Question Asking to Inform Preference Learning: A Case Study

Melinda Gervasio **Karen Myers**
AI Center, SRI International
333 Ravenswood Ave.
Menlo Park, CA 94025
{melinda.gervasio, karen.myers}@sri.com

Marie desJardins
Univ. of Maryland Baltimore County
1000 Hilltop Circle
Baltimore, MD 21250
mariedj@cs.umbc.edu

Fusun Yaman
BBN Technologies
10 Moulton Street
Cambridge, MA 02138
fyaman@bbn.com

Abstract

The prospect of teaching a computer how to perform a task through demonstration rather than programming has tremendous appeal. However, learning purely from a demonstration trace is a difficult challenge. One way to ease the learning problem is to supplement the demonstration with information provided by asking questions of the demonstrator. This paper presents a case study that explores how to instantiate a question asking framework to select questions for a particular type of learner used within learning by demonstration systems, namely a lexicographic preference learner. Experimental results show that, generally speaking, judicious question asking can improve learning performance. However, the study makes clear the importance of understanding the value of different types of information to learning in different contexts.

Introduction

There has been a resurgence of interest in the field of learning by demonstration in the last few years, motivated by the desire to make it easier for nonprogrammers to teach software systems tasks that are mundane and repetitive (Allen et al., 2007; Gervasio et al., 2008; Little et al., 2007). Given recent technical progress in the area, research has shifted from an initial focus on learning small, self-contained procedures for simple tasks to the harder problem of learning coherent bodies of problem-solving knowledge that can be reused on different but similar tasks.

For example, the POIROT project seeks to use learning by demonstration technology to develop a repertoire of reusable problem-solving knowledge to support medical evacuation planning (Burstein et al., 2008). A typical problem in this domain requires the scheduling of transport for patients at different locations to appropriate treatment facilities, taking into account criteria such as the type and severity of injury, treatment times, facility capabilities, and transit times. Learning problem-solving knowledge of this scope requires not only the trace generalization mechanisms pioneered in programming by demonstration research, but additional technologies for learning preferences over choices in the domain, conditional branches, and abstraction structures such as methods and loops.

The technical challenges inherent to learning procedural knowledge for rich problem spaces are substantial. A single demonstration sequence—or even a few of them—will generally lack the breadth to define the general case. A human demonstrator may be willing to provide a small number of examples, but generally will not provide enough for unambiguous learning of the intended generalized problem-solving knowledge. In addition, the demonstrator may make mistakes or may become distracted and include irrelevant or unnecessary steps in the demonstration, thus further complicating the learning process.

For these reasons, we believe that a successful learning by demonstration capability will need additional information from the demonstrator to simplify the learning problem. For example, the PLOW system relies on a running narrative by the demonstrator to focus the learning process (Allen et al., 2007). This type of mixed-initiative interaction is common in other subfields of AI that have sought to build technologies for real-world tasks (Bresina et al., 2005; Myers et al., 2003; Tecuci et al., 2007).

In prior work, we defined a framework called QUAIL (Question Asking to Inform Learning) for *asking questions* of the demonstrator as a way of supplementing the information provided by a demonstration trace (Gervasio and Myers, 2008). This framework is being developed in the context of the POIROT project, but applies generally to a range of learning by demonstration frameworks. That prior work defined a catalog of questions designed to inform the learning by demonstration process, covering areas such as the function and causality of elements in the demonstration trace, abstraction, alternatives and justifications, limitations on learned knowledge, and the process of learning. The work also defined a metalevel capability for managing question asking that reasons about gaps in the learned knowledge to identify appropriate questions. The metareasoning takes a limited rationality perspective in selecting questions to pose to the demonstrator, trading off the anticipated utility of the missing knowledge with the cost of obtaining it.

This paper reports on an experimental effort to understand how to instantiate the QUAIL question asking framework for an individual learning component. In particular, we describe a case study in which we applied our question asking framework to a particular learner, both to understand how to model questions for this style of learner, and to investigate different strategies for selecting questions to ask on the learner's behalf.

The specific learner that we consider is a system called CHARM (Yaman et al., 2008). CHARM learns a lexicographic preference model for ordering objects from training data consisting of pairs of the form $a<b$ indicating that a is at least as preferred as b. CHARM is being used within the POIROT system to learn several preference orders; here, we focus on its use for learning the order in which to process patients.

