There is one area where we actually see a deterioration by using the sub-transition method which can be infered from Figure 4. The variance in the averaged reward between neighbouring time steps is higher when compared to the vanilla HDQN method.

Conclusion: Based on our results on HDQN, we would like to conclude that the sub-transition method can significantly improve the learning efficiency and average performance of HRL algorithms. Also, for any HRL alorithm that follows the requirements presented by us, it's incorporation should be straightforward.

Future work: We hope to solve the high variance issue observed in the results using selective removal of sub-transitions via a quality metric for each tuple (Isele & Cosgun, 2018).

[1] Note that the sub-transition method can be extended and used even if these requirements are not met. But when these are met it becomes plug-and-play.

References

Barth-Maron, G., Hoffman, M. W., Budden, D., Dabney, W., Horgan, D., Tb, D., Muldal, A., Heess, N., and Lillicrap, T. Distributed distributional deterministic policy gradients. *arXiv preprint arXiv:1804.08617*, 2018.

Barto, A. G. and Mahadevan, S. Recent advances in hierarchical reinforcement learning. *Discrete event dynamic systems*, 13(1-2):41–77, 2003.

Bohn, B., Rieger, C., and Griebel, M. A representer theorem for deep kernel learning. *Journal of Machine Learning Research*, 20(64):1–32, 2019.

Buckner, R. L. The role of the hippocampus in prediction and imagination. *Annual review of psychology*, 61:27–48, 2010.

Dayan, P. and Hinton, G. E. Feudal reinforcement learning. In *Advances in neural information processing systems*, pp. 271–278, 1993.

Fruit, R., Pirotta, M., Lazaric, A., and Brunskill, E. Regret minimization in mdps with options without prior knowledge. In *Advances in Neural Information Processing Systems*, pp. 3166–3176, 2017.

Fujimoto, S., Van Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477*, 2018.

Gu, S., Lillicrap, T., Ghahramani, Z., Turner, R. E., and Levine, S. Q-prop: Sample-efficient policy gradient with an off-policy critic. *arXiv preprint arXiv:1611.02247*, 2016.

Guestrin, C., Lagoudakis, M., and Parr, R. Coordinated reinforcement learning. In *ICML*, volume 2, pp. 227–234. Citeseer, 2002.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

Isele, D. and Cosgun, A. Selective experience replay for lifelong learning. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

Jendele, L., Christen, S., Aksan, E., and Hilliges, O. Learning functionally decomposed hierarchies for continuous control tasks. *arXiv preprint arXiv:2002.05954*, 2020.

Kulkarni, T. D., Narasimhan, K., Saeedi, A., and Tenenbaum, J. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Advances in neural information processing systems*, pp. 3675–3683, 2016.

Levy, A., Konidaris, G., Platt, R., and Saenko, K. Learning multi-level hierarchies with hindsight. *arXiv preprint arXiv:1712.00948*, 2017.

Li, A. C., Florensa, C., Clavera, I., and Abbeel, P. Subpolicy adaptation for hierarchical reinforcement learning. *arXiv preprint arXiv:1906.05862*, 2019a.

Li, S., Wang, R., Tang, M., and Zhang, C. Hierarchical reinforcement learning with advantage-based auxiliary rewards. In *Advances in Neural Information Processing Systems*, pp. 1407–1417, 2019b.

Makar, R., Mahadevan, S., and Ghavamzadeh, M. Hierarchical multi-agent reinforcement learning. In *Proceedings of the fifth international conference on Autonomous agents*, pp. 246–253, 2001.

Mullally, S. L. and Maguire, E. A. Memory, imagination, and predicting the future: a common brain mechanism? *The Neuroscientist*, 20(3):220–234, 2014.

Nachum, O., Norouzi, M., Xu, K., and Schuurmans, D. Trust-pcl: An off-policy trust region method for continuous control. *arXiv preprint arXiv:1707.01891*, 2017.

Nachum, O., Gu, S. S., Lee, H., and Levine, S. Data-efficient hierarchical reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 3303–3313, 2018.

Omidshafiei, S., Pazis, J., Amato, C., How, J. P., and Vian, J. Deep decentralized multi-task multi-agent reinforcement learning under partial observability. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2681–2690. JMLR. org, 2017.

Parr, R. and Russell, S. J. Reinforcement learning with hierarchies of machines. In *Advances in neural information processing systems*, pp. 1043–1049, 1998a.

Parr, R. E. and Russell, S. *Hierarchical control and learning for Markov decision processes*. University of California, Berkeley Berkeley, CA, 1998b.

Peshkin, L., Kim, K.-E., Meuleau, N., and Kaelbling, L. P. Learning to cooperate via policy search. *arXiv preprint cs/0105032*, 2001.

Precup, D. and Sutton, R. S. Multi-time models for temporally abstract planning. In *Advances in neural information processing systems*, pp. 1050–1056, 1998.

Rafati, J. and Noelle, D. C. Learning representations in model-free hierarchical reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 10009–10010, 2019.

Schneider, J., Wong, W.-K., Moore, A., and Riedmiller, M. Distributed value functions. 1999.

Stone, P. and Veloso, M. Team-partitioned, opaque-transition reinforcement learning. In *Proceedings of the third annual conference on Autonomous Agents*, pp. 206–212, 1999.

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

Tang, H., Hao, J., Lv, T., Chen, Y., Zhang, Z., Jia, H., Ren, C., Zheng, Y., Meng, Z., Fan, C., et al. Hierarchical deep multiagent reinforcement learning with temporal abstraction. *arXiv preprint arXiv:1809.09332*, 2018.

Vezhnevets, A. S., Osindero, S., Schaul, T., Heess, N., Jaderberg, M., Silver, D., and Kavukcuoglu, K. Feudal networks for hierarchical reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3540–3549. JMLR. org, 2017.

Wei, X.-X., Prentice, J., and Balasubramanian, V. A principle of economy predicts the functional architecture of grid cells. *Elife*, 4:e08362, 2015.