Can predicate invention in meta-interpretive learning compensate for incomplete background knowledge?

Andrew Cropper and Stephen H. Muggleton

Department of Computing, Imperial College London
a.cropper13@imperial.ac.uk,s.muggleton@imperial.ac.uk

Abstract. In ILP, incomplete background knowledge can lead to lower predictive accuracies. In this paper, we investigate whether meta-interpretive learning (MIL) can compensate for incomplete background knowledge through predicate invention. We compare the effect of providing progressively fewer background predicates to two MIL implementations: Metagol$_{pi}$, which performs predicate invention, and Metagol$_{nopi}$, which does not perform predicate invention. Our results show that as the number of background predicates decreases, Metagol$_{pi}$ outperforms Metagol$_{nopi}$ in terms of predictive accuracies, indicating that predicate invention can compensate for incomplete background knowledge.

1 Introduction

In an ideal world, machine learning datasets would be complete. However, in reality, we are often faced with incomplete data, which can lead to lower predictive accuracies in both feature-based [4, 8] and relational [15, 12] machine learning. In ILP, incomplete background knowledge can be due to missing values or missing predicates [7]. There are several techniques for handling missing data (see [2] for an overview). In this paper, we investigate whether predicate invention can compensate for missing background predicates. To evaluate this, we compare two versions of Metagol [11, 10, 6, 1]: Metagol$_{pi}$, which performs predicate invention, and Metagol$_{nopi}$ which does not perform predicate invention. We also compare the results with Progol [9], a popular ILP system which does not perform predicate invention.

1.1 Motivating example

Imagine a robot in a two-dimensional space which can perform six dyadic actions: *left, right, forwards, backwards, grab,* and *drop*. The task is to learn how to move a ball from a start position (1,1) to an end position (3,3). For this task, Metagol$_{pi}$ learns the solution in figure 1, where *s1, s2*, and *s3* are invented high-level actions. Now suppose that the *right*, *forwards*, and *drop* actions are not in the background knowledge. In this scenario, Metagol$_{pi}$ learns the solution in figure 2 using the extra invented predicate *s4*, which inverts the invented high-level

action *s3*, thus allowing the actions *left*, *backwards*, and *grab* to indicate their inverse actions *right*, *forwards*, and *drops*. It is this form of predicate invention that we explore in this paper.

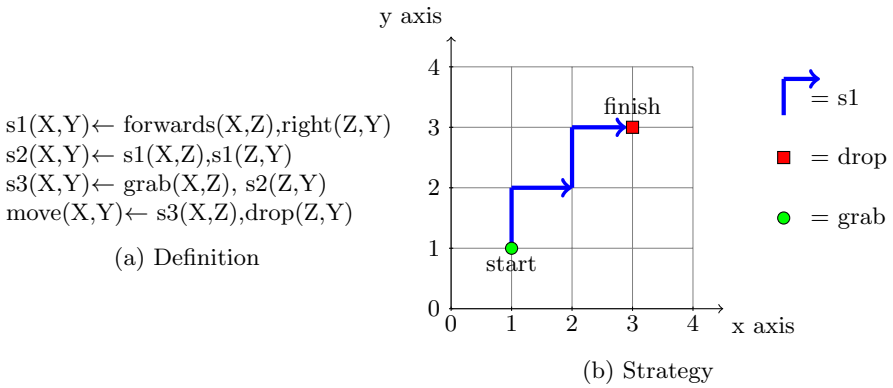


Fig. 1: Learning how to move ball with all six background predicates

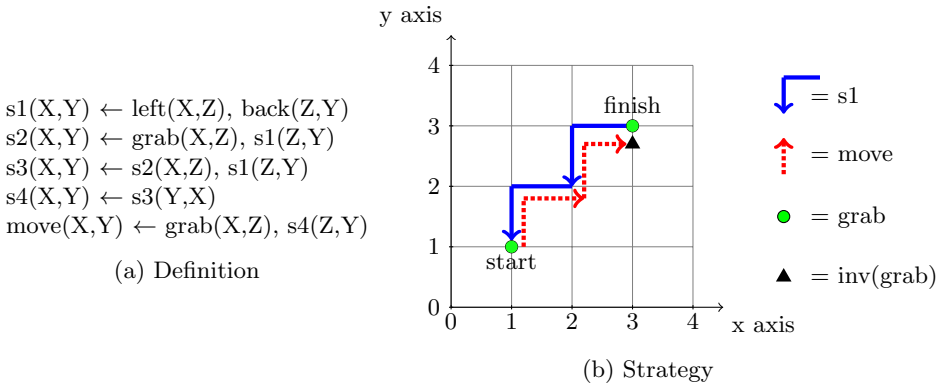


Fig. 2: Learning how to move ball with only three background predicates

1.2 Contributions

To our knowledge, this is the first study to investigate whether predicate invention can compensate for incomplete background knowledge. The main results are as follows:

– Metagol$_{pi}$ outperforms Metagol$_{nopi}$ when supplied with missing background predicates
– In some cases, Metagol$_{pi}$ maintains *respectable* predictive accuracies with only half of the original background predicates, i.e. predicative accuracies do not decrease in proportion to the decrease in background predicates

