Exploiting the Ordering of Observed Problem-solving Steps for Knowledge Base Refinement: an Apprenticeship Approach

Steven K. Donoho and **David C. Wilkins***
Department of Computer Science
University of Illinois
405 North Mathews Avenue
Urbana, IL 61801
donoho@cs.uiuc.edu, wilkins@cs.uiuc.edu

Abstract

Apprenticeship is a powerful method of learning among humans whereby a student refines his knowledge simply by observing and analyzing the problem-solving steps taken by an expert. This paper focuses on knowledge base (KB) refinement for classification problems and examines how the *ordering* of the problem-solving steps taken by an observed expert can be used to yield leverage in KB refinement. Questions examined include: What added information can be extracted from attribute ordering? How can this added information be utilized to identify and repair KB shortcomings? What assumptions must be made about the observed expert, and how important of a role do these assumptions play? The principles explored have been implemented in the SKIPPER apprentice, and empirical results are given for the audiology domain.

1 Introduction

Apprenticeship is a powerful method of learning among humans in which a student refines his knowledge by observing and analyzing the problem-solving steps of an expert. Many previous works in apprenticeship [Dent et al., 1992; Mahadevan et al., 1993; Mitchell et al., 1985; Redmond, 1992; Tecuci and Kodratoff, 1990; Wilkins, 1988] have taken steps toward harnessing the information provided by observing problem-solving steps. In this paper we focus on knowledge base (KB) refinement for classification problems and examine how the *ordering* of the intermediate steps of an observed expert can be used to yield leverage in KB refinement.

In the classical classification problem, the problem-solver is given an example consisting of a set of attributes and their corresponding values, and it must put the example in one of a pre-enumerated set of classes. For example, the problem-solver may be given a batch of attribute/value pairs describing a soybean plant and must classify that plant as having one of a pre-enumerated set of soybean plant diseases.

Consider a slightly different situation, though, in which the problem-solver is not given *all* the attribute/value pairs from the outset but rather must request attributes one at a time and make his classification decision once sufficient evidence is gathered. This situation would arise when it is too costly or otherwise unreasonable to simply be given all the attribute values. This situation applies to domains such as medical diagnosis in which all the patient's symptoms are not given at once but rather must be requested individually based on what the doctor knows about the patient so far. When a mechanic is troubleshooting a malfunctioning car, he does not run every test possible and then stop to examine his data and make his decision. Rather he checks one thing, and based on the result of that, he decides what to check next.

Thus the order in which attributes are requested reflects the internal problem-solving process going on in the mind of the observed expert. By watching the order in which a *superior* problem-solver requests attributes, we should be able to refine the KB of a *weaker* problem-solver. We will refer to the superior problem-solver which is being watched as the **observed expert** and the weaker problem-solver which is being refined as the **critiqued problem-solver**. While refining the critiqued problem-solver, we have full access to its KB, but our only interface with the observed expert is the visible actions he takes so as to require nothing more from the expert than to perform his normal work.

The approach we take to KB refinement is *knowledge acquisition within the context of a shortcoming in the knowledge base.* TEIRESIAS [Davis, 1979] introduced this approach whereby an expert

*This work supported by a DoD Graduate Fellowship, ONR grant N00014-88K-0124, and AFOSR grant F49260-92-J-0545.

would point out an expert system's failure, the expert would then guide TEIRESIAS in localizing the cause of the failure, and the expert would suggest a repair. A natural next step is to move toward automating the three tasks for which the expert was indispensable in TEIRESIAS. These tasks are as follows: detecting a KB shortcoming, localizing the shortcoming, and constructing a repair. TEIRESIAS solved these three subtasks by consulting with the expert. In this paper we attempt to push these three subtasks toward automation relying not on the expert's intervention but rather on simply observing his problem-solving steps and their ordering.

