Theoretical Analysis of SHIV

Michael Laskey[1], Jeffrey Mahler[1], Florian T. Pokorny[1], Anca D. Dragan[1] and Ken Goldberg[1,2]

I. Problem Statement

The goal is to learn a policy that matches that of the supervisor's while asking the supervisor for as few examples as possible. We are also interested in theoretically understanding how our query selection method effects the original DAgger convergence guarantees shown in Sec. IV.

Policies and State Densities. Following conventions from control theory, we denote by $\mathcal{X}$ the set consisting of observable states for a robot task, consisting, for example, of high-dimensional vectors corresponding to images from a camera, or robot joint angles and object poses in the environment. We furthermore consider a set $\mathcal{U}$ of allowable control inputs for the robot, which can be discrete or continuous. We model dynamics as Markovian, such that the probability of state $\mathbf{x_{t+1}} \in \mathcal{X}$ can be determined from the previous state $\mathbf{x}_t \in \mathcal{X}$ and control input $\mathbf{u}_t \in \mathcal{U}$:

$$p(\mathbf{x}_{t+1}|\mathbf{u}_t, \mathbf{x}_t, \ldots, \mathbf{u}_0, \mathbf{x}_0) = p(\mathbf{x}_{t+1}|\mathbf{u}_t, \mathbf{x}_t)$$

We assume a probability density over initial states $p(\mathbf{x}_0)$.

A trajectory $\hat{\tau}$ is a finite series of $T+1$ pairs of states visited and corresponding control inputs at these states, $\hat{\tau} = (\mathbf{x}_0, \mathbf{u}_0, ..., \mathbf{x}_T, \mathbf{u}_T)$, where $\mathbf{x}_t \in \mathcal{X}$ and $\mathbf{u}_t \in \mathcal{U}$ for $t \in \{0, \ldots, T\}$ and some $T \in \mathbb{N}$. For a given trajectory $\hat{\tau}$ as above, we denote by τ the corresponding trajectory in state space, $\tau = (\mathbf{x}_0, ..., \mathbf{x}_T)$.

A policy is a function $\pi : \mathcal{X} \to \mathcal{U}$ from states to control inputs. We consider a space of policies $\pi_\theta : \mathcal{X} \to \mathcal{U}$ parameterized by some $\theta \in \mathbb{R}^d$. Any such policy π_θ in an environment with probabilistic initial state density and Markovian dynamics induces a density on trajectories. Let $p(\mathbf{x}_t|\theta)$ denote the value of the density of states visited at time t if the robot follows the policy π_θ from time 0 to time $t-1$. Following [6], we can compute the average density on states for any timepoint by $p(\mathbf{x}|\theta) = \frac{1}{T}\sum_{t=1}^{T} p(\mathbf{x}_t|\theta)$.

While we do not assume knowledge of the distributions corresponding to: $p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$, $p(\mathbf{x}_0)$, $p(\mathbf{x}_t|\theta)$ or $p(\mathbf{x}|\theta)$, we assume that we have a stochastic real robot or a simulator such that for any state $\mathbf{x}_t$ and control $\mathbf{u}_t$, we can sample the $\mathbf{x}_{t+1}$ from the density $p(\mathbf{x}_{t+1}|\pi_\theta(\mathbf{x}_t), \mathbf{x}_t)$. Therefore, when 'rolling out' trajectories under a policy π_θ, we utilize the robot or a simulator to sample the resulting stochastic trajectories rather than estimating $p(\mathbf{x}|\theta)$ itself.

Objective. The objective of policy learning is to find a policy that minimizes some known cost function $C(\hat{\tau}) = \sum_{t=1}^{T} c(\mathbf{x}_t, \mathbf{u}_t)$ of trajectory $\hat{\tau}$. The cost $c : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$ is typically user defined and task specific. For example, in the task of inserting a peg into a hole, the distance between the peg's current and desired final state is often used [3].

