MC-TopLog: Complete Multi-clause Learning Guided by A Top Theory

Stephen Muggleton, Dianhuan Lin, and Alireza Tamaddoni-Nezhad

Department of Computing, Imperial College London

Abstract. Within ILP much effort has been put into designing methods that are complete for hypothesis finding. However, it is not clear whether completeness is important in real-world applications. This paper uses a simplified version of grammar learning to show how a complete method can make a difference to the grammar learning results compared to an incomplete method. Seeing the necessity of having a complete method for real-world applications, we introduce a method called $\top$-directed theory co-derivation, which is sound and complete with respect to the provided declarative bias. The proposed method has been implemented in the ILP system MC-TopLog and tested on the grammar learning example. Compared to Progol5, an efficient but incomplete ILP system, MC-TopLog has higher predictive accuracies, especially when the background knowledge is severely incomplete.

1 Introduction

As first pointed out by Yamaoto [15], hypotheses derivable from Progol [6] are restricted to those which subsume E relative to B in Plotkin's sense [12]. Progol's incompleteness can be characterized by deriving only single-clause hypotheses. Therefore we refer to entailment-incomplete methods of this type as single-clause learning, while entailment-complete methods as multi-clause learning.

Yamamoto uses the learning of the concept of odd-numbers to demonstrate Progol's incompleteness. His example involves recursion and mutually dependent predicates (odd and even), making it unclear whether only applications with these properties might be affected by this incompleteness. To the authors' knowledge it has not subsequently been demonstrated conclusively that the incompleteness of single-clause learning noticeably restricts the application of single-clause learners. It might reasonably be supposed that in real-world applications learned theories can always be built by sequentially adding single clauses to explain individual examples.

Since grammar learning is central to language translation software, automated booking systems and grammar checking for word processes, section 2 uses a simplified version of grammar learning, which is artificially designed and does not involve recursion or mutually dependent predicates, to show how a complete method can improve the learning results of an incomplete method. This is further demonstrated via experiments in section 4. More experiments with real-world applications can be found in [4], where target hypotheses are unknown for knowledge discovery tasks. The focus of this paper is to introduce a new complete approach called $\top$-directed theory co-derivation($\top$DTcD). The artificial example of grammar learning, whose target hypothesis is already known, is used as a running example for explaining our new approach.

$\top$DTcD extends $\top$DTD ($\top$-directed theory derivation) [5] based on the idea of constructing common generalisations of multiple examples. Common generalization was first introduced in Plotkin's Least General Generalization(LGG) [11] and Reynolds' Least Common Generalization (LCG) [14]. More recent methods of constructing common generalisations include Golem [8] and ProGolem [10]. However, they all suffer from similar type of incompleteness as in Progol. While all the existing complete methods (e.g. CF-Induction [2], XHAIL [13], IMPARO [3] and TAL [1]) use a single example as a seed for constructing generalisation. In this paper, we refer to methods generalising a single example as solo-generalization, as opposed to co-generlization that generalise multiple examples together.

$\top$DTD is proved in [5] to be complete for deriving hypotheses from a top theory. It extends $\top$DHD ($\top$-directed hypothesis derivation) implemented in TopLog [9], which is incomplete like Progol. That is why the system with $\top$DTD is named MC-TopLog (Multi-clause TopLog). $\top$DTD and $\top$DTcD correspond to two different learning modes in MC-TopLog: generalising single example or multiple examples. Inherited from $\top$DHD in TopLog, both $\top$DTD and $\top$DTcD are $\top$-directed methods, which use a logic program called the top theory to represent declarative bias. Compared to mode declarations used in other complete methods, a top theory has the advantage of encoding a strong declarative bias. More details about strong declarative biases are given in section 3.1

2 Multi-clause Learning vs. Single-clause Learning

Here we explain why Progol's incompleteness is characterized by single-clause learning. A hypothesis H will not be derived by Progol, unless it subsumes an example e relative to B in Plotkin's sense. This condition requires H to be a single clause, and this clause is used only once in the refutation of the example e. We define single-clause and multi-clause learning as follows in Definition 1. According to this definition, Yamamoto's example of learning odd-numbers is a multi-clause learning task. Because the hypothesis clause $odd(s(X)) \leftarrow even(X)$ is used twice when proving the positive example $odd(s(s(s(0))))$ with other clauses in B, therefore deriving such a clause is multi-clause learning even though the final hypothesis appear to be a single clause.

