Approximate Theory Formation: An Explanation-Based Approach

Thomas Ellman

27 May 1988
CUCS-347-88

Approximate Theory Formation: An Explanation-Based Approach

Thomas Ellman[1]
Department of Computer Science
Columbia University
New York, New York 10027
ellman@cs.columbia.edu

27 May 1988

Abstract

Existing machine learning techniques have only limited capabilities of handling computationally intractable domains. This research extends explanation-based learning techniques in order to overcome such limitations. It is based on a strategy of sacrificing theory accuracy in order to gain tractability. Intractable theories are approximated by incorporating simplifying assumptions. Explanations of teacher-provided examples are used to guide a search for accurate approximate theories. The paper begins with an overview of this learning technique. Then a typology of simplifying assumptions is presented along with a technique for representing such assumptions in terms of generic functions. Methods for generating and searching a space of approximate theories are discussed. Empirical results from a testbed domain are presented. Finally, some implications of this research for the field of explanation-based learning are also discussed.

1 Introduction

Current machine learning techniques face considerable difficulties when dealing with intractable domains. Standard explanation-based learning (EBL) methods apply only to domains for which a tractable domain theory is available [Mitchell et al. 86]. While similarity-based learning (SBL) can be applied to intractable domains, it does not fully exploit the background knowledge contained in an intractable domain theory. These limitations are significant for the science of machine learning due to the ubiquity of intractable domains. Problems of intractability arise in a variety of domains including games like chess, circuit design, job scheduling and many others. Machine learning techniques are needed for such domains because intractability prevents the available theories from being directly useful for solving problems.

This research is aimed at handling the intractable theory problem by developing new explanation-based learning methods. A program called POLLYANNA has been developed to experiment with such new EBL methods. POLLYANNA's learning strategy involves replacing exact intractable theories with approximate theories requiring fewer computational resources. Theories are approximated by explicitly introducing *simplifying assumptions* about the domain. The assumptions are useful because they greatly simplify the process of explaining observations or making inferences in a performance element. Although such assumptions are not strictly true in all situations, they may be correct in most typical cases. Even when not true, the assumptions may be sufficiently accurate so as to generate correct performance. In order to find such useful assumptions, POLLYANNA makes use of empirical information. Explanations of teacher-provided training examples are used to guide a search for accurate simplifying assumptions.

[1]Reprint of a paper appearing in the Proceedings of the Seventh National Conference on Artificial Intelligence, St. Paul Minnesota, August 1988.

The learning strategy used in POLLYANNA differs markedly from that of previous explanation-based learning programs. Prior EBL research has focused on compiling explanations into schemata [DeJong and Mooney 86; Mitchell et al. 86]. Some recent studies have investigated the role of simplifying assumptions for intractable domains [Chien 87; Bennett 87]; however, the assumptions are studied mainly in the context of schema formation. POLLYANNA is based on the belief that schema formation is a problem of secondary importance compared to the task of finding appropriate simplifying assumptions themselves. POLLYANNA's methodology does not preclude schema formation; however, it involves using explanations primarily for a different purpose. Explanations are used for the purpose of evaluating candidate assumptions. Assumptions are evaluated according to whether they shorten the process of building explanations, and whether they correctly explain many examples. By adopting assumptions according to their power to explain examples, POLLYANNA manifests a form of abductive inference. A more complete description of this approach is found in [Ellman 87].

2 A Methodology for Finding Simplifying Assumptions

The learning process embodied in POLLYANNA has been broken down into several distinct phases, enumerated in Figure 1. These phases correspond roughly to a generate and test framework for finding simplifying assumptions. The first step generates a set of candidate assumptions by systematically instantiating schemata from a predefined typology of simplifying assumptions. An example of such a typology is described in Figure 2. After generating candidate assumptions, the system selects various well-formed sets of assumptions and integrates them into the initial intractable theory. This produces a collection of approximate theories, organized in a lattice structured search space. In the final phase, the system conducts a search through the approximate theory space. Various approximate theories are invoked to explain teacher-provided training examples. The results are used to guide the search process. The general approach of using examples to guide a search through an approximate theory space is similar to methods described in [Keller 87; Mostow and Fawcett 87]; however, the methods used here to generate and search the theory space are different. The theory space generation and theory space search phases have been implemented. Implementation of the assumption generation phase is in progress.

