Optimal Sensor Scheduling via Classification Reduction of Policy Search (CROPS)

Doron Blatt and Alfred O. Hero
Department of Electrical Engineering and Computer Science
University of Michigan
Ann Arbor, Michigan, USA

***Abstract*—**

The problem of sensor scheduling in multi-modal sensing systems is formulated as the sequential choice of experiments problem and solved via reinforcement learning methods. The sequential choice of experiments problem is a partially observed Markov decision problem (POMDP) in which the underlying state of nature is the system's state and the sensors' data are noisy state observations. The goal is to find a policy that sequentially determines the best sensor to deploy based on past data, which maximizes a given utility function while minimizing the deployment cost. Several examples are considered in which the exact model of the measurements given the state of nature is unknown but a generative model (a simulation or an experiment) is available. The problem is formulated as a reinforcement learning problem and solved via a reduction to a sequence of supervised classification subproblems. Finally, a simulation and an experiment with real data demonstrate the promise of our approach.

I. Introduction

The advent of agile sensing systems that collect data through a variety of sensing modalities has brought about new and exciting challenges to the field of signal processing. Agile, multi-modal, sensing (see e.g. [16] and [13]) exploit the capability of controlling the data collection process. Examples of agile sensing systems include a radar that can control its beam direction, a land mine detector that can deploy radar or seismic sensors, or a LANDSAT satellite that can control the frequency band of its radar. The key element that differentiates agile sensing systems from other data collection systems is a resource allocation constraint that precludes using all sensor modalities at all times. We formulate agile sensing as an optimization in which the system must automatically select the best sensing modality based on past observations to maximize a given objective function while minimizing the data collection cost.

When formulated as a sequential choice[1] of experiments problem [8], the agile sensing problem consists of an episodic task that is divided into a sequence of decision epochs. Each episode begins as the first observation is collected. Then, at each subsequent decision epoch two decisions are made. The first one is to decide if the amount of information collected thus far is sufficient for making inference (detection or estimation) on the data with a desired accuracy or whether more observations are required. This first decision also determines the choices available at the second decision. If more observations are required, the next best sensor modality needs to be determined. If the information is deemed sufficient for inference, the final estimation or detection decision is made. Every sensor modality has an associated deployment cost and a decision rule must balance the expected information gain from a sensor deployment, which results in improved inference capabilities, with the deployment cost. The collection of decision rules, i.e., the sequence of mappings from past observations to the decision space, is called a policy and the goal is to find a policy that optimally trades-offs the overall average sensor deployment costs and the estimation or detection performance, e.g., mean squared estimation error or classification error rate.

The problem of finding optimal policies for sequential choice of experiments suffers from the curse-of-dimensionality [4] and scenarios in which a closed form solution for the optimal policy exists are rare. Past research has focused on the asymptotic regime in which one assumes a large number of data collection iterations (or sensor dwells) and low sensor deployment cost (see [15] and references therein). Another focus has

This research was partially supported by DARPA-MURI grant ARO DAAD 19-02-1-0262.

[1]The key difference from the related sequential design of experiment problem is that instead of adapting a set of continuous experiment parameters, here we choose from a finite set of fixed experiments.

been on "experiment sufficiency" – when is one experiment (or sensor modality) always better than another experiment (see [11] and references therein).

In this paper, we take a different approach. We assume that the underlying model is unknown and aim at finding *approximate* solutions to the optimal policy. In particular, in the absence of a model, optimal policies are approximated from data using a generative model, where data is generated by a simulator or collected in a field experiment. It is shown that this problem formulation falls into the class of reinforcement learning problems and the Classification Reduction of Policy Search (CROPS) methodology that has been recently proposed by the authors [7] is applied. Two case studies are reported as well. The first is the problem of finding sensor scheduling policies for land-mine detection. For this problem a simulator is used to generate data which is then used for policy search. The second problem is perform optimal waveform selection for a multi-band radar on a land classification satellite. In this application competitive policies are found from experimental LANDSAT data.

II. Problem Formulation

Let $X_1 \in \mathcal{X}_1, X_2 \in \mathcal{X}_2, \ldots, X_K \in \mathcal{X}_K$ be K random variables that correspond to the outputs of K sensors or K sensor modalities. Note that each of these random variables lies, in general, in a different space. We append each random variables with its index so that a value of an observation also indicates which sensor was used to collect it. Let $Y \in \mathcal{Y}$ be a discrete random variable that represent the state of nature whose value we try to predict. The presented results can also be applied when Y is a continuous random variables, whose value we try to estimate, but we focus on the detection problem for concreteness.

A policy π specifies which sensor to deploy first, say sensor k. Then based of the value of X_k the policy determines if an accurate prediction of Y is possible, and if so, what is the best prediction, or, otherwise, which is the next best sensor to deploy to collect additional data. This process continues until either a prediction of Y is made or all available sensors are deployed. We assume that each sensor can be applied at most once and hence, the total number observations is bounded by K. Therefore, a policy π is sequence of $K+1$ decision rules $\pi = [\pi_1, \pi_2, \ldots, \pi_{K+1}]$. This assumption is valid when the randomness in the process, e.g. the observation noise, is governed by clutter that cannot be averaged out by repeated measurements, rather than by thermal noise. Note that π_1 simply indexes the first sensor to deploy (excluding the possibility of predicting Y without taking any observations), and hence, $\pi_1 \in \{1, 2, \ldots, K\}$. Also note that π_{K+1} is used only if at all the decision epochs the decision was to defer the prediction of Y and deploy another sensor. The decision rule π_{K+1} is a map from $\mathcal{X}_1 \times \mathcal{X}_2 \times \ldots \times \mathcal{X}_K$ to $\mathcal{Y}$. If the objective is to try to minimize the detection error, then it is well known that the optimal map is the Bayes classifier [12]

$$\pi^*_{K+1}(x_1, x_2, \ldots, x_K) = \arg\max_{y \in \mathcal{Y}} \Pr\{Y = y | X_1 = x_1, X_2 = x_2, \ldots, X_K = x_K\}.$$