Definition 1. *Suppose N is the number of hypothesised clauses that appear in the refutation sequence for an example e. If $N = 1$, then it is single-clause learning; while it is multi-clause learning, if $N \geq 1$.*

For the learning problem shown in Fig. 1, to explain the whole set of examples E, Progol suggests a theory as $H_{incomplete} = \{h_1, h_2, h_3\}$, in which each single clause is derived from one example. However, a complete method will suggest a hypothesis $H_{complete} = \{h_4, h_5, h_6, h_7\}$[1], which has a smaller number of liter-

[1] At least two of the clauses in $H_{complete}$ are used in the refutation of each positive examples, thus $H_{complete}$ cannot be derived by an incomplete method.

Positive and Negative Examples E:

e_1:$s([an, unknown, alien, hits, the, house], [])$.

e_2:$s([a, small, boy, walks, a, dog], [])$.

e_3:$s([a, dog, walks, into, the, house], [])$.

e_4:$\neg s([dog, hits, a, boy], [])$.

Hypothesis language $\mathcal{L}$:

Predicates $=\{s, np, vp, det, noun, verb...\}$

Variables $=\{S_1, S_2, S_3, ...\}$

Constants $=\{a, the, ...\}$

Background Knowledge B:

b_1:$np(S1, S2) \leftarrow det(S1, S3), noun(S3, S2)$.

b_2:$vp(S1, S2) \leftarrow verb(S1, S2)$.

b_3:$vp(S1, S2) \leftarrow verb(S1, S3), prep(S3, S2)$.

b_4:$det([a|S], S)$. b_5:$det([an|S], S)$. b_{13}:$det([the|S], S)$.

b_6:$noun([dog|S], S)$. b_7:$noun([boy|S], S)$.

b_8:$noun([house|S], S)$. b_9:$noun([alien|S], S)$.

b_{10}:$verb([hits|S], S)$. b_{11}:$adj([small|S], S)$.

b_{12}:$prep([into|S], S)$.

Part of Hypothesis Space $\mathcal{H}$:

h_1:$s(S1, S2) \leftarrow det(S1, S3), S3 = [Word|S4], noun(S4, S5), vp(S5, S6), np(S6, S2)$.

h_2:$s(S1, S2) \leftarrow det(S1, S3), adj(S3, S4), noun(S4, S5), vp(S5, S6), np(S6, S2)$.

h_3:$s(S1, S2) \leftarrow np(S1, S3), S3 = [Word|S4], prep(S4, S5), np(S5, S2)$.

h_4:$s(S1, S2) \leftarrow np(S1, S3), vp(S3, S4), np(S4, S2)$.

h_5:$np(S1, S2) \leftarrow det(S1, S3), adj(S3, S4), noun(S4, S2)$

h_6:$verb([walks|S], S)$. h_7:$adj([unknown|S], S)$. h_8:$prep([unknown|S], S)$.

h_9:$np(S1, S2) \leftarrow det(S1, S3), prep(S3, S4), noun(S4, S2)$

Fig. 1: Grammar Learning Example

als while covers same number of examples, and therefore is more compressive than $H_{incomplete}$. This example shows the potential in real world for a complete method to improve learning results of an incomplete method.

3 MC-TopLog

A top theory $\top$ is the key part of a $\top$-directed method. The following subsections first introduce top theories, and then explain how to derive a hypothesis using a top theory, that is, the $\top$DTD algorithm. Finally, we explain the idea of how to restrict derivable hypotheses to common generalisations using $\top$DTcD.

