Dynamic Option Creation in Option-Critic Reinforcement Learning

Extended Abstract

Mateus B. Melchiades
Universidade do Vale do Rio dos Sinos
São Leopoldo, Brazil
mateusbme@edu.unisinos.br

Gabriel de O. Ramos
Universidade do Vale do Rio dos Sinos
São Leopoldo, Brazil
gdoramos@unisinos.br

Bruno C. da Silva
University of Massachusetts, Amherst
Amherst, United States
bsilva@cs.umass.edu

ABSTRACT

The options framework introduces the concept of temporal abstraction in MDPs by combining high level courses of action with primitive, single-step actions which can greatly improve planning and learning speeds. Throughout the past two decades, there has been active interest in autonomous option discovery, as well as determining what characterizes a good option. One example of such interest and advance is the Option-Critic Architecture. However, given that the ideal number of options for learning an optimal policy is not evident for most problems, Option-Critic's reliance on a fixed set of options proves as a limitation. In the present work, we propose an algorithm for creating options dynamically in training time, using the Fast-Planning Option-Critic implementation as a base. The Dynamic Option Creation algorithm (DOC) analyzes the variance in episodic returns when selecting each option to determine whether the learning process would benefit from a new option. Our method achieves similar per-episode returns as FPOC in the four-rooms environment, with the added benefit of discovering the ideal number of options automatically.

KEYWORDS

Reinforcement Learning; Options; Dynamic; Creation; Option-Critic

ACM Reference Format:
Mateus B. Melchiades, Gabriel de O. Ramos, and Bruno C. da Silva. 2025. Dynamic Option Creation in Option-Critic Reinforcement Learning: Extended Abstract. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 3 pages.

1 INTRODUCTION

The Options Framework [8] introduces the concept of temporally abstracted actions in Reinforcement Learning, which can reduce the time to learn an optimal policy for a given environment by abstracting desired courses of action into sub-policies, known as intra-option policies, each representing some higher level action. Despite its clear advantage, what characterizes a good option, as well as what should an option achieve, is still open for debate [1–7, 9]. We approach option discovery with the idea that options should complement each other, and that new options should be created if, and only if, the current set of options cannot cover the entire state space. If no option is capable of providing consistent returns for a region in the state space, we can assume such region is not covered and a new option is necessary for handling it. An inconsistent set of returns can be represented by the variance in accumulated rewards over multiple episodes. Given that accumulated reward tends to vary drastically as an agent first explores varied courses of action but stabilizes as training progresses, we can interpret a continuously high variance in returns as the agent struggling to learn parts of the environment.

We introduce Dynamic Option Creation (DOC): an algorithm capable of automatically scaling the number of options dynamically by observing the variance in accumulated rewards over time. If the variance in accumulated rewards fails to decrease as training progresses, then a new option is automatically created and initialized using experience replay from steps where previous options are unlikely to be selected. To the best of our knowledge, this is the first approach capable of creating options dynamically in Option-Critic.

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

2 DYNAMIC OPTION CREATION

The return of an option has variance, which can be influenced by the stochasticity of the environment, the stochastic nature of an option's policy, or the ongoing updates to the policy during the learning process. The first component affecting return variance—*exogenous variance*—cannot be controlled by the agent and is determined solely by the dynamics of the MDP itself. During the learning process, policies are typically initialized randomly which temporarily results in increased return variance as the policy is still being frequently updated. As the policy is updated towards convergence, the variance in returns will naturally decrease. For these reasons, we posit that variance in an option's return can serve as a proxy for epistemic uncertainty.

We propose a novel approach to assess the quality of an option by observing its variance over a specified time period. After n episodes, we interrupt training and run an evaluation process for m episodes where the agent acts greedily. We can define $\mathcal{V}_o \doteq \{\sigma^2_{R^o_p} | O_p = o,\ p \in \mathcal{P}\}$ as the variance in accumulated rewards for option o, where $\mathcal{P} = 0, 1, 2, \ldots, m$ is the set of all evaluation episodes and R^o_p is the accumulated reward for every step in p where the agent chose option o. By repeating this process enough times, we can build a set $\varsigma_o \doteq \{\mathcal{V}^1_o, \mathcal{V}^2_o, \mathcal{V}^3_o, \ldots\}$ containing the variances of multiple evaluation processes. We can then apply a definite integration over ς_o to observe the increase in variance over time. If we define the slope of some linear function as a threshold, we can use it to determine whether the variance is continuously higher than desired. We can determine the integral's slope for a