The domain and range of the decision rules for stages $2, \ldots, K$ depend on the sequence of sensors deployed up to the decision time. For example, if $\pi_1 = k$, then

$$\pi_2 : \mathcal{X}_k \rightarrow (\{1, 2, \ldots, K\} \setminus k) \bigcup \mathcal{Y}.$$

If $\pi_2(x_k) \in (\{1, 2, \ldots, K\} \setminus k)$ then the decision is to take another observation using sensor $\pi_2(x_k)$. Alternatively, if $\pi_2(x_k) \in \mathcal{Y}$, then the decision is that the amount of information is sufficient and $\pi_2(x_k)$ is the predictor of Y. Instead of explicitly defining the policy through a sequence of mappings whose domains and ranges depend of past decisions and observations, we let $Z = [X_1, \ldots, X_K]$ and define the policy π as a two-dimensional function of Z. Given, the value of Z, its first argument $[\pi(Z)]_1$ is the resulting sequence of sensors that were deployed prior to the final decision and its second argument $[\pi(Z)]_2$ is the prediction for Y. Note that in general, only a subset of the elements of Z are observable at the time the final decision is made.

Denote by $P_c(\pi) = \Pr\{[\pi(Z)]_2 = Y\}$ the probability of correctly predicting the value of Y based on the data collected according to the policy π, by $C([\pi(Z)]_1)$ the cost associated with the sequence of sensor deployments $[\pi(Z)]_1$, e.g., the number of sensor dwells, and by $\mathrm{E}\{C([\pi(Z)]_1)\}$ the expected cost. We assume that the cost of the deployment of a sequence of sensors is the sum of the costs of deploying each of the sensors, and hence, does not depend on the order of deployment. The optimal policy π^* is the policy that maximizes

$$P_c(\pi) - \lambda \mathrm{E}\{C([\pi(Z)]_1)\}, \quad (1)$$

where λ is a tuning parameter that trades off the cost of data collection and the cost of prediction error. Under certain regularity conditions, the optimal policy can be defined though backward induction (see e.g. [19]). However, when $\mathcal{X}_1, \ldots, \mathcal{X}_K$ are continuous or discrete and large, the solution becomes intractable. Furthermore, even when $\mathcal{X}_1, \ldots, \mathcal{X}_K$ are finite and relatively small, the backward induction iterations require computing expectations with respect to the joint distribution of Z and Y.

In this paper we allow $\mathcal{X}_1, \ldots, \mathcal{X}_K$ to be continuous or discrete and large, and consider the case in which the joint distribution of Z and Y is unknown. We assume that n realizations of (Z, Y) are available and the goal is find a policy that maximizes (1) based on this data set. Hence, this is a model free instance of the sequential choice of experiments problem as formulated in [8], which, to the best of our knowledge, has not been considered previously in the literature.

III. Partially Observable Markov Decision Processes and Reinforcement Learning

The field of reinforcement learning is centered around the challenge of designing agents that learn to act in a stochastic environment by interacting with it [21]. As the agent interacts with the environment it receives rewards, and the goal is to eventually learn through these rewards which actions maximize the future sum of rewards. A common mathematical model for reinforcement learning is the problem of finding the optimal policy for controlling a finite-horizon partially observable Markov decision process (POMDP) [14]. The formulation of our sequential choice of experiments problem as finite-horizon POMDP consists of several elements:

- **The decision epochs** determine the times in which the agent is to take an action. In the discrete model adopted here, decision epochs occur at $t = 0, \ldots, \tau$. At every decision epoch either another observation is collected, or a final prediction of Y is made. In the later case the processes terminates. Therefore, τ is a random variable that depends on the deployed policy and Z.
- The system's state is the realization of Y which is fixed throughout the episode.
- The state at time zero is a random variable with distribution D over $\mathcal{Y}$.
- The state of the system cannot be directly observed but instead after every decision epoch $t = 0, \ldots, \tau$, in which the decision is to collect another observation, a noisy **observation** O_t of the systems' state is collected. The domain and distribution of the observation depends on the underlying systems' state Y and the deployed sensor. Denote by $\overline{O}_t = [O_0, O_1, \ldots, O_t]$ the observations up to and including time $t < \tau$, and note that $\overline{O}_t$ is a subset of Z.
- At every decision epoch $0 \leq t \leq \tau$ the agent chooses an **action** a_t, based on the past observations, from a set of possible actions called the **action space** $\mathcal{A}_t$. Though not explicitly appearing in the notation, the set of available actions $\mathcal{A}_t$ may depend on the past actions. In our application, only actions that correspond to sensors that have not be previously deployed can be taken.
- There exists a termination action which ends the process, such as the action of making the prediction of Y.
- We note that even though in our formulation the state of the system is fixed throughout the episode, the results can be generalized to the case in which upon taking action a at state y, the system makes a transition to state y' according to a **transition probability** $P_{y,a}$. In other wards, it is possible to generalize to the case in which the system's states evolve as a Markov process. This generalization is important for cases in which sensor deployment may be sensed by the target and lead to changes in the target's state as in [13].
- A **reward** $r(Y, a)$ is received after each time an action is taken. When a sensor is deployed to collect another observation, $r(Y, a)$ is minus the cost of deploying sensor a regardless of the state of the system. When the final prediction is made a reward of one unit is received only if the prediction $a = \widehat{Y}(\overline{O}_{\tau-1})$ equals Y, i.e., $r(Y, a) = I(a = Y)$, where I is the indicator function that equals one when its argument is true and zero otherwise.
- A **policy** π is a sequence of decision rules, or mappings from past observations to the action spaces, which specifies the action to take at each decision epoch. The policy is composed of $K + 1$ decision rules $(\pi_0, \pi_1, \ldots, \pi_K)$, however, if the termination action is taken prior to decision epoch K then not all decision rules are executed.