2 Meta-interpretive learning

MIL [11, 10] is an ILP technique based on an adapted version of a Prolog meta-interpreter. Whereas a Prolog meta-interpreter normally derives a proof by repeatedly fetching first-order clauses whose heads unify with a given goal, a MIL learner additionally fetches higher-order metarules whose heads unify with the goal, and saves the resulting meta-substitutions to form a hypothesis. To illustrate the idea consider the following metarule:

Metarule	**Clause**
P(x,y) ← Q(x,z), R(z,y)	aunt(x,y) ← sister(x,z), parent(z,y)

The uppercase letters P, Q, and R denote existentially quantified higher-order variables. The lowercase letters x, y, and z are universally quantified first-order variables. In a proof, meta-substitutions for the existentially quantified variables are saved in an abduction store. For example, the higher-order substitution θ = {P/aunt, Q/sister, R/parent} applied to the metarule in the left column above allows the proof to complete. In this scenario, the higher-order ground atom *chain(aunt, sister, parent)* is saved in the abduction store. The substitution θ can be reconstructed and re-used in later proofs, allowing a form of inductive programming which supports both predicate invention and the learning of recursive definitions. Following the proof of a goal consisting of a set of examples, the hypothesised program is formed by applying the resulting meta-substitutions to their corresponding metarules, displayed in the right column above.

In MIL, predicate invention is implemented by adding Skolem constants representing new predicate symbols to the predicate signature. These can then be used as substitutes for the existentially quantified variables in metarules.

3 Experiments

The experiments focus on determining whether predicate invention in MIL can compensate for missing background predicates. We test the following null hypothesis:

Null hypothesis Metagol$_{pi}$ cannot achieve higher predictive accuracies than Metagol$_{nopi}$ when supplied with missing background predicates

3.1 Materials

We use two implementations of Metagol: Metagol$_{pi}$, which performs predicate invention, and Metagol$_{nopi}$, which does not perform predicate invention. We also compare our results with Progol5, a popular ILP system which does not perform predicate invention. To test the experimental hypotheses we use the following data sets:

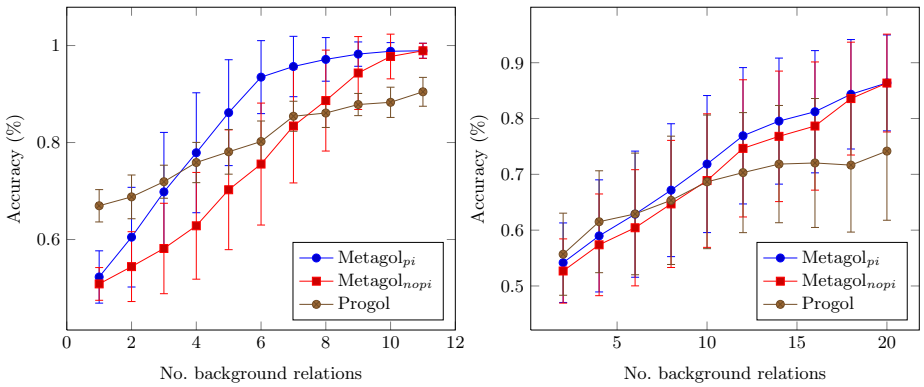
Hinton's kinship. The dataset from [5][1] contains 12 dyadic predicates and 104 examples. The predicates are uniformly distributed.

Custom kinship. This custom-built kinship dataset contains 21 dyadic predicates and 154 examples. The predicates are normally distributed.

3.2 Method

To explore the effect of missing background predicates, we randomly select p predicates to be included in the background knowledge. We then randomly select n training examples, half positive and half negative, of each target predicate and perform leave-one-out-cross-validation. Predictive accuracies are averaged over each target predicate over 50 trials.

3.3 Results



(a) Hinton's kinship dataset

(b) Our kinship dataset

Fig. 3: Predictive accuracies when learning kinship relations

Figure 1 shows that Metagol$_{pi}$ outperforms Metagol$_{nopi}$ on both datasets, except when all background predicates are present. Thus, the null hypothesis is rejected. The results displayed are for training sizes of $n = 10$. Results for other values of n are consistent but are omitted for brevity.

[1] https://archive.ics.uci.edu/ml/datasets/Kinship

4 Discussion

The rejection of the null hypotheses suggests that Metagol can compensate for missing background predicates through predicate invention. To illustrate this, consider learning the great-great-grandparent *(gggparent)* kinship relation using only the *mother* and *father* predicates. For this task, Metagol$_{pi}$ learns the following definition:

s2(X,Y) ← father(X,Y)
s2(X,Y) ← mother(X,Y)
s3(X,Y) ← s2(X,Z), s2(Z,Y)
gggparent(X,Y) ← s3(X,Z), s2(Z,Y)

Metagol$_{pi}$ invents both the *parent (s2)* and *grandparent (s3)* predicates given only examples of the *gggparent* predicate. To our knowledge, this level of nested predicate invention is beyond anything in the ILP literature.

