Learning Instantiation in First-Order Logic

Jelle Piepenbrock[1,2], Josef Urban[1], Konstantin Korovin[3], Miroslav Olšák[4], Tom Heskes[2], and Mikoláš Janota[1]

[1] CIIRC, Czech Technical University in Prague, Czechia
[2] Radboud University Nijmegen, The Netherlands
[3] University of Manchester, UK
[4] Institut des Hautes Études Scientifiques, France

Introduction The appearance of strong CDCL-based propositional (SAT) solvers has greatly advanced several areas of automated reasoning (AR). One of the directions in AR is thus to apply SAT solvers to expressive formalisms such as first-order logic, for which large corpora of general mathematical problems exist today. This is possible due to Herbrand's theorem, which allows reduction of first-order problems to propositional problems by instantiating variables. The core challenge is choosing the right instances from the typically infinite Herbrand universe. Instantiation is a powerful tool for formal reasoning with quantifiers.

In this work, we develop the first machine learning system targeting this task, addressing its combinatorial and invariance properties. In particular, we develop a new GNN2RNN architecture based on an invariant graph neural network (GNN) that learns from problems and their solutions independently of symbol names (addressing the abundance of skolems), combined with a recurrent neural network (RNN) that proposes for each clause its instantiations. The architecture is then trained on a corpus of mathematical problems and their instances produced by the iProver system, and its performance is evaluated in several ways. We show that the system can achieve high accuracy in predicting the right instances, and that it is capable of solving a large number of problems by educated guessing when combined with a SAT solver.

The power of instantiation is formalized by *Herbrand's theorem* [5], which states, roughly speaking, that within first-order logic (FOL), quantifiers can always be eliminated by the right instantiations. Herbrand's theorem further states that it is sufficient to consider instantiations from the *Herbrand universe*, which consists of terms with no variables (*ground terms*) constructed from the symbols appearing in the problem. This fundamental result has been explored in automated reasoning (AR) systems since the 1950s [2]. It means that once the right instantiations are discovered, we end up with a problem without quantifiers, which is typically easy to solve by state-of-the-art SAT solvers [12].

Methods Our starting point for instantiation in first-order logic is iProver. At the core of iProver is the Inst-Gen [4, 9] instantiation calculus, which can be combined with resolution and superposition calculi [3]. At a high level, the procedure works as follows. Given a set of first-order clauses S its propositional abstraction $S\perp$ is obtained by mapping all variables to a designated ground term $\perp$. A propositional solver is applied to $S\perp$ and it either proves that $S\perp$ is unsatisfiable and in this case the set of first-order clauses S is also unsatisfiable or shows that $S\perp$ is satisfiable and in this case returns a propositional model of the abstraction $S\perp$. This propositional model is analyzed if it can be extended to a full first-order model. If it cannot be extended then it is possible to show that there must be complementary literals in the model that are unifiable.

A major bottleneck is however the large number of generated instances, with only a few typically needed for the final proof. This motivates our work here: a trained predictor that proposes the most relevant instantiations can significantly help and complement the complete search procedures used by systems like iProver.

We construct a large corpus of instantiations by running iProver on 113 332 first-order ATP problems created by the AI4REASON project. They originate from the Mizar Mathematical Library (MML) [8] and are exported to first-order logic by the MPTP system [14]. All these problems have an ATP proof (in general in a high time limit) found by either the E/ENIGMA [11, 6] or Vampire/Deepire [10, 13] systems. Additionally, the problems' premises have been *pseudo-minimized* [7] by iterated Vampire runs. We use the pseudo-minimized versions because our focus here is on guidance rather than on premise selection.

We reimplement and modify the GNN architecture used in [6] to allow the network to produce partial instantiations for each clause by using a recurrent neural network (RNN) after running the GNN. The method computes instantiations level-wise, meaning that one head symbol is picked for each variable (if needed) in each clause, after which we add fresh variables and again ask for head symbols (see Figure 1).

$$\forall\ x\ \ z\ P(\ f(\ x\ ,\ z\ )\ )$$

$$\forall\ x_1\ x_2\ z_1\ P(\ f(\ \overbrace{t(\ x_1\ ,\ x_2\ )}^{t/2}\ ,\ \overbrace{g(\ z_1\ )}^{g/1}\ )\ )$$

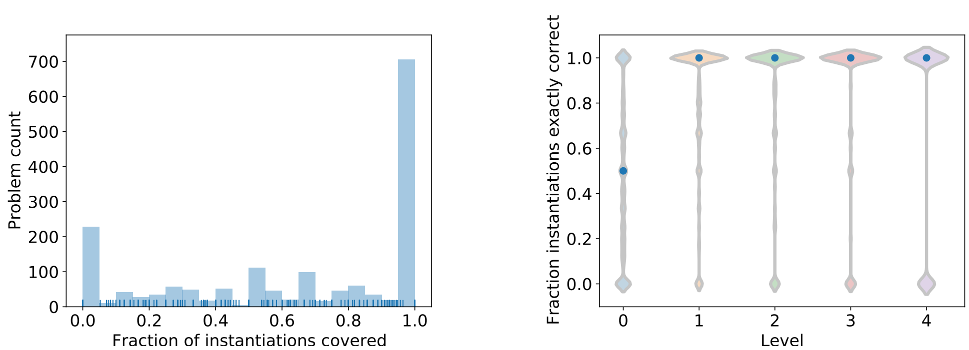
$$P(\ f(\ t(\ \underset{c/0}{c}\ ,\ \underset{c/0}{c}\ )\ ,\ g(\ \underset{e/0}{e}\ )\ )\ )$$

(1) instantiate x by head symbol t with arity 2 and z by g of arity 1 (going from $level_0$ to $level_1$)

(2) instantiate x_1, x_2, z_1 by constants c, c, and e, respectively (going from $level_1$ to $level_2$)

Figure 1: Term instantiation through incremental deepening. In the figure, there are two instantiation steps, one after the other.