window of data by first applying a linear regression to the values inside it and then derivating the result relative to x, which gives us the slope in the regression. When applying this logic to a small enough window, the loss in precision is marginal. We can define the definite integral of ς_o by using the trapezoidal rule and then applying linear regression to its result, thus obtaining:

$$\text{lsreg}\left(\sum_{k=1}^{|\varsigma_o|} \frac{\varsigma_{o_{k-1}} + \varsigma_{o_k}}{2}\right) \approx w_1 x + w_0 \tag{1}$$

The uncertainty factor ε_o for the current option is then defined by the derivative obtained from the regression in Equation 1:

$$\varepsilon_o \doteq (\, e\,|O|\,)^{-\frac{2}{f}} \left[\frac{d}{dx}\ \ w_1 x + w_0\right], \tag{2}$$

where the discount factor $(\, e\,|O|\,)^{-\frac{2}{f}}$ prevents options from being created too frequently, being f the number of times the algorithm has calculated the uncertainty factor but did not create a new option.

The uncertainty factor is then compared against the threshold for deciding whether a new option is necessary. If ε_o is above the threshold, it means that the variance's integral is continuously high. We initialize a new option o^+ with pessimistic (negative) values for the option-value function q_{o^+} with the goal of discouraging exploration. Given that we want o^+ to focus solely on states where previous options fail, exploration can lead it to *copy* behavior already learned by other options. We also set the interest function for o^+ as $i_{o^+}(s) = -\sum_o^{O\setminus o^+} i_o(s)\ \forall s \in \mathcal{S}$, so that it has greater interest in states where existing options do not and vice-versa.

After conditioning o^+ to give higher emphasis towards state spaces where previous options perform poorly, we apply experience replay to the new option before resuming training. Each element in the replay buffer represents a full episode and contains its history $\langle S, r, S', \perp\rangle$ as well as the episodic return $G_{1:\tau}$. For each step recorded in the buffer, we use the same option selection technique used in training to check whether the algorithm would select the newly created option. If the option most likely to be selected is o^+, we apply the experience step, otherwise, that step is skipped, thus conditioning i_{o^+} towards state spaces less visited by previous options. In DOC, we retain the same option selection process used by FPOC as our main focus is in creating options.

3 EXPERIMENTAL EVALUATION

We compare DOC using different threshold values against FPOC with varied number of options in the four-rooms environment [8], where each experiment ran 30 times with the same set of random seeds. The threshold is the most important parameter when configuring DOC given that it indicates how much variance over time is acceptable before creating a new option. If set too low, the algorithm may create options unnecessarily, while setting it too high can hinder learning speed. Whenever a new option is created, the episodic returns tend to stagnate or decrease for some episodes as the new option gets introduced into the training process. In general, the sooner an option is introduced, the smaller the impact in learning performance. Table 1 shows how many episodes on average each threshold value took to reach a mean accumulated reward of -26 over 10,000 episodes, which is close to the upper bound reached by all algorithms in this experiment.

Table 1: Average number of episodes until reaching a mean return of -26 for FPOC and DOC with different thresholds.

Algorithm	*Avg. Eps. to Convergence*	*Std Error*
DOC (L=800)	**1,078,092.86**	**9,209.72**
DOC (L=200)	1,152,632.14	8,818.80
DOC (L=1200)	1,171,682.14	11,673.91
FPOC (4 Options)	1,185,590.00	9,176.59
FPOC (2 Options)	1,236,082.76	10,528.60
FPOC (1 Option)	1,288,789.29	7,668.41

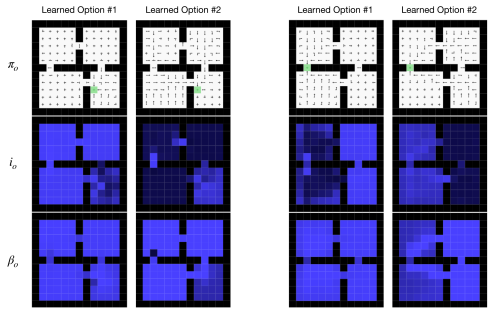

Figure 1: Policies, interest, and termination functions learned by DOC in two sample tasks.

