Strategy Extraction for Transfer in AI Agents

Doctoral Consortium

Archana Vadakattu
The University of Melbourne
Melbourne, Australia
vadakattua@unimelb.edu.au

ABSTRACT

We propose an approach to knowledge transfer for improved lifelong learning in AI agents, using behavioural strategies as a form of transferable knowledge, influenced by the human cognitive ability to develop strategies. A strategy is defined as a partial sequence of actions an agent can take to reach some predefined event of interest. This information acts as guidance or a partial solution that an agent can generalise and use to predict how to handle unknown observed phenomena. As a first step toward this goal, we present an approach for extracting strategies from an agent's existing knowledge that can be applied in multiple contexts. Our approach uses a combination of observed action frequency information with local sequence alignment techniques to find patterns of significance that form a strategy. We demonstrate our approach in two environments: Pacman; and a dungeon-crawling video game. Our evaluation serves as a promising first step towards efficient and robust generalisation to support lifelong learning across a wider class of tasks.

KEYWORDS

Strategy extraction; Autonomous agents; Game playing; Reinforcement learning

ACM Reference Format:
Archana Vadakattu. 2023. Strategy Extraction for Transfer in AI Agents: Doctoral Consortium. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 3 pages.

1 INTRODUCTION

The creation of artificial agents capable of operating autonomously in the real world is a long-standing research endeavour in AI. Achieving this level of autonomy requires flexibility for agents to perform in unfamiliar environments encountered over a lifetime, an ability that is lacking in existing systems. Observing human behaviour, cognitive scientists and psychologists have developed theories on how humans handle unfamiliar situations. For example, if performing a new task or perceiving an object we may not have seen before, we can make plausible default assumptions based on knowledge from similar situations. Lake et al. [8] make the following speculation about human players learning a new game – humans can grasp the basics of a game after just a few minutes of play because they are armed with extensive prior experience, which they intuitively leverage to give domain-specific knowledge. This allows a new player to infer information such as the general schema describing the goals of the game as well as the object types and their interactions.

In this work, we propose a method of knowledge transfer in AI agents based on the human cognitive ability to develop strategies in one context, that can be generalised and applied in other contexts. Existing work on transfer learning involves learning low-level information from a source task [3, 7, 15], however these approaches face limitations such as being restricted to certain classes of tasks due to learnt knowledge being insufficiently generalisable [12]. Suppose artificial agents can perform strategy synthesis, allowing them to better utilise their existing knowledge. This would significantly improve the agent's learning capabilities to handle a wide range of tasks with limited data. As a first step, we seek to obtain knowledge in the form of strategies from an agent's existing knowledge. We define a strategy as an abstract task structure that represents a partial plan to achieve a goal in some environment. For example, in the game of Pacman, a strategy may be to "collect a power-up to defeat a ghost". This could be abstracted to "collect an item to defeat an enemy". Defeating enemies is a common objective that appears in various games, making this strategy applicable in many contexts.

There are two key advantages to using strategies to support lifelong learning [5]. First, due to its partial nature, a strategy alone may not completely solve a problem, however it provides a base for a complete solution to multiple related problems. Our definition encompasses plans with a partial ordering of actions as well as plans with potentially unnecessary or missing actions. Second, the strategies we extract in this paper contain actions specific to a given game. We can broaden the applicability of these strategies through abstraction – for example, with the use of ontologies. These strategies provide a starting point for identifying a suitable course of action when an agent does not know how to achieve some goal.

2 CONTRIBUTION

For this research, we focus on behavioural strategies that describe an AI agent's behaviour in some context. We aim to examine the extent to which the transfer of behavioural strategies to new contexts improves the knowledge learnt by the agent, the learning time and real-time agent performance.

A key contribution of this work is our unique approach to strategy extraction by treating the problem as a sequential pattern mining task. Most existing work focuses on finding strategies for player modelling for applications such as learning an opponent's strategy, understanding a player's behaviour and finding a winning strategy [6, 11]. The types of strategies typically modelled are complete solutions, such as telling the player exactly how to reach the goal. Even if the game- or implementation-specific content is abstracted, complete strategies are highly unlikely to be applicable in other

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom.

games. Inspired by similar works such as [4] and [9], the strategies found by our method are a result of using sequence analysis methods to locate similar regions in sequences of action trajectories. The novelty lies in using a sequence alignment technique, the Smith-Waterman algorithm [13], which is more commonly known for comparing DNA string sequences. Strategy generalisation and transfer are left as future work.