A typical episode is a sequence

$$a_0 \to O_0 \to a_1(O_0) \to O_1 \to a_2(\overline{O}_1) \ldots \\ O_{\tau-1} \to a_\tau(\overline{O}_{\tau-1}) = \widehat{Y}(\overline{O}_{\tau-1}),$$

where a_0 is the first decision to deploy a sensor before any observations were collected, $O_0, O_1, \ldots, O_{\tau-1}$ are the observations whose domains and distributions depend on Y and the decisions $a_0, a_1, \ldots, a_{\tau-1}$, respectively, and $a_\tau(\overline{O}_{\tau-1})$ is a decision that the past observations are sufficient for making a prediction on Y, and it specifies the predictor $\widehat{Y}(\overline{O}_{\tau-1})$. The objective is to find a policy π that maximizes the expected sum of rewards:

$$V(\pi) = \mathrm{E}_\pi \left\{ \sum_{t=0}^{\tau} r(Y, \pi_t(\overline{O}_{t-1})) \right\}, \tag{2}$$

where the expectation is taken with respect to the joint distribution of Z and Y, which, through π, induce a distribution on the observations $O_0, O_1, \ldots, O_{\tau-1}$. The

expected sum of rewards $V(\pi)$ is called the value of the policy π.

It is well known that when the underlying joint distribution of the system state and the observations is known and the observations can take a finite number of possible values, it is possible to formulate the problems in terms of the information state and solve for the optimal policy [17]. In our setting, however, the joint distribution is unknown and the observations are, in general, continuous random variables. Approximating the optimal policy in this case is a classic problem in reinforcement learning. Here, we adopt the generative model assumption of [14]. Under this assumption, the initial distribution D and the distribution of the observations conditioned on the system state and the deployed sensor are unknown but it is possible to generate realizations of the system state Y according to D and observations conditioned on arbitrary state Y and deployed sensor. In particular, we assume that we have n realizations of the pair (Z, Y) denoted by $\{(Z_1, Y_1), (Z_2, Y_2), \ldots, (Z_n, Y_n)\}$. Note that given a realization (Z_1, Y_1) it is possible to generate the entire decision tree associated with the sequential choice of experiment problem. An example of the decision tree in a problem in which there are two sensors $K = 2$ and $\mathcal{Y} = \{0, 1\}$ is given in Figure 1. Given a realization (Z_1, Y_1) and a policy π, it is possible to follow the path that a system that uses π will follow and compute the sum of rewards for this realization. Prior to the prediction of Y, the rewards are minus the sensor deployment costs, and, at the prediction epoch, a unit reward is received only if $\widehat{Y}(\overline{O}_{\tau-1}) = Y_1$, where $\widehat{Y}(\overline{O}_{\tau-1})$ is chosen by following the path induced by π.

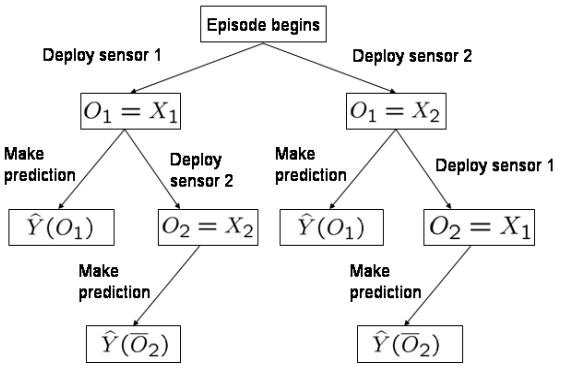

Fig. 1. A decision tree for a sequential choice of experiment problem with $K = 2$ and $\mathcal{Y} = \{0, 1\}$.

Now, consider a class of policies Π, i.e., each element $\pi \in \Pi$ is a sequence of decision rules $\pi = (\pi_0, \pi_1, \ldots, \pi_K)$. It is possible to estimate the value $V(\pi)$ (2) of any policy in the class from the set of trajectory trees by simply averaging the sum of rewards on each tree along the path that agrees with the policy [14]. A policy specifies the action to take at each decision epoch and so there is exactly one path in every tree that agrees with a given policy. Denote by $\widehat{V}^i(\pi)$ the observed sum of rewards on the i'th tree along the path that corresponds to the policy π. Then the value of the policy π is estimated by

$$\widehat{V}_n(\pi) = n^{-1} \sum_{i=1}^{n} \widehat{V}^i(\pi). \tag{3}$$

In [14], the authors show that with high probability (over the data set) $\widehat{V}_n(\pi)$ converges uniformly (over Π) to $V(\pi)$ with rates that depend on the VC-dimension of the policy class. This result motivates the use of policies π with high $\widehat{V}_n(\pi)$, since with high probability these policies have high values of $V(\pi)$.

