Learning Neuro-Symbolic Skills for Bilevel Planning

Tom Silver, Ashay Athalye Joshua B. Tenenbaum,
Tomas Lozano-Perez, Leslie Pack Kaelbling

Presenter: Noah T. Patton

October 25th 2022

Why Other Approaches Do Not Work

- Use premade policies [3]
- Do not use symbolic representations [4]
- Use premade skills and learn their policies [5]

Main Problem

Can a policy be effectively *learned* for neuro-symbolic skills in Bilevel Planning?

Motivation

- Bilevel planning allows reasoning about ‘what to do’ and ‘how to do it’ [2]
- Adding symbolic representation allows for use of efficient AI planning algorithms
- Learning the policy removes need for human to engineer policies for each skill in each task
- Learning neuro-symbolic skills can allow for efficient high-level planning
- Sequences of neuro-symbolic skills help with explainability of the agent’s decision process

Technical Challenge

- ❖ Abstracting states to symbolic representations is inherently lossy
- ❖ 2 Key Desires to overcome this
  - ➢ KD1: Skills can reach multiple physical versions of the same abstract state
  - ➢ KD2: Agent can consider multiple sequences of skills

Problem Setting: Environment

- X is the set of states
- Λ is the set of objects
- U is the action space
- f is the transition function $f: X \times U \to X$
- Ψ is the set of predicates

Problem Setting: Predicates

- A predicate is named
- Is defined over a tuple of object types
- A **ground atom** is a predicate that takes specific named objects
- A **lifted atom** uses variable place holders

Problem Setting: State Abstraction

- ❖ Abstract state is state represented in predicate form
- ❖ Formally: abstract(x) is the set of ground atoms which hold in state x

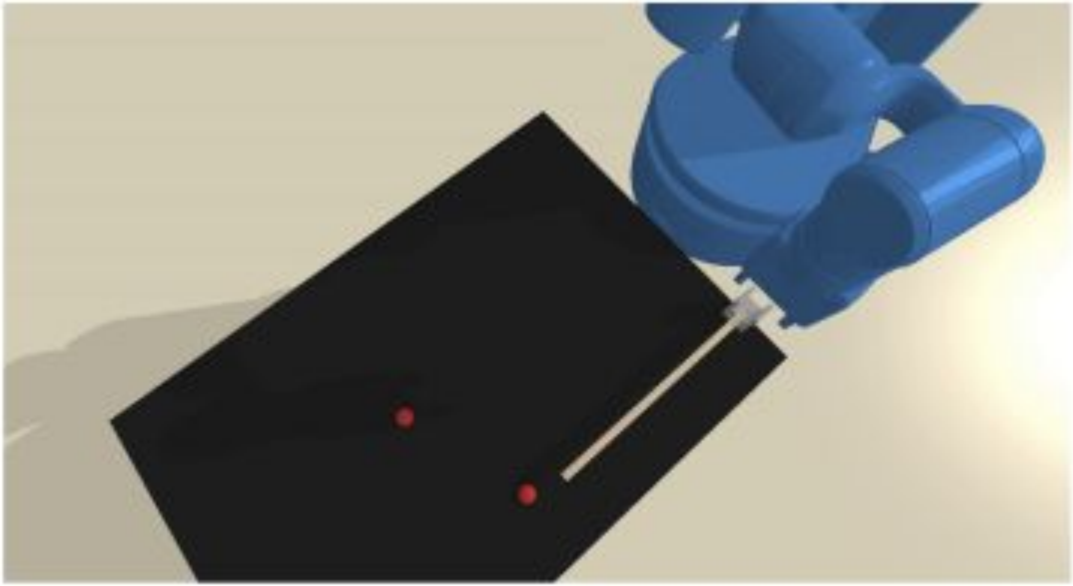
❖ **Note:**

Mapping is not

one-to-one

Problem Setting: Tasks

- T <O, x_0, g, H>
- O is the objects in the environment
- x_0 is the initial state
- g is the goal state
- H is the time horizon of the task