3.1 Top theories as declarative bias

As a first-order generalisation of mode declarations, which specifies the declarative bias of a learning problem, a top theory is another way to represent the declarative bias. For example, the top theory in Fig. 2(b) corresponds to the mode declaration in Fig. 2(a). On the other hand, a top theory can encode a strong declarative bias that can not be captured by any mode declarations. For example, knowing that a noun phrase always consists of a noun and a verb phrase always has a verb provides a strong language bias, which not only tells what predicates are allowed in the head or body of clauses, but also tells how they are connected. This information can not be represented by a mode declaration; while it can be encoded by a top theory as in Fig. 2(c). Such a top theory will avoid deriving clauses like $np(S1, S3) \leftarrow det(S1, S2), adj(S2, S3)$, which defines a noun phrase without a noun. More details about top theories are in [9].

3.2 $\top$-directed Theory Derivation ($\top$DTD)

Before a hypothesis H is derived, an example e cannot be explained by B, because of the missing clauses to be hypothesised. However, with a top theory, explanations for e are still obtainable according to equation 1. By extracting the top theories used in the explanations and translating them into their corresponding hypothesis clauses based on equation 2, we can derive hypotheses that

$modeh(1, s(+wlist, -wlist))$
$modeh(*, np(+wlist, -wlist))$
$modeh(*, vp(+wlist, -wlist))$
$modeb(1, noun(+wlist, -wlist))$
$modeb(1, verb(+wlist, -wlist))$
$modeb(*, np(+wlist, -wlist))$
$modeb(*, vp(+wlist, -wlist))$
$modeb(1, det(+wlist, -wlist))$ ...

$modeh(1, det([const| + wlist], -wlist))$
$modeh(1, noun([const| + wlist], -wlist))$
$modeh(1, verb([const| + wlist], -wlist))$

(a) Mode Declaration

Th_s:	$s(X, Y) \leftarrow \$body(X, Y).$
Th_{np}:	$np(X, Y) \leftarrow \$body(X, Y).$
Th_{vp}:	$vp(X, Y) \leftarrow \$body(X, Y).$
Tb_{noun}:	$\$body(X, Z) \leftarrow noun(X, Y), \$body(Y, Z).$
Tb_{verb}:	$\$body(X, Z) \leftarrow verb(X, Y), \$body(Y, Z).$
Tb_{np}:	$\$body(X, Z) \leftarrow np(X, Y), \$body(Y, Z).$
Tb_{vp}:	$\$body(X, Z) \leftarrow vp(X, Y), \$body(Y, Z).$
Tb_{det}:	$\$body(X, Z) \leftarrow det(X, Y), \$body(Y, Z).$
T_{end}:	$\$body(Z, Z).$
Ta_{det}:	$det([X\|S], S).$
Ta_{noun}:	$noun([X\|S], S).$
Ta_{verb}:	$verb([X\|S], S).$ ...

(b) Top Theory: Weak Declarative Bias

Th_s:	$s(X, Y) \leftarrow \$body(X, Y).$
Th_{np_noun}:	$np(X, Y) \leftarrow \$body(X, M1), noun(M1, M2), \$body(M2, Y).$
Th_{vp_verb}:	$vp(X, Y) \leftarrow \$body(X, M1), verb(M1, M2), \$body(M2, Y).$
	... (The rest are same as that in Fig. 2(b))

(c) Top Theory: Strong Declarative Bias

Fig. 2: Declarative Bias of Grammar Learning

meet equation 3. The soundness and completeness of ⊤DTD are proved in [5].

$$B \wedge \top_H \models e \ (e \in E^+) \tag{1}$$

$$\top_H \models H \tag{2}$$

$$B \wedge H \ \models e \ (e \in E^+) \tag{3}$$

Example 1. For the learning task in Fig. 1, one of the explanations for e_1 using clauses in ⊤ and B is a SLD-refutation sequence $R = [\neg e_1, Th_s, Tb_{np}, Th_{np\_noun}, Tb_{det}, b_5, Tb_{prep}, Ta_{prep}(unknown), T_{end}, b_9, T_{end}, Tb_{vp}, b_2, b_{10}, Tb_{np}, b_1, b_{13}, b_8, T_{end}]$. Using the extraction algorithm explained in [5], three derivation sequences can be extracted from R. They are: $D_1 = [Th_s, Tb_{np}, Tb_{vp}, Tb_{np}T_{end}]$, $D_2 = [Th_{np\_noun}, Tb_{det}, Tb_{prep}, T_{end}, T_{end}]$ and $D_3 = [Ta_{prep}(unknown)]$. By applying SLD-derivation and subsumption to the derivation sequences, $H_1 = \{h_4, h_8, h_9\}$ can be derived.

