Proc. Third International Machine Learning Workshop, Skytop, PA, June 1985.

ODYSSEUS: A LEARNING APPRENTICE

David C. Wilkins, William J. Clancey and Bruce G. Buchanan

Computer Science Department
Stanford University
Stanford, CA 94305

Abstract

Human specialists employ powerful learning methods during their apprenticeship training period, to augment their fledgling expertise. We describe a system under construction to allow an expert system to use some of these same methods.

The Odysseus learning apprenticeship program watches the observable actions of a specialist. Justifications are created for each action via a process of differential modeling between the specialist and an expert system. A learning opportunity occurs when no action justification is judged sufficiently plausible. This paper describes the three phases that Odysseus uses to learn via differential modeling: generation and justification of an expanded rule base, generation and ranking of action justifications for each observable action, and rationalization of any discrepancies to effect repair.

1. Introduction

An apprenticeship learning period is an important phase on the path to master expert status for human specialists. During this phase, an apprentice specialist *learns by watching master specialists* and *learns by doing problem solving* under the supervision of master specialists. Our research investigates how to give an expert system the benefits of an apprenticeship period.

This paper describes a computer program, Odysseus, that learns by watching specialists in the domain of medical diagnosis. The central task of Odysseus is to *rationalize* each observable action of a specialist during problem solving sessions. In medical diagnosis, these actions consist of all data requests made by a physician and the final diagnosis. Actions are rationalized by a process of *differential modeling* between the expert system and the specialist. Failure to find an adequate rationalization signals a possible deficiency in the expert system's domain or strategy knowledge. Using a taxonomy of deficiencies in conjunction with theoretical and experiential knowledge of the application domain, Odysseus automatically generates and tests conjectures to explain its inability to justify a specialist's action.

Beginning with an existing expert system, the apprenticeship learning process goes through three phases, to be described in the next three sections. These phases are (1) generation and justification of an expanded rule base, (2) generation and ranking of action justifications for each observable step of a problem-solving session, and (3) rationalization of discrepancies to effect repair.

2. Generating Rule Justifications

The Heracles expert system shell, a domain-independent version of Neomycin, is used by Odysseus for differential modeling. Neomycin (Heracles and a particular medical knowledge base) is a reorganization of the Mycin expert system that simulates the diagnostic process of experts [1]. It differs from Mycin in its factorization of the different types of knowledge originally contained in the Mycin rules, and its method of hypothesis-directed reasoning, which employs a body of domain-independent strategy knowledge and a strategy script for diagnosis. A taxonomy of abstract strategy goals permits comparison between the actions of Heracles and the specialist.

There are two ways in which an expert system must be augmented before differential modeling of a human specialist can commence. First, the set of heuristic rules must be replaced by a larger set obtained by induction over past problem solving cases. The original set of rules is adequate for problem solving but is too impoverished to explain the problem solving behavior of a community of specialists. Having Odysseus responsible for generating all initial and subsequent rules guarantees uniformity in the method of assigning strengths to rules. Second, all domain rules must be justified by theory or experience. Definitional, subsumption, and causal rules are justified by references to

textbook facts and causal models. Heuristic rules may be justified by theory and must always be justified by experience.

Heuristic rules are usually of the form $lhs \xrightarrow{cf} hypothesis$, where cf is a Mycin-type certainty factor. The induction part of Odysseus consists of a constrained rule generator and a candidate rule evaluator, that finds all *lhs* forms of the above rule template that meet given constraints of minimal rule generality (coverage), minimal rule specificity (discrimination), maximal rule colinearity (similarity), and maximal rule simplicity (number of conjunctions and disjunctions). Though the rule generation process replaces the entire heuristic rule base, preference is given to colinear forms of heuristic rules contained in the original rule base or found in textbooks. The initial rule set is necessarily incomplete; however, it bootstraps the differential modeling process that leads to its refinement. Rules that are not used for watching or problem solving are eventually pruned. In addition to the construction of an initial heuristic knowledge base, the induction program conjectures missing rules during the process of rationalizing discrepancies.

Heuristic rule sets have an unusual property: better rules do not necessarily lead to a better rule base, as all rules with a cf not equal to certainty contribute evidence toward false positive or false negative diagnoses for some cases. The best rule base is the element of the power set of good rules that yields a global minimum weighted error. The selection process can be modeled as an NP-complete bipartitite graph reduction problem, and we have developed a heuristic that yields a satisfactory weighted error. This optimization phase applied to the original training set of one hundred cases reduces the number of misdiagnoses from ten to zero.

3. Generating and Ranking Action Justifications

Once the rule justification phase is complete, differential modeling begins. For each observable action A_i of the specialist, Odysseus generates an *action justification set*: $J(A_i) = \{j_{i,1}, j_{i,2}, \ldots, j_{i,n}\}$. An action justification $j_{i,k}$ relates an action A_i to an abstract strategic goal G via a rule chain, that is, $A_i \supset R_1 \supset R_2 \supset \cdots \supset R_p \supset G$. All possible rule chains beginning with A_i to all possible goals are in $J(A_i)$. Using the Neomycin rule base, and inputting Neomycin's own actions to the action justification generator, the average size of $J(A_i)$ was ten and the maximum size was approximately one hundred. When an Odysseus-generated rule base for the Neomycin domain was used, these set sizes increased by a factor of four to five.