Also, our results show that Metagol$_{pi}$ generally outperforms Progol in terms of predicate accuracies, except in the case of few background predicates. This is because Metagol$_{pi}$ finds overly generalised definitions, due to the strong declarative bias imposed by the metarules. In contrast, Progol finds overly specialised definitions, due to the weaker declarative bias imposed by Progol's mode declarations.

5 Conclusions and further work

In this paper, we investigated whether predicate invention in MIL can compensate for missing background predicates. Our results show that as the number of background predicates decrease, Metagol$_{pi}$ outperforms Metagol$_{nopi}$ in terms of predictive accuracies, indicating that predicate invention can compensate for incomplete background knowledge.

We have so far implicitly suggested that incomplete background knowledge is due to error or other unavoidable reason. However, our results suggest a possible motivation for purposely removing background predicates. In [8] it was shown that within the H^2_2 hypothesis space[2] the number of programs of size n which can be built from p predicate symbols and m metarules is $O(p^{3n}m^n)$. Thus, if working with many background predicates, then it might be preferable to purposely remove half of them to make the problem more tractable, whilst maintaining *respectable* predictive accuracies. Specifically, halving the number of predicates reduces the number of programs to $O(2^{3n}m^n)$.

In some ways, purposely removing background predicates is analogous to dimensionality reduction, widely applied in other forms of machine learning [14], but which has so far been neglected in ILP research [3]. In [1], the authors used Plotkin's clausal reduction algorithm [13] to logically minimise a maximal set

[2] H^j_i consists of definite function-free logic programs with predicates of adicity at most i and with at most j atoms in the body of each clause

of metarules in a chained fragment of H_2^2. It seems reasonable to ask whether we can apply a similar technique to determine whether background predicates are redundant with respect to the metarules using Plotkin's algorithm. This involves applying the encapsulation theorem in [1] to convert first-order background primitives to higher-order clauses. Developing this theory is left for future work.

Acknowledgements

The first author acknowledges the support of the BBSRC and Syngenta in funding his PhD Case studentship. The second author would like to thank the Royal Academy of Engineering and Syngenta for funding his present five-year Research Chair.

References

1. Andrew Cropper and Stephen Muggleton. Logical minimisation of metarules within meta-interpretive learning. In *Proc. of the 24th International Conference on Inductive Logic Programming*, 2014. To appear.
2. Sašo Džeroski. Handling imperfect data in inductive logic programming. In *Proceedings of the Fourth Scandinavian Conference on Artificial Intelligence—93*, SCAI93, pages 111–125, Amsterdam, The Netherlands, The Netherlands, 1993. IOS Press.
3. Johannes Fürnkranz. *Dimensionality reduction in ILP: A call to arms.* Citeseer, 1997.
4. Zoubin Ghahramani and M.I. Jordan. Learning from incomplete data. Technical report, Lab Memo No. 1509, CBCL Paper No. 108, MIT AI Lab, 1995.
5. G E Hinton. Learning distributed representations of concepts. *Artificial Intelligence*, 40:1–12, 1986.
6. D. Lin, E. Dechter, K. Ellis, J.B. Tenenbaum, and S.H. Muggleton. Bias reformulation for one-shot function induction. In *Proceedings of the 23rd European Conference on Artificial Intelligence (ECAI 2014)*, Amsterdam, 2014. IOS Press. In Press.
7. Chunnian Liu and Ning Zhong. Rough problem settings for inductive logic programming. In *New Directions in Rough Sets, Data Mining, and Granular-Soft Computing*, pages 168–177. Springer, 1999.
8. Benjamin M Marlin. *Missing data problems in machine learning.* PhD thesis, University of Toronto, 2008.
9. S.H. Muggleton. Inverse entailment and Progol. *New Generation Computing*, 13:245–286, 1995.
10. S.H. Muggleton and D. Lin. Meta-interpretive learning of higher-order dyadic datalog: Predicate invention revisited. In *Proceedings of the 23rd International Joint Conference Artificial Intelligence (IJCAI 2013)*, pages 1551–1557, 2013.
11. S.H. Muggleton, D. Lin, N. Pahlavi, and A. Tamaddoni-Nezhad. Meta-interpretive learning: application to grammatical inference. *Machine Learning*, 94:25–49, 2014.
12. Stephen H Muggleton, Jianzhong Chen, Hiroaki Watanabe, Stuart J Dunbar, Charles Baxter, Richard Currie, José Domingo Salazar, Jan Taubert, and Michael JE Sternberg. Variation of background knowledge in an industrial application of ilp. In *Inductive Logic Programming*, pages 158–170. Springer, 2011.

13. G.D. Plotkin. *Automatic Methods of Inductive Inference*. PhD thesis, Edinburgh University, August 1971.
14. David Skillicorn. *Understanding complex datasets: data mining with matrix decompositions*. CRC press, 2007.
15. Ashwin Srinivasan, S Muggleton, and RD King. Comparing the use of background knowledge by inductive logic programming systems. In *Proceedings of the 5th International Workshop on Inductive Logic Programming*, pages 199–230. Department of Computer Science, Katholieke Universiteit Leuven, 1995.