Playgol: learning programs through play (extended abstract)

Andrew Cropper

University of Oxford

Abstract. Children learn through play. We introduce the analogous idea of *learning programs through play*. In this approach, a program induction system (the learner) is given a set of user-supplied *build* tasks and initial background knowledge (BK). Before solving the build tasks, the learner enters an unsupervised *playing* stage where it creates its own *play* tasks to solve, tries to solve them, and saves any solutions (programs) to the BK. After the playing stage is finished, the learner enters the supervised *building* stage where it tries to solve the build tasks and can reuse solutions learnt whilst playing. The idea is that playing allows the learner to discover reusable general programs on its own which can then help solve the build tasks. We implement our idea in *Playgol*, a new inductive logic programming system. Our experimental results suggest that playing can substantially improve learning performance.

1 Introduction

Children learn through play [17–19]. We introduce the analogous idea of *learning programs through play* [1]. In this approach, a program induction system (the learner) is given a set of user-supplied *build* tasks and initial background knowledge (BK). Whereas a standard program induction system would immediately try to solve the build tasks, in our approach the learner first enters an unsupervised *playing* stage. In this stage the learner creates its own *play* tasks to solve, tries to solve them, and saves any solutions (programs) to the BK. After the playing stage is finished, the learner enters the supervised *building* stage where it tries to solve the user-supplied build tasks and can reuse solutions learned whilst playing. The idea is that playing allows the learner to discover reusable general programs on its own which can then be reused in the building stage, and thus improve performance. For instance, if trying to learn sorting algorithms, a learner could discover the concepts of *partition* and *append* whilst playing which could then help learn *quicksort*.

To further illustrate our play idea, imagine a child that had never seen Lego before. Suppose you presented the child with Lego bricks and immediately asked them to build a (miniature) house with a pitched roof. The child would probably struggle to build the house without first knowing how to build a stable wall or how to build a pitched roof. Now suppose that before you asked

the child to build the house, you first left them alone to play with the Lego. Whilst playing the child may build animals, gardens, ships, or many other seemingly irrelevant things. However, the child is likely to discover reusable and general concepts, such as the concept of a stable wall. As we discuss in Section 2 in [1], the cognitive science literature shows that children can better learn complex rules after a period of play rather than solely through observation [17–19]. In this work, we explore whether a program induction system can similarly better learn programs after a period of play.

Our idea of using play to discover useful BK contrasts with most forms of program induction which usually require predefined, often human-engineered, static BK as input [15, 3, 12, 16, 5, 11, 10, 9]. Our idea is related to program induction approaches that perform multitask or meta learning [13, 6, 8, 7]. In these approaches, a learner acquires BK in a *supervised* manner by solving sets of user-provided tasks, each time saving solutions to the BK, which can then be reused to solve other tasks. In contrast to these supervised approaches, our play approach discovers useful BK in an *unsupervised* manner whilst playing. Playing can therefore be seen as an unsupervised technique for a learner to discover the BK necessary to solve complex tasks, i.e. a form of unsupervised bootstrapping for supervised program induction.

We claim that playing can improve learning performance. To support this claim, in [1], we make the following contributions:

- We introduce the idea of learning programs through play and show that playing can reduce the textual complexity of target concepts which in turn reduces the sample complexity of a learner.
- We implement our idea in *Playgol*, a new inductive logic programming (ILP) system based on meta-interpretive learning (MIL) [14, 15, 3, 2], uses Metagol [4], a MIL implementation, as the main learning algorithm.
- We experimentally show on two domains (robot planning and real-world string transformations) that playing can significantly improve learning performance.

References

1. Andrew Cropper. Playgol: learning programs through play. In *IJCAI 2019*. To appear.
2. Andrew Cropper. *Efficiently learning efficient programs*. PhD thesis, Imperial College London, UK, 2017.
3. Andrew Cropper and Stephen H. Muggleton. Learning higher-order logic programs through abstraction and invention. In *IJCAI 2016*, pages 1418–1424. IJCAI/AAAI Press, 2016.
4. Andrew Cropper and Stephen H. Muggleton. Metagol system. https://github.com/metagol/metagol, 2016.

5. Andrew Cropper and Stephen H. Muggleton. Learning efficient logic programs. *Machine Learning*, Apr 2018.
6. Eyal Dechter, Jonathan Malmaud, Ryan P. Adams, and Joshua B. Tenenbaum. Bootstrap learning via modular concept discovery. In *IJCAI 2013*, pages 1302–1309. IJCAI/AAAI, 2013.
7. Kevin Ellis and Sumit Gulwani. Learning to learn programs from examples: Going beyond program structure. In *IJCAI 2017*, pages 1638–1645. ijcai.org, 2017.
8. Kevin Ellis, Lucas Morales, Mathias Sablé-Meyer, Armando Solar-Lezama, and Josh Tenenbaum. Learning libraries of subroutines for neurally-guided bayesian program induction. In *NeurIPS 2018*, pages 7816–7826, 2018.
9. Kevin Ellis, Daniel Ritchie, Armando Solar-Lezama, and Josh Tenenbaum. Learning to infer graphics programs from hand-drawn images. In *NeurIPS 2018*, pages 6062–6071, 2018.
10. Richard Evans and Edward Grefenstette. Learning explanatory rules from noisy data. *J. Artif. Intell. Res.*, 61:1–64, 2018.
11. Sumit Gulwani. Automating string processing in spreadsheets using input-output examples. In *POPL 2011*, pages 317–330, 2011.
12. Mark Law, Alessandra Russo, and Krysia Broda. Inductive learning of answer set programs. In *JELIA 2014*, pages 311–325, 2014.
13. Dianhuan Lin, Eyal Dechter, Kevin Ellis, Joshua B. Tenenbaum, and Stephen Muggleton. Bias reformulation for one-shot function induction. In *ECAI 2014*, pages 525–530, 2014.
14. Stephen H. Muggleton, Dianhuan Lin, Niels Pahlavi, and Alireza Tamaddoni-Nezhad. Meta-interpretive learning: application to grammatical inference. *Machine Learning*, 94(1):25–49, 2014.
15. Stephen H. Muggleton, Dianhuan Lin, and Alireza Tamaddoni-Nezhad. Meta-interpretive learning of higher-order dyadic datalog: predicate invention revisited. *Machine Learning*, 100(1):49–73, 2015.
16. Peter Schüller and Mishal Benz. Best-effort inductive logic programming via fine-grained cost-based hypothesis generation - the inspire system at the inductive logic programming competition. *Machine Learning*, 107(7):1141–1169, 2018.
17. Laura E Schulz, Alison Gopnik, and Clark Glymour. Preschool children learn about causal structure from conditional interventions. *Developmental science*, 10(3):322–332, 2007.
18. Zi L Sim and Fei Xu. Learning higher-order generalizations through free play: Evidence from 2-and 3-year-old children. *Developmental psychology*, 53(4):642, 2017.
19. Zi Lin Sim, Kuldeep K. Mahal, and Fei Xu. Is play better than direct instruction? learning about causal systems through play. In *CogSci 2017*, 2017.