In [7] it is shown that while the task of finding the global optimum within a class of non-stationary policies may be overwhelming, the componentwise search, i.e., optimizing a single decision rule at a time, leads to single step reinforcement learning problems which can be reduced to a sequence of multi-class weighted classification problems. Multi-class weighted classification problems can be solved using re-sampling methods or heuristic extensions of methods for binary weighted classification (see [2] for both approaches). Below, it is shown how to convert a multi-action RL problem into a binary RL problem by introducing dummy decision epochs. Then, applying the method in [7] leads to a sequence of binary weighted classification problems that can be directly solved using off-the-self classification methods.

IV. A Nonlinear Gauss Seidel Approach

Suppose an initial policy is given and one wishes to improve upon it by optimizing one of the decision rules at a time while holding the rest fixed. In [7] it is shown that this component-wise search is equivalent to simple tree pruning operations. In particular suppose π_k is updated while the decision rules $(\pi_0, \ldots, \pi_{k-1})$ and $(\pi_{k+1}, \ldots, \pi_K)$ are held fixed. Since $(\pi_0, \ldots, \pi_{k-1})$ are held fixed, the path taken by the policy up to and including epoch $k-1$ will not change when we update π_k. Hence, it is possible to prune the tree from the top down to epoch k by removing the branches that do not agree with the actions taken according to $(\pi_0, \ldots, \pi_{k-1})$. Since $(\pi_{k+1}, \ldots, \pi_K)$ are held fixed, the path taken by the policy after taking each of the possible actions at epoch k are known and will not change when we update π_k. Hence, it is possible to prune the tree from $k+1$ to

the leaves by removing the branches that do not agrees with the actions taken according to $(\pi_{k+1}, \ldots, \pi_K)$. Furthermore, since by the second pruning the path that will be followed after taking each of the actions at decision epoch k is known, it is possible to obtain realizations of the sum of future rewards that results in taking each of the actions at decision epoch k. In mathematical programming, a component-wise optimization of a nonlinear function is often referred to as nonlinear Gauss-Seidel algorithm [5]. It is in this sense that we call the above component-wise policy search a nonlinear Gauss-Seidel approach.

This procedure is illustrated for the simple decision tree of Figure 1 in Figure 2. Note that since after taking an action at decision epoch 1 the path of the tree is fixed regardless of the policy (see Figure 1), there is no tree pruning, only reward propagation according to the value of $\widehat{Y}(\overline{O}_2)$. Specifically, after the reward propagation, we can observe that taking action 'make prediction' results in an immediate and final reward $I(\widehat{Y}(\overline{O}_1) = Y)$ and taking action 'deploy sensor 2' lead to an immediate reward of minus the deployment cost associated with sensor 2 plus the subsequent reward $I(\widehat{Y}(\overline{O}_2) = Y)$. Since at epoch 1 the future sum of rewards is determined for every action, the task of updating policy π_1 is a single step RL problem. Below it is shown that this problem is equivalent to a certain supervised learning problem. Before the conversion to supervised learning, we convert the RL problem into a binary RL problem.

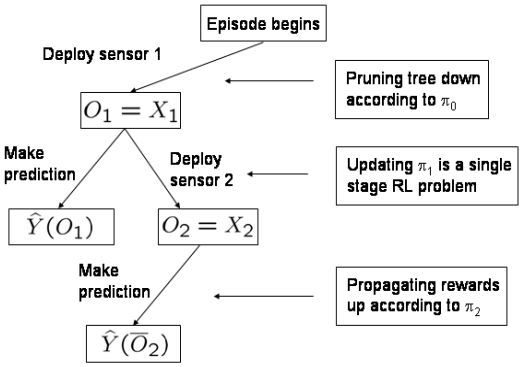

Fig. 2. Updating π_1 while holding π_0 and π_2 fixed.

V. From Multiple-Action Reinforcement Learning to Binary Reinforcement Learning

The nonlinear Gauss-Seidel approach of the previous section breaks the multi-stage search associated with the trajectory tree method into a sequence of single-stage RL subproblems. In [7] these single-stage RL subproblems were converted to multi-class weighted classification problems, which can then be solved using, e.g., re-sampling methods [2]. In this section it is shown that it is possible to convert a single-stage RL problem into multi-stage binary RL problem, apply the nonlinear Gauss-Seidel approach, and arrive at a sequence of binary single-stage RL subproblems.

Consider a single-stage RL problem with K possible actions. It is possible to describe any action as the answer to at most $\lceil \log_2(K) \rceil$ 'yes or no' questions, where $\lceil x \rceil$ is the smaller integer larger than or equal to x. Then, the single-stage RL problem is described by the decision tree associated with these binary decision epochs. Once an intermediate decision is made, it corresponds to a transition to the same state, i.e., the state does not evolve, but with a reduced (halved) action space. Only when the decision is between two actions, does the chosen action is executed and a state transition occurs. Figure 3 demonstrate converting a 4-action single-stage RL problem into a two-stage binary RL problem.

Finally, reapplying the nonlinear Gauss-Seidel algorithm of the previous section leads to a sequence of single-stage binary RL subproblems.

