Sifting through a large hypothesis space: Revisiting differentiable *learning through satisfiability* *

Stanisław J. Purgał[1], David M. Cerna[2,3], and Cezary Kaliszyk[1]

[1] University of Innsbruck, Innsbruck, Austria
`{stanislaw.purgal,cezary.kaliszyk}@uibk.ac.at`

[2] Czech Academy of Sciences Institute of Computer Science (CAS ICS), Prague, Czechia
`dcerna@cs.cas.cz`

[3] Research Institute for Symbolic Computation (RISC), Johannes Kepler University, Linz, Austria
`dcerna@risc.jku.at`

Abstract

A difficultly which must be addressed by inductive logical programming (ILP) systems is how to deal with the enormous space of plausible solutions. The majority of modern ILP systems approach this problem through the *meta-learning paradigm*, that is, only consider plausible solutions which are constructable from a set of *clause templates*. This approach has been adopted by investigations into neuro-symbolic ILP. Our investigation uses clause templates together with a variant of δILP, to expand the hypothesis space, rather than contract it. Our experiments support the following hypothesis: providing gradient descent with a larger solution space aids the discovery of explanatory hypotheses.

Inductive Logic Programming (ILP) [5] is a form symbolic machine learning approach which learns explanatory hypothesis from positive and negative evidence together with fixed background knowledge. These explanatory hypotheses take the form of a logic program. In contrast to statistical approaches to machine learning, ILP systems are data-efficient in that a complex hypothesis can be learned from only a few examples; in some cases, even a single example is sufficient. Additionally, these hypotheses tend to be human-readable and provide a route towards explainable AI. While there are many positive aspects of the approach, ILP systems ability to generalize is typically, negatively impacted by noisy input, and is limited to certain problem domains [1, 4].

Attempts to combine the flexibility and agnosticism to noise of statistical learning with the benefits of a firm logical foundation, forms the bedrock of the current investigations into *neuro-symbolic AI* [6]. In this abstract, we discuss our modification of a prominent approach to neuro-symbolic ILP, δILP [3]. This system is based on the *learning from satisfiability* ILP paradigm. In the case of δILP, the plausible hypothesis space is turned into a SAT problem where a model denotes a hypothesis. The hypothesis space is finite as a fixed program template is provided and the background knowledge is assumed to be ground and finite. This classical SAT problem can be transformed into a *soft* SAT problem by replacing the classical operators by *differentiable* ones. For example, Classical conjunction is replaced by the product *T-norm* [2], $X \wedge Y \equiv x * y$.

To understand our investigation we need to briefly introduce the structure of the *program templates* used by δILP. Clauses are assumed to be at most length 2, predicate definitions contain at most 2 clauses, and predicates may take at most 2 arguments. Each auxiliary predicate definition (including the predicate being learned) is associated with at most two *rule templates* defining the structure of its clauses. These rule templates state how many existential

*Supported by the ERC starting grant no. 714034 SMART, the Math$_{LP}$ project (LIT-2019-7-YOU-213) of the Linz Institute of Technology and the state of Upper Austria, Cost action CA20111 EuroProofNet.

variables occur within the clause and whether the predicate symbols occurring therein may be *extensional* (defined in the background) or *intensional* (derived during learning).

Each of the auxiliary predicate definition is associated with a matrix of weights where each entry denotes how strongly the system believes that a pair of clauses (respecting the rule templates) is the correct definition for the given predicate. This design choice is prohibitively expensive and significantly limits uses of the system due to the significant memory requirements. An alternative would be to assign a weight to each instantiation of the associated rule templates (so called *splitting* the definition), however, as discussed in Appendix F of [3], this approach is less effective for ILP.

In our investigation, we take definition splitting one step further and split not only the definitions (as was discussed in Appendix F of [3]), but also the individual rule templates. This entails that for each auxiliary predicate definition entries in the weight vector denote how strongly the system believes an instance of a predicate (i.e. $\mathit{father}(X, Y)$) is the correct choice for a particular position in a particular clause. This significantly reduces the memory requirement, but also goes far beyond the relaxations made by the Evans and Grefenstette [3] which they claim are less effective for ILP. To deal with this issue, instead of providing a program template which roughly matches the structure of the program we expect the system to find, We provide our modified δILP with many more auxiliary predicate then needed to construct the goal program. This is possible giving the memory saving resulting from splitting the weight matrix twice.

Let us consider the example $\mathit{fizz} \equiv \{X|X \in \mathbb{N} \wedge 0 = X(\mathit{mod}\ 3)\}$ from [3]. As background knowledge the authors provided the zero predicate and instances of the successor predicate up to 6 (i.e. $\mathit{succ}(0, 1)$,....). The positive examples are 0, 3 and 6 while the negative examples are all other natural numbers less than 6. This example posed a challenge for δILP and only 10% of the runs resulted in a mean squared error less than $1e - 4$. On the contrary, Up to 95% of our runs passed a validation phase regardless the mean squared error at the time of halting; The percentage is dependent on how many auxiliary predicates we allowed (see Figure 1).

This experiment together with a few others seem to contradict the exposition in Appendix F of [3]. However, it is not clear if these results can be further expended, nor how this can be generalized to handle more complex ILP task. An alternative approach to a more efficient search within the large search space would be the inclusion of supervised machine learning in the proposed framework. We leave these questions to future investigation.

```
 Possible explanatory Hypothesis
 for fizz example

fizz(X):- zero(X).
fizz(X):- p1(X,Y),p2(Y).
p1(X,Y):- succ(Y,Z),succ(Z,X).
p2(X):- succ(Y,X),fizz(Y).
```

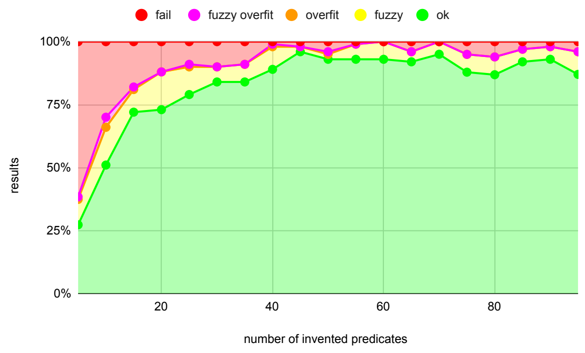

Figure 1: Percentage of runs finding a crisp solution which passes the validation phase.

References

[1] Andrew Cropper, Sebastijan Dumančić, Richard Evans, and Stephen H. Muggleton. Inductive logic programming at 30. *Machine Learning*, 111(1):147–172, Jan 2022.

[2] Francesc Esteva and Lluıs Godo. Monoidal t-norm based logic:towards a logic for left-continuous t-norms. *Fuzzy Sets and Systems*, 124(3):271–288, 2001. Fuzzy Logic.

[3] Richard Evans and Edward Grefenstette. Learning explanatory rules from noisy data. *Journal of Artificial Intelligence Research*, 61:1–64, January 2018.

[4] Mark Law, Alessandra Russo, and Krysia Broda. Inductive learning of answer set programs. In *Proceedings of Logics in Artificial Intelligence*, pages 311–325. Springer, August 2014.

[5] Stephen Muggleton. Inductive logic programming. *New Generation Computing*, 8(4):295–318, February 1991.

[6] Luc De Raedt, Sebastijan Dumancic, Robin Manhaeve, and Giuseppe Marra. From statistical relational to neuro-symbolic artificial intelligence. In *IJCAI 2020*, pages 4943–4950, 2020.