Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)

A POMDP Formulation of Proactive Learning

Kyle Hollins Wray and **Shlomo Zilberstein**
College of Information and Computer Sciences
University of Massachusetts
Amherst, MA 01003, USA
{wray,shlomo}@cs.umass.edu

Abstract

We cast the Proactive Learning (PAL) problem—Active Learning (AL) with multiple reluctant, fallible, cost-varying oracles—as a Partially Observable Markov Decision Process (POMDP). The agent selects an oracle at each time step to label a data point while it maintains a belief over the true underlying correctness of its current dataset's labels. The goal is to minimize labeling costs while considering the value of obtaining correct labels, thus maximizing final resultant classifier accuracy. We prove three properties that show our particular formulation leads to a structured and bounded-size set of belief points, enabling strong performance of point-based methods to solve the POMDP. Our method is compared with the original three algorithms proposed by Donmez and Carbonell and a simple baseline. We demonstrate that our approach matches or improves upon the original approach within five different oracle scenarios, each on two datasets. Finally, our algorithm provides a general, well-defined mathematical foundation to build upon.

Introduction

Active Learning (AL) techniques capture the process of taking an unlabeled dataset and labeling a selected subset by querying an omniscient oracle for labels (Cohn, Atlas, and Ladner 1994). In practice, however, active learning makes strong assumptions regarding the labeling process. Specifically, real world applications often involve multiple oracles, each of which may be reluctant to answer, incorrectly answer, and have data point-sensitive costs subject to a fixed budget (Attenberg and Provost 2011). Proactive Learning (PAL) captures all of these properties in a formal problem domain (Donmez and Carbonell 2008a). We present a Partially Observable Markov Decision Process (POMDP) solution for this inherently sequential optimization problem.

Within AL, multi-oracle (Ipeirotis et al. 2014; Yan et al. 2011), imprecise oracle (Golovin and Krause 2011; Ipeirotis et al. 2014), and cost-varying oracle (Culotta and McCallum 2005; Golovin and Krause 2011) scenarios have been explored separately in depth. Also, for single-oracle AL, both MDP (Lizotte, Madani, and Greiner 2003), POMDP (Jaulmes, Pineau, and Precup 2005), and other related methods (Golovin and Krause 2011) have been devised. These do not capture the entire multiple, fallible, reluctant, and cost-varying oracles found in the realistic PAL domain. As such, the MDP-like states, actions, observations, and transition functions differ markedly from ours. The distinct, but related, preference elicitation problem seeks to perform as few queries as possible while maximizing the belief of a user's preferences. POMDPs have been successfully implemented here; however, they do not maintain a belief over a dataset, model oracles, or manage a budget (Boutilier 2002).

Our primary contribution is an algorithm which maps a PAL problem directly to the true underlying sequential optimization problem using a POMDP. To the best of our knowledge, no one has proposed such a POMDP solution to PAL. We state and prove three propositions regarding the belief points and horizon required to rapidly produce high-quality solutions for the PAL POMDP using point-based methods. Additionally, we provide experimental evidence that demonstrates our approach either meets or exceeds the performance of the original algorithms and a random baseline.

The next section formally defines PAL, POMDPs, and our algorithm. Then, we provide a rigorous theoretical analysis of the point-based algorithm used to solve our PAL POMDP. Next, we present our experiments and discuss our findings. Finally, we conclude with a summary of our contributions.

An Automated Planning Approach

We formalize the general proactive learning problem and propose a general POMDP framework for solving it. To the best of our knowledge, neither has been previously formulated in this manner.

Proactive Learning Definition

Originally proposed by Donmez and Carbonell (2008a), proactive learning originally considered four relaxations to active learning. Their approach handled each of them separately. We state the problem in its most general form, simultaneously describing all four within one problem domain.

The Proactive Learning (PAL) problem is a tuple $\langle \mathcal{X}, \mathcal{Y}_l, \mathcal{O}, \mathcal{P}_r, \mathcal{P}_c, \mathcal{C} \rangle$. $\mathcal{X} = \mathcal{X}_l \cup \mathcal{X}_u$ is a dataset of d data points (zero-indexed), $\mathcal{X}_l$ denotes the d_l labeled data points with corresponding labels $\mathcal{Y}_l$, and $\mathcal{X}_u$ denotes the d_u unlabeled data points. $\mathcal{O}$ is a set of m oracles. $\mathcal{P}_r : \mathcal{X} \times \mathcal{O} \to [0, 1]$ denotes the probability that an oracle will respond for a data point. $\mathcal{P}_c : \mathcal{X} \times \mathcal{O} \to [0, 1]$ denotes the probability that an oracle's response is correct for a data point. These probabilities may be given (e.g., guarantees by oracles), estimated (e.g., cluster centroids), or a mix of both. $\mathcal{C} : \mathcal{X} \times \mathcal{O} \to \mathbb{R}^+$ is a cost function for an oracle labeling a data point.

The true underlying objective in PAL is to maximize classifier accuracy while minimizing cost. Unfortunately, we cannot directly measure classifier accuracy given that we do not necessarily have any labeled data for comparison. Thus, the objective of a PAL algorithm is to maximize a measure of the expected information gain, subject to a budget constraint $\beta \in \mathbb{R}^+$. Formally, let $\mathcal{S} \subseteq \mathcal{X} \times \mathcal{O}$ be a set of samples of data point-oracle pairs, and $\mathcal{V} : \mathcal{S} \to \mathbb{R}$ denote any value of information metric. Our optimization problem is to maximize $\mathbb{E}[\sum_{s \in \mathcal{S}} \mathcal{V}(s)]$ subject to $\sum_{\langle x,o \rangle \in \mathcal{S}} \mathcal{C}(x, o) \leqslant \beta$.

