Explanation Based Learning For Planning

Subbarao Kambhampati (Arizona State University)
Sungwook Yoon (PARC Labs)

November 20, 2008

Synonyms: Explanation-based generalization for planning; Speedup learning for planning.

Definition Explanation Based Learning (EBL) involves using prior knowledge to explain ("prove") why the training example has the label it is given, and using this explanation to guide the learning. Since the explanations are often able to pinpoint the features of the example that justify its label, EBL techniques are able to get by with much fewer number of training examples. On the flip side, unlike general classification learners, EBL requires prior knowledge (aka "domain theory/model") in addition to labeled training examples—a requirement that is not easily met in some scenarios. Since many planning and problem solving agents do start with declarative domain theories (consisting at least descriptions of actions along with their preconditions and effects), EBL has been a popular learning technique for planning.

Dimensions of Variation: The application of EBL in planning varies along several dimensions: whether the learning was for improving the speed and quality of the underlying planner [16, 7, 16, 11, 10, 21] or acquire the domain model [14]; whether it was was done from successes [11, 21] or failures [16, 9]; whether the explanations were based on complete/correct [16, 10] or partial domain theories [21], whether learning is based on single [16, 10, 11] or multiple examples [6, 15] (where, in the latter case, inductive learning is used in conjunction with EBL) and finally whether the planner whose performance

EBL aims to improve is a means-ends analysis one [16], partial-order planner [6] or a heuristic search planner [21].

EBL techniques have been used in planning both to improve search and to reduce domain modeling burden (although the former has received more attention by far). In the former case, EBL is used to learn "control knowledge" to speedup the search process [16, 10], or to improve the quality of the solutions found by the search process [6]. In the latter case EBL is used to develop domain models (e.g. action models) [14].

EBL for search improvement involves either remembering and reusing successful plans, or learning search control rules to avoid failing search branches. Other variations include learning effective indexing of stored cases from retrieval failures [9] and learning "adjustments" to the default heuristic used by the underlying search.

Another important issue is the degree of completeness/correctness of the underlying background theory used to explain examples. If the theory is complete and correct, then learning is possible from a single example. This type of EBL has been called "analytical learning." When the theory is partia, EBL still is effective in narrowing down the set of potentially relevant features of the training example. These features can then be used within an inductive learner. Within planning, EBL has been used in the context of complete/correct as well as partial domain models.

A final dimension of variation that differentiated a large number of research efforts is the type of underlying planner. Initially, EBL was used on top of means-ends analysis planners (c.f. PRODIGY [16]). Later work focused on partial order planners (e.g. [10, 6]). More recently, the focus has been on forward search state-space planners [21].

Learning from Success: Explanation-based Generalization When learning from successful cases (plans), the training examples comprise of successful plans, and the explanations involve proofs showing that the plan, as it is given, is able to support the goals. Only the parts of the plan that take part in this proof are relevant for justifying the success of the plan. The plan is thus "generalized" by removing extraneous actions that do not take part in the proof. Object identifiers and action orderings are also generalized as long as the generalization doesn't affect the proof of correctness [11]. The output of the learning phase is thus a variablized plan containing a subset of

the constraints (actions, orderings, object identity constraints) of the original plan. This is then typically indexed and used as a macro-operator to speed-up later search.

For example, given a planning problem of starting with an initial state where five blocks, A, B, C, D and E are on table, and the problem requires that in the goal state A must be on B and C must be on D, and a plan P that is a sequence of actions *pickup A, stack A on B, pickup E, putdown E, Pickup C, stack C on D*, the explanation-based learner might output the generalization "*do in any order* { *pickup x, stack x on y* } {*pick up z, stack z on w*}" for the generalized goals *on(x,y) and on(w,z)*, starting from a state where x, y, z and w are all on table and clear, and each of them denotes a distinct block.