A comparison of our approach to some related work in apprenticeship is given in section 2. Section 3 introduces how a KB shortcoming can be detected by analyzing the order in which attributes are requested. Section 4 explains how a KB shortcoming, once detected, can be localized using the context in which it was detected, and section 5 presents a method of generating KB repairs once the shortcoming is localized. The SKIPPER apprentice is an implementation which puts these three tasks together, and section 6 discusses an experiment showing how SKIPPER improves classification accuracy by refining a KB produced by the C4.5 machine learning program [Quinlan, 1993].

2 Related Work

The LEAP system [Mahadevan et al., 1993] works in the domain of digital circuit design giving an expert suggestions for how to decompose a high-level circuit specification into submodules. When the expert disagrees with a suggestion made by LEAP, LEAP learns from the alternative proposed by the expert. LEAP fits the definition of apprentice because rather than just examining a completed circuit as a whole, it analyzes the individual, fine-grained decomposition taken by the expert. But the order in which the expert takes these problem-solving steps is not used as a source of information.

The CAP program [Dent et al., 1992] assists in managing an individual's meeting calendar by predicting certain meeting details such as time, location, and duration from what it knows about the meeting such as the nature of the meeting and its attendees. The individual problem-solving steps taken by the expert (the calendar user) are often invisible. The expert examines in his head the meeting type, the department of the attendees, job title of the attendees, etc. and decides the details of the meeting; thus neither the fine-grained problem-solving steps nor their ordering are available as a source of information.

The ODYSSEUS project [Wilkins, 1988] holds many similarities to our work. The goal of both is to refine a classification KB by watching the actions of another problem-solver. The expert being observed in ODYSSEUS was required at each step to state what class he was focusing on. This forced the expert to articulate his strategies rather than just solve problems unhindered. Our method uses the context of the attributes requested so far to hypothesize what the observed expert is focusing on at any given point so as to avoid having to request it explicitly.

A closely related work in case-based reasoning is the CELIA system [Redmond, 1992]. CELIA detects KB shortcomings by predicting an expert's actions given the current problem-solving state and adds a new case when its predictions fail. Redmond explored repairing by taking hints from or asking questions of a competant instructor whereas our work uses attribute ordering as a means of guiding induction.

3 Detecting a KB Shortcoming

Detecting a KB shortcoming is synonymous with answering the question, "When does an action taken by the observed expert indicate that there is something missing from the critiqued problem-solver's knowledge base?" In short, the answer is that an action indicates something is missing from a KB when that action cannot be justified using that KB. Consider the following example. When a doctor is diagnosing a patient, the doctor asks the patient a series of questions to determine what disease the patient has. A medical student watching the doctor could probably give an explanation of why each question was asked because the student himself has a good body of medical knowledge. If the doctor asks a question, and the student cannot explain why the doctor asked it, then the student realizes that the doctor knows something that he does not know. He has detected a shortcoming in his knowledge. Likewise, the failure to explain an observed expert's action using a critiqued problem-solver's KB indicates a shortcoming in that KB.

In order to explain an observed action, though, we have to make some *a priori* assumptions about the observed expert. If we make no assumptions, the observed expert is totally unconstrained and could be requesting attributes randomly in which case all actions have the potential explanation, "the observed expert is acting randomly," and no shortcomings in the critiqued problem-solver's KB can be detected. One reasonable assumption to

make is that the observed expert always acts rationally, i.e. that he always has some reason or motive for requesting an attribute and therefore never takes an irrelevant action. This assumption is simple yet actually provides a great deal of leverage.