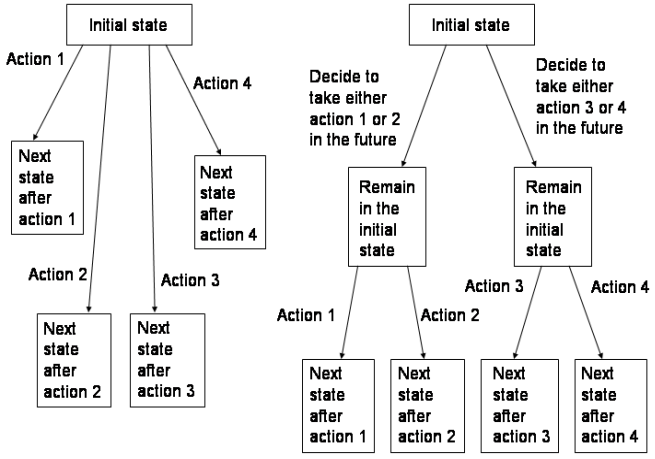

Fig. 3. Converting a 4 action single-stage RL problem into a two-stage binary RL problem.

VI. A Reduction From a Single Step Reinforcement Learning Problem to Weighted Classification

In this section we present the conversion of a single-step binary RL problem into a supervised learning problem, which is a special case of the classification reduction in [7]. The goal is to leverage techniques and theoretical results from supervised learning for solving the more complex problem of reinforcement learning [3]. To simplify to presentation we do not carry the heavy

notation of the previous section but rather introduce a simple generic notation to explain the conversion. Consider a single-step binary RL problem. An initial state $S_0 \in \mathcal{S}$ generated according to the distribution D is followed by one of 2 possible actions $A \in \{0,1\}$, which leads to a transition to state S_1 whose conditional distribution given that the initial state is s and the action is a is given by $P_{s,a}$. Given a class of policies Π, where a policy in Π is a map from $\mathcal{S}$ to $\mathcal{A}$, the goal is to find

$$\widehat{\pi} \in \arg\max_{\pi\in\Pi} \widehat{V}_n(\pi). \tag{4}$$

In this single stage problem the data are n realizations of the random element $\{S_0, S_1|0, S_1|1\}$, where $S_1|0$ (respectively $S_1|0$) is a realization of S_1 after taking action 0 (respectively 1) at state S_0. Denote the i'th realization by $\{s_0^i, s_1^i|0, s_1^i|1\}$. In this case, $\widehat{V}_n(\pi)$ can be written explicitly by

$$\widehat{V}_n(\pi) = \mathrm{E}_n \left\{ \sum_{l=0}^{1} r(S_0, l, S_1|l) I(\pi(S_0) = l) \right\}, \tag{5}$$

where $r(S_0, l, S_1|l)$ is the reward gained when taking action l at state S_0 and making a transition to state $S_1|l$, for a function f, $\mathrm{E}_n\{f(S_0, S_1|0, S_1|1)\}$ is its empirical expectation $n^{-1}\sum_{i=1}^n f(s_0^i, s_1^i|0, s_1^i|1)$, and $I(\cdot)$ is the indicator function taking a value of one when its argument is true and zero otherwise.

The following theorem shows that the problem of maximizing the empirical reward (5) is equivalent to a binary weighted classification problem.

Proposition 1: Given a class of policies Π and a set of n trajectory trees,

$$\arg\max_{\pi\in\Pi} \mathrm{E}_n \left\{ \sum_{l=0}^{1} r(S_1|l) I(\pi(S_0) = l) \right\} = \arg\min_{\pi\in\Pi}$$
$$E_n \left\{ |r(S_1|0) - r(S_1|1)| I(\pi(S_0) \neq \arg\max_k r(S_1|k)) \right\}$$

Proof: Take $L = 2$ in Proposition 1 in [7]. ■

The theorem implies that the maximizer of the empirical reward over a class of policies is the output of an optimal weights-dependent classifier for the data set:

$$\left\{ \left(s_0^i, \arg\max_{k\in\{0,1\}} r(s_1^i|k), |r(s_1^i|0) - r(s_1^i|1)| \right) \right\}_{i=1}^n,$$

where for each sample, the first argument is the example, the second is the label, and the third is a realization of the cost incurred when misclassifying the example. The implication is that a variety of supervised learning methods, such as k-nearest neighbors [9], neural networks [6], Boosting [10], and support vector machines [20], can be applied to solve the single-stage binary RL problem.

VII. Sensor Scheduling for Land-Mine Detection

This section reviews a sequential choice of experiment problem that arises in the design of unmanned land-mine detection vehicle. The vehicle carries three sensors for performing the detection: an EMI sensor, a ground penetrating radar (GPR), and an acoustic sensor. As can be seen in Figure 4, the sensors have different responses under different types of land-mines and clutter. In addition, deploying a sensor takes time and energy and hence not all sensors are deployed at every potential land-mine location. Upon reaching a new location, in which a land-mine is potentially present, a policy that trades of the cost of a sensor deployment and detection probability determines the first sensor to deploy. Based on the collected measurement, either a prediction regarding the presence of the land-mine is made or a second sensor is deployed. Finally, based on the output of the first two deployed sensors, either a prediction regarding the presence of the land-mine is made or a third sensor is deployed followed by the final prediction based on all three measurements. The goal is to maximize the probability of correct detection minus a constant $c > 0$ (1) times the number of sensor dwells.

Since there are a total of three sensors $Z = [X_1, X_2, X_3]$. The state space is binary $\mathcal{Y} = \{0,1\}$, where $Y = 0$ means no land-mine is present and $Y = 1$ indicates the presence of a land-mine. The decision tree associated with this problem is presented in Figure 5.

