Model Discovery for Efficient Search*

Jan Hůla[1 2] and Mikoláš Janota[1]

[1] Czech Technical University in Prague, Prague, Czech Republic
[2] University of Ostrava, Ostrava, Czech Republic

Motivation and overview. In this contribution, we focus on exploration strategies inspired by human cognition. In comparison to AI search and learning, humans show greater efficiency in terms of the number of states explored. This efficiency is mostly due to the fact that humans have aversion towards prolonged periods of random exploration and most of the time are following goal-oriented behavior. The goals often form a hierarchy and are proposed by the person themselves. Moreover, the person is constantly tracking the expected difficulty of achieving the given goal and, based on the ratio between this difficulty and the value of this goal, they can decide to abandon the goal and instead pursue another.

Goals are proposed and their difficulty is tracked using knowledge about a given domain. In completely novel domains, in which it seems that random exploration will not lead to success, the person typically starts by trying to create a mental model of the domain and then uses this mental model to explore more effectively. For example, if they have already learned that keys unlock doors and then find themselves in a locked room, they can temporarily set a goal to find a key.

Inspired by these observations, we design an agent which explores the state space by proposing various sub-goals to itself. First, these sub-goals are curiosity-driven and their aim is to incrementally learn a mental model of the domain. These sub-goals are later interleaved with the original goals and their sub-goals. Each goal and sub-goal is being searched for with a planner that leverages the mental model discovered thus far.

Related Work. Our work is related to a current trend of learning *world models* in model-based reinforcement learning [6, 5, 12]. Such world models are represented as a neural network which predicts state dynamics and rewards conditioned on the actions of the agent. The learned world model can later be used by the agent to decide how to act in an online fashion or even to learn the entire policy in an offline fashion. Our approach differs by the fact that in our case the model is represented as a set of concepts and declarative statements which can be used by the planner and learned incrementally. This different kind of learning [10, 1] overcomes many issues inherent to learned statistical models such as catastrophic forgetting [9], failure to generalize due to spurious correlations [8], inability to recompose old knowledge in novel ways [13], etc. Another trend related to our work is the *learnig to propose sub-goals* research direction where the goal is to train a neural network to propose sub-goals that are useful during the search for the original goal. [14, 3]. In our case, the sub-goal proposal is manually engineered. Our approach is also related to symbolic regression and ontology learning, where the goal is to discover a symbolic model that explains or summarizes the observed data [4, 2].

Problem Domain. We study our approach in the domain of logical puzzles and simple games. We chose this domain to avoid the complexity of real-world problems that are hard to model precisely. Our aim is to learn and understand basic principles of efficient exploration in

*The results were supported by the Ministry of Education, Youth and Sports within the program ERC CZ under the project POSTMAN no. LL1902.

the simplified domain with the hope that later we would be able to include these principles in a more complex approach scaling to realistic domains.

Concretely, we study exploration strategies in games written in the video game description language (VGDL) [11]. VGDL enables one to define games in a few lines of declarative statements. These declarative statements are divided into three categories: The first category corresponds to the definitions of entity types and their default behavior, the second category describes the interaction events between each pair of entity types, and the third category describes termination conditions. An example of a simple game defined in VGDL is visible in Figure 1. To instantiate a concrete game, a rectangular grid of symbols representing different entities is used. The VGDL engine parses this description into an instantly playable game.

We propose an agent that learns the mechanistic model of the game to explore more effectively. At each time step, our agent receives the current state of the board as input and outputs one of the possible actions. The agent has the VGDL engine together with a planner in his “head” but does not have access to the definition of the current game. Therefore, the initial exploration aims to recover this definition by discovering the types of entities present in the game and the effects of interaction between different pairs of entities. The recovered definition is constantly updated and used by the VGDL engine to produce a “mental simulation” of a possible evolution of the game. The planner is used to propose sub-goals which are either aimed to observe the result of an interaction between a pair of entity types or to get the agent closer to the original goal. After finding a “mental plan” for the currently schedule goal, the agent executes the plan in the real game. The discrepancy between the imagined dynamics and the real dynamics is used to update the mental model of the agent.

As mentioned at the beginning of this section, we consider this simplistic domain as an exploration playground for prototyping efficient search algorithms. We believe that the ability to discover abstractions (in our case entities) and relations between them (in our case interaction events) is a generally useful and interesting area of research, which is also related to theorem proving where one can see an analogy in concept and lemma discovery [7]. We hope that this work will inspire further research on model discovery for the purpose of efficient search and reasoning.

```
SpriteSet
    hole   > Immovable color=DARKBLUE
    avatar > MovingAvatar #cooldown=4
    box    > Passive
InteractionSet
    avatar wall > stepBack
    box avatar  > bounceForward
    box wall    > undoAll
    box box     > undoAll
    box hole    > killSprite
TerminationSet
    SpriteCounter stype=box    limit=0 win=True
```

Figure 1: An example of a VGDL definition of a game (Sokoban). The concrete game also requires a grid of symbols which defines the initial conditions of the game. SpriteSet contains definitions of entity types and their default behavior, InteractionSet describes what happens when two entities end up in a same position and TerminationSet describes termination conditions.

References

[1] Susan Carey. The origin of concepts. *Journal of Cognition and Development*, 1(1):37–41, 2000.

[2] Xinlei Chen, Abhinav Shrivastava, and Abhinav Kumar Gupta. Neil: Extracting visual knowledge from web data. *2013 IEEE International Conference on Computer Vision*, pages 1409–1416, 2013.

[3] Konrad Czechowski, Tomasz Odrzygóźdź, Marek Zbysiński, Michał Zawalski, Krzysztof Olejnik, Yuhuai Wu, Łukasz Kuciński, and Piotr Miłoś. Subgoal search for complex reasoning tasks. *Advances in Neural Information Processing Systems*, 34:624–638, 2021.

[4] Kevin Ellis, Catherine Wong, Maxwell Nye, Mathias Sable-Meyer, Luc Cary, Lucas Morales, Luke Hewitt, Armando Solar-Lezama, and Joshua B Tenenbaum. Dreamcoder: Growing generalizable, interpretable knowledge with wake-sleep bayesian program learning. *arXiv preprint arXiv:2006.08381*, 2020.

[5] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pages 2555–2565. PMLR, 2019.

[6] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

[7] Jónathan Heras, Ekaterina Komendantskaya, Moa Johansson, and Ewen Maclean. Proof-pattern recognition and lemma discovery in acl2. In *International Conference on Logic for Programming Artificial Intelligence and Reasoning*, pages 389–406. Springer, 2013.

[8] Robert Kirk, Amy Zhang, Edward Grefenstette, and Tim Rocktäschel. A survey of generalisation in deep reinforcement learning. *arXiv preprint arXiv:2111.09794*, 2021.

[9] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[10] Dharshan Kumaran, Demis Hassabis, and James L McClelland. What learning systems do intelligent agents need? complementary learning systems theory updated. *Trends in cognitive sciences*, 20(7):512–534, 2016.

[11] Tom Schaul. A video game description language for model-based or interactive learning. In *2013 IEEE Conference on Computational Inteligence in Games (CIG)*, pages 1–8. IEEE, 2013.

[12] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

[13] Paul Smolensky, R Thomas McCoy, Roland Fernandez, Matthew Goldrick, and Jianfeng Gao. Neurocompositional computing in human and machine intelligence: A tutorial. 2022.

[14] Josef Urban and Jan Jakubův. First neural conjecturing datasets and experiments. In *International Conference on Intelligent Computer Mathematics*, pages 315–323. Springer, 2020.