Machine Learning, 6, 197-204 (1991)

Book Review

Learning Search Control Knowledge: An Explanation-Based Approach, by Steven Minton. Kluwer Academic Publishers, 1988, 214 pages.

ROLAND J. ZITO-WOLF
Computer Science Department, Brandeis University, Waltham, MA 02254

1. Overview

Learning Search Control Knowledge is a detailed examination of the learning component of the PRODIGY problem-solving system. PRODIGY uses Explanation-Based Learning (EBL) to distill effective search-control knowledge from experience. This work significantly extends the application of EBL methods; moreover, its methodology and coverage make it an excellent example of empirical research. Several significant results are presented:

- It gives a definition of EBL as computing the weakest preconditions of an explanation. An algorithm, Explanation-Based Sepcialization (EBS), is presented and proved correct.
- The application of EBL is extended to meta-level concepts, concepts acquired across multiple examples, and learning from difficulty and failure as well as success.
- It introduces a definition of "operationality" based on the utility of a learned rule, and gives methods for evaluating it dynamically.

Minton's book methodically addresses each aspect of his research. The initial chapters present the research framework—the methods and terminology of EBL, the design of the PRODIGY system, and the concepts of operationality and utility. Each stage of PRODIGY's learning process—EBS to generate candidate rules, *compression* to reduce rule evaluation cost, and *utility evaluation* to identify effective rules—is discussed separately. The discussion focuses on the practical issues involved in implementing an EBL-based system, such as the selection of training examples and the cost/benefit tradeoffs in learning. The next three chapters show how these techniques are applied. A chapter is devoted to each type of meta-concept PRODIGY currently learns: success, failure, and goal interactions. Then follows a thorough empirical analysis of PRODIGY's learning performance, involving several hundred problems and multiple domains. The book concludes with a formal analysis of the EBS algorithm and its relation to other EBL methods, and an in-depth discussion of related work in EBL and machine learning. The book reproduces the bulk of the author's thesis, omitting only certain appendices. We first review the main points of the book with commentary, after which we discuss the larger context of the work.

2. Framework

PRODIGY is a STRIPS-like (Fikes & Nilsson, 1971) domain-independent problem solver based on means-end analysis. PRODIGY has a universal subgoaling structure (cf. SOAR (Laird et al., 1986)) such that all decisions it makes can be explicitly reasoned about. There are four decision types: which *node* of the search space to expand next, which *goal* to pursue at the curent node, which *operator* to use to achieve the chosen goal, and what *bindings* to use in instantiating that operator. Control rules fall into 3 classes—*selection*, *rejection* and *preference* of alternatives—yielding 12 types of rule in all. PRODIGY makes novel application of EBL to generate new rules for proposing and ordering choices.

In EBL, a training example is explained as an instance of a specified *target concept* and then generalized using that explanation as a guide. The result is a generalization of the training example that is logically consistent with the target concept. A complete and correct domain theory is required to support explanation. PRODIGY's version of EBL is called Explanation-Based Specialization. EBS uses the fact that the result of EBL is both a generalization (of the example) and a specialization (of the target concept); it specializes the target concept directly rather than generalizing the proof of the example.[1] The target is specialized by rewriting selected subexpressions in the same way they were expanded in the derivation of the example. The usage of the terms "generalization" and "specialization" can be confusing because it refers to a difference in method rather than result. Later in the book EBL and EBS are shown to produce equivalent results; however, EBS is easier to understand and implement. EBS makes clear that the resulting formula is correct: a strict specialization of the target concept.

EBL is usually viewed as creating a specialized version of the target concept satisfying some *operationality criterion*, meaning it is easier or more efficient to use. In practice, operationality is difficult to define; it is typically taken to mean "expressed in terms of primitive (and, by implication, efficient) predicates." Minton argues that such a syntactic criterion for operationality is inadequate to guarantee that learning will improve overall performance because it does not consider the cost of executing the learned rules. He proposes that operationality be identified with empirical *utility*, the (average) difference between the search effort saved by a rule and the cost of utilizing it.

3. Method

Chapters 4–6 describe PRODIGY's learning process. First, an OBSERVER module selects an appropriate example from a problem-solving trace. PRODIGY has four classes of target concepts: *successful choice*; *failing choice*; *goal interference*, where a choice undoes a previously satisfied goal or precondition; and *sole-alternative*, a decision point where all choices fail but one. Crossing these with the four decision types yields 16 possible meta-concepts to learn. Heuristics are associated with each concept type to control example selection, that is, to limit the number of examples processed. EBS is applied to each pair of example and target concept. The resulting rules are typically verbose; compression by partial evaluation and domain-dependent transformations substantially reduces their evaluation cost.