1. **Assumption Generation**: Generate candidate assumptions by instantiating schemata from a predefined typology of simplifying assumptions.
2. **Theory Space Generation**: Incorporate sets of simplifying assumptions into the domain theory to generate a space of approximate theories.
3. **Theory Space Search**: Search through the theory space to find simple, accurate theories.

Figure 1: Learning Phases in POLLYANNA

3 An Intractable Theory

The card game "hearts" has been chosen as a testbed domain for the POLLYANNA program.[2] The hearts domain theory is represented in terms of a collection of purely functional LISP expressions that are used to evaluate potential card choices in any game situation. The theory computes an evaluation

[2]Hearts is normally played with four players. Each player is dealt thirteen cards. At the start of the game, one player is designated to be the "leader". The game is divided into thirteen successive tricks. At the start of each trick, the leader plays a card. Then the other players play cards in order going clockwise around the circle. Each player must play a card matching the suit of the card played by the leader, if he has such a card in his hand. Otherwise, he may play any card. The player who plays the highest card in the same suit as the leader's card will take the trick and become the leader for the next trick. Each player receives one point for every card in the suit of hearts contained in a trick that he takes. The game objective is to minimize the number of points in one's score.

function ES(c,p,t) yielding the expected final game score for player (p) if he plays card (c) in trick (t). In order to compute this value, it is necessary to average over all ways the cards might be dealt, and perform a mini-max search for each possible deal. In practice this computation is hopelessly intractable. Each mini-max computation involves searching a large space of game trees to find a solution tree. Each solution tree is itself quite large, and the evaluation of each tree must be summed over a large number of possible deals. The hearts domain thus exhibits both types of intractability described in [Rajamoney and DeJong 87], i.e., a "large search space" and a "large explanation structure".

4 A Typology of Simplifying Assumptions

In order to implement POLLYANNA in the hearts domain, it has been necessary to identify the general types of simplifying assumptions that are useful for this domain. A partial typology of such assumptions is shown in Figure 2. The assumptions shown here are drawn from a longer list that was developed by analyzing protocols of hearts games played by humans. Verbal explanations of people's decisions were analyzed to extract and formalize the assumptions they implicitly contained. Some specific instances of these types of assumptions are shown in Figure 3. Although the typology was developed by studying the hearts domain, it is expected that future research will demonstrate its usefulness in other domains as well.

```
1. Invariance of Functions:
     F(x) = F(y) for all x and y.

2. Independence of Random Variables:
     Exp[x * y] = Exp[x] * Exp[y]

3. Equal Probability of Random Variables:
     Prob[var = value] = 1/|Range[var]|

4. Abstraction of the Problem State:
     Prob[x Given state] = Prob[x]
```

Figure 2: Typology of Simplifying Assumptions

1. Assume the expected number of points to be taken in all future tricks, EFTS(c,p,t), is invariant with respect to the card (c), the player (p) and the trick (t).
2. Assume the odds of winning a trick, P-WIN, are independent of the trick's expected point value, EXP-POINTS.
3. Assume the lead suit for trick number N is equally likely to be any suit.
4. Assume all cards in the deck remain unplayed, ignoring information about which cards have actually been played in the current problem state.

Figure 3: Specific Assumptions for Hearts

5 Representation of Simplifying Assumptions

In the course of implementing POLLYANNA, an important task has involved finding representations for the simplifying assumptions. The assumptions must be represented in a manner that allows them to interface with the initial intractable theory and to shorten the process of building explanations. In the POLLYANNA system, this problem has been handled by an approach based on polymorphism and generic functions [Stefik and Bobrow 86]. Each function appearing in the hearts domain theory is considered to be a generic function and is implemented in one or more versions. Some examples of functions with multiple versions are shown in Figure 4. This figure shows several different functions used in the hearts theory. Each of the functions exists in the two different versions shown, among other versions not shown. Each version of a generic function implements a different simplifying assumption, or set of simplifying assumptions. The various function definitions have been coded in terms of purely functional LISP expressions. Some of the function arguments are not shown. In particular, each function takes an additional argument that determines which version should be used.

```
ES(card):  (Expected Score)
 ES-0(c)  = Constant
 ES-1(c)  = ECTS(c)  + EFTS(c)

ECTS(card):  (Expected Current Trick Score)
 ECTS-0(c)  = Constant
 ECTS-1(c)  = P-WIN(c)*EXP-POINTS(c)

EFTS(card):  (Expected Future Tricks Score)
 EFTS-0(c)  = Constant
 EFTS-1(c)  = SUM(k)(HAND-{c})EXP-TAKE(k)

UC(state):  (Unplayed Cards)
 UC-0(s)  = DECK
 UC-1(s)  = DECK  - CARDS-PLAYED(s)
```

