Active Learning and Classification of Markov Decision Processes

Abheek Ghosh [1] **Ufuk Topcu** [2]

Abstract

We study the learning and classification of dynamic objects. Active classification, i.e., the sequential decision making process aimed at data acquisition for classification purposes, arises naturally in many applications, including medical diagnosis, intrusion detection, and object tracking. In this work, we study the problem of actively classifying dynamic objects with a finite set of Markov decision process (MDP) models. First, we give a sample-efficient PAC active learning algorithm that learns the models corresponding to the classes the objects. Second, building upon previous works on active classification algorithms for MDPs, we give a new classification objective and a corresponding algorithm that is robust to approximation errors in the learned MDPs. We prove that our learning and classification algorithms together achieve an approximately optimal classification accuracy.

1. Introduction

Consider an agent that has to actively interact with a dynamic object in an effort to correctly classify it. For example, a Mars exploration robot that has to classify the type of a rock among 5 categories by using the rock's pictures, weight, texture, and possibly by doing more involved physical and chemical tests. There is inherent randomness in the outcome due to the noisy sensors of the robot and the partial accuracy of the tests. Some of these tests can possibly change the state of the object (the rock) and increase or decrease the effectiveness of further tests. Also, some of the tests might have an associated cost, either due to its affect on the robot or the object.

As another example, consider a doctor who is trying to detect and cure the disease of a patient by prescribing diagnostic tests and treatments. The doctor doesn't exactly know the patient's disease, but she knows that the patient's disease belongs to a finite set. Observing the results of the diagnostic tests and the affects of the treatments, the doctor gains partial information about the disease. But these tests and treatments have associated costs: monetary expenses, toll on the patient's health, etc. Also, the state of the patient evolves with treatment and time.

The conventional approach for classification, which we refer to as *passive* classification, is an open-loop process that determines the class of the object of interest based on whatever data provided. While a challenging problem itself, such an approach is not suitable for the scenario described above. Passive classification cannot decide what, when and how to obtain data to best assist the classification task. As a result, excessive amount of data (for example, the tests in diagnosis scenario) may have to be collected. Furthermore, potentially uninformative data and ignorance of the impact of the data collection process on the object of interest may make the classification results less accurate (Wu et al., 2019).

Active classification (Hollinger et al., 2017), on the other hand, is a sequential decision-making process that has control over the data acquisition and interacts closely with the object of interest. Compared to passive classification, active classification is a more plausible approach in many practical classification applications, such as medical diagnosis (Hauskrecht & Fraser, 2000; Ayer et al., 2012), intrusion detection, and target tracking (Adhikari et al., 2017), where the object of interest has certain underlying dynamics belonging to a family of models.

Gaining insight from the previous examples, we model the classes of the dynamic objects using a family of Markov decision processes (MDPs). We assume that each object belongs to a finite set of classes, and each class corresponds to an MDP. All the MDPs share the same set of states and actions, the same cost function, and the same starting state. They differ in their transition functions. Intuitively, the states of the MDPs correspond to the states of the dynamic object as observed by the agent, e.g., the condition of the patient and the outcome of the tests as seen by the doctor. The actions corresponds to the agent's actions, e.g., the tests and the treatments as prescribed by the doctor. Initially, the agent has minimal information about a new object and

[1]Department of Computer Science, University of Texas at Austin, Austin, Texas, USA [2]Oden Institute for Computational Engineering and Sciences, University of Texas at Austin, Austin, Texas, USA. Correspondence to: Abheek Ghosh <abheek@utexas.edu>, Ufuk Topcu <utopcu@utexas.edu>.

Working Paper.

the object is at the initial state of the MDP. The agent takes actions and observes the transitions and the new states. After a small number of actions the agent has to classify the object, i.e., classify the MDP it was interacting with. The agent also accumulates cost through the process. The classification process directly affects the state evolution of the object by selecting actions and obtaining observations as a closed-loop system such that classification decisions can be reached more efficiently and accurately.

We make a few assumptions regarding our problem setup. First, as mentioned, we assume that the states, actions, and associated costs are known beforehand. This seems reasonable for the examples we mentioned, the doctor should know the available tests and treatments and their associated costs. The transition functions have to learned. Second, we assume that for each instance of an unknown dynamic object, the classification has to be done in a bounded number of actions, i.e., the sequence of transitions in the MDP should be of bounded length. In our example, the doctor has to classify the patient after a limited number of tests. Third, during learning also we follow the same bounded length restrictions as classification. We assume that the learning problem is episodic, i.e., in each episode, the agent gets a new sample (unknown dynamic object) and interacts with it for finite number of steps. At the end of the episode, the label of the object is revealed to the agent and the agent uses this label to estimate the model. This assumption also seems reasonable for our examples.

We break the problem into two main parts. First, how to learn the model, i.e., the MDPs analogous to the classes? Second, how to use the model to do accurate and efficient classification?

For the learning part, we take a probably approximately correct (PAC) (Valiant, 1984) learning approach. Our problem differs from usual reinforcement learning setup in two main ways. First, the agent's reward is the probability of correct classification, which, as we will see, depends upon transition probabilities of all the MDPs (not only the one that the agent is interacting with) along the sequence of states visited and actions taken by the agent. This motivates us to use a model-based approach instead of a model-free one. Second, as the agent doesn't know beforehand the MDP it is interacting with, we appropriately extend previous works on model-based learning algorithms to take care of this issue. If we had only one MDP, our algorithm would look and perform similar to the popular model-based learning algorithms (Fiechter, 1994; 1997; Kearns & Singh, 2002).

Previous works have addressed the classification problem where the MDPs are known (Wu et al., 2019). The agent wants to find a policy that has the highest probability of correct classification. We observe that this objective might amplify the approximation errors in the transition functions of the learned MDPs. We give an alternate classification objective that avoids this error amplification; algorithms from previous papers can be easily extended to optimize this objective.

TODO: Write more of related work, details contributions, conclusion, etc.