One general class of such proof schema involves showing that every top level goal of the planning problem as well as the precondition of every action are established and protected. Establishment requires that there is an action in the plan that gives that condition, and protection requires that once established, the condition is not deleted by any intervening action.

A crucial point is that the extent of generalization depends on the flexibility of the proof strategy used. Kambhampati & Kedar [11]) discuss a spectrum of generalization strategies associated with a spectrum of proof strategies, while Shavlik [20] discusses how the number of actions in the plan can also be generalized.

Learning from Failure When learning from the failure of a search branch, EBL starts by analyzing the plans at the failing nodes and constructing an explanation of failure. The failure explanation is just a subset of constraints in the plan at the current search node, which, in conjunction with domain theory ensures that no successful solution can be reached by further refining this plan. The explanations can range from direct constraint inconsistencies (e.g. ordering cycles), to indirect violation of domain axioms (e.g. the plan requiring both clear(B) and On(A,B) to be satisfied at the same time point). The explanations at the leaf nodes are "regressed" over the decisions in the search tree to higher level nodes to get explanations of (implicit) failures in these higher level nodes. The search control rules can then essentially recommend pruning any search node which satisfies a failure explanation.

The deep affinity between EBL from search failures and the idea of no-

good learning and dependency-directed backtracking in CSP is explored in [12]. As in dependency directed backtracking, the more succinct the explanation, the higher the chance of learning effective control rules. Note that effectiveness here is defined in terms of the match costs involved in checking whether the rule is applicable, and the search reductions provided when it is applicable. Significant work has been done to identify classes of failure explanation that are expected to lead to ineffective rules [7]. In contrast to CSP that has a finite depth search tree, one challenge in planning is that often an unpromising search node might not exhibit any direct failure with a succinct explanation, and is abandoned by the search for heuristic reasons (such as the fact that the node crosses a depth limit threshold). Strategies for finding implicit explanations of failure (using domain axioms), as well as getting by with incomplete explanations of failure are discussed in [10]. EBL from failures has also been applied to retrieval (rather than search) failures. In this case, the failure of extending a plan retrieved from the library to solve a new problem is used to learn new indexing schemes that inhibit that case from being retrieved in such situations [9].

Learning adjustments to Heuristics: Most recent work in planning has been in the context of heuristic search planners, where learning from failures doesn't work as well (since the heuristic search may change directions much before a given search branch ends in an explainable failure). One way of helping such planners is to improve their default heuristic [21]. Given a heuristic $h(s)$ that gives the heuristic estimate of state s, the aim in [21] is to learn an adjustment $\delta(s)$ that is added to $h(s)$ to get a getter estimate of $h^*(s)$–the true cost of state s. The system has access to actual plan traces (which can be obtained by having the underlying planner solve some problems from scratch). For each state s on the trace, we know the true distance of state s from the goal state, and we can also compute the $h(s)$ value with respect to the default heuristic. This gives the learner a set of training examples which are pairs of states and the adjustments they needed to make to the default heuristic meet the true distance. In order to learn the $\delta(s)$ from this training data, we need to enumerate the features of state s that are relevant to it needing the specific adjustment. This is where EBL come in. Specifically, one way of enumerating the relevant features is to explain why s has the default heuristic value it does. This, in turn, is done by taking

the features of the relaxed plan for state s. Since the relaxed plan is a plan that assumes away all negative interactions between the actions, relaxed plan features can be seen as features of the explanation of the label for state s in terms of a *partial domain theory* (one which ignores all the deletes of all actions).

EBL from Incomplete Domain Theories: While most early efforts for speed-up focused on complete and correct theories, several efforts also looked at speed-up learning from incomplete theories. The so-called Lazy EBL approaches [22, 3] work by first constructing partial explanations, and subsequently refine the over-general rules learned. Other approaches that use similar ideas outside planning include [15, 4]. As we noted above, the work by Yoon *et. al.* [21] can also be seen as basing learning (in their case of adjustments to a default heuristic function) w.r.t. a partial domain theory.

