An Empirical Comparison of Techniques for Learning Models of Markov Decision Processes

Todd Hester TODD@CS.UTEXAS.EDU
The University of Texas at Austin, 1 University Station C0500, Austin, Texas 78712

Peter Stone PSTONE@CS.UTEXAS.EDU
The University of Texas at Austin, 1 University Station C0500, Austin, Texas 78712

1. Introduction

Reinforcement learning (RL) studies the problem of finding effective solutions to sequential decision making problems (Sutton & Barto, 1998). For many agent-based applications, it is critical that an RL algorithm be very sample efficient: that it takes very few actions to learn an effective policy.

This paper is motivated primarily by the observation that the world is too large to explore exhaustively. Therefore, we would like an agent that can learn an effective policy without exploring every state. To achieve this goal, the agent should avoid exploring some states by generalizing knowledge it has learned from other states. In addition, we want to drive the agent to explore the states in which the model makes inaccurate predictions, so that it can learn a more accurate model.

In this work, we explore the possibility of using existing supervised learning techniques to build generalizable models of Markov Decision Processes (MDP) that provide information on the accuracy of their predictions that could be used to motivate an agent's exploration. We then empirically compare the predictions of these models across three example domains.

2. Models

We adopted the standard Markov Decision Process formalism for this work (Sutton & Barto, 1998). An MDP consists of a set of states S, a set of actions A, a reward function $R(s, a)$, and a transition function $P(s'|s, a)$. In each state $s \in S$, the agent takes an action $a \in A$. Upon taking this action, the agent receives a reward $R(s, a)$ and reaches a new state s'. The new state s' is determined from the probability distribution $P(s'|s, a)$. Model-based reinforcement learning methods learn a model of the domain and then simulate actions inside their models. The domain can be modeled by approximating its transition and reward functions.

For our agent to behave as desired, the technique for learning a model of the MDP should have the following properties:

- Generalizes predictions well to unvisited states
- Has some measure of confidence in the accuracy of its predictions (It knows what it knows)

Using a model with these properties, the agent can explore the states in which the model has low confidence, and learn a model about the world without exploring every state by generalizing its knowledge to unseen states.

We examined whether existing supervised learning techniques can be used to learn this model. Each model needs to predict the $P(s'|s, a)$, $R(s, a)$ and its confidence in its prediction $C(s, a)$ for each state-action pair. The confidence is used to compare the relative prediction accuracy of the model across different state-action pairs, but not across models. We compare against a tabular model as well.

We compared model learning techniques based on the following supervised learning algorithms (confidence measure in parenthesis):

- C4.5 Decision Tree (Quinlan, 1986) (C = number of instances in leaf)
- Committee of Decision Trees (confidence based on model agreement)
- Random Forest (Breiman, 2001) (confidence based on model agreement)
- Support Vector Machines (SVM) (Burges, 1998) (C = distance from decision boundary)

- Neural Networks (Rumelhart et al., 1986) (C = difference of predicted probabilities and 1.0)
- K Nearest Neighbors (Cover & Hart, 1967) (C = average distance of k nearest neighbors)
- Tabular (C = number of visits to state)

3. Experiments

We performed experiments comparing the seven different models on three domains. The domains were selected as examples of factored domains in which it should be possible to generalize knowledge across states. For each experiment, the model was trained on some number n of (s, a, s', r) transitions that were randomly sampled from the MDP. We then recorded the predictions of the model for every state-action in the MDP along with the model's confidence in each prediction. The model's predictions were then compared with the correct transitions in the MDP.

The first domain used in the experiments is the classic Taxi domain introduced by Dietterich (Dietterich, 1998). We also performed experiments in a gridworld domain we call the Castle domain, and a non-gridworld problem called the Lights domain.

4. Results

We tested each algorithm on the three domains while varying the number of training samples from 50 to 25600, doubling each time. We thresholded the confidence measures of the models to classify some predictions as *unknown*. Predictions with confidence below threshold were labeled as unknown and predictions with confidence over the threshold were tested and labeled as *correct* or *incorrect*.

By varying the confidence threshold over the entire range of confidences reported by the model, we built plots of the operating characteristics of each model. The data was plotted with the x axis representing the percentage of transitions classified as unknown, which varies from 0 to 100%. The y axis shows the percentage of the known predictions that were classified correctly.

Figure 1 shows the operating characteristics of the models after being trained on 50 sample transitions in the Taxi domain. Here the tree committee performed the best, followed by the single tree. The tabular and random forest models performed reasonably well. The SVM started improving rapidly after 800 samples, and it surpassed the tree methods as the best model by 6400 samples. By 25600 samples, the SVM model was nearly perfect, classifying more than 95% of the samples correctly at 0% unknown. We had similar results on the Castle and Lights domains, with the tree-based methods performing the best after a small number of samples, and SVM and tabular models doing the best later.

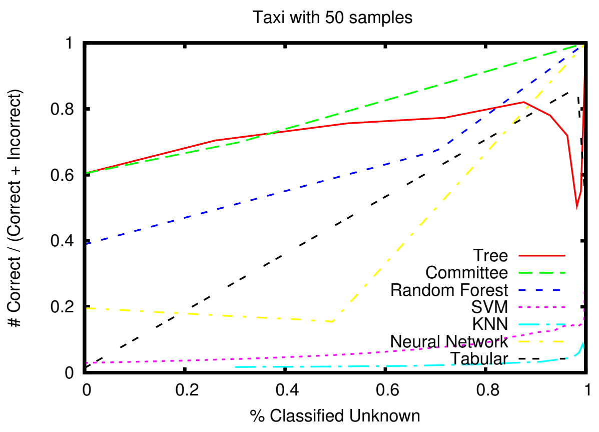

Figure 1. Operating characteristics of the models after training on 50 samples in the Taxi domain

5. Discussion

Our results showed that existing supervised learning techniques can be used as a model learning technique for an RL agent. Experiments across three domains show that the models generalize well and have sufficient knowledge of what they know. While these results look promising, the ultimate test of these methods is to use them to learn models in a model-based reinforcement learning algorithm.

References

Breiman, L. (2001). Random forests. *Machine Learning*, *45*, 5–32.

Burges, C. (1998). A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, *2*, 121–167.

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*, 21–27.

Dietterich, T. G. (1998). The MAXQ method for hierarchical reinforcement learning. *ICML* (pp. 118–126).

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*, 81–106.

Rumelhart, D., Hintont, G., & Williams, R. (1986). Learning representations by back-propagating errors. *Nature*, *323*, 533–536.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press.