2. Notation and Problem Definition

We denote the set of natural numbers $\{1, 2, \ldots, n\}$ as $[n]$.

In this work we use the following definition of a Markov decision process or MDP. An MDP is a quintuple $(S, \hat{s}, A, T, C)$ where,

- S: the set of states of size N_S.
- $\hat{s}$: the initial state, $\hat{s} \in S$.
- A: the set of actions of size N_A.
- T: the transition function over $S \times A \times S$.
- C: the cost function over $S \times A$.

We assume that there is a set of N_{MDP} MDPs and the agent has to learn and classify a sample MDP from this set. We denote this underlying set of MDPs unknown to the agent as $\mathbb{MDP}^* = \{MDP_i^*\}_{i \in [N_{MDP}]}$, where MDP_i^* defined as:

$$MDP_i^* = (S, \hat{s}, A, T_i^*, C)$$

Without loss of generality, we assume that the states S, initial state $\hat{s}$, actions A, and cost function C are shared by all the MDPs. The MDPs differ by their transition functions T_i^*s. This model is also known in the literature as a hidden-model MDP (Chadès et al., 2012), which is essentially a partially-observable MDP with the state space $(S \times [N_{MDP}])$ where the observation function reveals the S component but hides the $[N_{MDP}]$ component. We assume that the agent already knows S, $\hat{s}$, A, and C, and that there are N_{MDP} MDPs. But, it has to learn the transition functions T_i^* by exploration and optimize its final objective of accurate classification.

We model the problem to be episodic. At the start of each episode, an MDP is randomly sampled from $\mathbb{MDP}^*$ with probability $\{p^*_{MDP}(i)\}_{i \in [N_{MDP}]}$, where $p^*_{MDP}(i)$ corresponds to the probability of selecting MDP_i^*. This MDP, say MDP_ι^*, is given to the agent to actively interact for H time-steps with the constraint that the total cost must be bounded by D. We assume that for all the N_{MDP} MDPs, from any given state the agent can take a *Halt* action that takes it to a *Halt* state with probability 1 and cost 0, and the agent stays in this *Halt* state for the rest of the episode for all actions with cost 0. This allows the agent to deterministically avoid exceeding the cost bound D by effectively

terminating the episode. After interaction, the agent classifies the MDP and the identity of the MDP (the label ι) is revealed to the agent. It uses this label ι and the path traveled in the episode to improve its estimate of the transition function and improve its probability of correct classification in future. This is a reinforcement learning problem with a complex reward function.

We call the sequence of states and actions seen over an episode (or a part of an episode) a path. We denote a path of h time-steps as $P^{(h)} = (s^{(0)} = \hat{s}, a^{(0)}, s^{(1)}, a^{(1)}, \ldots, s^{(h)})$. We identify the set of states in the path as $S(P^{(h)}) = \{s^{(t)} | t \in \{0\} \cup [h]\}$ and the set of transitions in the path as $\tau(P^{(h)}) = \{(s^{(t-1)}, a^{(t-1)}, s^{(t)}) | t \in [h]\}$

Classification

An active classification algorithm ActClass takes input:

- An estimate of the MDPs, say $\mathbb{MDP} = \{MDP_i\}_{i \in [N_{MDP}]}$. Essentially, estimates T_is of the true transition functions T_i^*s, everything else is already known.
- A prior p over the MDPs, which is an estimate of p^*.

The output of the algorithm is a policy $\pi = \text{ActClass}(\mathbb{MDP}, p)$. This policy π takes input the prior p and the path $P^{(h)}$, and gives output the probability distribution over the next action $a^{(h)}$. We denote this probability as $\pi(a|p, P^{(h)})$ for $a \in A$.

Given a prior p and a path $P^{(h)} = (\hat{s}, a^{(0)}, \ldots, a^{(h-1)}, s^{(h)})$, we denote the belief that the agent is interacting with the ith MDP as $b(i|p, P^{(h)})$. Say, from the state $s^{(h)}$, the agent takes an action $a^{(h)} \in A$ and reaches a state $s^{(h+1)} \in S$. The extended path becomes $P^{(h+1)} = (P^{(h)}, a^{(h)}, s^{(h+1)})$; the updated belief $b(i|p, P^{(h+1)})$ can be calculated as:

$$b(i|p, P^{(h+1)}) = \frac{T_i(s^{(h)}, a^{(h)}, s^{(h+1)}) b(i|p, P^{(h)})}{\sum_{j \in [N_{MDP}]} T_j(s^{(h)}, a^{(h)}, s^{(h+1)}) b(j|p, P^{(h)})} \quad (1)$$

Equation 1 can be further expanded and rewritten using the prior p as:

$$b(i|p, P^{(h+1)}) = \frac{p(i) \prod_{t \in [h+1]} T_i(s^{(t-1)}, a^{(t-1)}, s^{(t)})}{\sum_{j \in [N_{MDP}]} p(j) \prod_{t \in [h+1]} T_j(s^{(t-1)}, a^{(t-1)}, s^{(t)})} \quad (2)$$

The above equations have been derived by applying conditional probability.