EBL to learn domain knowledge: Although most work in EBL for planning has been focused on speedup, there has also been some work aimed at learning domain knowledge (rather than control knowledge). Of particular interest is "operationalizing" a complex, if opaque, domain model by learning from it a simplified domain model that is adequate to efficiently solve an expected distribution of problems. The recent work by Levine and DeJong [14] is an example of such an effort.

EBL and Knowledge-level Learning: Although the focus of this article is on EBL as applied to planning, we need to foreground one general issue: whether EBL is capable of knowledge-level learning or not. A popular misconception of EBL is that since it depends on a complete and correct domain theory, no knowledge-level learning is possible, and speedup learning is the only possibility.[1] As we noted at the outset however, EBL is not required

[1]The origins of this misconception can be traced back to the very beginning. The two seminal articles on EBL in the very first issue of the Machine Learning journal differed profoundly in their interpretations of EBL. While Mitchell *et. al.* [19] assumed that EBL by default works with complete and correct theories (thus precluding any knowledge-level learning), DeJong *et. al* [14] provide a more general view of EBL that uses background knowledge–whether or not it is complete–to focus the generalization (and as such can be seen as a knowledge-based feature-selection step for a subsequent inductive learner).

to depend on complete and correct domain theories, and when it doesn't, knowledge level learning is indeed possible.

Utility Problem & its non-exclusive relation to EBL: As we saw above, much early work in EBL for planning focused on speed-up for the underlying planner. Some of the knowledge learned for speedup–especially control rules and macro-operators–can also adversely affect the search by increasing either the search space size (macros) and/or per-node cost (matching control rules). Clearly, in order for the net effect to be positive, care needs to be exercised as to which control rules and/or macros are stored. This has been called the "utility problem" [18] and significant attention has been paid to develop strategies that either dynamically evaluate the utility of the learned control knowledge (and forget useless rules) [17, 18], or select the set of rules that best serve a given distribution of problem instances [8].

Despite the prominent attention given to the utility problem, it is important to note the non-exclusive connection between EBL and utility problem We note that *any* strategy that aims to provide/acquire control knowledge will suffer from the utility problem. For example, utility problem also holds for inductive learning techniques that were used to learn control knowledge (c.f. [13]). In other words, it is not special to EBL but rather to the specific application task. We note that it is both possible to do speedup learning that is less suceptible to the utility problem (e.g. learn adjustments to heuristics [21]), and possible to to use EBL for knowledge-level learning [14].

Current Status: EBL for planning was very much in vogue in late eighties and early nineties. However, as the speed of the underlying planners increased drastically, the need for learning as a crutch to improve search efficiency reduced. There has however been a recent resurgence of interest, both in further speeding up the planners, and in learning domain models. Starting 2008, there is a new track in the International Planning Competition devoted to learning methods for planning. In the first year, the emphasis was on speedup learning. ObtuseWedge, a system that uses EBL analysis to learn adjustments to the default heuristic, was among the winners of the track. The DARPA integrated learning initiative, and interest in model-lite planning have also brought focus back to EBL for planning–this time with partial domain theories.

Additional Reading: The tutorial [23] provides an up-to-date and broader overview of learning techniques applied to planning, and contains significant discussion of EBL techniques. The paper [24] provides a survey of machine learning techniques used in planning, and includes a more comprehensive listing of research efforts that applied EBL in planning.

See Also: Speedup Learning, explanation-based learning.