As an example of the information that attribute ordering gives, consider the example shown in Figure 1(a) taken from the audiology domain. The observed expert first requests the attribute *age_gt_60* and receives the answer *true*. Knowing this he requests the attribute *history_nausea* and receives the answer *false*. He requests three more attributes and then halts and makes his decision classifying the patient as *cochlear_age* without requesting any more attributes. This tells us many things:

- Given that nothing at all is known, *age_gt_60* is a relevant attribute to request.
- The value of *history_nausea* is relevant to solving the problem even given that *age_gt_60* is known to be *true*. Otherwise, the observed expert would not have requested *history_nausea* since *age_gt_60* was already known to be *true*.
- The value of *history_noise* is relevant given that *age_gt_60* is *true* and *history_nausea* is *false*.
- The value of *air* is relevant given that *age_gt_60* is *true*, *history_nausea* is *false*, and *history_noise* is *false*.
- The value of *tymp* is relevant given that *age_gt_60* is *true*, *history_nausea* is *false*, *history_noise* is *false*, and *air* is *normal*.

So the ordering of attributes gives us information about "conditional relevancy" — the values of certain attributes are necessarily relevant given the values of certain other attributes.

As an example of how this information can be used to detect KB shortcomings, consider the simple set of rules to be critiqued in Figure 1(b). Using the problem-solving steps from Figure 1(a), when the attribute *age_gt_60* is requested, this action is explainable because the observed expert could be requesting that attribute to satisfy the premise of any of the four rules. When the value *true* is given, though, the premise of Rule2 becomes false; therefore, Rule2 is no longer relevant to solving this problem. Likewise, *history_nausea* is explainable because it is in the premise of Rule3 which is still relevant. Following that, *history_noise* is explainable because it is in the premise of Rule1 which is still relevant (but finding that *history_noise* is *false* causes Rule1 to become irrelevant). But when the attribute *air* is requested, a KB shortcoming is

Requested Attribute	Value Given
age_gt_60	*true*
history_nausea	*false*
history_noise	*false*
air	*normal*
tymp	*a*

Classified as *cochlear_age*

(a) A sequence of attributes requested by an observed expert.

Rule1: age_gt_60 = true ∧
history_noise = true ∧
tymp = a →
cochlear_age_and_noise

Rule2: age_gt_60 = false ∧
air = mild →
cochlear_unknown

Rule3: age_gt_60 = true ∧
speech = very_poor ∧
history_nausea = false →
cochlear_age_plus_poss_menieres

Rule4: age_gt_60 = true ∧
history_dizziness = false ∧
tymp = a →
cochlear_age

(b) The rule set to be critiqued.

Figure 1: The attribute request *air* in (a) is irrelevant using the KB in (b) because the premises of all rules which contain *air* are false by the time *air* is requested.

detected. The attribute *air* is found only in the premise of Rule2 which is no longer applicable since *age_gt_60* is known to be *true*; therefore, the attribute request *air* is unexplained. According to the critiqued KB, the attribute *air* is not relevant at this point in the problem yet the observed expert requested it, and the observed expert is assumed to only request relevant attributes! There must therefore be some knowledge which the observed expert possesses which is not in the critiqued KB. Thus the ordering of the attributes — specifically the fact that *air* was requested after *age_gt_60* was known to be *true* — has enabled the detection of a KB shortcoming. **The ordering allows the analysis of the relevancy of an attribute at a given time with respect to the critiqued KB, and this may be at odds with the relevancy indicated by the observed expert's actions.**

Assuming that the observed expert is rational is a weak yet generally applicable constraint. Stronger constraints can yield even more leverage.

A slightly more constraining assumption is that the observed expert uses a "test-hypothesis" strategy, i.e. he hypothesizes a class which he thinks the example at hand might belong to, and he requests attribute values which will either verify or disprove the his hypothesis. If there is sufficient evidence that the example does belong to the hypothesized class, the expert stops and reports his decision. If the evidence disproves his hypothesis, then the expert hypothesizes a new class and proceeds to verify or disprove it. The test-hypothesis assumption embodies the basic idea that the expert sticks with one train of thought rather than spuriously jumping from one line of reasoning to another.