Now, if the underlying MDP the agent is interacting with is MDP_ι^* and the policy used by the agent is π, the agent can compute the probability of seeing the path $P^{(H)}$ as:

$$prob(P^{(H)}|\iota, \pi) = \prod_{t \in [H]} \pi(a^{(t-1)}|p, P^{(t-1)}) T_\iota(s^{(t-1)}, a^{(t-1)}, s^{(t)}) \quad (3)$$

Observe that this is not the actual probability of seeing the path $P^{(H)}$, rather the probability as computed by the agent. The actual probability will depend upon T_ι^* rather than T_ι, and we denote it as:

$$prob^*(P^{(H)}|\iota, \pi) = \prod_{t \in [H]} \pi(a^{(t-1)}|p, P^{(t-1)}) T_\iota^*(s^{(t-1)}, a^{(t-1)}, s^{(t)}) \quad (4)$$

Similarly, if the agent knew p^* and T_i^* it would compute the belief, similar to equation (2), as:

$$b^*(i|p^*, P^{(H)}) = \frac{p^*(i) \prod_{t \in [H]} T_i^*(s^{(t-1)}, a^{(t-1)}, s^{(t)})}{\sum_{j \in [N_{MDP}]} p^*(j) \prod_{t \in [h+1]} T_j^*(s^{(t-1)}, a^{(t-1)}, s^{(t)})} \quad (5)$$

The cost of the path, which we want to be upper bounded by D, is:

$$cost(P^{(H)}) = \sum_{t \in [H]} C(s^{(t-1)}, a^{(t-1)}) \leq D \quad (6)$$

The agent wants to use a policy that maximizes the belief for the ith MDP when the underlying MDP is the ith MDP, i.e., when $\iota = i$, given that the policy follows the cost constraint. For a given π, the probability of correct classification as estimated by the agent is:

$$classify(\pi) = \sum_{i \in [N_{MDP}]} p(i) \sum_{P^{(H)}} b(i|p, P^{(H)}) \, prob(P^{(H)}|i, \pi) \quad (7)$$

The agent wants to maximize this classification probability:

$$\max_\pi classify(\pi) \quad (8)$$

ActClass finds a policy $\pi = \text{ActClass}(\mathbb{MDP}, p)$ that optimizes the objective (8) and satisfies the constraint (6). If $\mathbb{MDP}^*$ and p^* are known, then the algorithm computes the optimal policy $\pi^* = \text{ActClass}(\mathbb{MDP}^*, p^*)$. Previous works (Wu et al., 2019) gave an exponential time accurate algorithm and a Monte Carlo based approximation algorithm for ActClass. TODO: Write hardness proof and look into its approximations if we have space left in paper.

It is important to note that although $classify(\pi)$ is the classification probability that the agent optimizes for, the

actual probability of classification will depend upon the underlying dynamics of the MDPs. We denote this as:

$$classify^{\#}(\pi) = \sum_{i \in [N_{MDP}]} p^*(i) \sum_{P^{(H)}} b(i|p, P^{(H)})\, prob^*(P^{(H)}|i, \pi) \quad (9)$$

Here, the belief is using the estimates, but dynamics (the probability of seeing an MDP and a path) is using the true values. Also, the optimal classification probability for a policy π, where both the belief computation and dynamics are using the true values, can be written as:

$$classify^*(\pi) = \sum_{i \in [N_{MDP}]} p^*(i) \sum_{P^{(H)}} b^*(i|p^*, P^{(H)})\, prob^*(P^{(H)}|i, \pi) \quad (10)$$

$classify^*(\pi)$ is what ACTCLASS($\mathbb{MDP}^*, p^*$) optimizes for.

3. Active Learning and Sample Complexity

In this section, we discuss the model-based active learning algorithm and show that it achieves close to optimal accuracy in polynomial number of samples. We first describe the ACTLEARN algorithm that estimates p^* by p and T_i^*s by T_i. Then we do its analysis, and the majority of the paper is dedicated to that. We also modify the ACTCLASS algorithm to make it robust to approximation errors in p and T_is, and therefore make it suitable to be used in conjunction with a learning algorithm.

ACTLEARN

Our learning algorithm ACTLEARN estimates p^* by p and T_i^*s by T_is using the empirical distribution it sees during exploration. It maintains a count table, $count(i, s, a)$ for each $i \in [N_{MDP}]$, $s \in S$, and $a \in A$; this keeps count how many times the action a has been executed from state s for MDP i. For each MDP i, it also maintains a set of states S'_i that have been visited at least n_S times across all episodes that interacted with MDP i. At the beginning, S'_is are empty. For each episode, ACTLEARN uses the following routine:

1. Select the lowest index $i \in [N_{MDP}]$ that has not been selected n_{MDP} times in this step.

2. At the beginning of the episode, find a policy that takes the sequence of actions that leads to a state in $S \setminus S'_i$ with the highest probability, in less than H time steps and with cost bounded by D. Execute this policy whenever we are at a state in S'_i.

3. At any point in the episode, if we are at a state $s \in S \setminus S'_i$, then do *balanced wandering*, i.e., take the action from the state s least explored for the ith MDP.

4. At the end of the episode, the correct MDP ι is revealed. Update the estimates: $p, T_\iota, count(\iota, \ldots, \ldots)$ (used for balanced wandering), and S_ι. Note that these updates are done for ι and not i.

All the steps in the ACTLEARN, except finding the policy for step 2, are straight forward. Finding a cost-bounded optimal policy for reachability is an NP-hard problem[1]. But one can find dynamic programming based pseudo-polynomial time approximate solutions. (TODO: If seems relevant add the DP algorithm.)

TODO: Maybe define n_{MDP} and n_S before algorithm.

We now jump to the algorithm's analysis. At the end of each episode we get to see the true label of the MDP, this gives us $n_{MDP}N_{MDP}$ samples from p^* and allows us to accurately estimate it.

Lemma 1. *When* ACTLEARN *terminates, p is a good estimation of p^*, i.e., for all $i \in [N_{MDP}]$:*

$$p^*(i) - \epsilon_{MDP} \le p(i) \le p^*(i) + \epsilon_{MDP}$$

with probability of failure $\le 2e^{-2\epsilon^2_{MDP} n_{MDP} N_{MDP}}$. This gives us the corollary that if $p^*(i) \ge \beta_{MDP}$, we can get a multiplicative bound on $p(i)$:

$$p^*(i)(1 - \alpha_{MDP}) \le p(i) \le p^*(i)(1 + \alpha_{MDP})$$

where $\alpha_{MDP} = \epsilon_{MDP}/\beta_{MDP}$.