References

[1] Neeraj Bhatnagar, Jack Mostow: On-Line Learning from Search Failures. Machine Learning 15(1): 69-117 (1994)

[2] Daniel Borrajo, Manuela M. Veloso: Lazy Incremental Learning of Control Knowledge for Efficiently Obtaining Quality Plans. Artif. Intell. Rev. 11(1-5): 371-405 (1997)

[3] Steve A. Chien: Using and Refining Simplifications: Explanation-Based Learning of Plans in Intractable Domains. IJCAI 1989: 590-595

[4] William W. Cohen: Abductive Explanation-Based Learning: A Solution to the Multiple Inconsistent Explanation Problem. Machine Learning 8: 167-219 (1992)

[5] Gerald DeJong, Raymond J. Mooney: Explanation-Based Learning: An Alternative View. Machine Learning 1(2): 145-176 (1986)

[6] Tara A. Estlin, Raymond J. Mooney: Learning to Improve both Efficiency and Quality of Planning. IJCAI 1997: 1227-1233

[7] Oren Etzioni: A Structural Theory of Explanation-Based Learning. Artif. Intell. 60(1): 93-139 (1993)

[8] Jonathan Gratch, Steve A. Chien, Gerald DeJong: Improving Learning Performance Through Rational Resource Allocation. AAAI 1994: 576-581

[9] Laurie H. Ihrig, Subbarao Kambhampati: Storing and Indexing Plan Derivations through Explanation-based Analysis of Retrieval Failures. J. Artif. Intell. Res. (JAIR) 7: 161-198 (1997)

[10] Subbarao Kambhampati, Suresh Katukam, Yong Qu: Failure Driven Dynamic Search Control for Partial Order Planners: An Explanation Based Approach. Artif. Intell. 88(1-2): 253-315 (1996)

[11] Subbarao Kambhampati: A Unified Framework for Explanation-Based Generalization of Partially Ordered and Partially Instantiated Plans. Artif. Intell. 67(1): 29-70 (1994)

[12] Subbarao Kambhampati: On the Relations Between Intelligent Backtracking and Failure-Driven Explanation-Based Learning in Constraint Satisfaction and Planning. Artif. Intell. 105(1-2): 161-208 (1998)

[13] Christopher Leckie, Ingrid Zukerman: An Inductive Approach to Learning Search Control Rules for Planning. IJCAI 1993: 1100-1105

[14] Geoffrey Levine, Gerald DeJong: Explanation-Based Acquisition of Planning Operators. ICAPS 2006: 152-161

[15] Nicholas S. Flann, Thomas G. Dietterich: A Study of Explanation-Based Methods for Inductive Learning. Machine Learning 4: 187-226 (1989)

[16] Steven Minton, Jaime G. Carbonell, Craig A. Knoblock, Daniel Kuokka, Oren Etzioni, Yolanda Gil: Explanation-Based Learning: A Problem Solving Perspective. Artif. Intell. 40(1-3): 63-118 (1989)

[17] Shaul Markovitch, Paul D. Scott: The Role of Forgetting in Learning. ML 1988: 459-465

[18] Steven Minton: Quantitative Results Concerning the Utility of Explanation-based Learning. Artif. Intell. 42(2-3): 363-391 (1990)

[19] Tom M. Mitchell, Richard M. Keller, Smadar T. Kedar-Cabelli: Explanation-Based Generalization: A Unifying View. Machine Learning 1(1): 47-80 (1986).

[20] Jude W. Shavlik: Acquiring Recursive and Iterative Concepts with Explanation-Based Learning. Machine Learning 5: 39-40 (1990)

[21] Sungwook Yoon, Alan Fern and Robert Givan Learning Control Knowledge For Forward Search Planning JMLR, 9 (APR), 683–718, 2008

[22] Prasad Tadepalli: Lazy ExplanationBased Learning: A Solution to the Intractable Theory Problem. IJCAI 1989: 694-700

[23] Sungwook Yoon and Subbarao Kambhampati. Learning for Planning. Tutorial delivered at ICAPS 2007. *http://rakaposhi.eas.asu.edu/learn-plan.html*

[24] Terry Zimmerman, Subbarao Kambhampati: Learning-Assisted Automated Planning: Looking Back, Taking Stock, Going Forward. AI Magazine 24(2): 73-96 (2003)