PAL Algorithm Initialization

As stated above, the probabilities $\mathcal{P}_r$ and $\mathcal{P}_c$ are obtained in one of two ways: (1) initially given, or (2) acquired using a clustering method. We experiment with both scenarios.

In many proactive learning domains, these probabilities are given. For example, consider medical domains that require a company "oracle" to conduct lab work in order to label a data point. Any contract with the company to do this work will clearly state estimates for duration, and thus oracle reluctance, as well as probabilistic guarantees regarding the quality of the labels, and thus oracle correctness. Obviously, costs are provided for each oracular company.

Other domains require a pre-processing step to obtain estimates of these probabilities. We employ a similar clustering method as the original PAL algorithms (Donmez and Carbonell 2008a) and others (Wallace et al. 2011). In summary, we are given an initial clustering budget $\beta_c < \beta$. Using this budget, we run k-means, obtain $k \propto \beta_c$ data points closest to the cluster centroids, query the oracles to obtain labels, and train a classifier on any labeled data points yielding parameters $\hat{w}$. Then, for each unlabeled data point $x_i \in \mathcal{X}_u$ we now have $Pr(y_i|x_i, \hat{w})$. Using this and a distance metric $d(x_i) \in [0, 1]$ from the closest cluster centroid denoted as x_c with label y_c (possibly undefined, denoted as $\varnothing$), we compute our probabilities for oracle $o \in \mathcal{O}$:

$$\mathcal{P}_r(x_i, o) = \sigma((2[y_c \neq \varnothing] - 1)(1 - d(x_i))) \quad (1)$$

$$\mathcal{P}_c(x_i, o) = \sigma((2 \max_{y \in \mathcal{Y}} Pr(y|x_i, \hat{w}) - 1)(1 - d(x_i))) \quad (2)$$

with standard sigmoid function $\sigma(\cdot)$ and Iverson brackets $[\cdot]$. This assumes (strongly) that if an oracle does not respond, then it is likely to not respond for data points nearby.

POMDP Definition

A POMDP is represented by the tuple $\langle S, A, \Omega, T, O, R \rangle$ (Smallwood and Sondik 1973; Sondik 1978; Kaelbling, Littman, and Cassandra 1998). S is a set of n states, A is a set of m actions, and Ω is a set of z observations. T is a state transition function that captures the stochastic Markovian state transitions after each action is taken, with $T : S \times A \times S \to [0, 1]$ such that $T(s, a, s') \equiv Pr(s'|s, a)$. O is an observation function that stochastically presents an observation to the agent based on the action performed and the true underlying state that resulted from that action, with $O : A \times S \times \Omega \to [0, 1]$ such that $O(a, s', \omega) \equiv Pr(\omega|a, s')$. Finally, $R : S \times A \to \mathbb{R}$ is a function mapping state-action pairs to rewards such that $R(s, a) \in \mathbb{R}$. The sequential optimization process considers a number of time steps called the *horizon* h. Infinite horizon ($h = \infty$) POMDPs could be approximated using a finite horizon ($h \in \mathbb{N}$) or solved directly using a variety of methods (Amato, Bernstein, and Zilberstein 2007). The overall objective is to maximize the cumulative reward over the problem horizon, where rewards are discounted by a *discount factor* $\gamma \in [0, 1]$ per time step.

The decision maker or agent must select actions without knowing the true underlying state of the system. Instead, it maintains a *belief* over the possible state denoted $b \in \triangle^n$, or a set of r belief points over the standard n-simplex $\triangle^n$ denoted $B \subseteq \triangle^n$. At every time step, the agent takes action a at belief b and makes observation ω. This updates the belief over all possible successor states s' following:

$$b'(s'|b, a, \omega) = \eta O(a, s', \omega) \sum_{s \in S} T(s, a, s') b(s) \quad (3)$$

with normalizing constant $\eta = Pr(\omega|b, a)^{-1}$ (Kaelbling, Littman, and Cassandra 1998). This belief is a sufficient statistic for the entire *history* of actions and observations the agent has taken. For notational brevity, we often denote $b' = [b'(s_1|b, a, \omega), \dots, b'(s_n|b, a, \omega)]^T$.

A *policy* π maps beliefs to actions $\pi : B \to A$. With a policy, we define a *value function* as $V : B \to \mathbb{R}$, denoting the expected reward at beliefs. Value functions are piecewise linear and convex (Smallwood and Sondik 1973). We represent them by a set of α-*vectors* $\Gamma = \{\alpha_1, \dots, \alpha_i\}$ such that each $\alpha_i = [V(s_1), \dots, V(s_n)]^T$, with the value of each state denoted as $V(s_i)$. A policy is defined by attaching an action to each α-vector, compactly denoting $V(b) = \alpha_i \cdot b$ and $\pi(b) = a_{\alpha_i} \in A$. We may write the value function for an initial belief b, following policy π, at horizon h as:

$$V^h_\pi(b) = \mathbb{E}\Big[\sum_{t=0}^{h-1} \gamma^t R(b^t, \pi^t(b^t)) \Big| b^0 = b, \pi \Big]$$

The objective is to select a policy which maximizes the expected reward earned over time. Thus, the optimal policy follows selecting maximal α-vectors at each belief point (Kaelbling, Littman, and Cassandra 1998):

$$V^t(b) = \max_{a \in A} b \cdot r_a + \sum_{\omega \in \Omega} \max_{\alpha \in \Gamma^{t-1}} \sum_{s \in S} b(s) V^t_{sa\omega\alpha} \quad (4)$$

with $r_a = [R(s_1, a), \dots, R(s_n, a)]^T$ and:

$$V^t_{sa\omega\alpha} = \gamma \sum_{s' \in S} O(a, s', \omega) T(s, a, s') \alpha(s')$$

Let initial α-vectors be $\alpha(s) = \underline{R}/(1 - \gamma)$, with $\underline{R} = \min_{s \in S} \min_{a \in A} R(s, a)$, for all $s \in S$. This guarantees α-vectors weakly monotonically increase (Lovejoy 1991).