Referring again to Figure 1, we give an example of shortcoming detection assuming that the observed expert is using a test-hypothesis strategy. Again, *age_gt_60* can be explained because the problem-solver could be testing any of the four classes. When the attribute *history_nausea* is requested and the value *false* is received, we tentatively explain this by assuming that the observed expert is focusing upon the class *cochlear_age_plus_poss_menieres* and that the next attribute he will request is *speech* to complete the premise of Rule3. When this does not happen — when *history_noise* is requested next instead — we are forced to admit that the observed expert was not applying Rule3 when it requested *history_nausea* and furthermore that our tentative explanation for *history_nausea* no longer holds! The attribute request *history_nausea* is unexplained. When the observed expert requested *history_nausea*, he must have been using some knowledge which is absent from the critiqued KB. Thus imposing stronger assumptions on our observed expert allowed us to detect a KB shortcoming which was missed when the assumptions were weaker.

The assumptions made about the observed expert are not insignificant and should be given close examination. As mentioned above, when no assumptions are made, no shortcomings can be detected. As demonstrated in the two examples, the stronger the assumptions, the more leverage is available for detecting shortcomings. Yet these assumptions are not based on anything the problem-solver is observed to do but rather are *a priori* and provide a bias of sorts. Imposing stronger assumptions is not always better, though. If assumptions are too constraining, the observed expert will be expected to behave more rigidly than he actually does in reality. This leads to false positive shortcoming detection — KB shortcomings being detected when in fact there are none. Therefore, a tradeoff exists between detecting true shortcomings and avoiding the detection of nonexistent shortcomings.

So the ordering of attribute requests has proven valuable in the detection of KB shortcomings. Next we show how the detection process has already provided much of the information needed to localize the shortcoming.

4 Localizing the KB Shortcoming

In the previous section we explained a method of detecting KB shortcomings, but this method also takes us a long way toward localizing a shortcoming. The detection process gives us an **unexplained attribute**, the attribute request which could not be explained, and **focus facts**, the facts which were known at the time the unexplained attribute was requested. These focus facts deserve special attention because knowledge of them gave rise to the request of the unexplained attribute; therefore, they may be related to the unexplained attribute and to the shortcoming.

Furthermore, the focus facts can be used to identify a handful of **focus classes**, classes relevant to the shortcoming, because the known facts often rule out some classes leaving a subset of classes to focus upon. This is done using a set of labeled examples from a representative case library. All examples in the set which disagree with any of the focus facts are eliminated leaving a subset of examples which agree with all the facts known so far. The classes which these remaining examples belong to are the focus classes. So the shortcoming can be localized to a central attribute, a handful of focus facts, and a handful of focus classes.

5 Repairing the KB Shortcoming

Once the shortcoming has been localized, an attempt is made to repair it. Taking a rule-based approach, we try to repair the shortcoming by adding a rule of the form: $condition_1 \wedge condition_2 \wedge \ldots \wedge condition_N \longrightarrow class_i$ where each condition is an attribute/value pair such as *history_dizziness* = *true* or *temperature* > 102. Since the shortcoming has been localized to an unexplained attribute, a handful of focus facts, and a handful of focus classes, repairing the shortcoming consists of generating and empirically evaluating the rules formed from different combinations of these attributes and classes. While the localization process has narrowed down the number of attributes and classes, an exponential number of potential combinations still re-

main. The following paragraph describes a greedy approach to finding a good repair.

A repair can be generated by starting with a "seed repair" — a single-condition rule — and greedily adding other conditions to the rule. The single condition in the seed repair contains the unexplained attribute. For example if the unexplained attribute is *air*, then *air = mild*, *air = normal*, *air = severe*, etc. are each used in separate seed repairs. The class of a seed repair is any of the focus classes[1]. Each seed repair is empirically tested with respect to a set of labeled examples to see which examples satisfy the premise and for what percentage of those examples the repair gives the correct class. For example, of 150 training examples, 76 may satisfy the premise of the rule, and 28 of these 76 may match the class of the rule yielding a score of 28/76 = 36.8%. A set of new temporary repairs are then created by adding a condition to the seed repair. These conditions are derived from the focus facts[2]. If none of these new temporary repairs yield a higher score than the seed repair, then the seed repair is taken as the actual repair. If any of the temporary repairs are better, the best one becomes the new seed repair and the process is repeated.