Lemma 1. As we sample from $p^*(i)$ $n_{MDP}N_{MDP}$ times, by direct application of Hoeffding's inequality we get the result. □

As we took care of the p^*, let's focus on the learning of T_i^*s. To learn the T_i^* at some state $s \in S$, ACTLEARN should be able to visit s in the ith MDP enough times during the learning process. The first way that this can become difficult is when $p^*(i)$ is too low. In this case, as MDP i won't be seen enough times during learning, and therefore we won't

[1] Reduction from knapsack problem. Consider a knapsack problem with $W_1, W_2, \ldots, W_n$ weights and total weight bounded by W. We construct an MDP with: $n+2$ states $s_1, s_2, \ldots, s_n$, s_{good} and s_{bad}; 2 actions sel and rej. Say $B \ge \sum_{i \in [n]} W_i$ be some upper bound. The transition function T is defined as: $T(s_{good}, *, s_{good}) = 1$; $T(s_{bad}, *, s_{bad}) = 1$; $T(s_i, *, s_{i+1}) = \frac{n-i}{n-i+1}$, $T(s_i, sel, s_{good}) = \frac{W_i}{(n-i+1)B}$, $T(s_i, sel, s_{bad}) = \frac{B - W_i}{(n-i+1)B}$, $T(s_i, rej, s_{good}) = 0$, $T(s_i, rej, s_{bad}) = \frac{1}{n-i+1}$. The cost function is defined as: $C(s_i, sel) = W_i, C(s_i, rej) = 0$. Solving optimal policy to reach s_{good} with cost bounded by W for this MDP solves the knapsack problem.

be able accurately learn its T_i^*. We will restrict our focus on learning the properties of an MDP i when $p^*(i) \geq \beta_{MDP}$. We call such an MDP a *good MDP*. During evaluation, if we are given a *bad MDP* (not good MDP), then we are likely going to fail. Therefore β_{MDP} contributes to δ, the probability of (ϵ, δ)-PAC failure of ACTLEARN.

Even if MDP i is a good MDP, it might still be difficult to estimate T_i^* at state $s \in S$ if there is no policy π that leads to s in MDP i with probability high enough. We denote this probability by β_S and call a state s a *good state* for MDP i if it can be reached from $\hat{s}$ (initial state) using some policy, in less than H time-steps, with a cost bounded by D, and with a probability $\geq \beta_S$. Formally, a state s is a *bad state* (not good state) for MDP i if:

$$\max_{\pi} \sum_{\substack{P^{(H)} \\ s \in S(P^{(H-1)})}} prob^*(P^{(H)}|i,\pi) < \beta_S \qquad (11)$$

As we are working with N_{MDP} MDPs, we could have a different set of good states for each one of them. We denote the set of good states for MDP i by $S_i^* \subseteq S$. As we did with the bad MDPs, we will avoid trying to learn the transition function at the bad states. During evaluation this leads to another cause of failure. Therefore, β_S will also be a contributor to δ.

The sets S_i's that we created in the ACTLEARN algorithm are intended to approximate the good states S_i^*s. We hope to show that $S_i^* \subseteq S_i'$ with high probability.

Lemma 2. *When* ACTLEARN *terminates, the transition function from every state s in S_i' has been well estimated, i.e., for every $i \in [N_{MDP}]$, $s \in S_i'$, $a \in A$, $s' \in S$:*

$$T_i^*(s,a,s') - \epsilon_T \leq T_i(s,a,s') \leq T_i^*(s,a,s') + \epsilon_T$$

with probability of failure $\leq 2e^{-2\epsilon_T^2 \left\lfloor \frac{n_S}{N_A} \right\rfloor}$. This gives us the corollary that for $T_i^*(s,a,s') \geq \beta_T$, we can get a multiplicative bound:

$$T_i^*(s,a,s')(1-\alpha_T) \leq T_i(s,a,s') \leq T_i^*(s,a,s')(1+\alpha_T)$$

where $\alpha_T = \epsilon_T/\beta_T$.

Lemma 2. The criteria to add a state s to S_i' in ACTLEARN is that s has been explored at least n_S times. Also, before s is added to S_i', the actions a in s are selected by balanced wandering. Therefore, for every $i \in [N_{MDP}]$, $s \in S_i'$, $a \in A$, and $s' \in S$, $T_i^*(s,a,s')$ is sampled at least $\left\lfloor \frac{n_S}{N_A} \right\rfloor$ times. Using Hoeffding's inequality we get the required result. □

Another term we use is of a *good transition*: a transition (s,a,s') in MDP i good if it has a probability of occurrence at least β_T, i.e., $T_i^*(s,a,s') \geq \beta_T$. As shown in Lemma 2, good transitions will give us multiplicative bounds. These bounds on the transitions will be fruitful in giving approximation bounds for the probability of paths in T_i compared to T_i^*. The simple lemma below also helps for the same purpose.

Lemma 3. *For a positive integer n and a non-negative real number $x \leq 1/n$:*

$$(1+x)^n \leq 1 + 2xn$$
$$(1-x)^n \geq 1 - xn$$

This lemma can be proved by using Taylor's expansion. Proof is given in the appendix (Section 4).

In the following discussions, to be able to write the equations in a reasonable space we will suppress notation and write $\prod_{t\in[H]} T_i(s^{(t-1)}, a^{(t-1)}, s^{(t)})$ as $\prod T_i$, $\prod_{t\in[H]} \pi(a^{(t-1)}|p, P^{(t-1)})$ as $\prod \pi$, and $b(i|p, P^{(H)})$ as $b(i)$. In a similar way we write $\prod T_i^*$, $b^*(i)$, $\bar{b}(i)$, $\overline{b^*}(i)$, etc. The expansion will be obvious by the context where these terms appear.

The lemma below bounds the probability of seeing paths with bad transitions.