Each generated rule is then monitored during problem-solving. Those whose evaluation cost exceeds their estimated savings are discarded. Typically only about 20% of the rules generated are retained.

The next three chapters give examples of rule acquisition via each class of meta-concept. Generalizing a successful choice creates a *preference* rule that appraises choices. Explaining a failed choice generates a *rejection* rule to eliminate choices destined to fail. Goal-interactions generate preference rules rather than rejection rules since some goal interactions are unavoidable. Sole-alternatives generate *selection* rules that identify good choices. Domain-independent selection heuristics for each concept class identify salient examples, reject examples subsumed by others, and perform other filtering.

The reliance on selection heuristics and domain-dependent transformations are potential limitations of this technique. The selection heuristics are domain-independent; what guarantees that adequate examples will be selected? The rule transformations, on the other hand, represent domain-dependent knowledge that is required for effective learning, as the performance data (Chapter 10) clearly show. Inadequacies in selection and compression will lead to redundant, over-specific, or inefficiently expressed rules that reduce problem-solving efficiency. The utility monitor compensates somewhat by filtering out inefficient or redundant rules; unfortunately, ineffective rules can confuse the utility monitor by diluting the savings attributed to desirable rules, causing it to rate rules incorrectly. PRODIGY's dependence on domain-specific heuristics is not quantified.

4. Performance

Minton gives performance results for 3 domains—block-stacking, a STRIPS-like robot-and-rooms domain, and job-shop scheduling. Problems were generated randomly, using variable complexity parameters. In each domain, there was a learning phase using selected training problems, a settling phase for utility evaluation, and a testing phase using problems of gradually increasing complexity. EBS was used only in the first phase. PRODIGY achieved a consistent speedup by a factor of 2–3 across all problems using learned rules, close to that achieved by hand-coded rules. Comparison with macro-based learning showed EBS superior in two of the three domains, which is offered as evidence that directed learning is superior to macro caching. Examination of the contributions of compression and utility-evaluation shows that both are critical to PRODIGY's performance.

Interestingly, most useful rules were derived from failed choices or goal interactions rather than successful choices. This suggests that learning from success, the mainstay of EBL and macro-based methods, is inherently less effective than learning from failure, a position espoused for example by Schank (1982). However, macro-based learning systems have also achieved significant results (Korf, 1985; Iba, 1989). The discrepancy might be attributable to certain implementation limitations in PRODIGY's learning from success; for example, the example-selection heuristics used allowed learning only from *global* successes, whereas failure examples were less restricted. The distribution of rule types before utility evaluation is not given, so one cannot tell whether learning from success is less *effective* (poorer rules) or merely less *productive* (fewer candidate rules). Since rules compete to survive, it would be informative to examine the system's performance when restricted to a single class of learning—success, failure, or interactions.

Recent work by Etzioni (1990) offers an explanation. It is difficult to generate effective control rules from recursive examples, and explanations based on successes tend to be recursive, while explanations of failure do not. The observed effect is due to the *specific* problem domains used. The text attempts to characterize the domains for which EBS is appropriate, but the characterization given is not particularly operational. Etzioni's analysis shows that EBL is most appropriate in problem spaces that yield non-recursive explanations.

Learning control rules and learning macro-operators should be regarded as complementary, rather than competing, alternatives (Iba, personal communication). The latter tell the system what it *can* do, while the former tell the system what it *should* do. Each provides a way of reducing search—control rules by reducing the search *width*, and macro-learning by providing composed operators that reduce the search *depth*. In a framework like PRODIGY's (or SOAR's, for that matter) macros are at a disadvantage because they must be expressed as sequences of preferences. This introduces extra decision points between each step with choice criteria that are difficult to characterize in primitive terms—that is, without reference to the fact that one is "currently executing a macro."

The rationale for learning in the first place is the exponential cost of uninformed search. Control rules are effective if they reduce the branching factor at each choice point. This should yield an exponential performance improvement, easily distinguishable from constant factor improvements. The performance data presented demonstrate a significant and consistent speedup due to learning, but the expected exponential relationship between problem size and cost is not clearly demonstrated (Figures 10-3 and 10-4). Yet specific example problems in the text do exhibit exponential behavior. Is this due to the testing method, the presentation of the data, or PRODIGY itself? It makes it difficult to extrapolate from the data, as constant factors related to the implementation—compression efficiency and rule evaluation, for example—are harder to factor out. PRODIGY's domains to date are simple micro-worlds. It would be interesting to see if PRODIGY could be harnessed to a more complex domain (e.g., the 8 or 15 puzzle) where the exponential relationships would be more clearly evident.