After the set $J(A_i)$ is generated, the action justification ranking subsystem of Odysseus determines the likelihood that $J(A_i)$ contains $j_{i,s}$, the action justification of the specialist. This involves, first, ranking $j_{i,1}, \ldots, j_{i,n}$ in order of likelihood of being equal to the unknown $j_{i,s}$. An example of ranking rule is: given two elements of a $J(A_i)$, where A_i occurs early in the problem solving session, the $j_{i,k}$ that relates to the more general hypothesis is more likely to be $j_{i,s}$. Second, Odysseus must decide if the top ranked justification(s) is likely to be equal to $j_{i,s}$ — a very difficult problem, although the ability to "know that you don't know" why an action is taken is easy for human specialists who are watching other specialists.

On Neomycin test cases, the elements of $J(A_i)$ generated using the Neomycin rule set as a basis were correctly ranked. The specialist's (Neomycin's) justification was always the top-ranked justification. However, generating and ranking justifications became more difficult when protocols of actual physicians where used. Indeed, $j_{i,s}$ was not in the $J(A_i)$ generated from the Neomycin rule set 75% of the time. The solution to the incompleteness of $J(A_i)$ is the use of an initial induction phase to expand the rule base. The solution to inaccurate ranking is more complex ranking heuristics. A method being implemented to produce these is as follows.

Automatic generation of new ranking heuristics begins with the collection of protocols annotated with $j_{i,s}$ for each A_i. These protocols allow identification of the features of the problem solving session relevant to ranking elements of $J(A_i)$. These features fall into a number of broad categories: patterns of interpretation in the problem solving session, the distance between the expert system's preferred strategic action and the strategic action associated with each $j_{i,k}$, a user model that records individual diagnostic styles, and a history of past problem solving sessions. Each of these features may relate to features of $j_{i,k}$, such as its focus, rules, and strategic goal.

Given the set of problem features that contribute evidence to determining $j_{i,s}$ and the features

of elements of the solution set $J(A_i)$ to which these problem features can relate, the task of inferring the likelihood that a member of $J(A_i)$ is $j_{i,s}$ can be formulated as a heuristic classification problem. Odysseus generates a rule base for an expert system to solve this problem. Odysseus is fed training instances of the form: $\langle J(A_i), j_{i,s}, \Omega \rangle$, where Ω is a set of the dynamic feature-value pairs. An example of such a feature-value pair is $\langle focus_{i-2} \equiv focus_{i-1} \equiv focus_i, True \rangle$

4. Rationalizing Discrepancies

The lack of an adequate action justification for an action A_i signals a potential learning opportunity. Because heuristic rules are probabilistic and are rationalized in terms of all past problem solving sessions, it makes little sense to think of these rules as wrong. Rather, an unjustified A_i suggests a missing rule or undesirable probabilistic interaction in the rule set. The missing rule may be a specialization or generalization of an existing rule. To learn new heuristic domain knowledge, first the action justification subsystem provides a ordered goal set, $\gamma = \{G_1, G_2, ..., G_m\}$, of the goals to which the specialist's action most likely relates. An example of a goal is: $G_j = \langle task, focus \rangle = \langle confirm\ hypothesis, intracranial\ pressure \rangle$. Using a blackboard architecture representation, the Odysseus repair subsystem carries out a bi-directional search for a missing heuristic rule R_m such that $A_i \supset R_1 \supset \cdots \supset R_m \supset \cdots \supset R_t \supset G_j \in \gamma$. At each point where a missing rule could complete the chain, the induction part of the rule justification subsystem searches for such a rule. For each unjustified A_i, we assume that at most one rule is missing. The specialist is asked to validate the correctness and relevance of the new rule chain with R_m, and may also be asked for a statement of the goal being pursued. Knowledge of the goal can be used to reduce the size of γ, thus reducing the complexity of the search for R_m.

Correcting inadequacies due to incomplete or erroneous domain-independent strategy knowledge is beyond the capabilities of Odysseus. The system can detect this type of error while watching specialists, but the repair must be performed manually.

5. Summary

This paper provides an overview of the three phases used by Odysseus to automate knowledge acquisition for expert systems, and gives the results from implementing the first two phases. We are currently implementing the new method of ranking justifications and rationalizing discrepancies.

The Leap learning apprentice for circuit design justifies rules in terms of circuit theory, a strong theory of the domain [2]. By contrast, a weak theory underlies medical diagnosis, and Odysseus's justifications for rules rely strongly on experiential predictive power. In Leap, the exact input and output of each problem solving step is observable. The medical specialist's actions observed by Odysseus reflect more indirectly the problem-solving knowledge to be learned.

6. Acknowledgements

The apprenticeship learning framework described here owes much to discussions with Avron Barr, Jim Bennett, Tom Dietterich, Paul Scott, and Derek Sleeman. Marianne Winslett Wilkins gave insightful comments on an earlier draft and provided a graph-theoretic viewpoint on optimizing rule sets. For critiquing Odysseus during its development, we are grateful to doctors Larry Fagan, Kurt Kapsner, Randy Miller, Mark Musen, Roy Rada, and Ted Shortliffe.

This work was supported in part by NSF grant MCS-83-12148 and ONR/ARI contract N00014-79C-0302. Computational resources were provided by SUMEX-AIM (NIH grant RR 0078).

7. References

[1] Clancey, W. J. and Letsinger, R., "Neomycin: Reconfiguring a rule-based system for application to teaching," *Readings in Medical Artificial Intelligence*, Clancey, W. J. and Shortliffe E. H., (eds.), Addison-Wesley, 1984.

[2] Mitchell, T. M., Mahadevan, S., and Steinberg, L. I., "Leap: A learning apprentice for VLSI design," to appear in *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, August 1985.

[3] Wilkins, D. C., Buchanan, B. G., Clancey, W. J., "Inferring an expert's reasoning by watching," *Proceedings of the 1984 Conference on Intelligent Systems and Machines*, 1984, pp 51-59. (also HPP Report HPP-84-29, Stanford University, June 1984)