Lemma 4. *For any MDP i and any policy π, the probability of seeing a path $P^{(h)}$ that takes a transition (s,a,s') with $T_i^*(s,a,s') < \beta_T$ is less than $HN_S\beta_T$.*

Proof. From any state and action the number of transitions to next state with probability less than β_T is bounded by N_S, and therefore the total probability of such transitions by $N_S\beta_T$. Therefore, for any path of h time-steps, the total probability that such a transition is selected by the nature is bounded by $hN_S\beta_T \leq HN_S\beta_T$. □

Focusing only on the good states and good transitions, we give the next lemma to bound the error in approximation of the probability of a path.

Lemma 5. *The probability of a path $P^{(h)}$ ($h \leq H$) in MDP i that only passes through states in S_i' and uses transitions $T_i^*(s,a,s') \geq \beta_T$ can be bounded as:*

$$prob^*(P^{(h)}|i,\pi)(1-h\alpha_T) \leq prob(P^{(h)}|i,\pi) \leq prob^*(P^{(h)}|i,\pi)(1+2h\alpha_T)$$

Lemma 5.

$$\begin{aligned} prob(P^{(h)}|i,\pi) &= (\prod_{t\in[h]} \pi) \prod_{t\in[h]} T_i \\ &\text{(Using Lemma 2)} \\ &\leq (\prod_{t\in[h]} \pi) \prod_{t\in[h]} (T_i^*(1+\alpha_T)) \\ &\leq (\prod_{t\in[h]} \pi)(\prod_{t\in[h]} T_i^*)(1+\alpha_T)^h \\ &\text{(Using Lemma 3. } \alpha_T \leq 1/H) \\ &\leq (\prod_{t\in[h]} \pi)(\prod_{t\in[h]} T_i^*)(1+2h\alpha_T) \\ &\leq prob^*(P^{(h)}|i,\pi)(1+2h\alpha_T) \end{aligned}$$

For the other direction, with a similar sequence of arguments, we can prove that $prob(P^{(h)}|i,\pi) \geq prob^*(P^{(h)}|i,\pi)(1-h\alpha_T)$. □

With the bounds on the approximation error for p, T_is, and probability of paths in place, we now show that for good MDPs i all the good states are in S'_i.

Lemma 6. *For an MDP i with $p^*(i) \geq \beta_{MDP}$, if the probability of reaching a state $s \in S$ using any policy π is $\geq \beta_S$ (in less than H time-steps), then the probability of $s \notin S'_i$ is bounded by $e^{-2\epsilon^2_{MDP} n_{MDP}} + e^{-2\epsilon^2_S(\beta_{MDP}-\epsilon_{MDP})n_{MDP}}$.*

Proof. Fix an MDP i with $p^*(i) \geq \beta_{MDP}$. In ActLearn, there were n_{MDP} exploration attempts for every MDP including i. Among the attempts, the probability that less that $(p^*(i) - \epsilon_{MDP})n_{MDP}$ attempts ended up exploring MDP i is bounded (using Hoeffding's inequality) by $e^{-2\epsilon_{MDP} n_{MDP}}$. Now, let's focus on the case where this failure didn't occur and i was correctly explored at least $(p^*(i) - \epsilon_{MDP})n_{MDP} \geq (\beta_{MDP} - \epsilon_{MDP})n_{MDP}$ times.

Let us assume that some good state s is not in S'_i. For some policy, s has the probability of being reached more than β_S in MDP i, but when ActLearn terminates s is not in S'_i. As $s \in S \setminus S'_i$, the probability of reaching a state in $S \setminus S'_i$ is at least β_S. Over the course of the algorithm ActLearn, the set S'_i never decreases. Therefore, in Step 2 of ActLearn, the probability of reaching states not in S'_i has always been $\geq \beta_S$.

Observe that for calculating the probability of reaching $S \setminus S'_i$ only the transition functions at the states in S'_i matter. ActLearn Step 2 finds the policy that maximizes the success probability (the probability of reaching states in $S \setminus S'_i$). But, the optimization is done with the approximate transition functions T_is. Say π is the optimal policy found in ActLearn Step 2 and let its success probability as computed in ActLearn be x (using T_is). In a similar way, for π, let the actual success probability be x^* (using T_i^*s).

If we consider only the paths with good transitions, i.e., transitions (s,a,s') with $T_i^*(s,a,s') \geq \beta$, then π has a success probability $\geq \beta'_S - HN_S\beta_T$ (Lemma 4) with these paths. For paths with only good transitions, we can use Lemma 5, in the original MDP with T_i^*s, π has a success probability:

$$x^* \geq x(1-H\alpha_T) - HN_S\beta_T \geq x - H\alpha_T - HN_S\beta_T$$

Similarly, let the optimal policy found using the true transition functions T_i^* be π^*. Let π^* has a success probability of $y^* \geq \beta_S$ using T_i^* and y using T_i. With an argument similar to π (first removing paths with bad transitions and then using Lemma 5) we can show that

$$y \geq y^*/(1+2H\alpha_T) - HN_S\beta_T \geq \beta_S - 2H\alpha_T - HN_S\beta_T$$

As π is optimal for T_is, therefore $x \geq y$. Combining the inequalities for x, x^*, y, and y^* we get:

$$x \geq \beta_S - 3H\alpha_T - 2HN_S\beta_T$$

Again using Hoeffding's inequality, out of the $(\beta_{MDP} - \epsilon_{MDP})n_{MDP}$ correct exploration attempts the probability that less than $((\beta_{MDP} - \epsilon_{MDP})(\beta_S - 3H\alpha_T - 2HN_S\beta_T) - \epsilon_S)n_{MDP}$ attempts end up reaching states in $S \setminus S'_i$ and doing balanced wandering is bounded by $e^{-2\epsilon^2_S(\beta_{MDP}-\epsilon_{MDP})n_{MDP}}$.

In ActLearn, any state s can have at most n_S many balanced wanderings, after that the state moves to S'_i. As $((\beta_{MDP}-\epsilon_{MDP})(\beta_S-3H\alpha_T-2HN_S\beta_T)-\epsilon_S)n_{MDP} \geq N_S n_S$, all the good states are in S'_i. □

Together, all the previous lemmas prove that with high probability ActLearn accurately estimates the p^*, and for good MDPs and good states, the T_i^*s. The important piece remaining is to show that this approximated model can be used to achieve a close to optimal classification probability.