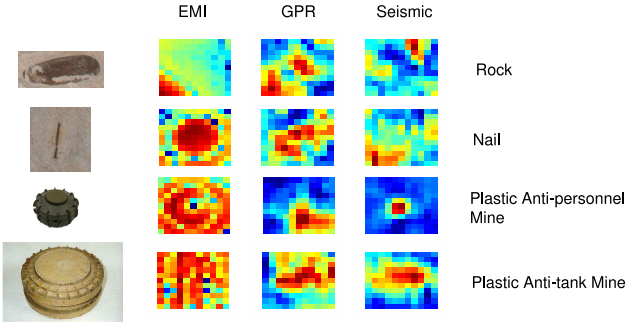

Fig. 4. Sensors signatures for several land-mine and clutter types.

Figure 7 summarizes the features extracted from each sensor and their expected signatures under different scenarios. In the simulation, one of the possible eight scenarios was first chosen randomly. Then, a realization of each of the features, which together compose Z, is generated as a Gaussian random variable with means 0, 0.5, or 1, corresponding to low, medium, or high, respectively. The covariances of sensors 1, 2, and 3, were $0.5I$, $0.45I$ and 0.1, respectively, where I is the

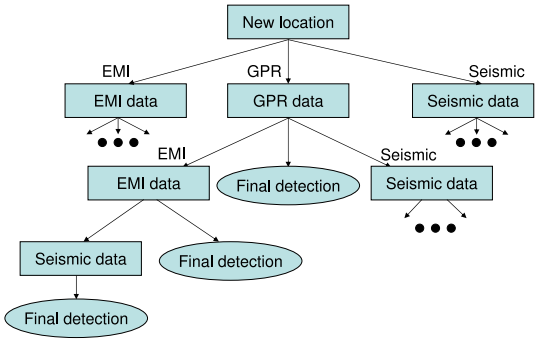

Fig. 5. The decision tree associated with the land-mine detection problem.

2-dimensional identity matrix. These values of means and covariances were chosen in correspondence with experiments that were conducted in a sand box [18]. Hence the marginal distribution of the vector of sensor outputs is a five-dimensional eight-component Gaussian mixture.

Before searching for the optimal sensor scheduling policy, the classifiers $\widehat{Y}(O_1), \widehat{Y}(\overline{O}_2), \widehat{Y}(\overline{O}_3)$ for all possible combinations of sensor selections

$$X_1, X_2, X_3,$$
$$(X_1, X_2), (X_1, X_3), (X_2, X_3),$$
$$(X_1, X_2, X_3)$$

were found by training two-layer feed-forward neural networks, each with ten input and two output nodes, on 1000 samples of (Z, Y). By testing the performance of these classifiers on a separate test set of 1000 samples, we found that the best single sensor to use for detecting a land-mine is the EMI sensor, that the two best fixed sensors are GPR plus the Seismic, and that in this scenario the classifier which is based on the output of all three sensors has a probability of correct detection of 0.887. The search for the optimal sensor scheduling policy was conducted while these classifiers remained fixed. In other words, only decisions regarding whether or not to deploy a sensor, and which sensor to deploy next were considered. Since the classifiers remained fixed during the policy search, once a decision to make prediction is made, the reward is gained according to the classifier output, without trying to further optimize its performance.

As explained above, the optimal policy was approximated by introducing dummy decision epochs, so that all the decisions are binary. We then performed the nonlinear Gauss-Seidel decomposition into a sequence of single-stage binary reinforcement learning problems. Each subproblem was then converted to a weighted classification problem that was solved by a weights-sensitive two-layer feed-forward neural network with seven input and two output nodes.

Figure 6 summarizes the results. The horizonal axis is the average number of sensor dwells and the vertical is the probability of correct detection. The three solid circles correspond to the performance of the best single sensor, best two sensors, and the performance when all three sensors are deployed, respectively. These points are connected by a solid line that corresponds to performance that can be achieved by randomly selecting one of these fixed sensor configurations. The crosses corresponds to the performance (estimated from a 1000 trail test set) obtained by the approximated optimal sensor scheduling policies. Each cross correspond to a different choice of c (1), ranging from $c = 0.2$ at the left lower corner and $c = 0$ at the outmost upper right cross. When $c = 0.2$ the price of taking more than a single measurement is too dear compared to the improvement in the probability of correct detection and the policy dictates making decision using only a single sensor. As c decreases, more and more observations are allowed. It is interesting to see that when c is zero, i.e, the sensor deployment cost is zero, the algorithm does not always deploy all three sensors, but achieves better performance than when all three sensors are always deployed. This happens since the classifiers used at the prediction stages are not the Bayes classifiers (in which more information can never worsen performance) but rather sub-optimal classifiers that were found by training neural networks. It is encouraging that by training the neural networks we found a policy that accounts for generalization errors at the predictor level and do not collect the third observation when that observation might lead to a worse prediction. In summary, it can be seen that through sensor scheduling it is possible to achieve better classification performance with fewer average number of sensor dwells. The actual sensor sequences taken under the possible eight scenarios when the policy whose performance cross is circled is presented in Figure 7. It is seen that the optimal policy dictates that the first deployed sensor is the GPR sensor even though the optimal single sensor is the EMI sensor. This is not surprising since an optimal sensor scheduling optimizes the future sum of rewards rather than choosing the sensor whose stand alone performance are the best. Furthermore, only when the underlying system state is a plastic anti-personal land-mine, which has the weakest signature, does the policy dictate using all three sensors. In other cases, two sensors are sufficient for the land-mine detection.

