Automation Intelligence for the Smart Environment

G. Michael Youngblood, Edwin O. Heierman, Lawrence B. Holder, and Diane J. Cook
Department of Computer Science and Engineering
The University of Texas at Arlington
Arlington, TX 76019-0015
{youngbld,heierman,holder,cook}@cse.uta.edu

Abstract

Scaling AI algorithms to large problems requires that these algorithms work together to harness their respective strengths. We introduce a method of automatically constructing HHMMs using the output of a sequential data-mining algorithm and sequential prediction algorithm. We present the theory of this technique and demonstrate results using the MavHome intelligent environment.

1 Introduction

An important component of an intelligent environment is to anticipate actions of a human inhabitant and then automate them. The decision of which action to execute must be correct in order to avoid creating excess work for humans in the form of correcting wrong automated actions and performing manual actions.

We examine the problem of learning human inhabitant behavioral models in the MavHome intelligent environment and using this to automate the environment. An event in the environment is described by the time of the event, the device/sensor zone, the device/sensor number, the new value of the device or sensor, the source of the vent (e.g., sensor network, powerline controller), and the inhabitant initiating the event (if known).

2 Solution Strategy

To automate the environment, we collect observations of manual inhabitant activities and interactions with the environment. We then mine sequential patterns from this data using the ED sequence mining algorithm. Finally, a hierarchical Markov model is created using low-level state information and high-level sequential patterns, and is used to learn an action policy for the environment.

2.1 Mining Sequential Patterns Using ED

Our data mining algorithm, ED, mines sequential patterns from observed activities. Data is processed incrementally and sequential patterns are mined according to their ability to compress the data using the Minimum Description Length principle. Periodicity (daily, every other day, weekly occurrence) of episodes is detected using autocorrelation and included in the episode description. If the instances of a pattern are highly periodic (occur at predictable intervals), the exact timings do not need to be encoded and the resulting pattern yields even greater compression value.

2.2 Predicting Activities Using ALZ

To predict inhabitant activities, we borrow ideas from text compression. By predicting inhabitant actions, the home can automate or improve upon anticipated events that inhabitants would normally perform in the home. Our Active LeZi (ALZ) algorithm [Gopalratnam and Cook, 2005] approaches this problem from an information-theoretic standpoint. ALZ incrementally parses the input sequence into phrases and, as a result, gradually changes the order of the corresponding Markov model that is used to predict the next symbol in the sequence. Frequency of symbols is stored along with phrase information in a trie, and information from multiple context sizes are combined to provide the probability for each potential symbol as being the next one to occur. In our experiments, ALZ proved to be a very accurate sequential predictor. However, accuracy is further improved when the task is restricted by ED to only perform predictions when the current activity is likely to be part of a frequently-occurring pattern.

2.3 Decision Making Using ProPHeT

Work in decision-making under uncertainty has popularized the use of Hierarchical Hidden Markov Models and Partially Observable Markov Decision Processes. Recently, there have been many published hierarchical extensions that allow for the partitioning of large domains into a tree of manageable POMDPs [Pineau *et al.*, 2001; Theocharous *et al.*, 2001]. Although the Hierarchical POMDP is appropriate for an intelligent environment domain, current approaches generally require *a priori* construction of the HPOMDP. Given the large size of our domain, we need to seed our model with structure automatically derived from observed inhabitant activity data.

Unlike other approaches to creating a hierarchical model, our decision learner, ProPHeT, actually automates model creation by using the ED-mined sequences to represent the abstract nodes in the higher levels of the hierarchy. Lowest-level states correspond to an environment state representation together with an ALZ-supplied prediction of the next inhabitant action. To learn an automation strategy, the agent explores the effects of its decisions over time and uses this experience within a reinforcement learning framework to form control policies which optimize the expected future reward. The current version of MavHome receives negative reinforcement when the inhabitant immediately reverses an automation decision (e.g., turns the light back off) or an automation decision contradicts user-supplied safety and comfort constraints (e.g., do not let the temperature exceed 100 degrees).