In our problem, we do not have access to the cost function itself. Instead, we only have access to a supervisor that can achieve a desired level of performance on the task. The supervisor provides the robot an initial set of N stochastic demonstration trajectories $\{\tilde{\tau}^1, ..., \tilde{\tau}^N\}$. which are the result of the supervisor applying this policy. This induces a training data set $\mathcal{D}$ of all state-control input pairs from the demonstrated trajectories.

We define a 'surrogate' loss function as in [6], $l : \mathcal{U} \times \mathcal{U} \to \mathbb{R}$, which provides a distance measure between any pair of control values. In the continuous case, we consider $l(\mathbf{u}_0, \mathbf{u}_1) = ||\mathbf{u}_0 - \mathbf{u}_1||_2^2$, while in the discrete case $l(\mathbf{u}_0, \mathbf{u}_1) = 1$ if $\mathbf{u}_0 \neq \mathbf{u}_1$ and $l(\mathbf{u}_0, \mathbf{u}_1) = 0$ otherwise.

Given a candidate policy π_θ, we then use the surrogate loss function to approximately measure how 'close' the policy's returned control input $\pi_\theta(\mathbf{x}) \in \mathcal{U}$ at a given state $\mathbf{x} \in \mathcal{X}$ is to the supervisor's policy's control output $\tilde{\pi}(\mathbf{x}) \in \mathcal{U}$. The goal is to produce a policy that minimizes the surrogate loss relative to the supervisor's policy.

Following [6], our objective is to find a policy π_θ minimizing the expected surrogate loss, where the expectation is taken over the distribution of states induced by the policy across any time point in the horizon:

$$\min_\theta E_{p(\mathbf{x}|\theta)}[l(\pi_\theta(\mathbf{x}), \tilde{\pi}(\mathbf{x}))] \tag{1}$$

If the robot could learn the policy perfectly, this state density would match the one encountered in user examples. But if the robot makes an error, that error changes the distribution of states that the robot will visit, which can lead to states that are far away from any examples and difficult to generalize to [5]. This motivates iterative algorithms like DAgger, which iterate between learning a policy and the supervisor providing feedback. The feedback is in the form of control signals on states sampled from the robot's new distribution of states.

II. Risky and Safe States

Providing correct control inputs for (or "labeling") all states encountered at each iteration can impose a large burden on the supervisor. Instead of asking the supervisor for labels at all visited states, SHIV uses a measure of risk to actively decide whether a label is necessary.

In contrast to the standard measure risk based purely on variance, we define a state as "risky" if: 1) it lies in an area with a low density of previously trained states, which can

[1] Department of Electrical Engineering and Computer Sciences; {mdlaskey,iamwesleyhsieh,ftpokorny,anca}@berkeley.edu, staszass@rose-hulman.edu

[2] Department of Industrial Engineering and Operations Research; goldberg@berkeley.edu

[1−2] University of California, Berkeley; Berkeley, CA 94720, USA

cause the current policy to mis-predict the supervisor and incur high surrogate loss [9], or 2) the surrogate loss, or training error, of the current policy at the state is high, so that the policy does not model the supervisor's control inputs correctly. States that are not classified as "risky" are deemed "safe" .

The amount of data needed to estimate the density, scales exponentially in the dimension of the state space [4]. Thus, to evaluate risk in high-dimensional state spaces we use a modified version of the technique known as the One Class SVM that estimates a regularized boundary of a user defined quantile on the training data in $\mathcal{X}$ [7].