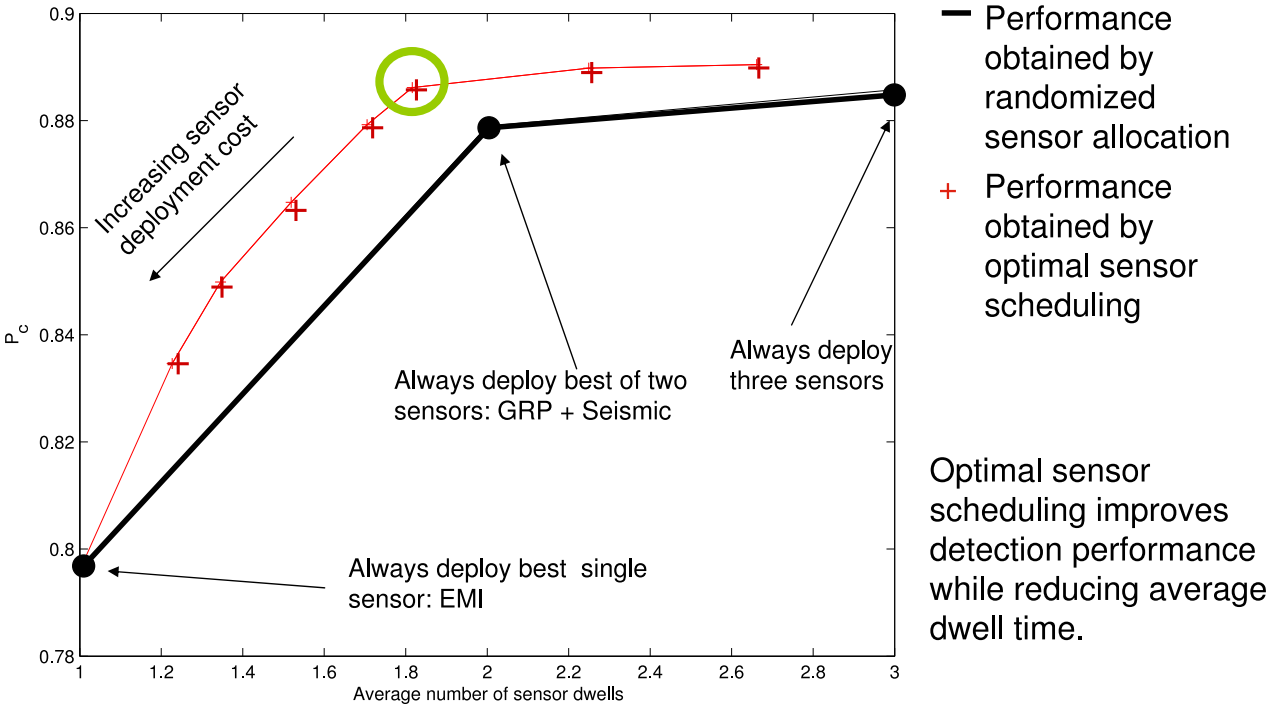

Fig. 6. Performance of sensor-scheduling-based detection compared to detection under optimal fixed sensor allocations.

		Object Type								
		1	2	3	4	5	6	7	8	Feature
		M-AT	M-AP	P-AT	P-AP	Cltr-1	Cltr-2	Cltr-3	Bkg	Description
Sensor	EMI (1)	High	High	Medium	High	High	Low	Low	Low	Conductivity
		High	High	High	Medium	Medium	Low	Low	Low	Size
	GPR (2)	High	Medium	High	Medium	Low	Low	Low	Low	Depth
		High	Medium	High	Medum	High	High	High	Low	RCS
	Seismic (3)	High	Medium	High	Medium	Medium	Medium	Low	Low	Resonance
Optimal sequence for mean state		2 3 D	2 1 D	2 3 D	2 1 3 D	2 3 D	2 3 D	2 3 D	2 3 D	

Fig. 7. Sensor mean responses under various scenarios. M-Metal, P-Plastic, AP-Anti personal, AT-Anti tank, Cltr-1-Hallow metal clutter, Cltr-2-Hallow non-metal clutter, Cltr-3-Non-metal non-hallow clutter, Bkg-Background.

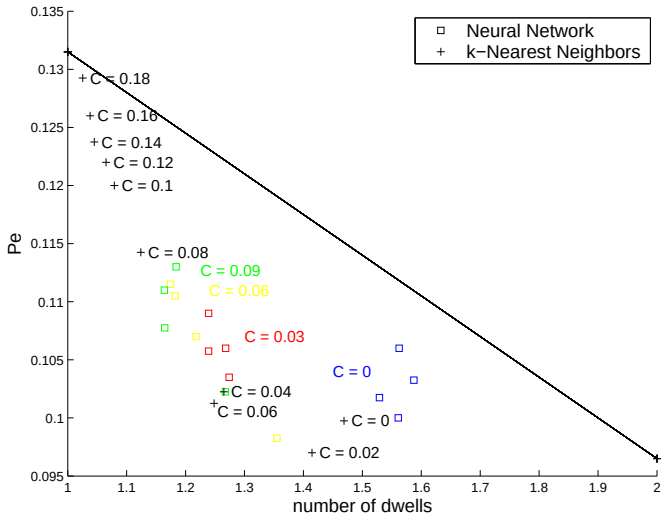

Fig. 8. Performance of sensor scheduling algorithm for the land monitoring satellite problem.