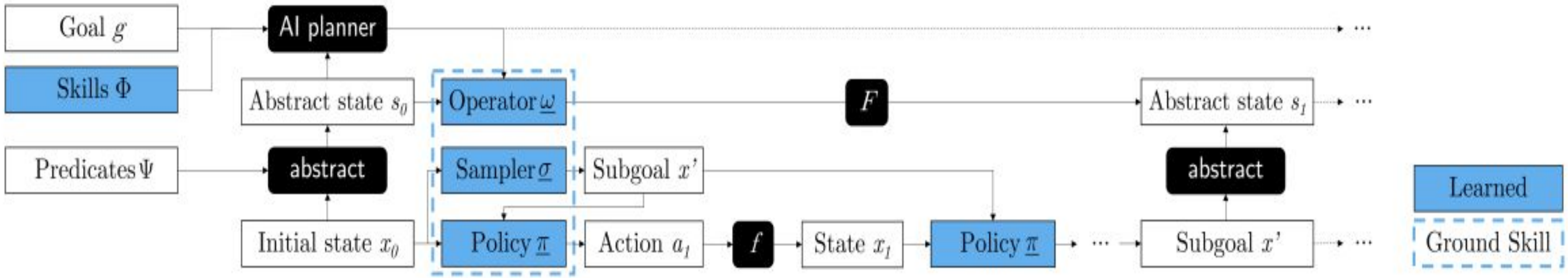


Stick Button [1]

Related Work & Limitations of Prior Work

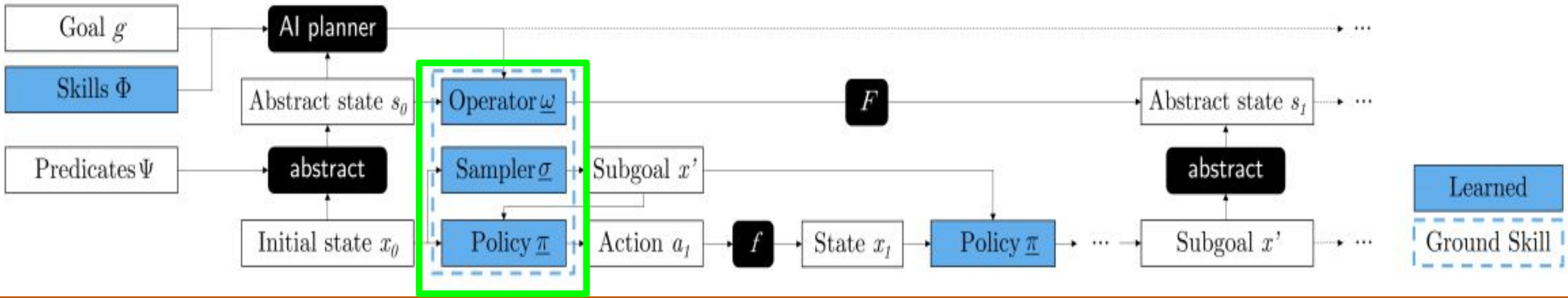
- Earlier paper by the same authors is very similar to this work but they used manually designed policies for each skill [3]
- Deep skill chaining paper learns skills with learned policies but does not represent them symbolically [4]
- SDRL represents skills symbolically but does not learn skills wholistically, only their policies [5]
  - Also does not address KD1 and KD2

Proposed Approach: High-Level



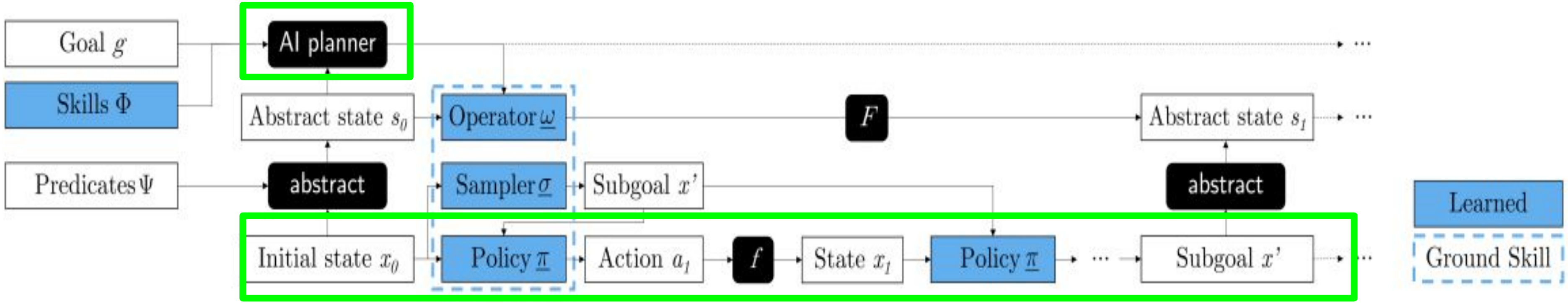
Proposed Approach: Neuro-Symbolic Skills