We consider the problem of estimating the quantile level-sets of a distribution P on a set $\mathcal{X}$ by means of a finite set of independent and identically distributed samples $\mathbf{x}_1, ..., \mathbf{x}_n \in \mathcal{X}$. In most general terms, the quantile function for P and subject to a class of measurable subsets $\mathcal{G}$ of $\mathcal{X}$ is defined by

$$U(\gamma) = \inf\{\lambda(G) : P(G) \geq \gamma, G \in \mathcal{G}\}\ 0 < \gamma \leq 1 \qquad (2)$$

$\lambda : \mathcal{G} \rightarrow \mathbb{R}$ above denotes a volume measure. Suppose furthermore that $G : [0,1] \rightarrow \mathcal{G}$ assigns a set $G(\gamma) \in \mathcal{G}$ that attains the infinum measure (i.e. volume) for each $\gamma \in [0,1]$ (this set is in general not necessarily unique). $G(\gamma)$ denotes a set of minimum measure $G \in \mathcal{G}$ with $P(G(\gamma)) \geq \gamma$.

To handle distributions defined on high-dimensional spaces $\mathcal{X}$, work by Scholköpf et al. represents the class $\mathcal{G}$ via a kernel k as the set of half-spaces in the support vector (SV) feature space [7]. By minimizing a support vector regularizer controlling the smoothness of the estimated level set function this work derives an approximation of the quantile function described in Eq. 2.

Let $\Phi : \mathcal{X} \rightarrow \mathcal{F}$ denote the feature map corresponding to our exponential kernel, $k(\mathbf{x}_0, \mathbf{x}_1) = e^{-||\mathbf{x}_0 - \mathbf{x}_1||^2/2\sigma^2}$, mapping the observation space $\mathcal{X}$ into a Hilbert space $(\mathcal{F}, \langle , \rangle)$ such that $k(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle$.

The One Class SVM proposed by [7] determines a hyperplane in feature space $\mathcal{F}$ maximally separating the input data from the origin:

$$\underset{w \in \mathcal{F}, \xi \in \mathbb{R}, \rho \in \mathbb{R}}{\text{minimize}} \frac{1}{2}||w||^2 + \frac{1}{vn}\sum_{i}^{n} \xi_i - \rho \qquad (3)$$
$$\text{s.t } \langle w, \Phi(x_i) \rangle \geq \rho - \xi_i,\ \xi_i \geq 0.$$

Here, the parameter ν controls the penalty or 'slack term' and $1 - \nu$ is equivalent to γ [10] in the quantile definition, Eq. 2, as the number of samples increases. The decision function, determining point membership in the approximate quantile levelset is given by $g(\mathbf{x}) = \text{sgn}(\langle w, \Phi(x) \rangle - \rho)$. Here, for $x \in \mathcal{X}$, $g(x) = 0$ if x lies on the quantile levelset, $g(x) = 1$ if x is strictly in the interior of the quantile super-levelset and $g(x) = -1$ if x lies strictly in the quantile sub-levelset.

The dual form of the optimization yields a Quadratic Program that has worst case computational complexity of $O(n^3)$. However, Schölkopf et al. developed an improved optimization method that has empirically been shown to scale quadratically [7],which we use. In the dual, the decision function is given by $g(\mathbf{x}) = \text{sgn}(\sum_{i=1}^{N} \alpha_i k(\mathbf{x}_i, \mathbf{x}) - \rho)$ where α_i corresponds to the dual variables. However, even when sufficient data is available, the associated control inputs may be inconsistent or noisy and a resulting policy optimizing Eq. 6 may still incur a large surrogate loss. To account for this, we propose a modification to the One Class SVM:

$$y_i = \begin{cases} 1 & : l(\pi_\theta(\mathbf{x}_i), \mathbf{u}_i) \leq \varepsilon \\ -1 & : l(\pi_\theta(\mathbf{x}_i), \mathbf{u}_i) > \varepsilon \end{cases} \qquad (4)$$

Where, in the case when l denotes discrete $0-1$ loss, we set $\varepsilon = 0$, while in the continuous L_2 loss case, ε is a user defined threshold specifying allowable surrogate loss. We use y_i to modify the One Class SVM decision function as follows:

We divide up our data in to two sets those correctly classified: $\mathcal{D}_s = \{\{\mathbf{x}_i, \mathbf{u}_i\} \in \mathcal{D}_k, y_i = 1\}$ and those states incorrectly classified: $\mathcal{D}_r = \{\{\mathbf{x}_i, \mathbf{u}_i\} \in \mathcal{D}_k, y_i = -1\}$ A separate One-Class SVM is then trained on each set of states, (D_s and D_r) and providing measures of the level sets, g_s and g_r. Specified by parameters (ν, σ) and (ν_r, σ_r), respectively.