3.3 ⊤-directed Theory Co-Derivation (⊤DTcD)

The design of ⊤DTcD is based on the fact that if a theory is common to multiple examples E, then refutation proofs of each example in E using that common theory will have the same structure, that is, the proofs are the same except the instantiations of variables. Those same-structure refutation proofs can be combined by combining corresponding arguments. It is the combined proof that forces the co-generalised examples to be proved using the same non-ground rules. The first step of ⊤DTcD is to combine examples to be generalised together into a compound goal. Then prove that compound goal using clauses in B and ⊤, which is similar to that in ⊤DTD.

The next question is how to choose the examples to be generalized together. Rather than randomly sample a pair of examples as that in ProGolem, ⊤DTcD start with all examples, while those do not fit are filtered out along the derivation of a refutation proof. At the end of a refutation, not only a hypothesis is derived, but also the maximum set of examples that can be explained by that hypothesis.

Example 2. For all the positive examples in Fig. 1, the $\top$DTcD method first combines them into a compound example like s([[an,unknown,alien,hits,the,house],[a, small,boy,walks,a,dog],[a,dog,walks,into,the,house]],[[],[],[]]), and then proves this compound example using clauses in B and $\top$. In this way, we can derive the hypothesis $H_2 = \{h_4, h_5, h_7\}$ that co-generalises examples e_1 and e_2. While e_3 is filtered out during the derivation of this hypothesis, since the second word 'dog' in e_3 is known to be a noun, rather than adjective, which does not fit into the compound proof that derives H_2. On the other hand, non-common generalisations are pruned. For example, the hypothesis $H_1 = \{h_4, h_8, h_9\}$ derived when generalising e_1 alone is no longer derivable because both e_2 and e_3 have their second words known as non-prepositions according to the given background knowledge.

4 Experiments

The null hypotheses to be empirically investigated in the study are: (a) compared to an incomplete method, a complete method does not derive hypotheses with higher predictive accuracies. (b) the search space of a co-generalisation method has the same size as that of a solo-generalisatoin method.

Materials MC-TopLog and Progol5 [7] are the two ILP systems used in the experiments. The background knowledge B for each learning task is generated by randomly removing certain number of clauses from the complete theory, and those left-out clauses form the corresponding target hypothesis. Examples are similar to those in Fig. 1. All materials used in the experiments can be found at http://ilp.doc.ic.ac.uk/mcTopLog.

Methods The null hypothesis(a) was investigated by comparing learning performances of MC-TopLog and Progol5[7] when a randomly chosen subset of the complete theory were left out. For each size of leave-out, we sampled ten times and the results were averaged. Leave-one-out cross validation was used to measure the predictive accuracies. The null hypothesis(b) was studied by comparing the search spaces of $\top$DTcD and $\top$DTcD. The search space is measured by the number of hypothesised theories generated for all positive examples.

Results Fig. 3 show that the predictive accuracies of MC-TopLog is higher than that of Progol. Their differences in accuracies increase when more clauses are left-out. While the predictive accuracies of $\top$DTcD are the same as that of $\top$DTD, so that the two lines overlap. Fig. 4 shows the improvement of $\top$DTcD over $\top$DTD in terms of efficiency. The solid line denotes the learning mode of singleEx which applies $\top$DTD to generalize a single example; while the dash line corresponds to the learning mode of multipleExs which applies $\top$DTcD to generalize multiple examples together. As shown in Fig. 4, the search space is reduced by more than half when the learning mode switches from singleEx to multipleExs. Therefore, the two null hypotheses are both refuted by the results.