3 METHOD

We consider an agent that has been trained to play a game through reinforcement learning (RL) [14]. A single-player game environment consists of states, actions and rewards. A game trajectory is a finite sequence of consecutive actions, rewards and states of the environment before and after an action is taken by the player. A *subtrajectory* is a trajectory containing a subset of the elements from another trajectory, maintaining the same temporal ordering. A subtrajectory may not be equivalent to a subsequence; elements from the original trajectory can be dropped. For example, given a trajectory of the form a, b, c, a valid subtrajectory is a, c as well as a, b and b, c.

We adapt Smith-Waterman to perform pairwise comparisons on trajectories and return a subtrajectory. Smith-Waterman aims to find a similar *subsequence* when comparing two sequences, explicitly placing gap tokens between consecutive items in the result if they do not occur consecutively in the two sequences being compared. We are interested in identifying the important similarities (actions) between two trajectories, and their relative temporal ordering. We do not care whether gaps should be placed in the resulting subtrajectory, or how many, in the context of our strategy extraction objective. It is assumed that gaps may exist between any consecutive actions in a subtrajectory.

Given a policy for playing a game, our approach first discovers *events of interest.* These events occur when playing and represent goals or sub-goals that an agent could use a strategy to achieve. We use a method of detection based on rewards, however this can be replaced with more sophisticated approaches that do not rely solely on the reward function [1, 10]. For each event of interest found, we then find collections of trajectories by simulating the agent when following the learnt policy. We collect trajectories in which the event occurs (positive) and does not occur (negative). The positive and negative trajectories are used to compute values for each available action to indicate their likelihood of being in a strategy. Actions that appear more often in positive trajectories are more likely to be part of a strategy.

Positive trajectories are clustered based on their actions to ensure that we can also identify different strategies for achieving the same event of interest. Algorithm 1 outlines our method for finding strategies for a given event of interest. The shortest trajectory (i.e., with the least number of actions) is selected for each cluster. A pairwise comparison between this trajectory, and all others in the cluster, is performed using the Smith-Waterman algorithm. The output of each pairwise comparison becomes a candidate strategy.

4 EXPERIMENTS

We evaluate the performance of our proposed method on two custom environments implemented in OpenAI Gym [2]. We developed a version of Pacman as a 2D environment and an exploration game, “Dungeon Crawler” where the agent’s goal is to navigate through a maze to find a key and then escape through a door. The agent must avoid monsters in its search for the key or kill monsters by collecting a weapon (gun or sword).

Algorithm 1 Find Strategies for an Event of Interest

```
Input: $C$ (clusters of trajectories), $\mathcal{L}$ (action likelihoods)
Output: Strategy set, $\mathcal{S}$
Let $\mathcal{S} = \emptyset$.
for $c \in C$ do
    $t_c = ShortestTrajectory(c)$
    for $t_j \in c \setminus \{t_c\}$ do
        $result \leftarrow SmithWaterman(t_c, t_j, \mathcal{L})$
        $\mathcal{S} \leftarrow \mathcal{S} \cup \{result\}$
    end for
end for
return $\mathcal{S}$
```

To test the robustness of the extraction approach, we executed 50 runs for each environment under the same conditions. We saved the resulting strategies and total counts of how many times each strategy was found across all the runs. The predominant strategies, those with the highest ‘Found’ percentage, are what we expected for each event of interest.

With a dataset size of 100 trajectories in each of the positive and negative trajectory sets, we obtain the following strategy in the form of an action sequence for Pacman when the specified event of interest is “kill a ghost":

“collect power-up", “kill a ghost"

In Dungeon Crawler, we specified the event as “kill a monster" and observe the following strategies which are predominant:

“collect gun", “kill a monster"

“collect sword", “kill a monster"

The predominant strategy in Pacman appears in all 100% of experiment runs, and both of the Dungeon Crawler strategies are found 98% of the time. Our method also detects variations on these strategies that are reasonable candidates based on the environment.