Figure 1 shows the learned intra-option policies for each dynamic option alongside their interest and termination functions for two different goal positions. By observing the policies on the left, we can see that DOC created a second option that focuses on navigating towards the goal room, while the first option was responsible for reaching the goal from inside the room itself. This behavior can be considered meaningful as both policies represent some well-defined course of action. Meanwhile, when the goal is located at a door, we can see that each option covers a different half of the environment. We attribute this behavior to the fact that the goal is between two rooms, so we can consider both the top and bottom-left rooms as the goal room. From this perspective, the option on the left focuses on navigating towards the goal, while the option on the right goes towards the goal room, which is the same behavior observed when the goal is inside a single room.

4 CONCLUSION

This work proposed a novel method for dynamically creating options in training time in Option-Critic algorithms. When compared to FPOC, our approach learns options only as necessary with no increase in training steps needed for convergence. Despite the benefits mentioned above, our method has limitations outside the scope of the current work, such as the reliance on a human-provided uncertainty threshold and its tabular nature, which can limit its applicability in more complex environments.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their valuable feedback. This research was partially supported by Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq (grants 313845/2023-9, 443184/2023-2, and 445238/2024-0).

REFERENCES

[1] Tom Croonenborghs, Kurt Driessens, and Maurice Bruynooghe. 2008. *Learning Relational Options for Inductive Transfer in Relational Reinforcement Learning*. Lecture Notes in Computer Science, Vol. 4894. Springer Berlin Heidelberg, Berlin, Heidelberg, 88–97. https://doi.org/10.1007/978-3-540-78469-2_12

[2] Dongge Han and Sebastian Tschiatschek. 2022. Option Transfer and SMDP Abstraction with Successor Features. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, Vienna, Austria, 3036–3042. https://doi.org/10.24963/ijcai.2022/421

[3] George Konidaris and Andrew Barto. 2007. Building Portable Options: Skill Transfer in Reinforcement Learning. In *Ijcai*, Vol. 7. 895–900.

[4] Amy McGovern and Andrew G Barto. 2001. Automatic Discovery of Subgoals in Reinforcement Learning using Diverse Density. *Computer Science Department Faculty Publication Series. 8.* (2001).

[5] Ishai Menache, Shie Mannor, and Nahum Shimkin. 2002. Q-Cut—Dynamic Discovery of Sub-goals in Reinforcement Learning. In *Machine Learning: ECML 2002*, Gerhard Goos, Juris Hartmanis, Jan Van Leeuwen, Tapio Elomaa, Heikki Mannila, and Hannu Toivonen (Eds.), Vol. 2430. Springer Berlin Heidelberg, Berlin, Heidelberg, 295–306. https://doi.org/10.1007/3-540-36755-1_25

[6] Özgür Şimşek and Andrew Barto. 2008. Skill characterization based on betweenness. In *Advances in neural information processing systems*, Vol. 21. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2008/file/934815ad542a4a7c5e8a2dfa04fea9f5-Paper.pdf

[7] Martin Stolle and Doina Precup. 2002. Learning Options in Reinforcement Learning. In *Abstraction, Reformulation, and Approximation*, G. Goos, J. Hartmanis, J. Van Leeuwen, Sven Koenig, and Robert C. Holte (Eds.). Vol. 2371. Springer Berlin Heidelberg, Berlin, Heidelberg, 212–223. https://doi.org/10.1007/3-540-45622-8_16

[8] Richard S. Sutton, Doina Precup, and Satinder Singh. 1999. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence* 112, 1–2 (Aug 1999), 181–211. https://doi.org/10.1016/S0004-3702(99)00052-1

[9] Yi Wan and Richard S. Sutton. 2022. Toward Discovering Options that Achieve Faster Planning. *arXiv* arXiv:2205.12515 (Sep 2022). http://arxiv.org/abs/2205.12515 arXiv:2205.12515 [cs].