Figure 4: Multiple Versions of Generic Functions

Important representation issues arise upon comparing the typology of simplifying assumptions (Figure 2) to the actual LISP implementation of the assumptions (Figure 4). In some cases, the LISP definitions represent straightforward implementations of simplifying assumptions from the typology. For example, the definition "EFTS(card) = Constant" is a direct application of the assumption that a function is independent of its arguments. Other definitions are semantically equivalent to assumptions from the typology, but are syntactically quite different. This indicates that the task of generating such assumptions may involve significant issues of theory reformulation, as noted in [Mostow and Fawcett 87].

Several advantages result from the technique of representing assumptions in terms of generic functions. To begin with, it helps in dealing with problems of inconsistency that arise when strictly untrue assumptions are added to the initial intractable theory. When an approximate version of a function F is added to the theory, the inconsistency is avoided if the original definition of F is removed at the same time. This technique also provides a convenient mechanism for determining when a set of assumptions is complete. A set of assumptions is complete when there is a definition for each function referenced in the set.

6 Generating a Space of Approximate Theories

In order to generate a space of approximate domain theories, POLLYANNA systematically combines various versions of the generic functions. For this purpose, the theory space generator is provided with a list of versions of each generic function. The generator is also provided with a relation partially ordering the versions of each generic function. More specifically, the relation PRIMITIVE-REFINEMENT(F0,F1) indicates that F0 implements a strictly stronger set of assumptions than F1, i.e., the assumptions of F0 logically imply the assumptions of F1. For example, the relation PRIMITIVE-REFINEMENT(ES-0,ES-1) indicates that version zero uses a strictly stronger set of assumptions than version one. At present this relation is coded by hand; however, it is expected that future research will demonstrate that it can be generated automatically.

The theory space is generated by a process that extends the PRIMITIVE-REFINEMENT(F0,F1) relation among generic functions into a relation, REFINEMENT(T0,T1), among theories. The space is generated by beginning with the simplest version of the top level function ES-0. This represents the root of the theory space. Refinements of this simple theory are generated by repeatedly applying the following rule. Any theory T-old can be refined into a theory T-new, by replacing some generic function version F0 with a new version F1 such that PRIMITIVE-REFINEMENT(F0,F1) holds. If F1 references a new generic function G not yet defined in the theory, the simplest version of G is added to make the refined theory complete. Thus the root theory using ES-0 can be refined into a theory using ES-1. Since ES-1 references ECTS and EFTS, the simplest versions of these functions are added to make the theory complete. The theory can then be further refined by substituting new versions of ECTS or EFTS. This process creates a lattice of theories, organized by the relation REFINEMENT. Whenever REFINEMENT(T0,T1) holds, T0 uses a strictly stronger set of assumptions than T1. It is expected that the REFINEMENT relation serves also to approximately order the theories according to costs of computation. Preliminary measurements indicate this is indeed the case.

7 Searching a Space of Approximate Theories

A number of different algorithms have been developed in POLLYANNA for searching the approximate theory space. The algorithms all use an "optimistic" strategy of starting at the lattice root, i.e., the simplest theory in the space, and moving to more complex theories only when simpler ones are contradicted by training examples. This strategy is achieved by using the REFINEMENT relation to constrain the order in which theories are examined. The search algorithms differ mainly in the goal conditions and control strategies that are used. One version takes an error rate threshold as input, and searches for a theory of minimal computational cost that meets the specified error rate. A best first search algorithm is used to control the search, always taking a theory of minimal cost to expand next into its refinements.[3] Both the costs and the error rates are measured empirically, by using candidate theories to explain a set of teacher-provided training examples. The examples are processed in batches, i.e., the system tests each theory against the entire example set before moving on to refined theories. It is worth noticing that the search is facilitated by the lattice organization of the theory space. The space is structured so that costs of computation increase monotonically along paths in the lattice. This allows the search algorithm to terminate upon expanding the first theory meeting the error rate threshold, since more refined theories will have equal or greater computational costs.