Our experimental evaluation shows that, by and large, question answering can be an effective mechanism for improving learning performance in CHARM. However, the study makes clear the importance of understanding the value of different types of information for learning in different contexts. It also yielded some surprising results with respect to the knowledge nonmonotonicity, best summarized by the phrase "a little knowledge can be a dangerous thing".

The paper is organized as follows. We begin with brief descriptions of the question asking and preference learning frameworks. Next, we present the experimental design and results followed by a discussion of related work. We conclude with a summary of the findings and some directions for future work.

Question Selection Framework

Within a learning by demonstration setting, it is natural to expect that the demonstrator will focus on interacting with the system and so be tolerant of questions. Nevertheless, it is important to impose restrictions on question asking to keep interactions to a reasonable level. In addition, different questions could be expected to provide different levels of value to the learning process, where value is likely to be correlated with context.

For these reasons our question asking framework, QUAIL, formulates the question selection task in terms of a cost-benefit analysis that draws on models of *utility* and *cost* for individual question instances. Details of our approach can be found in (Gervasio and Myers, 2008); of greatest relevance for this case study is the fact that question-answering budgets are limited.

QUAIL includes two possible control strategies for managing question selection. An *asynchronous* control strategy supports a continuous model of question asking, meaning that questions can be posed throughout the learning process. In contrast, a *synchronous* approach allows questions to be asked only at one (or possibly more) designated points during learning.

Asynchronous strategies have the potential to provide greater impact as information can be obtained when needed during learning. They also permit the possibility of conditional question asking, where the answer to one question can inform the choice of subsequent questions. However, asynchronous strategies present challenges in trading off the value that can be obtained by asking a question at a given point in time with the potential opportunity cost of not being able to ask other questions later because of resource limitations. For this reason, our case study focuses on the simpler case of synchronous selection. In particular, we consider an approach where questions are selected from a pool of possible questions at a single point in time and answered as a unit.

The synchronous question selection problem can be formulated as follows. We assume the following functions defined for a collection of questions Q.

$$Cost(Q) = \sum_{q \in Q} Cost(q)$$

$$Utility(Q) = \sum_{q \in Q} Utility(q)$$

Definition (Synchronous Question Selection). Given a collection of questions $Q=\{q_1 \ldots q_n\}$ and a budget B, determine a subset $Q' \subseteq Q$ with $Cost(Q') \leq B$ such that there is no $Q'' \subset Q$ for which $Cost(Q'') \leq B$ and $Utility(Q'') > Utility(Q')$.[1]

This framing of synchronous question selection maps directly to the *knapsack* problem (Kellerer et al., 2005). Although the knapsack problem is NP-complete, there are pseudo-polynomial time dynamic programming algorithms that can generate solutions with $O(nB)$ runtime (Garey and Johnson, 1979). Given the limited budgets to be imposed on question asking (to avoid excessive interactions with the demonstrator), efficient question selection is possible using these algorithms.

Preference Learning in CHARM

The lexicographic preference learning problem addressed by CHARM can be characterized as follows: given a collection of objects with an associated set of attributes and a set of observations of the form $a<b$ indicating that object a is at least as preferred as object b, learn a partial order on the object attributes that is consistent with the observations. The partial order in turn defines a preference model for the full set of objects.

CHARM (Yaman et al., 2008) approximates the target preference model by constructing and reasoning with a collection of models that are consistent with the observations. Basically, it learns a partial order on the attributes such that every linearization will be consistent with the observations.

For simplicity, assume that all attributes are binary and that the preferred value of each attribute is known. Given objects a and b and the partial order, the preferred object is

[1] This formulation of the problem assumes that the questions in Q are independent of each other, i.e., obtaining the answer to one question does not impact the utility of answering the others.

determined through a *voting scheme* as follows: Among their attributes that differ, those that have the smallest rank (and are hence the most salient) in the partial order vote to choose the preferred object. The object that has more preferred values among the voting attributes is declared to be the preferred one. If the votes are equal, then the objects are equally preferred.

CHARM maintains the minimum possible rank for every attribute that does not violate an observation with respect to the voting scheme explained above. Initially, all attributes are considered equally important (rank of 1). The algorithm loops over the set of observations until the ranks converge. At every iteration and for every pair, the voting predicts a winner using the current partial order. If the prediction is correct, then the ranks stay the same. Otherwise, the ranks of the variables that voted for the wrong object are incremented, thus reducing their importance.