The bad MDPs and the bad states lead to high error in the classification probability. This error is because the transition function at these difficult to reach states are used to compute the belief $b^*(i|p^*, P^{(H)})$. Look at equation (5): to compute $b^*(i|p^*, P^{(H)})$ the transitions along the path $P^{(H)}$ for all the MDPs are used, not just i. Even if MDP i is good, some other MDP j might be bad, and therefore we might have large errors in T_j. Even if all the MDPs are good, still the sets S'_i and S'_j need not be same. It is possible that the path $P^{(H)}$ may contain all good states for MDP i and a very high probability of occurring, but it may contain a bad state for MDP j, which can lead to high error in $T_j(s,\ldots)$, and therefore $b(i|p, P^{(H)})$.

To circumvent this problem, we tweak the belief computation formula $b(i|p, P^{(H)})$ used by the classification algorithm. For this tweak, we need an approximate set of good MDPs, good states, and good transitions.

- A good MDP i has $p^*(i) \geq \beta_{MDP}$. We don't know p^*, but using p we can approximate the set of good MDPs as:

$$\mathcal{M} = \{i|p(i) \geq \beta_{MDP} + \epsilon_{MDP}, i \in [N_{MDP}]\} \quad (12)$$

  Using Lemma 1, excluding the small probability of failure, we know that $\mathcal{M}$ contains all the MDPs that have $p^*(i) \geq \beta_{MDP}+2\epsilon_{MDP}$ and avoids all the MDPs that have $p^*(i) < \beta_{MDP}$. In other words, $\mathcal{M}$ has most of the good MDPs and avoids all the bad MDPs.

- We already have an estimate for the set of good states for each MDP i: the sets S'_i. The transition functions at all these states have been well estimated with high probability.

- The transition functions are well estimated only for good MDPs and good states. We define the set of approximately good transitions for an MDP i as:

$$\tau_i = \{(s, a, s')|T_i(s, a, s') \geq \beta_T + \epsilon_T, \\ s \in S'_i, a \in A, s' \in S, i \in \mathcal{M}\} \quad (13)$$

  Using Lemma 2, excluding the small probability of failure, we know that τ_i avoid all the bad transition ($T^*_i(s, a, s') < \beta_T$) and contains most of the good transitions ($T^*_i(s, a, s') \geq \beta_T + 2\epsilon_T$) for the approx-good MDPs $i \in \mathcal{M}$ and approx-good states $s \in S'_i$.

For any $j \in [N_{MDP}]$, if j is estimated to be a bad MDP, i.e., $j \notin \mathcal{M}$, or if $P^{(H)}$ contains a bad state or a bad transition for MDP j, i.e., $S(P^{(H)}) \not\subseteq S'_j$ or $t\tau(P^{(H)}) \not\subseteq \tau_j$, we replace the $(p(j)\prod T_j)$ in the formula for $b(i|p, P^{(H)})$ by 0. Formally, for path $P^{(H)} = (s^{(0)}, a^{(0)}, \ldots, a^{(H-1)}, s^{(H)})$ we define:

$$prod(P^{(H)}|j, p, \mathcal{M}, S'_j, \tau_j) = \\ \begin{cases} 0; \text{ if } (j \notin \mathcal{M}) \vee (S(P^{(H)}) \not\subseteq S'_j) \vee (\tau(P^{(H)}) \not\subseteq \tau_j) \\ p(j)\prod_{t\in[H]} T_j(s^{(t-1)}, a^{(t-1)}, s^{(t)}); \text{ otherwise} \end{cases} \quad (14)$$

We write $prod(P^{(H)}|j, p, \mathcal{M}, S'_j, \tau_j)$ as $prod(P^{(H)}|j, p)$, suppressing the notation for $\mathcal{M}$, S'_j, τ_j. We can compute the transformed belief formula $\overline{b}$ as:

$$\overline{b}(i|p, P^{(H)}) = \frac{prod(P^{(H)}|i, p)}{\sum_{j\in[N_{MDP}]} prod(P^{(H)}|j, p)} \quad (15)$$

Note that the formula above is undefined if $prod(P^{(H)}|j, p) = 0$ for all j. In that case, we replace $\overline{b} = 0$. In a similar fashion to $\overline{b}$, we define $\overline{b^*}$ (using the same sets $\mathcal{M}$, S'_j, τ_j); and by replacing b with $\overline{b}$ and b^* with $\overline{b^*}$ we define $\overline{classify}$, $\overline{classify^{\#}}$, and $\overline{classify^*}$. Finally, our classification algorithm will be $\overline{\text{ACTCLASS}}$ that finds a policy π that optimizes for $\overline{classify}(\pi)$ instead of $classify(\pi)$.

The following lemma shows that $\overline{b}$ is not an underestimate of b for approximately good MDPs, states, and transitions.

Lemma 7. *For an MDP $i \in \mathcal{M}$ and a path $P^{(H)}$ s.t. $S(P^{(H)}) \subseteq S'_i$ and $\tau(P^{(H)}) \subseteq \tau_i$, we have:*

$$\overline{b}(i|p, P^{(H)}) \geq b(i|p, P^{(H)})$$

Lemma 7. The proof is direct from the definition of $\overline{b}$. If the given conditions $i \in \mathcal{M}$, $S(P^{(H)}) \subseteq S'_i$, and $\tau(P^{(H)}) \subseteq \tau_i$ are satisfied then the numerator of $\overline{b}(i|p, P^{(H)})$ and $b(i|p, P^{(H)})$ are exactly same. The denominator of $\overline{b}(i|p, P^{(H)})$ is at most that of $b(i|p, P^{(H)})$. This gives us the required result. □

We now state the main theorem of the paper.