Figure 2 gives an example of the repair process. The localization information in Figure 2(a) gives *air = mild* ⟶ *cochlear_age* as one of the possible seed repairs (other seed repairs would also exist, but for this example we will only examine this one). Its accuracy on the training examples for which the premise is true is 36.8%. In Cycle 1 three temporary repairs are created each by adding a new condition derived from a focus fact. Adding the new condition reduces the number of examples for which the premise is true, and a higher percentage of these examples may match the specified class. The best of the temporary repairs in Cycle 1 has a higher percent accuracy than the seed repair and therefore is chosen to become the new seed repair. Similarly, in Cycle 2 another conjunct is added, but in Cycle 3 the added conjunct does not improve accuracy; therefore, the current seed repair is selected as the final repair and is added to the KB.

While apprenticeship techniques provide good guidance for finding one attribute related to the shortcoming (the unexplained attribute), a weakness which becomes apparent in the repair stage is that these techniques give little guidance as to which of the previous attribute requests may be related to the shortcoming. An attribute chosen for the final repair may have been requested in close proximity to the unexplained attribute or it

[1]There are *#_of_unexplained_attribute_values* * *#_of_focus_classes* seed repairs.

[2]For nominal attributes the focus fact can be used exactly. For numeric attributes, the condition is a range including the specified value.

Unexplained attribute: *air*
Focus facts: *age_gt_60 = true*, *history_nausea = false*, *history_noise = false*
Focus classes: cochlear_age_plus_poss_menieres, cochlear_age

(a) The shortcoming localization information

Seed repair:	air = mild ⟶ cochlear_age	36.8%
Cycle 1		
Temp repair 1:	air = mild ∧ age_gt_60 = true ⟶ cochlear_age	65.9%
Temp repair 2:	air = mild ∧ history_nausea = false ⟶ cochlear_age	37.5%
Temp repair 3:	air = mild ∧ history_noise = false ⟶ cochlear_age	45.4%
Temp repair 1 becomes the new seed repair.		
Cycle 2		
Temp repair 1:	air = mild ∧ age_gt_60 = true ∧ history_nausea = false ⟶ cochlear_age	70.3%
Temp repair 2:	air = mild ∧ age_gt_60 = true ∧ history_noise = false ⟶ cochlear_age	92.6%
Temp repair 2 becomes the new seed repair.		
Cycle 3		
Temp repair 1:	air = mild ∧ age_gt_60 = true ∧ history_noise = false ∧ history_nausea = false ⟶ cochlear_age	92.3%
No repairs better than seed repair.		
Final repair:	air = mild ∧ age_gt_60 = true ∧ history_noise = false ⟶ cochlear_age	92.6%

(b) Three cycles of the repair generation process.

Figure 2: The shortcoming localization information in (a) guides the greedy repair construction in (b). New conditions are added to the seed repair, and the best one becomes the new seed repair.

may have been the very first attribute requested. Position in the ordered sequence seems to give little help in the repair stage forcing the use of weak search techniques.

6 Experiments

The SKIPPER program puts together the three tasks of apprenticeship: shortcoming detection, localization, and repair. The assumptions that SKIPPER makes about the observed expert are that it is rational and is using a test-hypothesis strategy as discussed in section 3. First, the order of attributes in an observed problem-solving session are examined, and all the unexplained attributes are found. Next, the *last* unexplained attribute in the step sequence is selected as the focus of shortcoming localization. Why the last unexplained attribute? Any unexplained attribute could have been chosen, but the last one has the most specific context since more attributes were requested before it and therefore provides the most localization information. Next, the best repairs are generated and are added to the KB, and then the whole process is repeated. When no more shortcomings can be repaired using the set of training examples, SKIPPER goes through a KB pruning stage in which unhelpful rules (rules whose removal does not decrease the overall accuracy on the training set) are removed from the KB.