It is easy to generalize this algorithm for the case where the preferred value of a binary attribute is unknown. The idea is to have two ranks per attribute, each representing the importance of one attribute value. For attributes with multi-values where there is a monotonic order on the values, the same generalization can be used to learn the preferred direction on the monotonic order. For example if the attribute represents time, then CHARM can learn whether *later* is more preferred than *earlier* or vice versa.

If the observations do not contain any hidden ties (i.e., two objects are equally preferred but the demonstrator arbitrarily picked one over the other) then CHARM is guaranteed to converge to the correct lexicographic preference model (given enough data). Note that hidden ties are possible only when there are irrelevant attributes. Alternatively, if the preferred value of each relevant attribute is known then the algorithm is guaranteed to converge to the correct model regardless of the existence of arbitrarily broken ties in the observations. The learning algorithm has a mistake-bound of $O(n^2)$, where n is the number of attributes, because each mistake increases the sum of the potential ranks by at least 1 and the sum of the ranks in any lexicographic model is $O(n^2)$.

Case Study Details

The case study focused on a particular preference learning problem, namely that of learning a lexicographic preference model for ordering patients based on training data extracted from a demonstration trace. Here, we describe specializations of the learning and question asking frameworks for this problem.

Within POIROT, the training data that underpins the preference learning is implicit in the demonstration trace. Training instances consist of pairs of the form $P1<P2$ indicating that patient $P1$ is at least as preferred as patient $P2$. These training instances are extracted from demonstration traces by considering the order in which the demonstrator processed patients.

Features for Learning

As noted above, the preference learner bases its generalizations on features of the objects that it learns to rank. Patients in POIROT's medical evaluation planning domain are described by eight attributes:

- TriageCode: indication of the severity of the injury (one of five predefined values)
- WoundType: type of injury (one of ten predefined values)
- PersonClass: category for personnel (one of seven predefined values)
- ReadyForTransport: time at which the patient is available for transport
- LatestArrivalTime: latest time at which the patient should reach the assigned medical facility
- Special Needs: any special equipment or personnel required by the patient (e.g., ventilator, translator)
- Origin: location of the patient
- Destination: location to which the patient is to be brought

As discussed earlier, CHARM requires that the attribute values be ordered, and learns whether higher or lower values are preferred for each attribute. This requirement eliminates the last three features, leaving the first five features for learning a preference model for patient ordering.

Question Categories

(Gervasio and Myers, 2008) defines a comprehensive collection of questions to support question asking for learning by demonstration. Not all of those questions are relevant for preference learners. For example, a portion of the questions target generalization of structure from demonstration traces, addressing concepts such as causal linkage and the extraction of loops and methods.

The QUAIL and CHARM teams jointly agreed that the questions listed in Figure 1 would be the most natural to consider within the case study. Q1 asks whether a given patient should be handled before another, and so establishes that the first patient is at least as preferred as the second (but may not be more preferred). Q2 asks whether one of the five patient attributes defined above is relevant to the ordering. Q3 asks whether a given attribute is more important than another for learning the ordering. Q4 asks whether, for a given attribute, one value is considered more important than another.

CHARM currently interprets negative answers to these questions as positive answers to the inverse question. So, for example, an answer of 'no' to the question "Is Attr1 more important than Attr2?" is taken to mean that Attr2 is more important than Attr1. Because of this, the experiment could not include questions for which a negative answer could have meant that Attr1 and Attr2 have the same rank in the learned preference model.

Q1. *Object ordering:* Should <*patient1*> be handled before <*patient2*>?

Q2. *Attribute relevance:* Is <*attribute*> relevant to the ordering?

Q3. *Attribute ordering:* Is <*attribute1*> more important than <*attribute2*>?

Q4. *Attribute value ordering:* For <*attribute*>, is <*value1*> more important than <*value2*>?

Figure 1. Question Types for CHARM Experimental Evaluation

As noted above, the question selection mechanisms in the QUAIL framework support a general model of costs for questions. The intent behind the model was to associate costs in a way that roughly reflected the ‘cognitive burden’ imposed on the demonstrator to answer them. After consideration of the questions in Figure 1, we decided that the questions were (by and large) equivalent in cognitive burden and hence a uniform cost model should be adopted for the experimental evaluation.[2]

Question Selection

Question selection uses the knapsack algorithm described earlier. However, given the adoption of the uniform cost model for questions, selection effectively reduces to choosing questions in decreasing utility order, with randomization applied to questions with equal utility.