We then define the overall decision function as:

$$g_\sigma(\mathbf{x}) = \begin{cases} 0 & : g_s(\mathbf{x}) == 1 \text{ and } g_r(\mathbf{x}) == -1 \\ -1 & : \text{otherwise} \end{cases} \qquad (5)$$

points are deemed risky if $g_\sigma(\mathbf{x}) \neq 0$. Practically, this modification corresponds to 'carving out holes' in the estimated quantile super-levelset such that neighborhoods around states with $y_i = -1$ are excluded from the super-levelset.

The decision function parametrization consists of the kernel bandwidth σ in g_s. We treat σ as a "risk sensitivity" parameter (and study its implications in Section **??**). For two reasons: 1)The expected number of examples, after a policy roll out, the supervisor can be asked is $T * \int_{\mathbf{x}} \mathbf{1}(g_\sigma(\mathbf{x}) == 0) p(\mathbf{x}|\theta) d\mathbf{x}$. Thus, smaller σ corresponds to asking for more examples. 2) A relation exists between how smooth the supervisor's policy, $\tilde{\pi}$ and how many examples are needed to learn it. Thus, a large σ can be dangerous for policies with sharp variation because it will treat points as safe that are really risky.

III. SHIV:Svm-based reduction in Human InterVention

Both SHIV and DAgger [6] solve the minimization in Eq. 1 by iterating two steps: 1) compute a θ using the training data $\mathcal{D}$ thus far, and 2) execute the policy induced by the current θ, and ask for labels for the encountered states. However, instead of querying the supervisor for every new state, SHIV actively decides whether the state is risky enough to warrant a query after the policy roll out.

A. Step 1

The first step of each iteration k computes θ_k that minimizes surrogate loss on the current dataset $\mathcal{D}_k = \{(x_i, u_i) | i \in \{1, \ldots, M\}\}$ of demonstrated state-control pairs (initially just the set $\mathcal{D}$ of initial trajectory demonstrations):

$$\theta_k = \arg\min_{\theta} \sum_{i=1}^{M} l(\pi_\theta(\mathbf{x}_i), \mathbf{u}_i). \qquad (6)$$

This sub-problem is a supervised learning problem, solvable by estimators like a support vector machine or a neural net. Performance can vary though with the selection of the estimator [8]

B. Step 2

The second step SHIV and DAgger rolls out their policies, π_{θ_k}, to sample states that are likely under $p(\mathbf{x}|\theta_k)$.

What happens next, however, differs between SHIV and DAgger. For every state visited, DAgger requests the supervisor to provide the appropriate control/label. Formally, for a given sampled trajectory $\hat{\tau} = (\mathbf{x}_0, \mathbf{u}_0, ..., \mathbf{x}_T, \mathbf{u}_T)$, the supervisor provides labels $\tilde{\mathbf{u}}_t$, where $\tilde{\mathbf{u}}_t \sim \tilde{\pi}(\mathbf{x}_t) + \epsilon$, where ϵ is a small zero mean noise term, for $t \in \{0, \dots, T\}$. The states and labeled controls are then aggregated into the next data set of demonstrations $\mathcal{D}_{k+1}$:

$$D_{k+1} = \mathcal{D}_k \cup \{(\mathbf{x}_t, \tilde{\mathbf{u}}_t) \| t \in \{0, \dots, T\}\}$$

SHIV only asks for supervision on states for which are risky, or $g_\sigma(\mathbf{x}) \neq 0$:

$$D_{k+1} = \mathcal{D}_k \cup \{(\mathbf{x}_t, \tilde{\mathbf{u}}_t) \| t \in \{0, \dots, T\}, g(\mathbf{x}_t) = -1\}$$

Steps 1 and 2 are repeated for K iterations or until the robot has achieved sufficient performance on the task[1].