Experiments were run using the standardized audiology dataset [Jergen, 1987] which has 69 attributes, 24 classes, and contains 226 examples. For each experiment N examples were used as a training set and the $226 - N$ remaining examples were used as a validation set. First, the C4.5 program used the training examples to create an initial KB of rules, and the accuracy of this initial KB was tested on the validation set. Next, SKIPPER refined this initial KB using the same training examples, and the accuracy of the final refined KB was checked using the validation set. Training sets were selected randomly from the pool of 226 examples.

The observed expert used in the experiments was a set of rules generated by C4.5 using all 226 audiology examples available. Because this master rule set was generated using all 226 examples, the knowledge contained in it can be viewed as the gold standard as far as classifying these 226 examples is concerned, and it thus serves as an "synthetic expert." An ordered sequence of requested attributes can be generated for a given example by observing the order in which the master rule set would request attributes in solving that example. During each run of the apprentice, such an ordered sequence was created for each example in the training set and was used in refining the KB.

Training Set Size	Initial KB (percent accuracy)	Refined KB (percent accuracy)	Improvement
5	21.3±4.8	38.9±7.8	17.6±9.2
10	25.6±12.1	44.1±9.8	18.5±15.6
25	55.2±7.2	62.9±6.0	7.7±8.4
50	59.6±8.1	71.7±7.6	12.1±8.3
75	67.3±5.0	77.2±4.4	9.9±2.6
100	67.9±6.6	82.1±3.9	14.2±8.4
125	72.3±4.4	82.3±5.0	10.0±4.8
150	75.0±5.3	85.4±5.0	10.4±5.5

Table 1: Accuracies and improvements for experiments with the standardized audiology set.

Experiments were run using training sets ranging in size from 5 up to 150, and the results are summarized in Table 1 and Figure 3. Percent accuracies are given along with their 99% confidence ranges. The "Improvement" column simply reflects the net percent accuracy gained. Each result is an average taken over 10 independent runs.

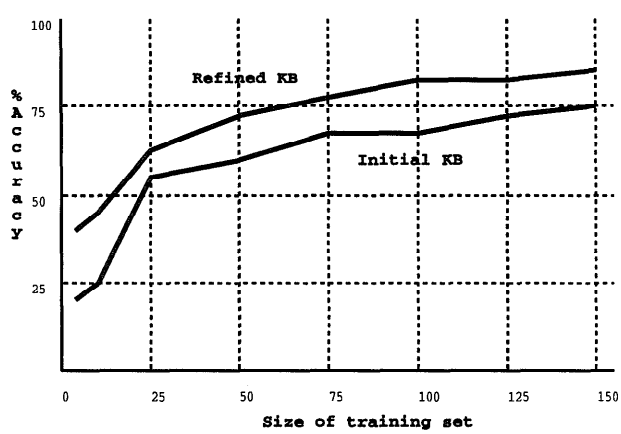

Figure 3: Accuracy of the initial KB and refined KB as a function of the training set size.

7 Discussion

An apprenticeship system refines its knowledge base by observing and analyzing the intermediate problem-solving steps taken by an expert (a generalization of the definition suggested in [Mitchell et al., 1985]). This paper has focused upon classification and specifically has sought what leverage can

be attained from attribute ordering — knowing the order in which an expert requested attribute values while classifying an example. This ordering yields leverage because it presents each action within the context in which that action was taken. An attribute request can be analyzed from the critiqued problem-solver's point of view with respect to what was and wasn't known at the time of the request. Since the apprentice attempts to explain *why* the expert takes certain actions, *a priori* assumptions must be made about the expert and his problem-solving strategy. These assumptions provide a crucial bias for KB shortcoming detection. The power of attribute ordering is that it does not rely on empirical calculations to discover attribute/class relationships. Rather, attribute/class relationships are suggested by attribute ordering and are only verified empirically thus requiring less empirical evidence.