Results We first evaluate the trained GNN2RNN by measuring the overlap of the predicted instantiations on the unseen test problems at each level. The system manages to predict correct instantiations for a large part of the set, see Figure 2a. In particular, about for 700 out of 1682 problems, the predictions include the exact instances used in the iProver proof. Figure 2b



(a) (b)

Figure 2: **a**: Histogram of the fraction of needed instantiations predicted for unseen test set problems. **b**: Violin plot of the fraction of instantiations correctly predicted, split by how many symbol levels from the base problem the problem was. Blue dot is the median of each group.

shows the results per level, which reveals an interesting pattern: the system is much better at predicting the instances for levels 1–4 (almost fully correct), when the first head symbol of each term is already determined by the proof instance. Next, we combine GNN2RNN with EGround and PicoSAT [1] to see if the proposed ground instances are already propositionally unsatisfiable. The fraction of problems that PicoSAT finds unsatisfiable after one top-down GNN2RNN step at level$_i$ is 21%, 80%, 80%, 83% and 80% respectively. Again, we see that picking the first head symbol for each variable is the hardest, but the system performs well for the subsequent symbol choices.

Acknowledgements This work was partially supported by the European Regional Development Fund under the Czech project AI&Reasoning no. CZ.02.1.01/0.0/0.0/15_003/ 0000466 (JP, JU), Amazon Research Awards (JP, JU) and by the Czech MEYS under the ERC CZ project *POSTMAN* no. LL1902 (JP, MJ).

References

[1] Armin Biere. Picosat essentials. *J. Satisf. Boolean Model. Comput.*, 4(2-4):75–97, 2008.

[2] Martin Davis. The early history of automated deduction. In *Handbook of Automated Reasoning*, pages 3–15. Elsevier and MIT Press, 2001.

[3] André Duarte and Konstantin Korovin. Implementing superposition in iProver (system description). In Nicolas Peltier and Viorica Sofronie-Stokkermans, editors, *Automated Reasoning - 10th International Joint Conference, IJCAR 2020, Paris, France, July 1-4, 2020, Proceedings, Part II*, volume 12167 of *Lecture Notes in Computer Science*, pages 388–397. Springer, 2020.

[4] Harald Ganzinger and Konstantin Korovin. New directions in instantiation-based theorem proving. In *18th IEEE Symposium on Logic in Computer Science (LICS 2003), 22-25 June 2003, Ottawa, Canada, Proceedings*, pages 55–64. IEEE Computer Society, 2003.

[5] Jacques Herbrand. *Recherches sur la théorie de la démonstration.* Doctorat d'état, La Faculté des Sciences de Paris, 1930.

[6] Jan Jakubuv, Karel Chvalovský, Miroslav Olsák, Bartosz Piotrowski, Martin Suda, and Josef Urban. ENIGMA anonymous: Symbol-independent inference guiding machine (system description). In Nicolas Peltier and Viorica Sofronie-Stokkermans, editors, *Automated Reasoning - 10th International Joint Conference, IJCAR 2020, Paris, France, July 1-4, 2020, Proceedings, Part II*, volume 12167 of *Lecture Notes in Computer Science*, pages 448–463. Springer, 2020.

[7] Cezary Kaliszyk and Josef Urban. Learning-assisted automated reasoning with Flyspeck. *J. Autom. Reasoning*, 53(2):173–213, 2014.

[8] Cezary Kaliszyk and Josef Urban. MizAR 40 for Mizar 40. *J. Autom. Reasoning*, 55(3):245–256, 2015.

[9] Konstantin Korovin. Inst-Gen - A modular approach to instantiation-based automated reasoning. In Andrei Voronkov and Christoph Weidenbach, editors, *Programming Logics - Essays in Memory of Harald Ganzinger*, volume 7797 of *Lecture Notes in Computer Science*, pages 239–270. Springer, 2013.

[10] Laura Kovács and Andrei Voronkov. First-order theorem proving and Vampire. In Natasha Sharygina and Helmut Veith, editors, *CAV*, volume 8044 of *LNCS*, pages 1–35. Springer, 2013.

[11] Stephan Schulz. System description: E 1.8. In Kenneth L. McMillan, Aart Middeldorp, and Andrei Voronkov, editors, *LPAR*, volume 8312 of *LNCS*, pages 735–743. Springer, 2013.

[12] João P. Marques Silva, Inês Lynce, and Sharad Malik. Conflict-driven clause learning SAT solvers. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pages 131–153. IOS Press, 2009.

[13] Martin Suda. Improving ENIGMA-style clause selection while learning from history. In *CADE*, volume 12699 of *Lecture Notes in Computer Science*, pages 543–561. Springer, 2021.

[14] Josef Urban. MPTP 0.2: Design, implementation, and initial experiments. *J. Autom. Reasoning*, 37(1-2):21–43, 2006.