Learning Discrete State Abstractions With Deep Variational Inference

Ondrej Biza, Robert Platt, Jan-Willem van de Meent and Lawson L.S. Wong
Northeastern University

Discrete State Abstractions and Bisimulations

We are interested in working with states represented as **images** and encoding them as **symbols**. As an example, you can think of an agent controlling a robotic arm. It should learn symbols that represent different configurations of objects in the workspace, with focus on features relevant to the task it is solving.

Bisimulations are a particular type of state abstraction that preserves the underlying dynamics of the environment [1]. For instance, in a grid world where the agent receives a reward for visiting the right-most column, the row it is located in is irrelevant (Figure 1, left). Hence, we can find a bisimulation that reduces the number of states (Figure 1, right).

Learning Bisimulations

Our method learns bisimulations by predicting the state-action values and the transition dynamics of the environment. First, we encode an image into a **continuous latent vector *z***, from which we then predict a state-action value *y* (Figure 2). The continuous embedding *z* is then compressed into a **discrete state $\bar{s}$**.

We train this two-layer hierarchy of continuous and discrete encodings using the deep variational bottleneck method [2].

$$\mathbb{E}_{q(s,s',z,z',a,y)}\left[I(y;z) - \beta I((s,s');(z,z')|a)\right]$$
$$\geq \mathbb{E}_{q(z,a)}\left[\log p(y|z,a)\right] - \beta D_{KL}(q(z,z'|s,s',a) \parallel p(z,z'|a))$$

The first term encourages the model to predict state-action values *y*, and the second term is a loss for the transition model. Here, we encode a transition in the environment (s, a, s') as a transition in the continuous latent space (z, a, z') using the encoder q(z | s). Then, we model this latent transition with a Hidden Markov Model p(z, z' | a). We treat the components of the HMM as the discrete states $\bar{s}$.

The HMM gives us discrete states and a transition model. If we specify the reward function over the discrete states, we get a discrete Markov Decision Process that simulates the larger ground MDP with image states.

Northeastern University
Khoury College of Computer Sciences

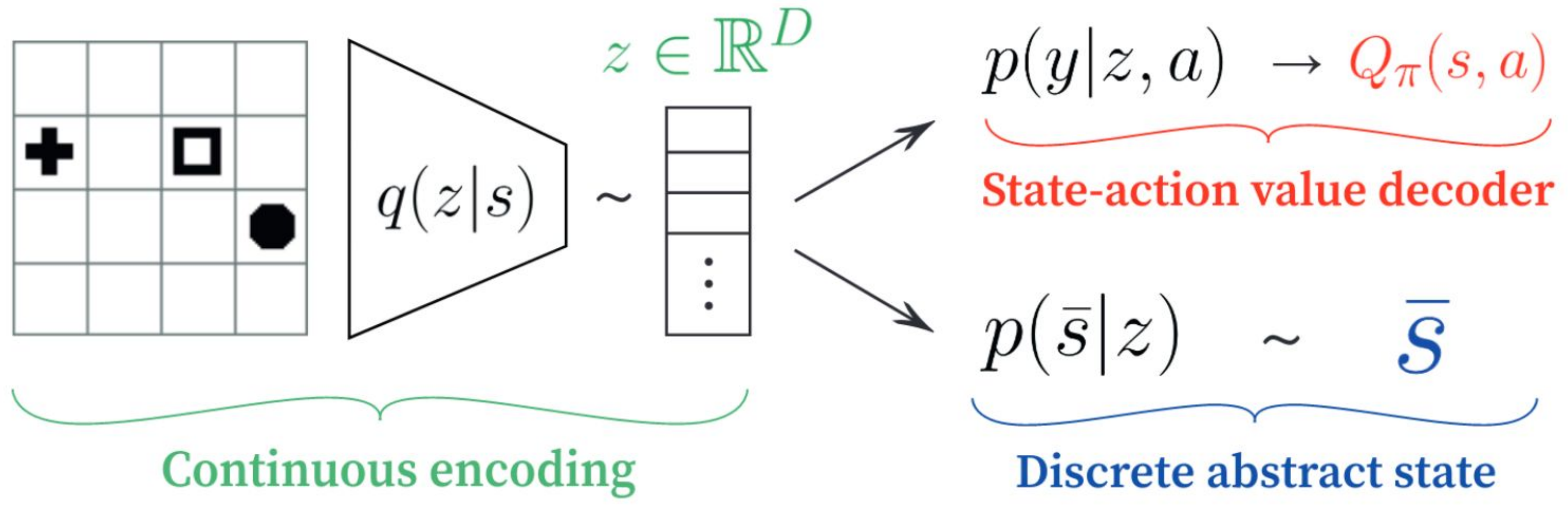
$z \in \mathbb{R}^D$
$q(z|s)$
~
$p(y|z,a)$ → $Q_\pi(s,a)$
State-action value decoder
$p(\bar{s}|z)$ ~ $\bar{s}$
Continuous encoding
Discrete abstract state

Figure 2: Inference in our model. We first take a state, represented as an image here, and encode it as a continuous vector z (green). Then, we predict state-action values from z for each action a (red) and further encode z into a discrete abstract state $\bar{s}$ using our prior (blue).