The Proactive Learning POMDP Model

We begin by ordering the data points in $\mathcal{X}_u$ so that during policy execution we initially select the most informative data

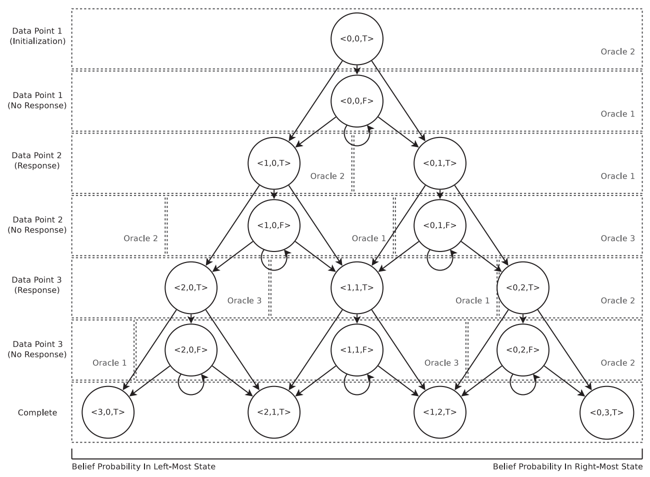

Figure 1: Example of PAL POMDP with $d_u = 3$ data points. States denote the current dataset's correct and incorrect labelings, as well as if the last query's oracle responded. Arrows denote the non-zero probabilistic state transitions, with duplicates for each specific action (oracle query) omitted for clarity. Boxes visually represent an example policy, mapping a range of beliefs regarding the *true* correctness of the dataset to an oracle selection action.

points to label, progressively selecting less informative ones at each time step until we either run out of data points or, more likely, the allotted budget. The order follows by selecting the oracle which will receive the highest utility, then selecting the data point which yields the highest utility given this fixed oracle. Formally, each $x \in \mathcal{X}_u$ and $o \in \mathcal{O}$, let $\hat{U}(x, o) = \mathcal{P}_r(x, o)\mathcal{P}_c(x, o)\mathcal{V}(x, o)/\mathcal{C}(x, o)$ be the utility. Given current ordered set $\mathcal{X}_o^{t-1} = \{x^1, \ldots, x^{t-1}\}$, the data point for step t denoted x^t is:

$$o^t = \operatorname*{argmax}_{o \in \mathcal{O}} \max_{x \in \mathcal{X}_u \backslash \mathcal{X}_o^{t-1}} \hat{U}(x, o)$$

$$x^t = \operatorname*{argmax}_{x \in \mathcal{X}_u \backslash \mathcal{X}_o^{t-1}} \hat{U}(x, o^t)$$

Importantly, this procedure only selects the *order*, which is connected to the POMDP state structure below. Hereafter, we will assume that $\mathcal{X}_u$ is reassigned to the ordering of $\mathcal{X}_o^{d_u}$.

We propose a mapping from a PAL problem to a POMDP. Figure 1 provides a visual explanation of the mapping, in addition to an example representation of a policy. The high-level idea is to represent the process of constructing a correctly labeled dataset as a sequential optimization problem using a POMDP. States in the POMDP capture the *quality* of the labelings within the dataset. As such, they include both the number of correctly and incorrectly labeled data points. In order to properly adjust the policy based on oracle responses, we also record if the previous oracle responded or not as part of the state. Formally, for d_u unlabeled data points, $S = \{\langle c, i, r\rangle | c, i \in \mathbb{N}, r \in \{\mathcal{T}, \mathcal{F}\}, 0 \leqslant c + i < d_u\} \cup \{\langle c, i, \mathcal{T}\rangle | c, i \in \mathbb{N}, c + i = d_u\}$. For example, $\langle 3, 2, \mathcal{F}\rangle \in S$, means the dataset has 3 correct labels, 2 incorrect labels, and there was no response from the previously queried oracle. The final dataset's correctness state does not need $\mathcal{F}$, since it is complete upon reaching the end.

Actions within our model directly correspond to which oracle should be queried, i.e., $A = \mathcal{O}$. Observations are simply if we observed an oracle response, i.e., $\Omega = \{\mathcal{T}, \mathcal{F}\}$. This simplified set of observations works due to the important structure of the subsequent belief update, which takes into consideration the state transition function T and observation transition function O (detailed below). These two probability functions combined with this definition of observations, yield the exact desired result: Certainty about the size of the labeled dataset and previous oracle response, and uncertainty about the true labelings within the current labeled dataset. This lets us map beliefs over this to optimal actions, i.e., select the best oracle to label the next data point.

The state transition function $T(s, a, s')$, for states $s, s' \in S$ and action $a \in A$, captures both the probability of an oracle responding and the probability it successfully labels the data point. Formally, for $s = \langle c, i, r\rangle$ and $s' = \langle c', i', r'\rangle$:

$$T(s, a, s') = \tag{5}$$
$$\begin{cases} 1 - \mathcal{P}_r(x_{c+i}, a) & \text{if } c{=}c', i{=}i', r'{=}\mathcal{F} \\ \mathcal{P}_r(x_{c+i}, a)\mathcal{P}_c(x_{c+i}, a) & \text{if } c{+}1{=}c', i{=}i', r'{=}\mathcal{T} \\ \mathcal{P}_r(x_{c+i}, a)(1 - \mathcal{P}_c(x_{c+i}, a)) & \text{if } c{=}c', i{+}1{=}i', r'{=}\mathcal{T} \\ 1 & \text{if } c{=}c', i{=}i', c{+}i{=}d_u \\ 0 & \text{otherwise} \end{cases}$$