This work has focused only on repairing a KB by adding new rules. Often, though, it may be desirable to mend a slightly imperfect rule [Ourston and Mooney, 1990]. SKIPPER inelegantly handles this situation by adding new rules and then pruning away useless rules as a final stage. The information contained in problem-solving step ordering could be used to point out imperfect rules. For example, if an attribute request is unexplainable, but that attribute is in the premise of a rule previously deemed irrelevant, then perhaps the false condition in that rule should be deleted or altered. Attribute ordering appears to provide little information, though, for the decision of *which* type of repair should be made: adding a new rule or altering an existing rule.

The experiments in this paper were designed to evaluate the improvement in accuracy that could be attained when an observed expert closely follows the assumed problem-solving strategy. ODYSSEUS [Wilkins, 1988] assumed a much more sophisticated problem-solving strategy and learned from a human expert, but it simply assumed that its knowledge of the expert's strategy was complete and correct. It could not be determined whether any deficiencies in ODYSSEUS's performance were due to flaws in the basic apprenticeship approach or flawed assumptions about the expert's strategy. We assumed the observed expert used a test-hypothesis strategy, and the synthetic expert used in the experiments did in fact use this strategy. These experiments bolster confidence in the basic apprenticeship approach. The future direction of this work is to scale the principles learned about attribute ordering up to a more sophisticated model of problem-solving such as is used in NEOMYCIN [Clancey, 1985], CELIA [Redmond, 1992], and ODYSSEUS.

Acknowledgements

We would like to thank Tom Ioerger and Yong Ma for helpful comments on earlier drafts of this paper.

References

[Clancey, 1985] Clancey, W. J. (1985). Heuristic classification. *Artificial Intelligence*, 27:289–350.

[Davis, 1979] Davis, R. (1979). Interactive transfer of expertise: Acquisition of new inference rules. *Artificial Intelligence*, 12:121–158.

[Dent et al., 1992] Dent, L., Boticario, J., McDermott, J., Mitchell, T., and Zabowski, D. (1992). A personal learning apprentice. In *Proceedings of the 1992 National Conference on Artificial Intelligence*, pages 96–103, San Jose, CA.

[Jergen, 1987] Jergen (1987). Original owner of the audiology dataset.

[Mahadevan et al., 1993] Mahadevan, S., Mitchell, T., Mostow, J., Steinberg, L., and Tadepalli, P. (1993). An apprentice-based approach to knowledge acquisition. *Artificial Intelligence*, 64(1):1 – 52.

[Mitchell et al., 1985] Mitchell, T. M., Mahadevan, S., and Steinberg, L. I. (1985). LEAP: A learning apprentice for VLSI design. In *Proceedings of the 1985 IJCAI*, pages 573–580, Los Angeles, CA.

[Ourston and Mooney, 1990] Ourston, D. and Mooney, R. (1990). Changing the rules: A comprehensive approach to theory refinement. In *Proceedings of the 1990 National Conference on Artificial Intelligence*, pages 815–820.

[Quinlan, 1993] Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning.* Morgan Kaufmann Publishers.

[Redmond, 1992] Redmond, M. (1992). *Learning by Observing and Understanding Expert Problem Solving.* PhD thesis, Georgia Inst. Tech.

[Tecuci and Kodratoff, 1990] Tecuci, G. and Kodratoff, Y. (1990). Apprenticeship learning in imperfect domain theories. In Kodratoff, Y. and Michalski, R. S., editors, *Machine Learning: An Artificial Intelligence Approach, Volume III*, pages 514–552. San Mateo: Morgan Kaufmann.

[Wilkins, 1988] Wilkins, D. C. (1988). Knowledge base refinement using apprenticeship learning techniques. In *Proceedings of the 1988 National Conference on Artificial Intelligence*, pages 646–651, Minneapolis, MN.