Planning in a Simple Manipulation Domain

We test our method in a four by four gridworld with objects of various shapes placed in the cells; an agent can pick and place them. We instantiate eight different tasks, such as stacking objects on top of each other or placing them in a row (Figure 5).

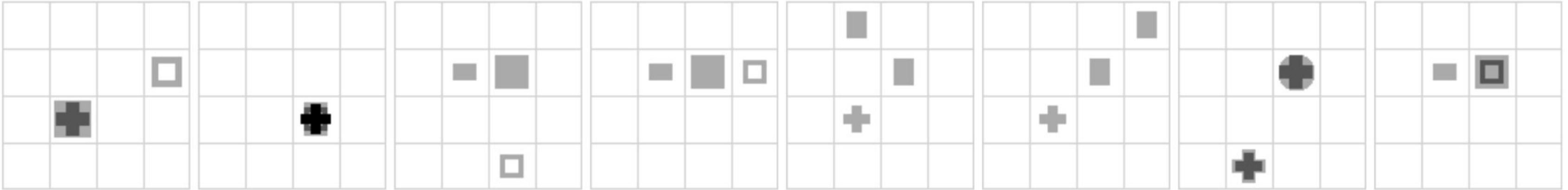

Figure 5: Goal states for tasks in Shapes World. There are four types of objects–pucks, boxes, squares and pluses–placed in a grid world. From left to right we have examples of goal states for two objects stacking, three objects stacking, two objects in a row, three objects in a row, two objects diagonal, three objects diagonal, two and two objects stacking, stairs from three objects.

Our model is trained on one or two source tasks, and then we test its ability to plan for unseen tasks. To do that, we use the learned discrete MDP learned by the Hidden Markov Model, and specify a reward function over discrete states with a reward of one for the goal state of the new task.

The success rates for planning for different tasks are reported below. Our model with 1k abstract states can solve tasks with two objects perfectly (e.g. 2S means stack of two objects). It reaches around 80 - 90% success rate for tasks with three objects and 30 - 40% on 2&2S, which involves making two stacks of two objects.

The main limitation of bisimulation is that it enumerates each possible configuration of objects; hence, it scales poorly with the complexity of the task.

Source tasks	2S	3S	2R	3R	2&2S	3ST	2D	3D
2S	**99.9** ±0.1	-	98.6 ±0.5	-	-	-	**98** ±0.7	-
2S, 2R	99.2 ±0.9	-	**99.9** ±0.1	-	-	-	81.5 ±3.1	-
3S	90 ±2.6	**98.2** ±0.8	75.7 ±2.7	40.8 ±14.7	-	61.9 ±7.5	67 ±3.6	24.9 ±7.7
3ST	74.4 ±3.5	21.8 ±6.4	**98.8** ±0.2	73.9 ±4.4	-	**98.8** ±0.4	83.6 ±2.7	39.3 ±4.2
3S, 3R	93.8 ±2.2	88.8 ±3.3	91.5 ±1.5	**88.4** ±2.7	-	74.8 ±6.1	**86** ±3.4	**65.2** ±6.6
3S, 3ST	**98.1** ±1.2	97.8 ±1.2	98.1 ±1.7	75.2 ±4.2	-	92.2 ±1.8	84.1 ±4.3	51.3 ±6.9
2&2S	76.9 ±8.2	16.2 ±2.5	65.8 ±3.6	24.9 ±4	**46.2** ±7.1	4.7 ±2.4	46.6 ±5.5	12.2 ±2.8
2&2S, 3S	**92.8** ±2.7	**33.9** ±3.7	67.6 ±4.4	30.8 ±3.2	38 ±1.7	9.6 ±2.6	51.5 ±5.1	**16.8** ±3.4
2&2S, 3R	61.8 ±5.4	18.4 ±3.1	71.6 ±3.1	**70.7** ±4.8	33.9 ±7.6	10.3 ±4.2	50.4 ±3	10.1 ±1.1
2&2S, 3ST	71.5 ±7.6	24.4 ±3.6	**75.6** ±4.4	29.6 ±2.1	36.1 ±4.4	**33.7** ±4.5	**53.4** ±4.9	15 ±2.4