As part of the experimentation, we were interested in investigating the impact of different question utility models on performance. For simplicity, we based utility models on question type. We considered five different utility profiles for questions, which can be characterized as follows.

- *Uniform*: assigns the same utility to all questions
- *Object Ordering*: assigns highest utility to questions of type Q1
- *Attribute Ordering*: assigns highest utility to questions of type Q2
- *Attribute Relevance*: assigns highest utility to questions of type Q3
- *Attribute Value Ordering*: assigns highest utility to questions of type Q4

As discussed further below, no more than five questions are asked in the experiment. Because at least five questions can be posed for each question type, additional details of the utility models used are irrelevant here.

Answer Incorporation into CHARM

CHARM incorporates answers to questions in three different ways. The answers to object ordering questions (Q1) are simply transformed into additional training data. The answers to attribute relevance (Q2) and attribute-value ordering (Q3) questions are used at the rank initialization phase. For instance, if an answer from QUAIL states that *attribute1* is not relevant then the rank for all values of *attribute1* is set to more than the number attributes (less importance than any other attribute). This effectively prevents *attribute1* from voting for any objects, thus leaving it out of the decision process. Similarly, the less preferred value of an attribute is assigned a high rank that disables the attribute-value in the voting process.

The answers to attribute order queries (Q4) are compiled into a set of constraints. A constraint representing “*attribute1 is more important than attribute2*” is violated iff the minimum rank for any value of *attribute1* is greater than or equal to the minimum rank for any value of *attribute2* (note that a lower rank suggests more importance). After the learning algorithm converges, i.e. the ranks of the attribute-values do not change, the algorithm iterates over the set of constraints. Every time CHARM detects a violated constraint, CHARM minimally modifies the ranks to satisfy the constraint. Basically, CHARM will increment the rank of *attribute2* values (hence decrease their importance) beyond the minimum rank among all *attribute1* values. After the modification the algorithm goes back to iterating over the data until ranks converge to a point where either all constraints are satisfied or exits after detecting an inconsistency.

Experiment Design

The overall objective for the experiment was to evaluate the ability of QUAIL’s question asking facility to improve preference learning within CHARM. In particular, the experiment was designed to investigate how different question selection strategies impact CHARM’s ability to learn lexicographic preference models for ordering POIROT patients, compared to CHARM running without question asking.

In real use, questions selection by QUAIL would be presented to the human demonstrator for answering. To facilitate experimentation, answers for the individual target preference models were precompiled into an oracle that QUAIL could query at runtime.

[2] The original experiment plan called for the inclusion of set-oriented counterparts to the questions in Figure 1 that involved selecting values from a collection of candidates. For example, the set-oriented variant of Q2 is *Which attributes in <attribute-set> are relevant to the ordering?* Such questions were to have higher associated costs, given the additional mental effort necessary to perform the selection task compared to the simpler yes/no counterpart to the question. It was not possible to develop mechanisms to absorb the answers for those questions into CHARM for the initial round of experiments but those questions will be considered in the future.

The experimental hypotheses for the case study can be summarized as follows.

- H1. The additional knowledge provided by question answering will improve CHARM's performance.
- H2. Higher question budgets will improve performance.
- H3. Different utility models will result in different questions being selected and differing performance.
- H4. Question answering will provide more value when learning more complex preference models.

Experimental Setup

The target lexicographic preference models for the evaluation were generated randomly, varying the number of relevant attributes and the ordering between them. Patients for the ten training and test sets were drawn randomly from a pool of 186 patients defined in the POIROT database.

A range of different target preference models was investigated corresponding to learning problems of differing complexity. Complexity was modulated by varying both the amount of training data made available and the number of attributes that were relevant to defining the preference model.

Learning performance was measured in terms of CHARM's accuracy in predicting preferences between pairs of patients. More precisely, the accuracy was computed as the ratio of ties and non-ties between the pairs of patients that CHARM predicted correctly to the total number of patient pairs.