POLLYANNA has been tested on several different sets of training examples. One set was designed to reflect a strategy of leading cards of minimal rank. The system was led to a goal theory asserting that ES(c,p,t) = C1 * P-WIN(c,p,t) + C2. This approximate theory uses a non-trivial version of P-WIN, the probability of winning the current trick. It ignores the expected point value of the current trick by assuming that EXP-POINTS(c,p,t) is a constant (C1). Another example set was designed to reflect a strategy of leading cards of minimal point value. The system was led to a goal theory asserting that ES(c,p,t) = C1 * EXP-POINTS(c,p,t) + C2. This theory ignores the odds of winning the trick by assuming that P-WIN(c,p,t) is a constant (C1). Instead it focuses on the expected trick point value, by using a non-trivial version of EXP-POINTS. These results indicate that POLLYANNA can be led to adopt different and inconsistent sets of assumptions depending on the examples provided.

POLLYANNA produces data to illustrate the tradeoff between accuracy and tractability, as shown in Figure 5. This graph was generated during the second of the two runs described above. Each point on

[3]An alternate search goal finds a theory of minimal error rate meeting a computational cost threshold. The alternate control strategy chooses theories of minimal error rate to expand next.

the graph represents a single approximate theory. The horizontal axis indicates the average running time of the theory, measured in terms of the number of function calls needed to evaluate all the choices in a given example situation. The vertical axis measures the "false good" error rate of the theory in the following way: If G is the true set of optimal cards in some example situation, and G' is the set of cards considered "optimal" by the approximate theory, then the false good rate is FG = |G'-G|/|G'|. This represents the probability that a card chosen randomly from G' will actually be wrong. The vertical axis measures FG averaged over all examples from the training set.

