Learning Equational Theorem Proving

Jelle Piepenbrock[1,2], Tom Heskes[1], Mikoláš Janota[2], and Josef Urban[2]

[1] Radboud University, Nijmegen, The Netherlands
[2] Czech Technical University, Prague, Czech Republic

Abstract

We develop Stratified Shortest Solution Imitation Learning (3SIL) to learn equational theorem proving in a deep reinforcement learning (RL) setting. The self-trained models achieve state-of-the-art performance in proving problems generated by one of the top open conjectures in quasigroup theory, the Abelian Inner Mapping (AIM) Conjecture. To develop the methods, we first use two simpler arithmetic rewriting tasks that share tree-structured proof states and sparse rewards with the AIM problems. On these tasks, 3SIL is shown to significantly outperform several established RL and imitation learning methods. The final system is then evaluated in a standalone and cooperative mode on the AIM problems. The standalone 3SIL-trained system proves in 60 seconds more theorems (70.2%) than the complex, hand-engineered *Waldmeister* system (65.5%). In the cooperative mode, the final system is combined with the *Prover9* system, proving in 2 seconds what standalone *Prover9* proves in 60 seconds.

Automated theorem proving has been applied in the theory surrounding the Abelian Inner Mapping Conjecture, known as the AIM Conjecture [3]. This is one of the top open conjectures in quasigroup theory. Work on the conjecture has been going on for more than a decade. Automated theorem provers use hundreds of thousands of inference steps when run on problems from this theory. In this work, we train a machine learning model to guide proof decisions.

We use a dataset of theorems generated by this conjecture as a testbed for our machine learning methods [1]. The dataset comes with a simple prover called AIMLEAP that can take machine learning advice.[1] We use this system as a reinforcement learning environment. AIMLEAP keeps the state and carries out the cursor movements and tree rewrites.

The AIM conjecture concerns specific structures in *loop theory* [3]. A loop is a quasigroup with an identity element. A quasigroup is a generalization of a group that does not preserve associativity. Currently, work in this area is done using automated theorem provers such as *Prover9* [4, 3]. The *Prover9* theorem prover is especially suited to this approach because of its well-established *hints* mechanism [7]. The dataset is derived from this *Prover9* approach and contains around 3500 theorems that can be proven with the definitions and lemmas [1].

There are 177 possible actions in the environment. Three actions are cursor movements, where the cursor can be moved to an argument of the current position. The other actions all rewrite the current term at the cursor position with various axioms, definitions and lemmas that hold in the AIM context.

To develop a method that can solve equational theorem proving problems, we considered two simpler arithmetic tasks, which also have a tree-structured input and a sparse reward structure: Robinson arithmetic and polynomial arithmetic. In both cases, the task is to normalize a mathematical expression to one specific form. The learning environments incorporate two existing datasets. For the Robinson arithmetic normalization task, we use a dataset that was constructed for reinforcement learning experiments in the interactive theorem prover HOL4 [2]. For the polynomial normalization task, we employ a dataset introduced for experiments in symbolic rewriting using recurrent language models [5].

[1] https://github.com/ai4reason/aimleap

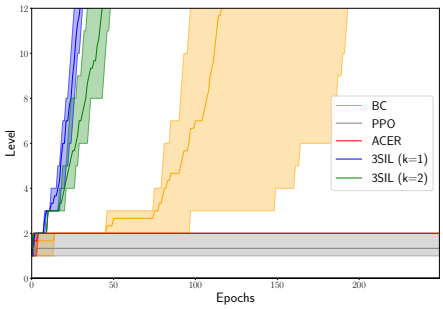

Figure 1: Comparison of methods on the Robinson arithmetic task. Y-axis denotes progress on a task curriculum, X-axis denotes the amount of training epochs.

Robinson arithmetic is a simple arithmetic theory. In the task, we are asking the agent to calculate the value of the expression. As an example, $S(S(0)) + S(0)$, representing $2 + 1$, needs to be rewritten to $S(S(S(0)))$. The setup for this RA normalization task is modeled after [2]. Based on the arithmetic tasks, we developed a method, called *stratified shortest solution imitation learning* (3SIL), which performed better than the baseline algorithms. In Figure 1, we show the relative performance of several methods on the Robinson arithmetic task. We compare standard reinforcement learning algorithms, such as PPO [6] and ACER [8] and behavioral cloning (BC), with our method 3SIL. On these tasks, our method outperforms the RL baselines, as shown in Figure 1. Both variants of our method advance more quickly through the curriculum than the baseline algorithms.

The method was then tested on the more difficult AIM theorem proving task. In Table 1, we show the performance of a model trained on the AIMLEAP task within the AIMLEAP environment. We compare with state-of-the-art theorem provers and observe that the model can outperform Waldmeister. In further experiments, we also observe that assisting Prover9 with the learned model can improve its performance. We let the model rewrite the starting state and add the rewritten states as lemmas to the Prover9 input. In this cooperative mode, the final system is combined with the *Prover9* system, proving in 2 seconds what standalone *Prover9* proves in 60 seconds. The results are shown in Table 2. This setup also outperforms E (shown in Table 1).

Table 1: Theorem proving performance on the hold-out test set in fraction of problems solved. Means and standard deviations are the results of evaluations of 3 different models from 3 different training runs.

Method	Success Rate
E (60s)	0.802
Waldmeister (60s)	0.655
Prover9 (60s)	0.833
Model (1x)	0.586 ± 0.029
Model (60s)	0.702 ± 0.015

Table 2: Prover 9 theorem proving performance on the hold-out test set when injecting lemmas suggested by the learned model. *Prover9*'s performance increases when using the model's suggested lemmas.

Method	Success Rate
Prover9 (1s)	0.715
Prover9 (2s)	0.746
Prover9 (1s) + Model (1s)	0.841 ± 0.019

In conclusion, we show that equational theorem proving in loop theory can be assisted by learned neural network models. In the future, we will explore whether we can automatically select the best previous proofs to learn from to accelerate the learning process.

References

[1] Chad E Brown, Bartosz Piotrowski, and Josef Urban. Learning to advise an equational prover. *Artificial Intelligence and Theorem Proving*, 2020.

[2] Thibault Gauthier. Deep reinforcement learning in HOL4. *arXiv preprint arXiv:1910.11797*, 2019.

[3] Michael Kinyon, Robert Veroff, and Petr Vojtěchovský. Loops with abelian inner mapping groups: An application of automated deduction. In *Automated Reasoning and Mathematics*, pages 151–164. Springer, 2013.

[4] W. McCune. Prover9 and Mace. 2010.

[5] Bartosz Piotrowski, Josef Urban, Chad E Brown, and Cezary Kaliszyk. Can neural networks learn symbolic rewriting? *ICML Workshop on Learning and Reasoning with Graph-Structured Data*, 2019.

[6] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[7] Robert Veroff. Using hints to increase the effectiveness of an automated reasoning program: Case studies. *J. Autom. Reason.*, 16(3):223–239, 1996.

[8] Ziyu Wang, Victor Bapst, Nicolas Heess, Volodymyr Mnih, Remi Munos, Koray Kavukcuoglu, and Nando de Freitas. Sample efficient actor-critic with experience replay. *International Conference on Learning Representations*, 2016.