IV. Theoretical Analysis

We recap the analysis of DAgger and state the main result and then introduce SHIV's analysis. Note, we denote a change in notation $l_i(\mathbf{x}) = l(\pi_{\theta_i}(\mathbf{x}), \tilde{\pi}(\mathbf{x}))$ where $l_i(\mathbf{u}) : \mathcal{U} \to \mathbb{R}$. The change is for convenience, since the only time l is used in the analysis is with respect to the supervisor.

DAgger Analysis Analysis of DAgger models the problem as online learning, where an algorithm must provide a policy π_n at iteration n which incurs loss $E_{p(\mathbf{x}|\theta_n)}[l_n(\pi_n)]$. After observing this loss the algorithm can provide a different policy π_{n+1} for the next iteration which will incur loss $E_{p(\mathbf{x}|\theta_{n+1})}[l_{n+1}(\pi_{n+1})]$. The loss functions $E_{p(\mathbf{x}|\theta_{n+1})}[l_{n+1}(\pi_{n+1})]$ may vary in an unknown fashion over time [6].

DAgger assumes a no-regret algorithm, i.e. an algorithm that produces a series of policies $\pi_1, \pi_2, ..., \pi_N$ such that the average regret with respect to the best policy in hindsight goes to 0 as N goes to ∞.

$$\frac{1}{N}\sum_{i=1}^{N} E_{p(\mathbf{x}|\theta_i)}[l_i(\pi_\theta(\mathbf{x}))] - \min_{\pi \in \Pi} \frac{1}{N}\sum_{i=1}^{N} E_{p(\mathbf{x}|\theta_i)}[l(\pi(\mathbf{x}))] \leq \gamma_N$$

for $\lim_{N \to \infty} \gamma_N = 0$. Many no-regret algorithms guarantee that γ_N is $\tilde{O}(\frac{1}{N})$ (e.g. when l is strongly convex).

Assume $l(\mathbf{u})$ is strongly convex and bounded over Π. Let $\epsilon_N = \min_{\pi \in \Pi} \frac{1}{N} \sum_{i=1}^{N} E_{p(\mathbf{x}|\theta_i)}[l(\pi(\mathbf{x}))]$ be the true loss of the best policy in hindsight. Then the following holds in the infinite sample case (infinite number of trajectories at each iteration):

Theorem 4.1: For DAgger if N is $\tilde{O}(T)$ there exists a policy $\hat{\pi} \in \hat{\pi}_{1:N}$ such that $E_{p(\mathbf{x}|\theta_N)}[l_N(\mathbf{x})] \leq \epsilon_N + O(1/T)$

Note, a bound on the total expected surrogate loss of a policy during roll out can be computed by multiplying the right hand side by a factor of T.

SHIV Analysis In SHIV, a state is queried or not queried via a decision function g_σ the boundaries of this decision function is parameterized by ν and σ. We are interested in bounding the worst case effect this has on the original DAgger analysis or Theorem 4.1.

For SHIV's analysis we make the additional assumptions of an L_0-Lipschitz supervisor policy, $\tilde{\pi}$ and a class of L_1-Lipschitz learned policy π_θ. The assumption of an L_0-Lipschitz supervisor's policy can be justified by noting that a lot of robotic systems are tele-operated and have bounded change in controls. For example, in the surgical robot setting, a surgeon usually makes small and low velocity motions when tele-operating during surgery [2].

The assumption of an L_1-Lipschitz learned policy is common for any learned function that is differentiable everywhere and has derivative of bounded magnitude, common techniques like linear or kernelized regression are capable of producing such a function [8].

We also assume a Radial Basis Function (RBF) kernel, $k(x_0, x_1) = \exp(-||x_0 - x_1||^2/\sigma^2)$, is used in the decision function $g_\sigma(\mathbf{x})$. Our experiments on several different domains suggest that an RBF kernel works well in practice.