5 CONCLUSION AND FUTURE WORK

We have proposed an approach for extracting strategies from learned agent policies. Our results, when demonstrated on video game trajectories, showcase the ability of this method to identify reasonable strategy candidates in different contexts. We are able to use sequence analysis to find useful causal information, which we can use to form strategies. Preliminary results showcase the ability of this method to identify reasonable strategy candidates in different contexts. In future work, we will utilise the strategies obtained from this method and look at generalisation techniques to support lifelong learning across a wider class of tasks. In particular, we will consider generalisation via abstraction, changing only the content of a strategy when lifting it to a more general context, leaving us with flexibility in the choice of data structures used. Ultimately, we envision this work could address issues around generalisability present in current state-of-the-art autonomous artificial agents and make them deployable in real-world scenarios.

REFERENCES

[1] Mehran Asadi and Manfred Huber. 2007. Effective Control Knowledge Transfer through Learning Skill and Representation Hierarchies. In *IJCAI'07: Proceedings of the 20th International Joint Conference on Artifical intelligence*. AAAI, India, 2054–2059.

[2] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. OpenAI Gym. arXiv:1606.01540

[3] Tim Brys, Anna Harutyunyan, Matthew E Taylor, and Ann Nowé. 2015. Policy Transfer using Reward Shaping. In *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, Turkey, 181–188.

[4] Zhengxing Chen, Magy Seif El Nasr, Alessandro Canossa, Jeremy Badler, Stefanie Tignor, and Randy Colvin. 2015. Modeling Individual Differences through Frequent Pattern Mining on Role-Playing Game Actions. In *Proceedings of the 11th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*. AAAI, Santa Cruz, CA, 6.

[5] Zhiyuan Chen and Bing Liu. 2018. Lifelong machine learning, second edition. *Synthesis lectures on artificial intelligence and machine learning* 12, 3 (Aug. 2018), 1–207.

[6] Niklas Een, Alexander Legg, Nina Narodytska, and Leonid Ryzhyk. 2015. SAT-Based Strategy Extraction in Reachability Games. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, Vol. 29. aaai.org, Austin, TX, 3738–3745.

[7] George Konidaris and Andrew Barto. 2007. Building portable options: Skill transfer in reinforcement learning. *IJCAI* 7 (2007), 895–900.

[8] Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. 2017. Building machines that learn and think like people. *Behavioral and Brain Sciences* 40 (2017), e253.

[9] Cécile Low-Kam, Chedy Raïssi, Mehdi Kaytoue, and Jian Pei. 2013. Mining Statistically Significant Sequential Patterns. In *2013 IEEE 13th International Conference on Data Mining*. IEEE, Dallas, TX, 488–497.

[10] Sujoy Paul, Jeroen van Baar, and Amit K. Roy-Chowdhury. 2019. Learning from trajectories via subgoal discovery. *Advances in Neural Information Processing Systems* 32 (2019), 8409–8419.

[11] Pier Giuseppe Sessa, Ilija Bogunovic, Maryam Kamgarpour, and Andreas Krause. 2020. Learning to Play Sequential Games versus Unknown Opponents. *Advances in neural information processing systems* 33 (2020), 8971–8981.

[12] Murray Shanahan and Melanie Mitchell. 2022. Abstraction for Deep Reinforcement Learning. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, Lud De Raedt (Ed.). International Joint Conferences on Artificial Intelligence Organization, Vienna, 5588–5596.

[13] Temple F Smith and Michael S Waterman. 1981. Identification of Common Molecular Subsequences. *Journal of molecular biology* 147, 1 (1981), 195–197.

[14] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement Learning: An Introduction* (2nd ed.). MIT Press, Cambridge, Massachusetts.

[15] Lisa Torrey, Trevor Walker, Jude Shavlik, and Richard Maclin. 2005. Using Advice to Transfer Knowledge Acquired in One Reinforcement Learning Task to Another. In *Machine Learning: ECML 2005*. Springer Berlin Heidelberg, Portugal, 412–424.