- **Think of these as abstract actions to plan over**
- Composed of
  - Argument object variables
  - A Symbolic Operator
  - A Subgoal Conditioned Policy
  - A Subgoal Sampler
- Abstract transition function F



Proposed Approach: Bilevel Planning

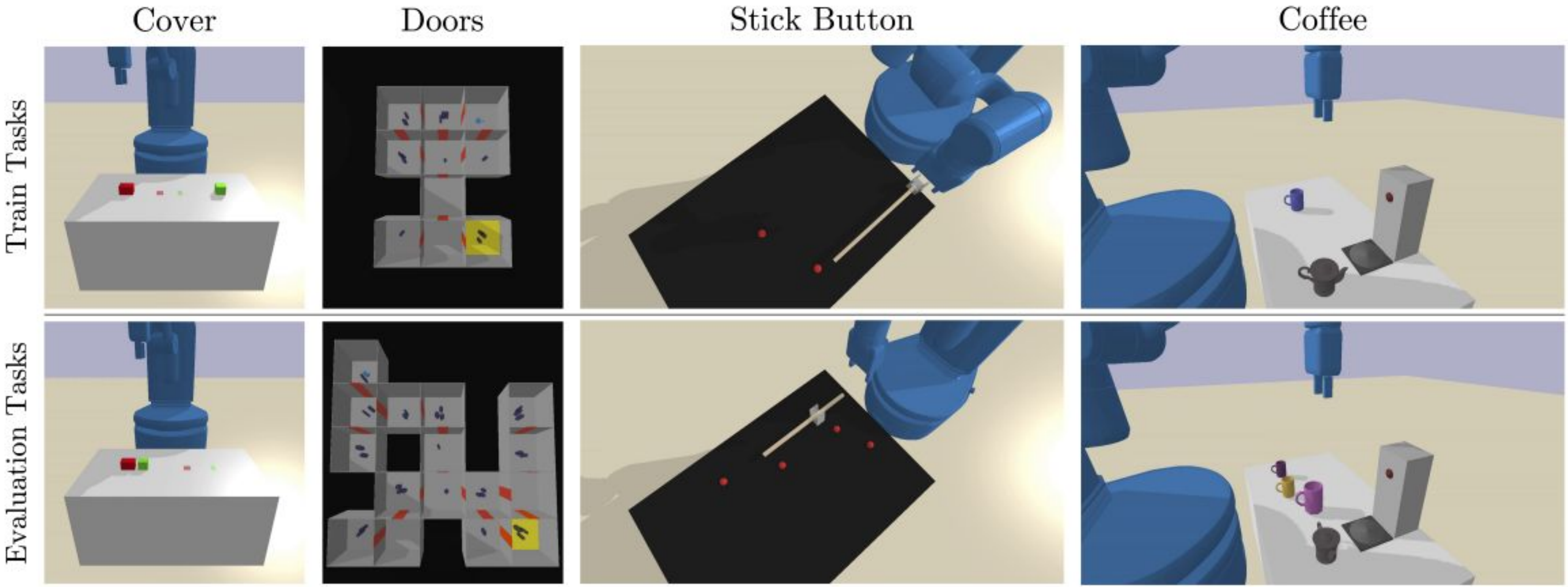
- Outer level generates k sequences of skills
  - Uses A* heuristic search
- Inner level iterates over candidate skill sequence from goal to current state
  - Uses subgoal sampler to generate subgoal for each skill
  - Uses this subgoal in goal conditioned policy to generate an action sequence



Proposed Approach: Learning Neuro-Symbolic Skills From Demonstrations

- Demonstration is are sequence of states and actions
- Preprocess Demonstrations
  - Segment based on predicates
  - Partition into similar skills
  - Lift to variable representations of predicates
- Operator Learning
  - Preconditions and Add/Delete effects follow from lifted representations
- Policy Learning
  - Supervised learning to learn map of demonstration states to actions
- Subgoal Sampler Learning
  - Supervised learning to learn mapping from initial state to subgoal state