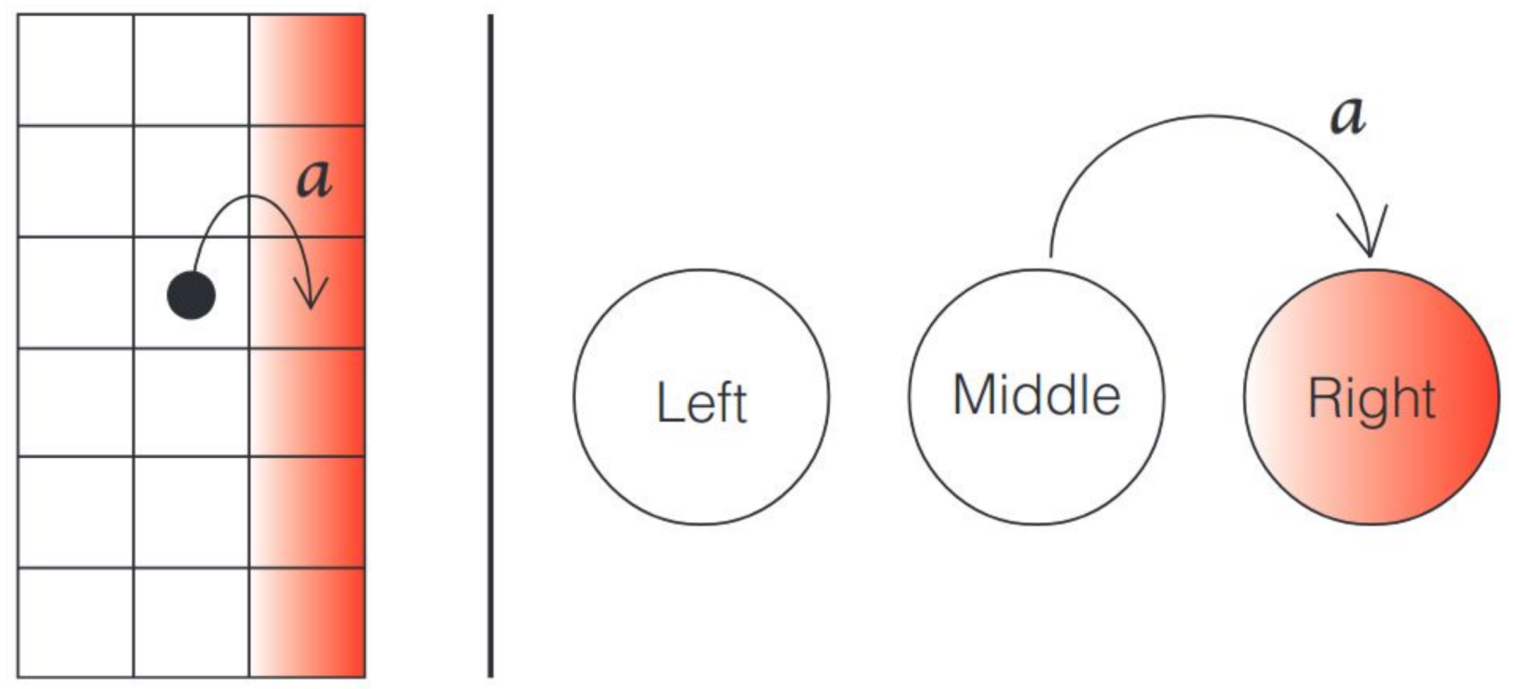

Figure 1: Example of bisimulation abstraction. The Column World (left) has 3 columns and 30 rows (we only show 6 rows); the agent travels between adjacent cells (Lehnert and Littman 2018). Since the agent gets a reward 1 for being in the right column (red) and 0 otherwise, it is irrelevant in which row it is located. Hence, the environment can be simulated by an MDP with three states (right).

Playing Mini Atari Games

We use learned discrete state abstractions to represent policies of simple versions of Atari games [3]. DQN is a model-free baseline, Mean Q keeps state-action values for each of the 1000 found discrete states, and VI uses Value Iteration to plan in the discretized MDP. VI only works in Freeway, whereas Mean Q performs well in three out of four games.

Game	DQN	Mean Q	VI	Random
Breakout	14	19.08 ±11	0.83 ±0.39	0.66
Space Invaders	55	20.52 ±2.95	5.81 ±3.12	3.06
Freeway	54	36.21 ±8.08	34.95 ±8.46	0.2
Asterix	20	0.53 ±0.1	0.49 ±0.08	0.5

References

[1] Givan, R.; Dean, T.; and Greig, M. 2003. Equivalence notions and model minimization in Markov decision processes.Artificial Intelligence 147(1): 163 – 223. ISSN 0004-3702. Planning with Uncertainty and Incomplete Information.

[2] Alemi, A. A.; Fischer, I.; Dillon, J. V.; and Murphy, K. 2017.Deep Variational Information Bottleneck. In 5th International Conference on Learning Representations, ICLR 2017,Toulon, France, April 24-26, 2017, Conference Track Proceedings.

[3] Young, K.; and Tian, T. 2019. MinAtar: An Atari-inspired Testbed for More Efficient Reinforcement Learning Experiments. CoRR abs/1903.03176.

https://arxiv.org/abs/2003.04300