Theorem 8. *When* ACTLEARN *terminates, with probability at least $(1 - \delta)$:*

$$\overline{classify^{\#}}(\pi) \geq classify^*(\pi^*) - \epsilon \quad (16)$$

where $\pi = \overline{\text{ACTCLASS}}(\mathbb{MDP}, p)$ *and* $\pi^* = \text{ACTCLASS}(\mathbb{MDP}^*, p^*)$.

Before we jump into the proof, let's see why this theorem proves that ACTLEARN solves the learning problem. The left hand side of inequality (16) is the performance of the $\overline{\text{ACTCLASS}}$ algorithm that we use for classification, which is at most ϵ less than the optimal performance given on the right. We prove Theorem 8 in parts using the lemmas below.

Lemma 9. *For any policy π:*

$$classify^*(\pi) \leq \overline{classify}(\pi)\frac{(1 + \alpha_{MDP})(1 + \alpha_T)^H}{(1 - \alpha_{MDP})^2(1 - \alpha_T)^{2H}} \\ + N_{MDP}(\beta_{MDP} + 2\epsilon_{MDP}) + \beta_S + \beta_T + 2\epsilon_T$$

Lemma 9. From the definition of $classify^*(\pi)$ we have:

$$classify^*(\pi) = \sum_i p^*(i) \sum_{P^{(H)}} b^*(i) \prod \pi \prod T^*_i$$

Separating the summation into MDPs in $i \in \mathcal{M}$ and $i \notin \mathcal{M}$. We know that $b^*(i) \leq 1$ and $\sum_{P^{(H)}} \prod \pi \prod T^*_i = 1$. Also, for $i \notin \mathcal{M}$, from the definition of $\mathcal{M}$, we have that $p^*(i) <$

$\beta_{MDP} + 2\epsilon_{MDP} \implies \sum_{i \notin \mathcal{M}} p^*(i) < N_{MDP}(\beta_{MDP} + 2\epsilon_{MDP})$. So, we get:

$$\sum_{i \notin \mathcal{M}} p^*(i) \sum_{P^{(H)}} b^*(i) \prod \pi \prod T_i^* \leq N_{MDP}(\beta_{MDP}+2\epsilon_{MDP}) \tag{17}$$

From now onward, let's focus on approx-good MDPs $i \in \mathcal{M}$. Let's split the summation between paths $P^{(H)}$ where $S(P^{(H)}) \subseteq S'_i$ and $S(P^{(H)}) \nsubseteq S'_i$.

Using Lemma 6 and the facts $b^*(i) \leq 1$ and $\sum_{i \in \mathcal{M}} p^*(i) \leq 1$, we have:

$$\sum_{i \in \mathcal{M}} p^*(i) \sum_{\substack{P^{(H)} \\ S(P^{(H)}) \nsubseteq S'_i}} b^*(i) \prod \pi \prod T_i^* \leq \beta_S \tag{18}$$

From now onward, we restrict our focus on paths through only approx-good states, $S(P^{(H)}) \subseteq S'_i$. Let's split the summation into paths $P^{(H)}$ where $\tau(P^{(H)}) \subseteq \tau_i$ and $\tau(P^{(H)}) \nsubseteq \tau_i$. From the definition of τ_i, we know that the probability of seeing a path with $\tau(P^{(H)}) \nsubseteq \tau_i$ is bounded by $\beta_T + 2\epsilon_T$. So, we get

$$\sum_{i \in \mathcal{M}} p^*(i) \sum_{\substack{P^{(H)} \\ S(P^{(H)}) \subseteq S'_i \\ \tau(P^{(H)}) \nsubseteq \tau_i}} b^*(i) \prod \pi \prod T_i^* \leq \beta_T + 2\epsilon_T \tag{19}$$

From now onward, we further restrict our focus to paths through only approx-good transitions, $\tau(P^{(H)}) \subseteq \tau_i$. As we have the three required conditions ($i \in \mathcal{M}$, $S(P^{(H)}) \subseteq S'_i$, and $\tau(P^{(H)}) \subseteq \tau_i$) for Lemma 7, we get

$$b^*(i) \leq \overline{b}^*(i) \tag{20}$$

Using the inequalities from equations (17), (18), (19), and (20) we have

$$classify^*(\pi) \leq \sum_{i \in \mathcal{M}} p^*(i) \sum_{\substack{P^{(H)} \\ S(P^{(H)}) \subseteq S'_i \\ \tau(P^{(H)}) \subseteq \tau_i}} \overline{b^*}(i) \prod \pi \prod T_i^*$$

$$+ N_{MDP}(\beta_{MDP} + 2\epsilon_{MDP}) + \beta_S + \beta_T + 2\epsilon_T \tag{21}$$

The summation in the right hand side of above equation is exactly equal to $\overline{classify^*}(\pi)$, we get:

$$\implies classify^*(\pi) \leq \overline{classify^*}(\pi) + N_{MDP}(\beta_{MDP} + 2\epsilon_{MDP}) + \beta_S + \beta_T + 2\epsilon_T \tag{22}$$

As we are using $\overline{b}$, and we already have $i \in \mathcal{M}$, $S(P^{(H)}) \subseteq S'_i$, and $\tau(P^{(H)}) \subseteq \tau_i$, we can use the multiplicative bounds in the Lemmas 1 and 2. We appropriately replace p^* with p and T^* with T to get,

$$\overline{classify^*}(\pi) \leq \overline{classify}(\pi) \frac{(1+\alpha_{MDP})(1+\alpha_T)^H}{(1-\alpha_{MDP})^2(1-\alpha_T)^{2H}} \tag{23}$$

□

Lemma 10. *For any policy π:*

$$\overline{classify}(\pi) \leq \overline{classify^{\#}}(\pi)(1+\alpha_{MDP})(1+\alpha_T)^H \tag{24}$$

Lemma 10. As we are working with $\overline{b}$, we can restrict our focus to $i \in \mathcal{M}$, $S(P^{(H)}) \subseteq S'_i$, and $\tau(P^{(H)}) \subseteq \tau_i$. In other cases, $\overline{b}(i|p, P^{(H)}) = 0$. Using the multiplicative bounds for p and T from Lemma 1 and 2, as we did in Lemma 9, we get the required result. □

Combining the lemmas, we now prove the main theorem.