5. Theory and related work

The theoretical analysis appears in Chapter 11, wherein EBS is shown to be correct and to subsume other EBL methods. As previously explained, EBS views EBL as a method for concept specialization, reversing the usual conception of EBL as generalization. This makes clear that the result of EBL is a strict specialization of the target concept. One seeks the most general such formula, one with the *weakest preconditions*, subject to the requirement of expression in operational terms. Proofs are outlined that EBS does precisely this. The proofs are not complex, though they are complicated by confusing terminology and some typographical errors. A complete proof is given in Minton (1988).

Minton's reconstruction of EBL is important because there is much dispute over precisely what generalizations EBL algorithms make. He shows that, given that one's generalization of the training example is required to be correct with respect to the domain theory, so that "explanation" becomes synonymous with "derivation," EBS generates the most inclusive generalization that retains the derivation structure of the training example. This establishes

a strict upper bound on what an explanation-based learner may produce. Since EBS achieves this bound, EBS subsumes all EBL methods that guarantee correctness. EBS has the additional advantage that specialization is an inherently simpler process. Minton argues that even if external constraints lead one to desire *less* general results than EBS produces, the process is still best conceived of as specialization.

Recasting EBL as specialization of the target formula exposes the close relationship of EBS (and other EBL techniques using proof as explanation) to logic program optimization by partial evaluation (Bjorner et al., 1988). This equivalence has been made precise (Kedar-Cabelli & McCarty, 1987; van Harmelan & Bundy, 1988), a result that is not adequately appreciated in the literature. There is a good deal yet to be said about this relationship. For example, logic research is highly conscious of the difficulties introduced by operators like NOT, FOR-ALL, and EXISTS. Such operators render the database non-monotonic, such that new information can reduce as well as expand the space of valid derivations. The EBS algorithm in fact treats these predicates specially—expressions inside FOR-ALL and EXISTS are expanded but the FOR-ALL and EXISTS predicates themselves are not specialized, while for NOT neither the predicate *nor* the contained expression is specialized. This fact is presented without explanation. Logic-program optimization research is more advanced in the treatment of complex terms, recursively-defined predicates, constraints, and selective expansion of subexpressions (Smith & Hickey, 1990). Conversely, techniques developed for EBL may be applicable to logic programming, as suggested in Harmelan & Bundy (1988). PRODIGY's methods of dynamic utility evaluation and the notion of directing partial evaluation by examples (Prieditis & Mostow, 1987) could be of relevance.

Chapter 12 discusses related work. It points out that EBL, STRIPS' MACROPS, macro-learning, and chunking in production-systems (e.g., SOAR) are all fundamentally methods of restating existing knowledge based on the problem-solver's experience. They share the limitation of conceiving of learning as "performance improvement" as opposed to "knowledge acquisition." In PRODIGY's case, the scope of the learned rules is broadened by its access to a theory of both the domain and the problem-solver itself. The problem-solver's behavior is conducive to axiomatization since it is a designed artifact, making meta-level learning an especially good choice of domain for EBL. PRODIGY demonstrates that this can produce significant improvements in performance. Yet this kind of learning is inherently unable to absorb knowledge that is "new," not implied by existing knowledge, such as inductive generalizations. Minton's rigorous characterization of EBS makes it easy to see that EBL at the meta-level does not escape this limitation; the search control knowledge it acquires must always be justified by the theory of the domain and problem-solver.

6. Discussion

PRODIGY convincingly demonstrates the advantages of knowledge-directed learning. Knowledge allows it to learn in a natural way from failure and self-observation as well as from success. Explanations distinguish independent aspects of a problem-solving episode, which is important when acquiring multiple rules from a single example. PRODIGY thus requires fewer examples and is less dependent on the structure of the training set. Within the EBL and machine-learning community, the improved characterization of EBL via EBS

should lead to a better understanding of the scope of EBL and further its theoretical development vis-a-vis work in logic programming and optimization. PRODIGY provides a foundation for further learning research, such as the work of Etzioni (1990) on problem-space structure and of Carbonell and Gil (1987) on learning via experimentation.