CHARM was trained on a single problem defined over five patients, and then tested on four problems defined over five patients each. This was designed to mimic the actual learning scenario in POIROT, where CHARM would learn from data from a single demonstration for use in future situations. Training data consisted of pairwise preference information for (a subset of) the five patients in the problem specification, corresponding to what would be extracted from a demonstration trace where the expert user schedules (a subset of) the patients in a particular order. For each preference model, we report performance as the average over the ten runs on the ten randomly generated training and test sets.

Results

Description of Graphs

Figure 2 presents the main results from the experimental evaluation. The four graphs show learning performance for six different types of runs: the baseline consisting of CHARM running on its own, along with CHARM running with question asking enabled for the five question utility models described earlier.

The figures along the top show the results when only a subset of the five possible patient attributes was relevant to defining the preference model; the figures along the bottom show results when all five attributes were relevant. The figures on the left show the case in which only two of the five patients were seen in training data, while the figures on the right correspond to cases in which data on all five patients was seen. For each of these four cases, evaluations were done in which one, two, three, and five questions were selected.[3]

The variations in number of relevant patient attributes and number of patients in the training data were introduced to vary the complexity of the underlying learning problem. As can be seen by the baseline performance in these graphs, increasing the number of patients seen results in simpler problems. However, although one would expect that increasing the number of relevant attributes to learning the model would yield a more difficult problem; in fact, that is only the case when all five patients are seen in the training data. In particular, CHARM's baseline performance was weakest in the case where all five patient attributes were relevant but only two of the five patients were seen (Figure 2c) while baseline performance was strongest in the case where all attributes were relevant and all patients were seen (Figure 2d).

Discussion of Results

The experiments (generally) validate our hypothesis that limited question answering enables CHARM to learn more accurate preference models. The one case where this claim does not hold is in the graph on the bottom right, which corresponds to the easiest problem considered (i.e., has the highest baseline score). In this particular situation, the learning is already fairly successful (approximately 82% accuracy). Appropriately targeted question asking should still enable greater performance but clearly requires more sophisticated strategies than considered here.

The results also (generally) show that performance improvements increase when more questions are allowed, as evidenced by the fact that the non-baseline results generally trend upward as more questions are asked.

With respect to the hypothesis that question answering will provide more value on more difficult learning tasks, the results are inconclusive. As noted above, question answering provided little incremental value for the easiest case on the bottom right while faring much better on the other problems. However, the biggest performance gains (choosing the best results over all utility models) were seen on the problems with baseline scores of 70% and 75% accuracy, with good but smaller gains on the hardest problem with a baseline of 63% accuracy.

One key finding was the need for richer utility models for questions. We had hypothesized before running the experiments that attribute relevance questions (type Q2) would provide the greatest benefit for learning. As it turned

[3] Figures 2b and 2d do not show data for the Object Ordering utility model because the observations given to CHARM for these cases contained the data that those questions could provide.