Experimental Setup: Domains



[1]

Experimental Setup: Baselines

- BPNS *No Subgoal*
- GNN-Meta
- GNN-Meta *No Subgoal*
- GNN BC
- Samples=1
- Abstract Plans =1

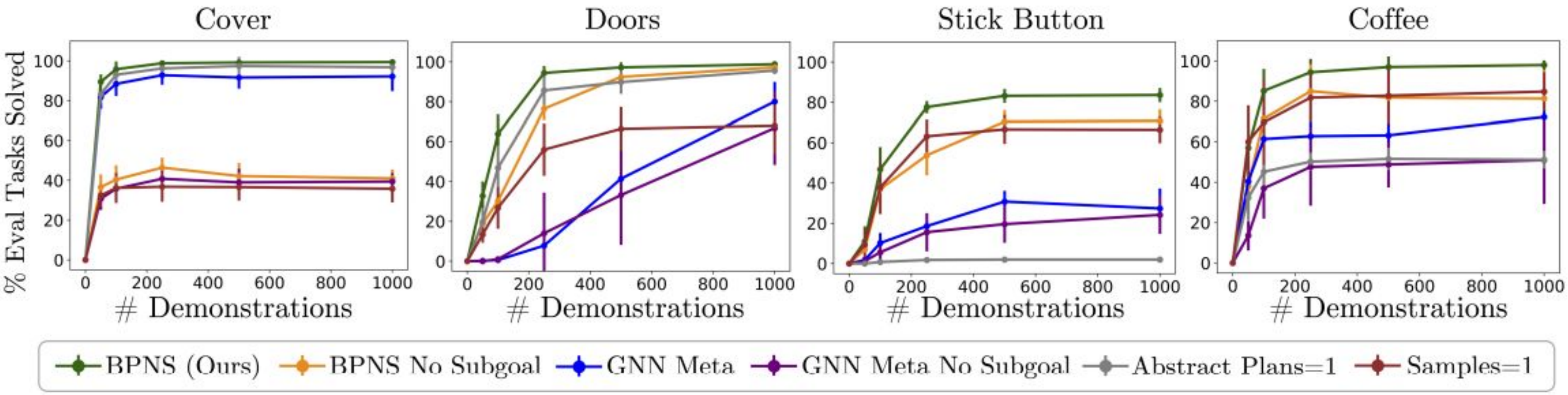
Experimental Setup: Questions to Answer

- **Q1**: How many train tasks are required to solve holdout tasks well?
- **Q2**: How well does it generalize to unseen numbers of objects?
- **Q3**: Can this learn skills which complement general purpose skills?
- **Q4**: How important is the ability to sample multiple sub-goals for KD1?
- **Q5**: How important is it to be able to generate multiple abstract plans for KD2?

Experimental Setup: Evaluation Metric

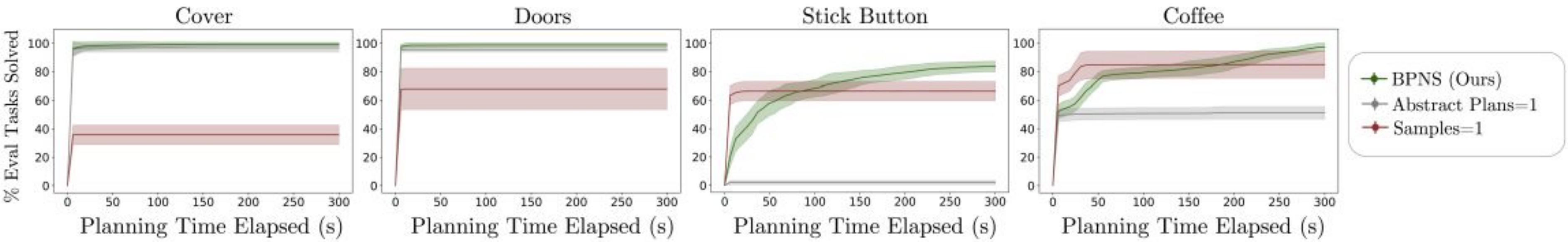
How many evaluation tasks can be solved

Experimental Results: Main Results



[1]

Experimental Results: Time Analysis with Ablations



[1]