Observation transition function $O(a, s', \omega)$, for action $a \in A$, successor $s' \in S$, and observation $\omega \in \Omega$, only needs to inform the agent if the oracle responded. For $s' = \langle c', i', r'\rangle$:

$$O(a, s', \omega) = [\omega = r'] \tag{6}$$

Lastly, the reward function $R(s, a)$, for state $s \in S$ and action $a \in A$, is built upon the original utility function proposed by Donmez and Carbonell (2008a), which essentially uses the value of information divided by the cost. For this value of the information gained by labeling, $\mathcal{V}(x)$ for a data point $x \in \mathcal{X}_u$, we use the same uncertainty weighted density score as Donmez and Carbonell. This metric assumes that data points within the same region are relatively clustered and share the same label. Thus, we first define a neighborhood of indexes N_x over x that are within some threshold distance τ: $N_x = \{i \in \{0, \ldots, d_u{-}1\} | \|x - x_i\| < \tau\}$. $\mathcal{V}(x)$ is defined in Equation 7 below, with weights $\hat{w}$ and entropy $H(y_k | x_k, \hat{w})$ using the probability of labeling data point x_k as y_k following the model so far (see initialization section) (Donmez and Carbonell 2008b). The reward weighs $\mathcal{V}$, $\mathcal{C}$, and a ratio of correctly versus incorrectly labeled points, in addition to a penalty of $\epsilon < 1$ *only* if they select a reluctant oracle when they just failed to receive a label. Formally, for state $s = \langle c, i, r\rangle$ and action a:

$$R(s, a) = \frac{c + 1}{i + 1} \frac{\mathcal{V}(x_{c+i})}{\mathcal{C}(x_{c+i}, a)} \epsilon(s, a) \tag{7}$$

$$\mathcal{V}(x) = \sum_{k \in N_x} \exp(-\|x - x_k\|^2) H(y_k | x_k, \hat{w})$$

with $\epsilon(s, a) = \epsilon$ only if $\mathcal{P}_r(x_{c+i}, a) < 1$ and $r = \mathcal{F}$ as described above; $\epsilon(s, a) = 1$ otherwise.

Algorithm 1 Proactive Learning POMDP: Initially unknown oracles; thus, it requires clustering.

```
Require: ⟨$\mathcal{X}_u, \mathcal{X}_l, \mathcal{Y}_l$⟩: The unlabeled and labeled datasets.
Require: $\mathcal{O}$: The set of oracles that may be queried.
Require: ⟨$\beta_c, \beta$⟩: The initial clustering budget and entire budget.
 1: $\langle \mathcal{P}_r, \mathcal{P}_c, \mathcal{C} \rangle \leftarrow init\_pal(\mathcal{X}_u, \mathcal{X}_l, \mathcal{Y}_l, \mathcal{O}, \beta_c, \beta)$
 2: $\langle S, A, \Omega, T, O, R \rangle \leftarrow init\_pomdp(\mathcal{X}_u, \mathcal{O}, \mathcal{P}_r, \mathcal{P}_c, \mathcal{C})$
 3: $\pi \leftarrow solve(S, A, \Omega, T, O, R)$
 4: $b(s) \leftarrow [s = \langle 0, 0, \mathcal{T} \rangle], \quad \forall s \in S$
 5: $i \leftarrow 0$
 6: $c_t \leftarrow \beta_c$
 7: while $c_t < \beta$ or $\mathcal{X}_u = \emptyset$ do
 8:     $\langle y, c \rangle \leftarrow query(x_i, \pi(b))$
 9:     $\omega \leftarrow \mathcal{F}$
10:     if $y \neq \emptyset$ then
11:         $\langle \mathcal{X}_u, \mathcal{X}_l, \mathcal{Y}_l \rangle \leftarrow \langle \mathcal{X}_u \backslash \{x_i\}, \mathcal{X}_l \cup \{x_i\}, \mathcal{Y}_l \cup \{y\} \rangle$
12:         $i \leftarrow i + 1$
13:         $\omega \leftarrow \mathcal{T}$
14:     end if
15:     $b(s) \leftarrow b'(s|b, \pi(b), \omega), \quad \forall s \in S$
16:     $c_t \leftarrow c_t + c$
17: end while
18: return $\langle \mathcal{X}_u, \mathcal{X}_l, \mathcal{Y}_l \rangle$
```

The entire PAL POMDP procedure is detailed in Algorithm 1. The functions $init\_pal(\cdot)$ and $init\_pomdp(\cdot)$ implement the previous two sections, respectively. The function $solve(\cdot)$ solves the POMDP, returning a policy π, using Point-Based Value Iteration (PBVI) (Pineau, Gordon, and Thrun 2003). Note another Iverson bracket on Line 4. The function $query(\cdot)$ queries an oracle for a data point label.

A variant of Algorithm 1 reorders all future data points upon each successful query after updating $\mathcal{P}_r$ and $\mathcal{P}_c$ (Equations 1 and 2). We then re-solve this modified POMDP on every step. Note that the current belief b over the dataset does not change, since the prior data points' ordering remains fixed. This process is obviously computationally expensive. In practice, however, domains which have a long real-world time delay for oracle queries (e.g., biological experiments) allow plenty of time to re-solve the POMDP.

Theoretical Analysis and Optimizations of Point-Based Value Iteration

Exact solutions to POMDPs require defining a *policy tree*. These trees have a height equal to the horizon h, with each node's branching factor equal to the number of possible observations z. Each node in this tree is assigned an action to take given its history of observations. In practice, this makes exact solutions intractable for anything but the smallest of POMDPs; in fact, POMDPs are PSPACE-hard (Papadimitriou and Tsitsiklis 1987). Instead, point-based methods were developed which operate over a fixed set of belief points, with additional methods for intelligently adding new belief points to the set (Pineau, Gordon, and Thrun 2003). We briefly describe the point-based method, then establish three properties of our PAL POMDP that enable us to define B and h in order to quickly produce high-quality solutions.