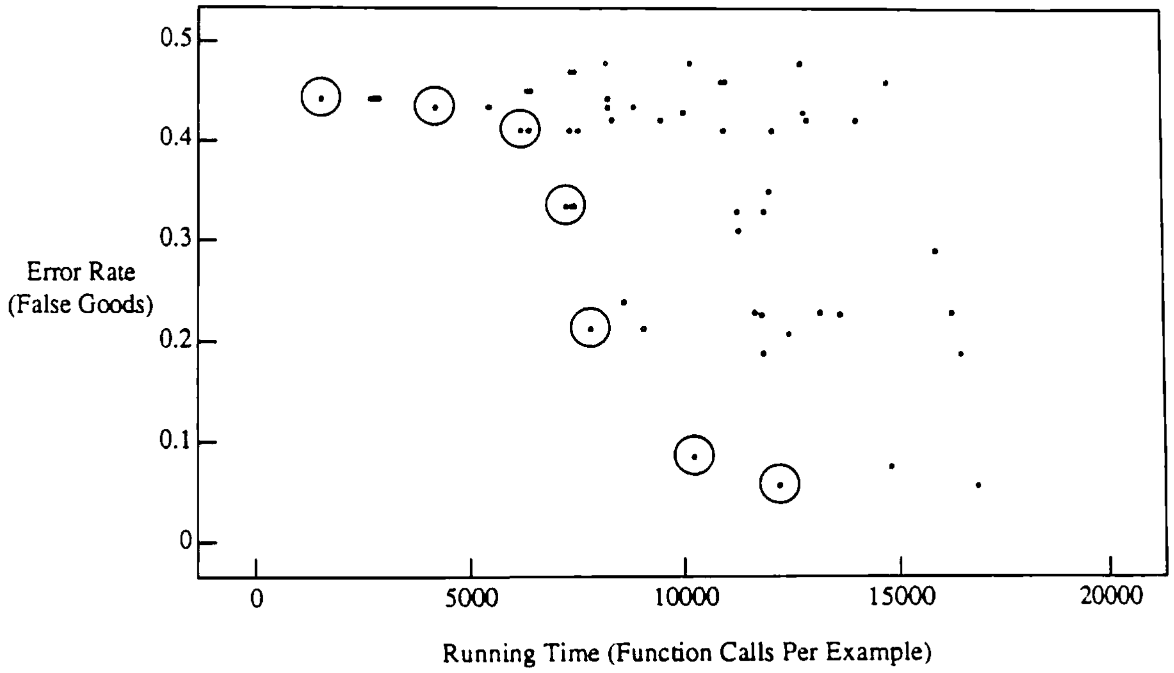

Figure 5: Tradeoff between Accuracy and Tractability

The circled points in Figure 5 correspond to theories that are Pareto optimal. Each circled point cannot be improved in running time except at the price of a greater error rate. Likewise each circled point cannot be improved in error rate, except at the price of increasing the running time. In order to choose among the Pareto optimal points, to find the right combination of accuracy and tractability, the system must be provided with contextual knowledge [Keller 87] defining its performance objectives.

8 Conclusion

A new viewpoint on explanation-based learning is suggested by the methodology used in POLLYANNA. Prior EBL research has equated "explanation" with "logically sound proof" [Mitchell et al. 86]. POLLYANNA is based on a weaker notion of explanation, i.e., a proof based on simplifying assumptions. The POLLYANNA methodology is also distinguished by the fact that it uses explanations in a manner different from previous EBL systems. Prior research has focused on compiling explanations into schemata.. The approach described here does not preclude schema formation; however, it uses explanations for a more important task. Explanations are used in a process of abductive inference to guide a search for simplifying assumptions. Depending on the examples provided, this technique can find different, inconsistent simplifying assumptions. It is therefore immune to a criticism leveled at other EBL systems, i.e., they only "compile" existing knowledge and do not change when viewed from the "knowledge level" [Dietterich 86]. The new strategy extends EBL beyond techniques for compilation of knowledge to become a process of substantive theory revision.

9 Acknowledgments

Many thanks to Michael Lebowitz for numerous useful discussions about the research described in this paper. Thanks also to Jack Mostow for comments on a draft of this paper and to Doree Seligman for help with graphics.

References

[Bennett 87] Bennett, S. W. Approximation in Mathematical Domains. Technical Report UILU-ENG-87-2238, University of Illinois, Urbana-Champaign, Illinois, 1987.

[Chien 87] Chien, S. A. Simplifications in Temporal Persistence: An Approach to the Intractable Domain Theory Problem in Explanation-Based Learning. Technical Report UILU-ENG-87-2255, University of Illinois, Urbana-Champaign, Illinois, 1987.

[DeJong and Mooney 86] DeJong, G. and Mooney, R. ''Explanation-Based Learning: An Alternative View.'' *Machine Learning 1*, 2, 1986, pp. 145 - 176.

[Dietterich 86] Dietterich, T. G. ''Learning at the Knowledge Level.'' *Machine Learning 1*, 3, 1986, pp. 287 - 315.

[Ellman 87] Ellman, T. P. Explanation-Based Methods for Simplifying Intractable Theories: A Thesis Proposal. Technical Report CUCS-265-87, Columbia University, New York, New York, 1987.

[Keller 87] Keller, R. The Role of Explicit Contextual Knowledge in Learning Concepts to Improve Performance. Technical Report ML-TR-7, Rutgers University, New Brunswick, New Jersey, 1987. PhD Thesis.

[Mitchell et al. 86] Mitchell, T. M., Keller, R. M. and Kedar-Cabelli, S. T. ''Explanation-Based Learning: A Unifying View.'' *Machine Learning 1*, 1, 1986, pp. 47 - 80.

[Mostow and Fawcett 87] Mostow, J., Fawcett, T. Forming Approximate Theories: A Problem Space Model. Technical Report ML-TR-16, Rutgers University, New Brunswick, New Jersey, 1987.

[Rajamoney and DeJong 87] Rajamoney, S., DeJong, G. The Classification, Detection and Handling of Imperfect Theory Problems. Proceedings of the Tenth International Joint Conference on Artificial Intelligence, Milan, Italy, 1987.

[Stefik and Bobrow 86] Stefik, M. and Bobrow, D. ''Object-Oriented Programming: Themes and Variations.'' *AI Magazine 6*, 4, 1986, pp. 40 - 62.