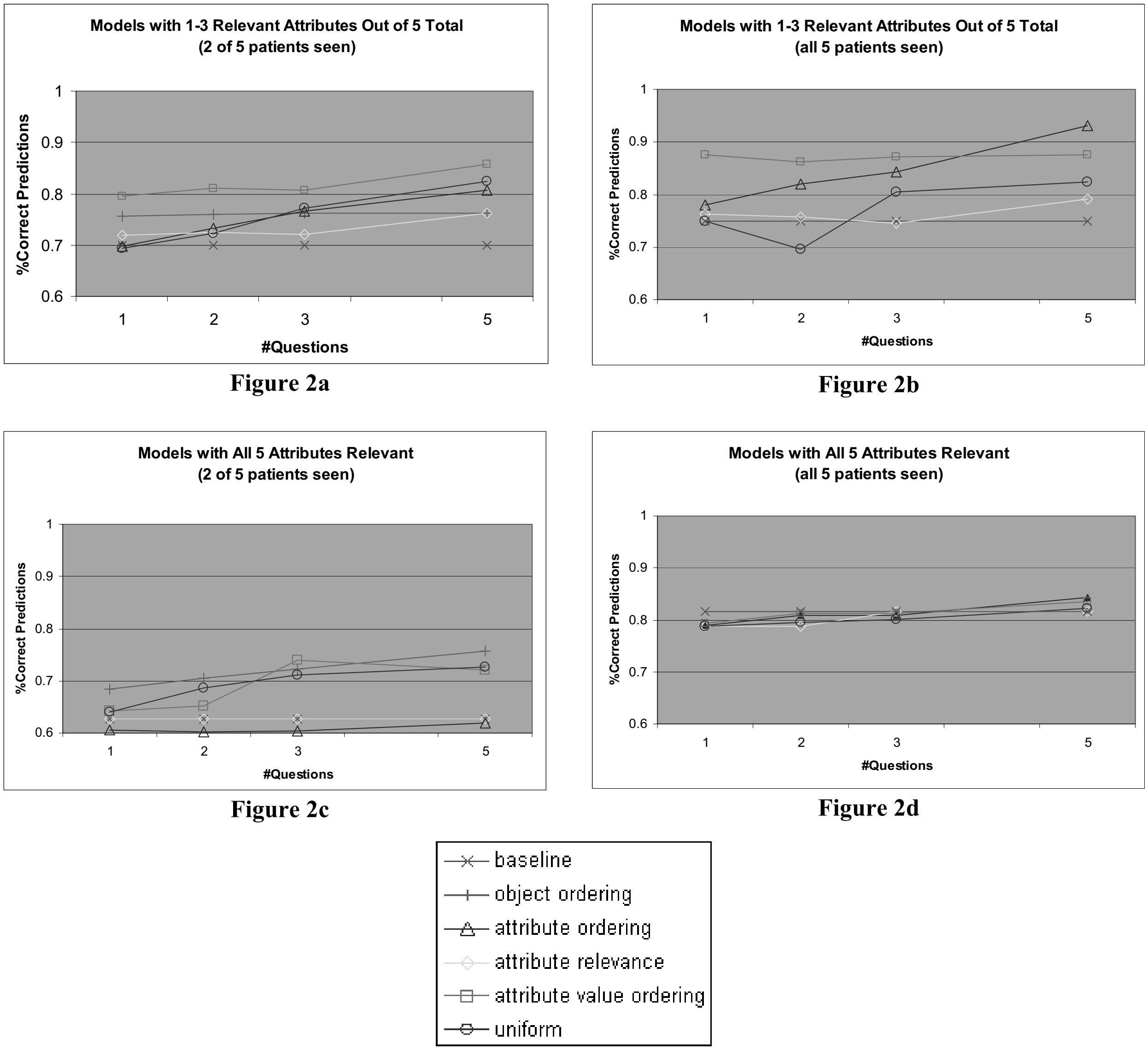

Figure 2a

Figure 2b

Figure 2c

Figure 2d

Figure 2. Experimental Results

out that was not the case. In particular, the results showed that questions of type Q4 (attribute value ordering) provided the greatest value overall. More generally, none of the utility models that we investigated proved to be superior in all cases. Our sense is that trying to knowledge engineer high-quality utility models will prove to be difficult. Moving forward, it would seem valuable to explore the use of learning techniques to create the models, drawing on features that would include properties of individual questions, information about previously asked questions, and the current state of the learned model.

Another unexpected result in the analysis was that the use of only one or two questions sometimes led to decreased learning performance. This negative result derives from the specific way in which CHARM incorporates answers from QUAIL. In particular, on questions of attribute ordering, given the information that *attr1* is more important than *attr2*, the rank of *attr2* is lowered in the preference model. This works if *attr2* also happens to be less important than the other attributes, but not if it is more important, as more often was the case in our experiments. With the right set of answers, CHARM could impose the correct relative ordering on the attributes, improving its performance over the baseline. This problem might thus be overcome by favoring questions that provide information that 'links' to other known preference information. More generally, this phenomenon raises an interesting problem for managing question selection that we had not anticipated originally, as we had expected that POIROT learners would behave monotonically with respect to answer information.

Related Work

Our work is closely related to the *active learning* paradigm which, in its most general sense, lets the learning algorithm request labels for unlabeled instances. Active learning has been shown to improve the performance of learners in the context of classifiers (Cohn et al., 1994; Balcan et al., 2006; Dasgupta, 2005) and grammar induction (Angluin, 1987). Without budget restrictions, one can perform a binary search over all models using a sequence of queries. Our work limits the number of queries by imposing a budget on question asking. Furthermore, our question repertoire is richer than just membership queries.