VIII. Waveform Selection for Land Monitoring Satellite

In this section, the optimal sensor scheduling algorithm is applied to real data for the problem of waveform selection for a LANDSAT land monitoring satellite. The satellite collects a radar backscatter on a patch of land and the goal is to classify the land type based on the returned signal. Given a new probing location, the satellite can transmit one of four possible waveforms. The different waveforms correspond to different frequency bands. Therefore, $Z = [X_1, X_2, X_3, X_4]$. Each of the observations $X_1, \ldots, X_4$ is a 9-dimensional vector taking values in $[0, 255]^9$, and hence, Z is a 36-dimensional vector. There are six land types, and hence $\mathcal{Y} = \{1, 2, \ldots, 6\}$. In the public data set [1], there are 4435 points in the training set and 2000 in the test set. For a more detailed explanation of the problem see [12] chapter 13. In this section we explore using sensor scheduling for reducing the number of waveform (frequency band) transmissions. In particular, we find policies that select the first best two frequency bands and based on the outcome determine if the remaining frequency bands are required, or whether the first two bands provide sufficient information for classifying the land type. Hence, at the first decision epoch there are six possible actions leading to siz possible measured pairs of frequency bands:

$$\{[X_1, X_2], [X_1, X_3], [X_1, X_4], \ldots$$
$$[X_2, X_3], [X_2, X_4], [X_3, X_4]\}.$$

The land type classifiers are the k-nearest neighbors algorithm with k set to 5, as recommended in [12] for the non-sequential problem. Two classifiers for the policy search were considered. The first is a $[7, 5, 2]$ feed-forward weights-sensitive neural network. The second is a weights-sensitive k-nearest neighbor, where $k = 30$. The performance are summarized in Figure 8. The crosses correspond to the performance of policies that were found by weights-sensitive k-nearest neighbor classifiers as c ranges from 0 to 0.18. The squares correspond to the performance of policies that were found by weights-sensitive $[7, 5, 2]$ feed-forward neural networks for four values of c. To study the effect of the initial network weights distribution, for each value of c, the neural networks training was initiated at four random weights selections, leading to four resulting policies. As can be seen, under both learning configurations it is possible to obtain a range of trade-offs between sensor deployment cost and classification performance. Particularly, the policy learned by the k-nearest neighbor classifier with $c = 0.02$ almost achieves the same performance as when all sensor modalities are used, but with a significant reduction in deployment cost. From comparing the performance of the k-nearest neighbor classifier based policy with the one based on the neural networks it is seen that the performance achieved by the two architectures are comparable.

IX. Conclusions

Sensor scheduling for controlling agile sensing systems was formulated as a sequential choice of experiments problem and solved via a reduction of the associated RL problem to a sequence of supervised learning problems. The method was applied to both real and synthetic data – land mine detection and LANDSAT terrain classification. Finally, the authors would like to thank Jay Marble and Raviv Raich for helpful discussions.

References

[1] The landsat data set. `http://www.niaad.liacc.up.pt/old/ statlog/ datasets/ satimage/ satimage.doc.html`.

[2] N. Abe, B. Zadrozny, and J. Langford. An iterative method for multi-class cost-sensitive learning. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 3–11, 2004.

[3] A. G. Barto and T. G. Dietterich. Reinforcement learning and its relationship to supervised learning. In J. Si, A. Barto, W. Powell, and D. Wunsch, editors, *Handbook of learning and approximate dynamic programming*. John Wiley and Sons, Inc, 2004.

[4] R. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, 1957.

[5] D. P. Bertsekas. *Nonlinear programming: second edition*. Athena Scientific, Belmont, MA, 1999.

[6] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford, Great Britain, 1995.

[7] D. Blatt and A. O. Hero. From weighted classification to policy search. In *18th Annual Conference on Neural Information Processing Systems (NIPS)*, 2005.

[8] M. H. DeGroot. *Optimal Statistical Decisions*. McGraw-Hill, New York, 1970.

[9] L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1996.

[10] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to Boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[11] P. K. Goel and J. Ginebra. When is one experiment always better than another. *The statistician*, 52(4):515–537, 2003.

[12] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning*. Springer-Verlag, New York, 2001.

[13] C. Kreucher K. Kastella and A. Hero. Sensor management using an active sensing approach. *Signal Processing*, 85(3):607–624, March 2005.

[14] M. Kearns, Y. Mansour, and A. Ng. Approximate planning in large POMDPs via reusable trajectories. In *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 2000.

[15] R. W. Keener. Local information and the design of sequential hypothesis tests. *Journal of Statistical Planning and Inference*, 130(1-2):111–125, 2005.

[16] V. Krishnamurthy. Algorithms for optimal scheduling and management of hidden markov model sensors. *IEEE Trans. Signal Process.*, 50(6):1382–1397, June 2002.

[17] M. Littman L. P. Kaelbling and A. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101, 1998.

[18] J. Marble, D. Blatt, and A. Hero. Confirmation sensor scheduling using a reinforcement learning approach. In *SPIE Defense and Security Symposium*, Orlando, Florida, April 2006. to appear.

[19] M. L. Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, Inc, 1994.

[20] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT, Press, 2002.

[21] R. S. Sutton and A. G. Barto. *Reinforcement Learning*. MIT Press, 1998.