Point-Based POMDP Solvers

Point-Based Value Iteration (PBVI) was proposed by Pineau, Gordon, and Thrun (2003). It operates over the set of α-vectors Γ^t, updating each one in the set at each time step t. Over these time steps, we only focus on a fixed set of belief points B. This is commonly defined using intermediate variables Γ_b and $\Gamma_{a\omega}$:

$$\Gamma^t_{a\omega} = \{[V^t_{s_1 a\omega\alpha}, \ldots, V^t_{s_n a\omega\alpha}]^T, \forall \alpha \in \Gamma^{t-1}\}$$

$$\Gamma^t_b = \{r_a + \sum_{\omega\in\Omega} \operatorname*{argmax}_{\alpha\in\Gamma^t_{a\omega}} \alpha \cdot b, \forall a \in A\}, \quad \forall b \in B \tag{8}$$

$$\Gamma^t = \{\operatorname*{argmax}_{\alpha\in\Gamma^t_b} \alpha \cdot b, \forall b \in B\} \tag{9}$$

Theoretical Analysis

The quality of the solution returned by this approach depends heavily on which belief points are chosen for B. Common algorithms explore the policy tree (i.e., reachable belief points) to define a B which best represents the belief space, to obtain the highest (and thus most accurate) values. Our specific PAL POMDP enables us to define three strong properties regarding the reachable belief points and the horizon required for PBVI. First we define helpful variables.

Let $b^0 \in \triangle^n$ be the initial belief from Algorithm 1, Line 4. Let $\hbar = \langle a^0, \omega^1, a^1, \omega^2, \ldots, a^{h-1}, \omega^h \rangle$ be any history. Let $b^h \in \triangle^n$ be the resultant belief applying Equation 3 at each time step in history $\hbar$, starting with initial belief b^0. Finally, let $S_{d\omega} = \{s \in S | s = \langle c, i, r \rangle, c + i \neq d \vee r \neq \omega\}$ be the set of states which are not at data point index d or did not have previous oracle response ω (or both).

Our first proposition describes the structure of our belief points, following any history of actions and observations. Importantly, it means that the agent's uncertainty of the true state space is *only* over the dataset accuracy it built so far.

Proposition 1 (Guarantee: Belief Is Always Over Dataset Correctness)**.** For any history $\hbar$, with $d^h = |\{i \in \{1, \ldots, h\} | \omega^i = \mathcal{T}\}|$, if $s \in S_{d^h\omega^h}$, then $b^h(s) = 0$.

Proof. By induction on h.
Base Case: h=0. Thus, $b^0 = b^h$ with $s^0 = \langle 0, 0, \mathcal{T} \rangle$ we have $d^h = 0$ and implicit $\omega^0 = \mathcal{T}$. This yields $S_{d^0\omega^0} = S \backslash \{\langle 0, 0, \mathcal{T} \rangle\}$. By Algorithm 1, Line 4, $b^0(s) = 0$ for all $s \neq \langle 0, 0, \mathcal{T} \rangle$, i.e., $s \in S_{d^0\omega^0}$. Thus, the base case is shown to hold true.

Inductive Step: Assume true for $h-1$ (induction hypothesis), must show that for h, for all $s' \in S_{d^h\omega^h}$ (given by $\hbar$) that $b^h(s') = 0$. Given our history $\hbar$, and thus a^{h-1} and ω^h, we apply Equation 3 to b^{h-1} and prove b^h equals zero for all $s' \in S_{d^h\omega^h}$:

$$b^h(s'|b^{h-1}, a^{h-1}, \omega^h) = \eta O(a^{h-1}, s', \omega^h) \sum_{s\in S} T(s, a^{h-1}, s') b^{h-1}(s)$$

Two cases (by $S_{d^h\omega^h}$): (1) $r' \neq \omega^h$, and (2) $c' + i' \neq d^h$.

Case 1: $r' \neq \omega^h$. Thus, by Equation 6, we have $O(a^{h-1}, s', \omega^h) = 0$. Therefore, $b^h(s') = 0$.

Case 2: $c'+i' \neq d^h$. We assume $r' = \omega^h$, otherwise $b^h(s') = 0$. Thus, $O(a^{h-1}, s', \omega^h) = 1$ and we eliminate values in which b^{h-1} is zero (induction hypothesis):

$$b^h(s'|b^{h-1}, a^{h-1}, \omega^h) = \eta \cdot 1 \cdot \sum_{s \in S} T(s, a^{h-1}, s') b^{h-1}(s)$$
$$= \eta \sum_{s \in S \setminus S_{d^{h-1}\omega^{h-1}}} T(s, a^{h-1}, s') b^{h-1}(s)$$

We must show that for all remaining $s = \langle c, i, r \rangle \in S \setminus S_{d^{h-1}\omega^{h-1}}$, that T or b^{h-1} is zero. By definition of $S_{d^{h-1}\omega^{h-1}}$, we know $c + i = d^{h-1}$ and $r = \omega^{h-1}$. By definition of $S_{d^h\omega^h}$, we know $c' + i \neq d^h$ or $r' \neq \omega^h$. By definition of d^h, we also know $d^{h-1} \leqslant d^h$, and since $c' + i' \neq d^h$ it implies $d^{h-1} < d^h$ must be true. This, in turn, implies $\omega^h = \mathcal{T}$ by definition of d^h. Three possible sub-cases: (1) $c' + i' < d^{h-1}$, (2) $c' + i' > d^h$, and (3) $c' + i' = d^{h-1}$.