Theorem 8. As $\pi = \overline{\text{ActClass}}(\mathbb{MDP}, p)$, π maximizes $\overline{classify}(\pi)$, and therefore

$$\overline{classify}(\pi^*) \leq \overline{classify}(\pi) \tag{25}$$

where $\pi^* = \text{ActClass}(\mathbb{MDP}^*, p^*)$. Applying Lemma 9 to $\overline{classify}(\pi^*)$ and Lemma 10 to $\overline{classify}(\pi)$ we get:

$$classify^*(\pi^*) \leq \overline{classify^{\#}}(\pi) \frac{(1+\alpha_{MDP})^2(1+\alpha_T)^{2H}}{(1-\alpha_{MDP})^2(1-\alpha_T)^{2H}} + N_{MDP}(\beta_{MDP} + 2\epsilon_{MDP}) + \beta_S + \beta_T + 2\epsilon_T$$

Assuming $\alpha_{MDP} \leq 1/2$ and $\alpha_T \leq 1/2H$, using Lemma 3 and then taking a loose upper bound we get:

$$\frac{(1+\alpha_{MDP})^2(1+\alpha_T)^{2H}}{(1-\alpha_{MDP})^2(1-\alpha_T)^{2H}} \leq 1 + 48\alpha_{MDP} + 48H\alpha_T$$

And therefore: $classify^*(\pi^*) \leq \overline{classify^{\#}}(\pi) + 48\alpha_{MDP} + 48H\alpha_T + N_{MDP}\beta_{MDP} + 2N_{MDP}\epsilon_{MDP} + \beta_S + \beta_T + 2\epsilon_T$.

For δ, we use union bound and add all the probabilities of failure that we saw in the analysis. □

TODO: Discussion about the classification algorithm, the approximation algorithm. Related work, conclusion, better intro, etc.

References

Adhikari, U., Morris, T. H., and Pan, S. Applying hoeffding adaptive trees for real-time cyber-power event and intrusion classification. *IEEE Transactions on Smart Grid*, 9 (5):4049–4060, 2017.

Ayer, T., Alagoz, O., and Stout, N. K. Or forum—a pomdp approach to personalize mammography screening decisions. *Operations Research*, 60(5):1019–1034, 2012.

Chadès, I., Carwardine, J., Martin, T. G., Nicol, S., Sabbadin, R., and Buffet, O. Momdps: a solution for modelling adaptive management problems. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

Fiechter, C.-N. Efficient reinforcement learning. In *Proceedings of the seventh annual conference on Computational learning theory*, pp. 88–97, 1994.

Fiechter, C.-N. Expected mistake bound model for on-line reinforcement learning. In *ICML*, volume 97, pp. 116–124, 1997.

Hauskrecht, M. and Fraser, H. Planning treatment of ischemic heart disease with partially observable markov decision processes. *Artificial Intelligence in Medicine*, 18 (3):221–244, 2000.

Hollinger, G. A., Mitra, U., and Sukhatme, G. S. Active classification: Theory and application to underwater inspection. In *Robotics Research*, pp. 95–110. Springer, 2017.

Kearns, M. and Singh, S. Near-optimal reinforcement learning in polynomial time. *Machine learning*, 49(2-3):209–232, 2002.

Valiant, L. G. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

Wu, B., Ahmadi, M., Bharadwaj, S., and Topcu, U. Cost-bounded active classification using partially observable markov decision processes. In *2019 American Control Conference (ACC)*, pp. 1216–1223. IEEE, 2019.

4. Appendix

Lemma 3. Expanding $(1+x)^n$:

$$
\begin{aligned}
&= 1 + \frac{xn}{1!} + \frac{x^2 n(n-1)}{2!} + \frac{x^3 n(n-1)(n-2)}{3!} + \dots \\
&\leq 1 + xn + \frac{x^2 n^2}{2} + \frac{x^3 n^3}{2^2} + \frac{x^4 n^4}{2^3} + \dots \\
&= 1 + xn(1 + \frac{xn}{2} + (\frac{xn}{2})^2 + (\frac{xn}{2})^3 + \dots) \\
&\leq 1 + xn(1 + \frac{1}{2} + (\frac{1}{2})^2 + (\frac{1}{2})^3 + \dots) \\
&\leq 1 + 2xn
\end{aligned}
$$

The reasons for first equality: Taylor expansion; first inequality: $(n-k) \leq n$ and $k! \geq 2(k-1)$; second equality: reorganizing the same series; second inequality: given $xn \leq 1$; third inequaility: sum of the geometric series.

Now expanding $(1-x)^n$:

$$
\begin{aligned}
&= 1 - \frac{xn}{1!} + \frac{x^2 n(n-1)}{2!} - \frac{x^3 n(n-1)(n-2)}{3!} + \dots \\
&= 1 - xn + \frac{x^2 n(n-1)}{2!}(1 - \frac{x(n-2)}{3} \\
&\qquad + \frac{x^2(n-2)(n-3)}{4.3} - \dots) \\
&\geq 1 - xn + \frac{x^2 n(n-1)}{2!}(1 - \frac{xn}{2} - \frac{x^2 n^2}{2^2} - \frac{x^3 n^3}{2^3} - \dots) \\
&\geq 1 - xn + \frac{x^2 n(n-1)}{2!}(1 - \frac{1}{2} - \frac{1}{2^2} - \frac{1}{2^3} - \dots) \\
&\geq 1 - xn + \frac{x^2 n(n-1)}{2!}(1-1) = 1 - xn
\end{aligned}
$$

The reasons for the equalities and inequalities are the same as the proof for $(1+x)^n$.

□