5 Conclusions and Future work

The simplified version of grammar learning shows the importance of having a complete method, even for learning problems without recursion and mutually dependent predicates. Both $\top$DTD and $\top$DTcD are sound and complete for

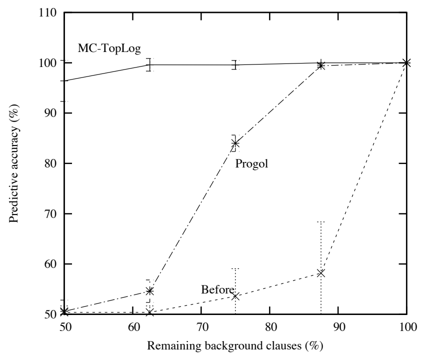

Fig. 3: Preditive Accuracies

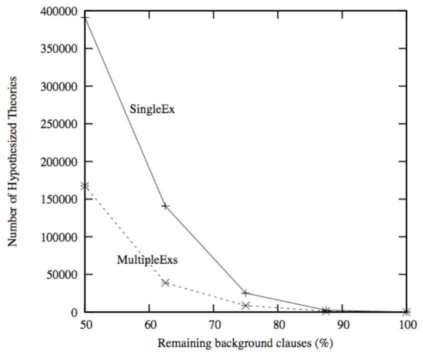

Fig. 4: Comparison of Search Spaces

deriving hypotheses, but ⊤DTcD is more efficient than ⊤DTD, while the improvement in efficiency is not a trade-off for its predictive accuracy. We intend to compare ⊤DTcD to other complete methods like CF-induction in future work. More experiments on different data sets will be given in the longer version paper.

References

1. D. Corapi, A. Russo, and E. Lupu. Inductive logic programming as abductive search. In *ICLP2010 Technical Communications*, Berlin, 2010. Springer-Verlag.
2. K Inoue. Induction as consequence finding. *Machine Learning*, 55:109–135, 2004.
3. T. Kimber, K. Broda, and A. Russo. Induction on failure: Learning connected Horn theories. In *LPNMR 2009*, pages 169–181, Berlin, 2009. Springer-Verlag.
4. D. Lin, J. Chen, H. Watanabe, S.H. Muggleton, and et al. Does multi-clause learning help in real-world applications? 2011. Submitted to ILP11.
5. Dianhuan Lin. Efficient, complete and declarative search in inductive logic programming. Master's thesis, Imperial College London, September 2009.
6. S.H. Muggleton. Inverse entailment and Progol. *New Generation Computing*, 13:245–286, 1995.
7. S.H. Muggleton and C.H. Bryant. Theory completion using inverse entailment. In *ILP-00*, pages 130–146, Berlin, 2000. Springer-Verlag.
8. S.H. Muggleton and C. Feng. Efficient induction of logic programs. In *ALT90*, pages 368–381, Tokyo, 1990. Ohmsha.
9. S.H. Muggleton, J. Santos, and A. Tamaddoni-Nezhad. Toplog: ILP using a logic program declarative bias. In *ICLP 2008*, pages 687–692, 2008.
10. S.H. Muggleton, J. Santos, and A. Tamaddoni-Nezhad. Progolem: a system based on relative minimal generalisation. In *ILP09*, pages 131–148. Springer-Verlag, 2010.
11. G.D. Plotkin. A note on inductive generalisation. In B. Meltzer and D. Michie, editors, *Machine Intelligence 5*, pages 153–163. Edinburgh University Press, 1969.
12. G.D. Plotkin. *Automatic Methods of Inductive Inference.* PhD thesis, Edinburgh University, August 1971.
13. Oliver Ray. Nonmonotonic abductive inductive learning. *Journal of Applied Logic*, 7(3):329–340, 2009.
14. J.C. Reynolds. Transformational systems and the algebraic structure of atomic formulas. In B. Meltzer and D. Michie, editors, *Machine Intelligence 5*, pages 135–151. Edinburgh University Press, Edinburgh, 1969.
15. A. Yamamoto. Which hypotheses can be found with inverse entailment? In N. Lavrač and S. Džeroski, editors, *ILP97*, pages 296–308. Springer-Verlag, 1997.