By Equation 5, sub-case (1) has $T(s, a^{h-1}, s') = 0$, since $c' + i' < d^{h-1} = c + i$ implies all four non-zero conditions cannot occur; informally, it describes transitioning to a state with less data points. For sub-case (2), $c' + i' > d^h > d^{h-1} = c + i$, which again implies all four non-zero conditions cannot occur; informally, it describes skipping data points, because s and s' would strictly be two data point indexes away from one another. Lastly, sub-case (3) implies $c' + i' = d^{h-1} = c + i$. Since $r' = \omega^h = \mathcal{T}$, again, all four non-zero conditions cannot occur, namely one ($\omega^h = \mathcal{T}$) and four ($d^{h-1} < d^h \leqslant d_u$); informally, it describes a contradiction in which it receives a response but does not transition. In each sub-case, $T(s, a^{h-1}, s') = 0$, thus $b^h(s') = 0$.

Case 1 and 2 are proven: $b^h(s') = 0$ for all $s' \in S_{d^h\omega^h}$.

The Base Case and Inductive Step are proven, thus by induction for any history $\hbar$, for all $s \in S_{d^h\omega^h}$, $b^h(s) = 0$. □

The second proposition has four results. First, it solidifies a bound on the size of any belief point vector's potential non-zero values. Second, this is bounded by a value proportional to the number of data points or square of the states, such that as the POMDP grows, the maximal number of the non-zero values in a belief point is asymptotically smaller. Third, the size of the POMDP itself grows *quadratically* with the number of data points, *not exponentially*. Lastly, the application of this insight enables us to compute dot products with the belief point vectors b over just *potential* non-zero elements (as in Equations 4, 8, and 9). We found this to vastly improve the performance of PBVI.

Proposition 2 (Bound: Reachable Belief Size; POMDP Size)**.** For any history $\hbar$ and its belief $b^h \in \triangle^n$, the number of non-zero elements in b^h is bounded: $|\{s \in S | b^h(s) > 0\}| \leqslant d_u = \sqrt{n} - 1$.

Proof. By Proposition 1, if $s \in S_{d^h\omega^h}$, then $b^h(s) = 0$.

$$|\{s \in S | b^h(s) > 0\}| = |\{s \in S \setminus S_{d^h\omega^h} | b^h(s) > 0\}|$$
$$\leqslant |\{s \in S \setminus S_{d^h\omega^h} | b^h(s) \geqslant 0\}| \leqslant |S \setminus S_{d^h\omega^h}|$$
$$= |\{\langle c, i, r \rangle \in S | c + i = d^h \wedge r = \omega^h\}|$$
$$\leqslant |\{\langle c, i, r \rangle \in S | c + i = d^h\}| \leqslant d_u$$

Let us define a function which maps a data point index $k \in \{0, \ldots, d_u\}$ to how many states have that index: $f(k) = |\{\langle c, i, r \rangle \in S | c + i = k\}|$. By definition of states S:

$$n = \sum_{k=0}^{d_u} f(k) = 2(1 + 2 + \cdots + d_u) + (d_u + 1)$$
$$= 2\frac{d_u(d_u + 1)}{2} + (d_u + 1) = (d_u + 1)^2$$

Therefore, $|\{s \in S | b^h(s) > 0\}| \leqslant d_u = \sqrt{n} - 1$. □

Our third proposition determines the horizon h to select for PBVI in order to guarantee h is large enough to capture the entire labeling process.

Proposition 3 (Bound: Required Horizon)**.** Let $\underline{\mathcal{P}}_r = \min_{x \in \mathcal{X}_u} \min_{o \in \mathcal{O}} \mathcal{P}_r(x, o)$. For an $h \geqslant (1 - \epsilon) d_u / \underline{\mathcal{P}}_r$, state $s^h = \langle c^h, i^h, r^h \rangle \in S$ will have $c^h + i^h = d_u$ with probability $1 - \epsilon$.

Proof. Given history $\hbar$, the action sequence induces a Markov process with respect to the true underlying states (with $s^0 = \langle 0, 0, \mathcal{T} \rangle$). The probabilities $Pr(s^{t+1} | s^t, a^t)$ vary following equation 5. A lower bound over all time steps (possible data points) and oracles (possible actions) reduces the process to a Binomial distribution $Z \sim Binomial(h, \underline{\mathcal{P}}_r)$, in which the number of "successes" maps to the data point index up to d_u. We apply Markov's inequality in order to determine h:

$$1 - \epsilon = Pr(Z \geqslant d_u) \leqslant \frac{\mathbb{E}[Z]}{d_u} = \frac{h \underline{\mathcal{P}}_r}{d_u} \Rightarrow h \geqslant (1 - \epsilon) \frac{d_u}{\underline{\mathcal{P}}_r} \; \square$$

Experimentation

We create similar experiments to those of Donmez and Carbonell for two of the well-known UCI datasets they used: Adult and Spambase (Lichman 2013). Adult contains 32544 people each with 14 features. The objective is to classify which individuals make over \$50K a year. Spambase contains 4601 emails each with 57 features. The objective is to classify which emails are spam. We consider five *oracle scenarios*: (1-3) Original #1-#3, (4-5) Complex #1-#2. In all cases, we vary the budget β from 3 to 21 with an interval of 3, and randomly sample 100-300 data points for $\mathcal{X}_u$ and use the remaining, up to 10000, for our test set. We use a support vector machine for the final classifier, training with the proactively learned $\mathcal{X}_l$ and $\mathcal{Y}_l$. Importantly, we only select the top 40 points for our POMDP's $\mathcal{X}_u$ due to their complexity ($n = (d_u + 1)^2 = 1681$). If a budget remains after 40 labelings, then we execute another POMDP with the next 40 points, and so on, until the budget was exhausted. Each configuration of dataset, oracle scenario, and budget is run 10 times and averaged to produce the final accuracy results shown in Figure 2. PBVI uses horizon $2d_u$ (increasing it only improves performances) and discount factor 0.9. Our implementation uses Python 3.4.3 with scikit-learn 0.16.1, NumPy 1.9.2, and SciPy 0.15.1, run on an Intel(R) Core(TM) i7-4702HQ CPU at 2.20GHz, 8GB of RAM, and a Nvidia(R) GeForce GTX 870M. We leverage a high-performing GPU-based implementation of

PBVI using CUDA(C) 6.5 (Wray and Zilberstein 2015a; 2015b).