A topic deserving further discussion is learning bias. There are two main sources of bias here. First, EBL is underconstrained: every expansion of the target concept leading to a derivation of the example qualifies as a generalization of that example. Whatever basis the system uses to choose among these constitutes an implicit bias toward certain forms of expression. For example, the method described in Mitchell et al. (1986) expands all non-primitive predicates, while DeJong and Mooney (1986) suggest expanding only those predicates where the specific expansion is necessary to the derivation as a whole. Consider PRODIGY's idiosyncratic specialization schemas for the predicates NOT, FOR-ALL, and EXISTS. The effect on the resulting rules deserves discussion; the data provided are equivocal because the examples given contain few occurrences of these predicates. This form of bias remains a significant open issue for EBL: at what level of generality is a learned concept most usefully expressed? In the classic SAFE-TO-STACK example (see Mitchell et al. (1986)), is it best to specialize the WEIGHT predicate or leave it to be derived dynamically? Little evidence has been presented that expansion to primitives whenever possible is an optimal strategy. PRODIGY addresses this issue to an extent by merging rules with the same left-hand side (conclusion), converting multiple specific rules to a single more general one.

A second source of bias is the training data. Bias was deliberately introduced into PRODIGY's training sets (p. 147) because:

> [The] learning process was sensitive to both the ordering and choice of training problems. Therefore... the sequences of training problems were biased by gradually increasing their difficulty. The sensitivity to problem difficulty arises because the system typically generates better explanations (i.e., higher in utility) for simpler problems.

Empirical investigation of the effect of training bias would have been welcome. This issue is significant because one will not always have available an optimal training set. There may be few examples, the examples will not necessarily include simple cases, and learning will almost certainly continue after the training period ends. While PRODIGY can in theory learn multiple concepts from a single problem-solving episode, we are told that extended or otherwise complex examples cause it to learn overly specific, hence ineffective rules. The most useful rules are generated from the simplest examples.

Let us step back and examine the framework within which the research was conducted. EBS is the culmination of an approach advocated in Mitchell et al. (1986). It attempted to formalize a model of learning where explanation would separate the important from the incidental; in the process explanations were identified with proofs. Minton takes the additional step toward formalization of replacing the elusive notion of operationality with the measurable one of utility. EBS makes clear both the scope and the limitations of this approach.

The notion of learning that is explanation-based is important, but the identification of "explanation" with "proof" is restrictive. While it bounds the problem and simplifies the implementation, it significantly limits what is learnable. It denies the system whole areas

of knowledge that human problem-solvers routinely draw on, and for that matter learn largely *from experience*: knowledge of which parts of an explanation are worth specializing and which not; heuristic knowledge about the consequences of actions; inductive knowledge about categories in the world; and new representations with which to characterize the world (e.g., ABOVE as a generalized ON-TOP-OF). The original notion of operationality similarly sought to capture certain ideas—appropriateness of expression, conciseness, expression at an appropriate level of generality—which utility measures only incidentally.

PRODIGY hews to a model of planning that aspires to internalize a complete, correct, and predictive world model. Many domains, and most interactions with the world at large, involve a more complex relationship between the learner and the world. They are characterized by unbounded amounts of potentially relevant knowledge, a partial or intractable domain theory, and uncertainty and unpredictability of events. Several recent lines of research address this problem (e.g., *case-based reasoning*, Kolodner & Simpson (1989); *adaptive planning*, Alterman (1988); *routines*, Agre (1988); *situated action*, Suchman (1987)). They focus on the need to make decisions in spite of inadequate or inconsistent knowledge—a concomitant of an understanding of the world that is always evolving. In such domains the requirement for a complete and correct domain theory to support explanation is untenable. There is more to understanding than explanation; there is also accommodation and interpretation. Goals are elaborated through an interactive process of accommodation, during which those same goals guide interpretation of what the agent encounters (Alterman & Zito-Wolf, 1990). These issues fit poorly into PRODIGY's framework.

Further research is necessary. These approaches need not be incompatible, if a way can be found to decouple the notion of "explanation" from that of "proof." DeJong and Mooney (1986) understood this problem when they suggested an alternative view of EBL based on matching to schemas rather than logical derivation, and a contextual rather than static notion of operationality. The work of Pazzani (1988) shows that inductive learning can form a basis for EBL. The case-based planner CHEF (Hammond, 1986) shows that EBL can be used for understanding and classifying cases. The heuristic aspects of human reasoning (Kahneman et al., 1982) suggest that what we truly need is a bridge between explanation and induction.