Among other work, active model selection is similar to our approach (Madani et al., 2004). It tries to determine a sequence of tests with different costs that would differentiate between possible models. In contrast, our evaluations are based on a synchronous approach where the selection of a question is independent of the answers to other questions. QUAIL can act as the question asking agent for multiple learners, drawing on a shared budget. For this reason, QUAIL adapts a question selection mechanism that is independent of the underlying learning tasks. The question utility model acts as a bridge between QUAIL and a learner, and lets QUAIL select a set of questions to improve overall system performance.

Conclusions

Informed question asking capabilities have the potential to enable learning by demonstration technology to work effectively on much more complex problems than is possible today, through the judicious injection of knowledge to supplement demonstration traces. To date, however, there has been relatively little effort focused on understanding how question asking can be used within learning systems, as well as assessments of the impact that question asking can have on learning performance.

The case study reported in this paper takes a first step toward furthering our knowledge of how to design and operationalize question asking facilities for a particular class of learner used in learning by demonstration systems, namely a learner for lexicographic preference models. The results show that, generally speaking, question asking can improve learning performance on this class of problems. However, it also reveals the importance of tailoring question asking strategies to the characteristics of individual learners to ensure the full benefits of question asking. Finally, the results highlight the importance of developing good utility models to guide the question selection process.

Future directions for this work include performing similar studies for other sorts of learners in isolation and in aggregate, and developing methods to learn high-quality question utility models.

Acknowledgments. This work was supported by the Defense Advanced Research Projects Agency and the United States Air Force through BBN Technologies Corp. on contract number FA8650-06-C-7606.

References

Allen, J., Chambers, N., Ferguson, G., Galescu, L., Jung, H., Swift, M., and Taysom, W. 2007. PLOW: A collaborative task learning agent. *Proc. 22nd Conference on Artificial Intelligence*, Vancouver, Canada.

Angluin, D. 1987. Learning regular sets from queries and counter-examples. *Information and Computation,* 75(2):87-106.

Balcan, M., Beygelzimer, A., and Langford, J. 2006. Agnostic active learning. *Proc. ICML*, 65-72.

Bresina, J., Jonsson, A., Morris, P., and Rajan, K. 2005. Activity planning for the Mars exploration rovers. *Proc. 15th International Conference on Automated Planning and Scheduling.* Monterey, CA.

Burstein, M., Laddaga, R., McDonald, D., Benyo, B., Roberston, P., Hussain, T., Brinn, M., McDermott, D. 2008. POIROT - Integrated Learning of Web Service Procedures. *Proc. AAAI-08,* 1274-1279.

Cohn, D., Atlas, L., and Ladner, R. 1994. Improving Generalization with Active Learning. *Mach. Learn.* 15 (2): 201-221.

Dasgupta, S. 2005. Coarse sample complexity bounds for active learning. *Advances in Neural Information Processing Systems*.

Garey, M. R., and Johnson, D. S. Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, 1979.

Gervasio, M., Lee, T. J., and Eker, S. 2008. Learning email procedures for the desktop. *Proc. AAAI 2008 Workshop on Enhanced Messaging,* Chicago, IL.

Gervasio, M., and Myers, K. 2008. Question Asking to Inform Procedure Learning. *Proc. AAAI-08 Workshop Metareasoning: Thinking about thinking.*

Kellerer, H., Pferschy, U., and Pisinger, D. 2005. *Knapsack Problems*. Springer Verlag.

Little, G., Lau, T., Cypher, A., Lin, J., Haber, E., and Kandogan, E. 2007. Koala: Capture, share, automate, personalize business processes on the Web. *Proc. CHI,* San Jose, CA.

Madani, O., Lizotte, D.J., and Greiner, R. 2004. Active model selection. *Proc. UAI.* 357-365.

Myers, K. L., Jarvis, P., Tyson, W. M., and Wolverton, M. J. 2003. A mixed-initiative framework for robust plan sketching. *Proc.13th International Conference on Automated Planning and Scheduling*. Trento, Italy.

Tecuci, G., Boicu, M., and Cox, M. T (eds.) 2007. *AI Magazine*, Special Issue on Mixed-initiative Assistants, Volume 28(2).

Yaman, F., Walsh, T.J., Littman, M.L., and desJardins, M. 2008. Democratic approximation of lexicographic preference models. *Proc. ICML,* 1200-1207.