Discussion of Results: Question Answers

- **Q1**: How many train tasks are required to solve holdout tasks well?
  - 100-250
- **Q2**: How well does it generalize to unseen numbers of objects?
  - Well
- **Q3**: Can this learn skills which complement general purpose skills?
  - Yes
- **Q4**: How important is the ability to sample multiple sub-goals for KD1?
  - Quite
- **Q5**: How important is it to be able to generate multiple abstract plans for KD2?
  - Quite

Discussion of Results: Strengths & Weaknesses

- Strengths:
  - Compares to and out-performs GNN-Meta which is pretty relevant and recent solution
  - Compare in environments that are not seemingly fabricated to inherently be better than compared baselines
- Weaknesses:
  - GNN-Meta didn't really seem to perform that well on the harder tasks
    - Perhaps GNN-Meta was a cherry picked baseline

Critique & Limitations

- Say at the beginning that they want to evaluate the number of tasks before wall clock timeout, but they never actually check that
  - Makes me wonder if it was actually slower but more successful on evaluation tasks?
- Assumes fully observable and deterministic states – limits applications
- Evaluating the predicates in real world scenarios is non-trivial

Future Work for Paper

- **Incorporate closed loop with other work that learns predicates from skills with this learning skills from predicates, to try to invent new skills/predicates**
  - **Authors have already begun work on this showing that in the cover environment that new predicates can be generated only starting with the goal predicate of Covering. It should be noted that some important predicates like Holding were not generated**
- **Try to evaluate predicates in real scenarios rather than assuming oracle predicate evaluator**
- **Try to learn the set of symbols concurrently?**

Extended Readings

- Leveraging Approximate Symbolic Models for Reinforcement Learning via Skill Diversity [6]
- Learning Symbolic Operators for Task and Motion Planning [3]
- Learning Multi-Object Symbols for Manipulation with Attentive Deep Effect Predictors [7]
- Neurosymbolic Learning for Robust and Reliable Intelligent Systems [8]
- Safe Neurosymbolic Learning with Differentiable Symbolic Execution [9]

Summary

- Want to learn, rather than construct, policies for neuro-symbolic skills for bilevel planning
- Allows for efficient AI planning algorithms, but state abstractions are lossy
- Policies for neuro-symbolic skills were hand-made, if any part of the skill was learned at all
- Able to address some of the loss that occurs by abstracting physical states to predicate representations, through sub-goal sampling and considering more than one skill sequence
- Generalizes to varying numbers of objects in the environment – skill predicates are robust to changes in number of variables it might have to consider

References

[1] T. Silver, A. Athalye, J. B. Tenenbaum, T. Lozano-Perez, and L. P. Kaelbling, “Learning neuro-symbolic skills for bilevel planning,” arXiv preprint arXiv:2206.10680, 2022.

[2] C. R. Garrett, R. Chitnis, R. Holladay, B. Kim, T. Silver, L. P. Kaelbling, and T. Lozano-Perez. Integrated task and motion planning. Annual Review of Control, Robotics, and Autonomous Systems, 4, May 2021.

[3] T. Silver, R. Chitnis, J. Tenenbaum, L. P. Kaelbling, and T. Lozano-Perez. Learning symbolic operators for task and motion planning. In 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 3182–3189. IEEE, 2021.

[4] A. Bagaria and G. Konidaris. Option discovery using deep skill chaining. In International Conference on Learning Representations (ICLR), 2019.

[5] D. Lyu, F. Yang, B. Liu, and S. Gustafson. SDRL: interpretable and data-efficient deep reinforcement learning leveraging symbolic planning. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, pages 2970–2977, 2019.

[6] L. Guan, S. Sreedharan, and S. Kambhampati. “Leveraging approximate symbolic models for reinforcement learning via skill diversity,” arXiv preprint arXiv:2202.02886, 2022.

[7] A. Ahmetoglu, E. Oztop, E. Ugur, “Learning Multi-Object Symbols for Manipulation with Attentive Deep Effect Predictors,” arXiv preprint arXiv:2208.01021, 2022.

[8] J. P. Inala, “Neurosymbolic Learning for Robust and Reliable Intelligent Systems,” thesis, 2022.

[9] C. Yang, S. Chaudhuri, “Safe Neurosymbolic Learning with Differentiable Symbolic Execution,” ICLR, 2022