We compare our algorithm with the three original decision-theoretic algorithms designed for a reluctant, fallible, and cost-varying oracles denoted as PAL #1, #2, and #3, respectively (Donmez and Carbonell 2008a). PAL #1 selects oracles using a utility of $\mathcal{P}_r(x,o)\mathcal{V}(x)/\mathcal{C}_{round}$, for data point x and oracle o. $\mathcal{C}_{round}$ is for reluctant oracles, aggregating the cost spent so far trying to label the same data point. Preference shifts if an oracle is continually queried but fails to respond. PAL #2 selects oracles using a utility of $\mathcal{P}_c(x,o)\mathcal{V}(x)/\mathcal{C}(x,o)$. PAL #3 selects oracles using a utility of $\mathcal{V}(x) - \mathcal{C}(x,o)$. We also include a random baseline.

In Original #1-#3, we re-implement the original three scenarios and their oracles using the process by Donmez and Carbonell. Original #1 has a normal and reluctant oracle, Original #2 has a normal and fallible oracle, and Original #3 has a normal and cost-varying oracle. These three scenarios assume $\mathcal{P}_r$ and $\mathcal{P}_c$ are unknown, requiring a clustering step using $\beta_c = \min\{5, \beta/2\}$. The normal and other oracles cost 0.5 and 0.125, respectively. We observe that our method has similar or improved performance in comparison with PAL #1-#3. This is to be expected, since both the PAL POMDP and PAL #i algorithms were designed to handle the oracle within scenario Original #i. This also supports the accuracy of our implementation of original PAL #1-#3, reproducing their above-baseline performance. Furthermore, it demonstrates that the cluster approach properly estimates $\mathcal{P}_r$ and $\mathcal{P}_c$ (initially unknown) for use with our algorithm. These scenarios, however, are quite simple, serving as a first comparison of individual PAL #1-#3 and our PAL POMDP.

In Complex #1-#2, we seek to truly compare the algorithms by removing the noise introduced by these initial clustering methods which estimate the probabilities, since they share the initial clustering step regardless. (Let $\langle X, Y, Z\rangle$ denote $\mathcal{P}_r(x,o) \in X$, $\mathcal{P}_c(x,o) \in Y$, and $\mathcal{C}(x,o) \in Z$ for all $x \in \mathcal{X}$ and $o \in \mathcal{O}$.) In Complex #1, we create four randomly assigned $\langle [0.25, 1], [0.25, 1], [0.01, 1]\rangle$. This truly represents a realistic scenario with heavily mixed oracles, each with its own benefits and drawbacks for different data points. Complex #1 for both Adult and Spambase strongly show the capability of our method. Finally, in Complex #2, we create three oracles: $\langle\{1\}, \{1\}, \{1\}\rangle$, $\langle\{0.5\}, \{0.01\}, \{0.01\}\rangle$, and $\langle\{0.01\}, \{0.5\}, \{0.01\}\rangle$. This extreme example illustrates our POMDP PAL algorithm's success in solving the true sequential optimization problem which weighs all oracle information, as well as the drawbacks of the previous algorithms. It is evident from both datasets that the previous algorithms select either the severely fallible or reluctant oracles. Our approach outperforms PAL #1-#3 as well as the random baseline, demonstrating overall that it builds upon the original algorithms and is able to select the correct oracle each time.

Conclusion

We propose a general solution for proactive learning which casts the problem as a POMDP, taking into account the observed history in order to optimally select oracles and construct a high quality labeled dataset. This approach simultaneously captures all aspects of proactive learning (multiple reluctant, fallible, cost-varying oracles) under one unified mathematical framework. We provide three propositions which assure point-based methods will quickly produce quality policies for large datasets. Our algorithm is compared with the original three PAL algorithms, as well as a simple baseline, on two distinct datasets. We explore five oracle configurations, demonstrating our algorithm builds upon the original three, outperforming them when presented with realistic reluctant, fallible, and cost-varying oracles.