3 Environments

All of the algorithms described here are implemented in MavHome and are being used to automate two environments, shown in Figure 1. The MavLab environment contains work areas, cubicles, a break area, a lounge, and a conference room. MavLab is automated using 54 X-10 controllers and the current state is determined using light, temperature, humidity, motion, and door/seat status sensors. The MavPad is an on-campus apartment hosting a full-time student occupant. MavPad is automated using 25 controllers and provides sensing for light, temperature, humidity, leak detection, vent position, smoke detection, CO detection, motion, and door/window/seat status sensors.

Figure 1: The MavLab (left) and MavPad (right) environments.

4 Case Study

As an illustration of these techniques, we have evaluated a week in an inhabitant's life with the goal of reducing the manual interactions in the MavLab. The data was generated from a virtual inhabitant based on captured data from the MavLab and was restricted to just motion and lighting interactions which account for an average of 1400 events per day. We trained ALZ and ED on real data and then repeated a typical week in our ResiSim simulator to determine if the system could automate the lights throughout the day in real-time.

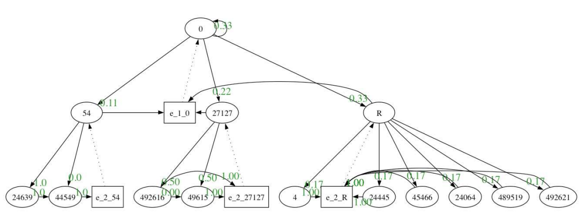

Figure 2: ProPHeT generated HHMM with production nodes abstracted.

ALZ processed the data and converged to 99.99% accuracy after 10 iterations through the training data, and accuracy was 54% on test data. When automation decisions were made using ALZ alone, interactions were reduced by 9.7% on average. Next, ED processed the data and found 3 episodes to use as abstract nodes in the HPOMDP, as shown in Figure 2. The HHMM model with no abstract nodes reduced interactions by 38.3%, and the combined-learning system (ProPHeT bootstraped using ED and ALZ) was able to reduce interactions by 76%, as shown in Figure 3.

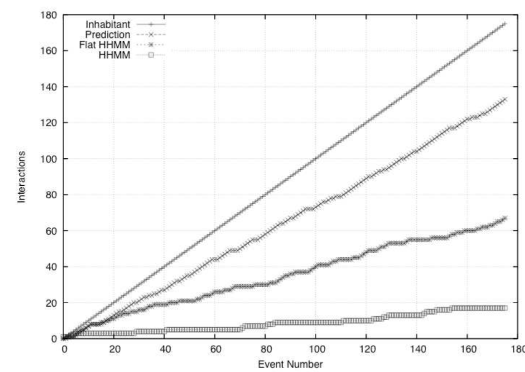

Figure 3: Interaction reduction.

Experimentation in the MavPad using real inhabitant data has yielded similar results. In this case, ALZ alone reduced interactions from 18 to 17 events, the HPOMDP with no abstract nodes reduced interactions by 33.3% to 12 events, while the bootstrapped HPOMDP reduced interactions by 72.2% to 5 events.

In this research we have shown that learning algorithms can successfully automate an intelligent environment. We see that synergy between these algorithms can improve performance, as ED-produced abstractions in the hierarchy coupled with a prediction produced by ALZ improved automation performance for ProPHeT. A full system deployment in the MavPad is currently being conducted.

References

[Gopalratnam and Cook, 2005] K Gopalratnam and D J Cook. Online sequential prediction via incremental parsing: The Active LeZi algorithm. *IEEE Intelligent Systems*, 2005.

[Pineau *et al.*, 2001] J. Pineau, N. Roy, and S. Thrun. A Hierarchical Approach to POMDP Planning and Execution, 2001. Workshop on Hierarchy and Memory in Reinforcement Learning (ICML).

[Theocharous *et al.*, 2001] G. Theocharous, K. Rohanimanesh, and S. Mahadevan. Learning Hierarchical Partially Observable Markov Decision Processes for Robot Navigation, 2001. IEEE Conference on Robotics and Automation.