7. Summary

Learning Search Control Knowledge is a well-written and thorough discussion of both the nature of explanation-based learning and its application to problem-solving. The discussion is replete with detail, stocked with examples and well-supported by empirical evidence. It is recommended both for its comprehensive overview and comparison of EBL-based efforts and for its detailed discussion of EBL implementation issues and techniques. This is the clearest and most understandable description of EBL theory and implementation which I have encountered. Newcomers will welcome its overview of EBL and its literature; practitioners will find the discussion of the learning system refreshingly clear. The result is a well-rounded discussion with something to offer to everyone concerned with machine learning.

Acknowledgments

Many thanks to Glenn Iba for discussions on macro-learning, to Don Smith and Tim Hickey for extensive help with partial evaluation, and to Rick Alterman for suggesting this project and for critical reading. Pat Langley, Steve Minton and Oren Etzioni provided valuable comments. Any unreasonable opinions are, of course, solely my responsibility.

Notes

1. The complete description of EBS is spread out over several chapters, especially Specialization, Proofs, and Related Work; it may be helpful to skim these before reading the book in detail.

References

Agre, P.E. (1988). *The dynamic structure of everyday life* (Technical Report AI-TR 1085). Cambridge, MA: MIT Artificial Intelligence Laboratory.

Alterman, R. (1988). Adaptive planning. *Cognitive Science Journal, 12*, 393–421.

Alterman, R., & Zito-Wolf, R. (1990). Planning and understanding: Revisited. *Proceedings of 1990 AAAI Spring Symposium*.

Bjorner, D., Ershov, A.P., & Jones, N.D. (1988). *Partial evaluation and mixed computation*. North-Holland.

Carbonell, J.G., & Gil, Y. (1987). Learning by experimentation. *Proceedings of the Fourth International Workshop on Machine Learning*. Cambridge, MA: Morgan Kaufmann.

DeJong, G., & Mooney, R. (1986). Explanation-based learning: An alternative view. *Machine Learning, 1*, 145–176.

Etzioni, O. (1990). Why PRODIGY/EBL works. *Proceedings of the Eighth National Conference on Artificial Intelligence* (pp. 916–922).

Fikes, R.E., & Nilsson, N.J. (1971). STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence 2*, 189–208.

Hammond, K.J. (1986). CHEF: A model of case-based planning. *Proceedings of AAAI-86* (pp. 267–271).

Iba, G.A. (1989). A heuristic approach to the discovery of macro-operators. *Machine Learning, 3*, 285–317.

Kahneman, D., Slovic, P., & Tversky, A. (1982). *Judgement under uncertainty: Heuristics and biases.* Cambridge University Press.

Kedar-Cabelli, S.T., & McCarty, L.T. (1987). Explanation-based generalization as resolution theorem-proving. *Proceedings of the Fourth International Workshop on Machine Learning* (pp. 383–389). Cambridge, MA: Morgan Kaufmann.

Kolodner, J.L., & Simpson, R.L. (1989). The MEDIATOR: Analysis of an early case-based problem solver. *Cognitive Science, 13*, 507–549.

Korf, R.E. (1985). Macro-operators: A weak method for learning. *Artificial Intelligence, 26*, 35–77.

Laird, J.E., Rosenbloom, P., & Newell, A. (1986). Chunking in SOAR: The anatomy of a general learning mechanism. *Machine Learning, 1*, 11–46.

Minton, S. (1988). *Learning search control knowledge: An explanation-based approach* (Technical Report CMU-CS-88-133). Pittsburgh, PA: Carnegie-Mellon University.

Mitchell, T., Keller, R., & Kedar-Cabelli, S. (1986). Explanation-based generalization: A unifying view. *Machine Learning, 1*, 47–80.

Pazzani, M.J. (1988). *Learning causal relationships: An integration of empirical and explanation-based learning methods* (Technical Report UCLA-AI-88-10). Los Angeles, CA: University of California, Los Angeles.

Prieditis, A.E., & Mostow, J. (1987). PROLEARN: Toward a Prolog interpreter that learns. *Proceedings of AAAI-87* (pp. 494–498).

Schank, R. (1982). *Dynamic memory*, Cambridge University Press.

Smith, D.A., & Hickey, T. (1990). *Partial evaluation of CLP(FT)* (Technical Report CS-90-148). Waltham, MA: Brandeis University.

Suchman, L.A. (1987). *Plans and situated actions*. Cambridge University Press.

van Harmelan, F., & Bundy, A. (1988). Explanation-based generalisation = Partial evaluation. *Artificial Intelligence, 36*, 401–412.