In future work, we hope to introduce richer aspects of or-

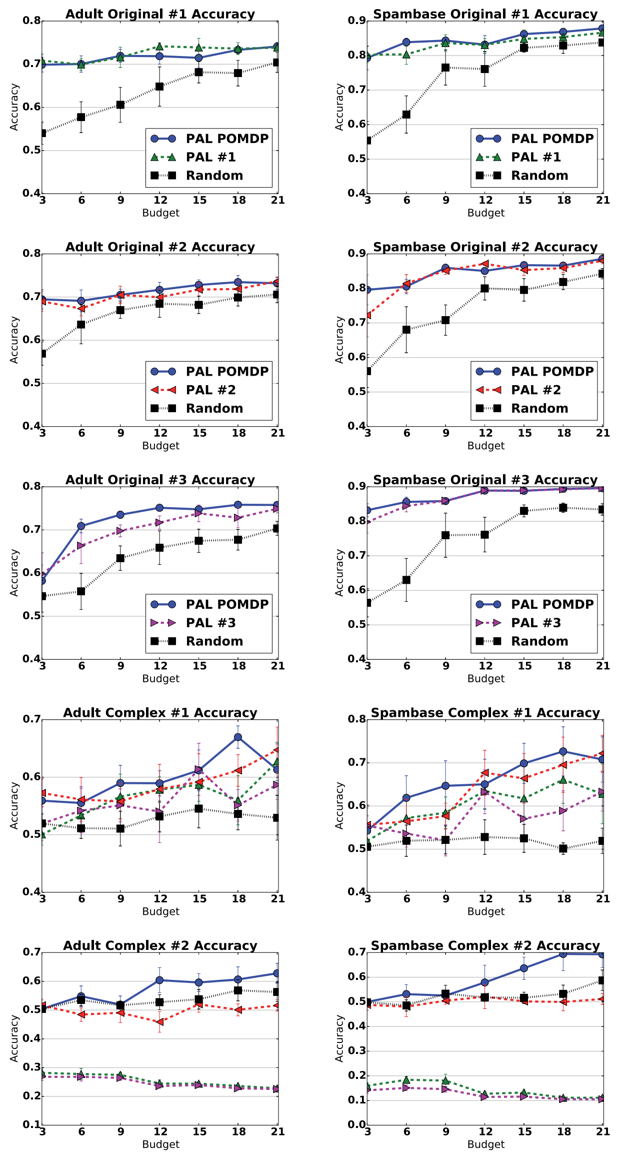

Figure 2: Mean accuracies with 95% confidence intervals for Adult (left) and Spambase (right) with five oracle scenarios: Original #1-#3 and Complex #1-#2 (top to bottom). Complex #1-#2's algorithms are marked the same as Original #1-#3 (legend omitted for clarity).

acles (e.g., responses), data point selection (e.g., relabeling), and integrate this into other domains (e.g., robotics). Our model's data point ordering $\hat{U}$, clustering step, and reward function components are purposefully built from the PAL algorithms' aspects (Donmez and Carbonell 2008a). Considering different methods within these components is a fertile ground for future exploration. Additionally, our experiments are purposefully designed to be the same as the original paper. This guarantees the accuracy of our implementations and allows for comparison with more difficult scenarios (Complex #1-#2). We will investigate additional application domains with this solid foundation established. Another extension of our work is to use a constrained POMDP to explicitly incorporate the budget; however, this alone presents numerous interesting facets which can be explored beyond the scope of this paper. Finally, we will provide our source code so that others may more easily build upon this work to design and implement high caliber proactive learners.

Acknowledgments We thank the anonymous reviewers for their helpful comments. Also, we thank Luis Pineda for his valuable comments. This work was supported in part by the National Science Foundation grant number IIS-1405550.

References

Amato, C.; Bernstein, D. S.; and Zilberstein, S. 2007. Solving POMDPs using quadratically constrained linear programs. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, 2418–2424.

Attenberg, J., and Provost, F. 2011. Inactive learning? Difficulties employing active learning in practice. *ACM SIGKDD Explorations Newsletter* 12(2):36–41.

Boutilier, C. 2002. A POMDP formulation of preference elicitation problems. In *Proceedings of the 18th AAAI Conference on Artifical Intelligence*, 239–246.

Cohn, D.; Atlas, L.; and Ladner, R. 1994. Improving generalization with active learning. *Machine Learning* 15(2):201–221.

Culotta, A., and McCallum, A. 2005. Reducing labeling effort for structured prediction tasks. In *Proceedings of the 20th AAAI Conference on Artifical Intelligence*, 746–751.

Donmez, P., and Carbonell, J. G. 2008a. Proactive learning: Cost-sensitive active learning with multiple imperfect oracles. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM)*, 619–628.

Donmez, P., and Carbonell, J. G. 2008b. Paired sampling in density-sensitive active learning. In *Proceedings of the 10th International Symposium on Artificial Intelligence and Mathematics (ISAIM)*.

Golovin, D., and Krause, A. 2011. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research (JAIR)* 42:427–486.

Ipeirotis, P. G.; Provost, F.; Sheng, V. S.; and Wang, J. 2014. Repeated labeling using multiple noisy labelers. *Data Mining and Knowledge Discovery* 28(2):402–441.

Jaulmes, R.; Pineau, J.; and Precup, D. 2005. Active learning in partially observable Markov decision processes. In *Proceedings of the 16th European Conference on Machine Learning (ECML)*, 601–608.

Kaelbling, L. P.; Littman, M. L.; and Cassandra, A. R. 1998. Planning and acting in partially observable stochastic domains. *Journal of Artificial Intelligence Research (JAIR)* 101(1):99–134.

Lichman, M. 2013. UCI machine learning repository. URL: http://archive.ics.uci.edu/ml.

Lizotte, D. J.; Madani, O.; and Greiner, R. 2003. Budgeted learning of naive-Bayes classifiers. In *Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence (UAI)*, 378–385.

Lovejoy, W. S. 1991. Computationally feasible bounds for partially observed Markov decision processes. *Operations Research* 39(1):162–175.

Papadimitriou, C. H., and Tsitsiklis, J. N. 1987. The complexity of Markov decision processes. *Mathematics of Operations Research* 12(3):441–450.

Pineau, J.; Gordon, G.; and Thrun, S. 2003. Point-based value iteration: An anytime algorithm for POMDPs. *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI)* 3:1025–1032.

Smallwood, R. D., and Sondik, E. J. 1973. The optimal control of partially observable Markov processes over a finite horizon. *Operations Research* 21(5):1071–1088.

Sondik, E. J. 1978. The optimal control of partially observable Markov processes over the infinite horizon: Discounted costs. *Operations Research* 26(2):282–304.

Wallace, B. C.; Small, K.; Brodley, C. E.; and Trikalinos, T. A. 2011. Who should label what? Instance allocation in multiple expert active learning. In *Proceedings of the SIAM International Conference on Data Mining (SDM)*, 176–187.

Wray, K. H., and Zilberstein, S. 2015a. Multi-objective POMDPs with lexicographic reward preferences. In *Proceedings of the 24th International Joint Conference of Artificial Intelligence (IJCAI)*, 1719–1725.

Wray, K. H., and Zilberstein, S. 2015b. A parallel point-based POMDP algorithm leveraging GPUs. In *AAAI Fall Symposium on Sequential Decision Making for Intelligent Agents (SDMIA)*, 95–96.

Yan, Y.; Rosales, R.; Fung, G.; and Dy, J. G. 2011. Active learning from crowds. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 1161–1168.