In order to prove the Theorem 4.4, we need the following lemmas.

Lemma 4.2: Given an L_0-Lipschitz supervisor policy $\tilde{\pi}$ and L_1-Lipschitz learned policy π_θ, define $L = \max(L_0, L_1)$. With maximum regression error on the dataset η_N, where $\eta_N = \max_{\mathbf{x} \in \mathcal{D}_N} ||\pi_\theta(\mathbf{x}_i) - \tilde{\pi}(\mathbf{x})||_2^2$. Define $L_w : \mathcal{X} \to [0, \infty)$ as $L_w(\mathbf{x}) = \min_{\mathbf{x}_i \in \mathcal{D}} ||\mathbf{x}_i - \mathbf{x}||_2^2$, which is the shortest Euclidean distance a point is to those in the dataset. The surrogate loss function $l(\pi_\theta(\mathbf{x}), \tilde{\pi}(\mathbf{x})) = ||\pi_\theta(\mathbf{x}) - \tilde{\pi}(\mathbf{x})||_2^2$ is bounded as follows:

$$l(\pi_\theta(\mathbf{x}), \tilde{\pi}(\mathbf{x})) \leq (2L)L_w(\mathbf{x}) + \eta_N$$

Proof: Given a point $\mathbf{x}$, define $\mathbf{x}_h = \text{argmin}_{\mathbf{x}_i \in \mathcal{D}} ||\mathbf{x} - x_i||$ or the point closest to it in the dataset.

$$\begin{aligned}
&||\pi_\theta(\mathbf{x}) - \tilde{\pi}(\mathbf{x})||_2^2 \\
&\leq ||\pi_\theta(\mathbf{x}) - \pi_\theta(\mathbf{x}_h) + \tilde{\pi}(\mathbf{x}) - \tilde{\pi}(\mathbf{x}_h) + \max_{\mathbf{x} \in \mathcal{D}_N}(\pi_\theta(\mathbf{x}_i) - \tilde{\pi}(\mathbf{x}))||_2^2 \\
&\leq ||\pi_\theta(\mathbf{x}) - \pi_\theta(\mathbf{x}_h)||_2^2 + \eta_N + ||\tilde{\pi}(\mathbf{x}) - \tilde{\pi}(\mathbf{x}_h)||_2^2 \\
&\leq (2L)L_w(\mathbf{x}) + \eta_N
\end{aligned}$$

Line two is the sum of how much the learned policy differs from the training point, how much the supervisor policy differs from the training point and the maximum training error at the current iteration. Line three is a result of the triangle

[1] In the original DAgger the policy rolled out was stochastically mixed with the supervisor, thus with probability β it would either take the supervisor's action or the robots. The use of this stochastically mix policy was for theoretical analysis. In practice, it is recommended to set $\beta = 0$ to avoid biasing the sampling [1], [6]

inequality and then applying the definition of η_N. Line four is applying the definition of Lipschitz continuity and the definition of L.

■

Lemma 4.3: Given a decision function $g_\sigma(\mathbf{x})$, parameterized by ν and σ and an RBF kernel. Define the constant F as the maximum squared Euclidean distance of a point inside the decision boundary from the states g_σ was trained on:

$$F = \max_{\mathbf{x}\in\mathcal{X}}\min_{\mathbf{x}_i\in\mathcal{D}} ||\mathbf{x}_i - \mathbf{x}||_2^2 \\ \text{s.t. } g_\sigma(\mathbf{x}) = 0.$$

We can bound F by the following:

$$F \leq \sigma^2 \log(\frac{1}{\rho\nu})$$

Proof:

We will prove Lemma 4.3 by increasing the set size of $g_\sigma(\mathbf{x})$ by a series of upper bounds and then bounding the increase set in terms of the hyper parameters ν and σ.

$$F = \max_{\mathbf{x}\in\mathcal{X}}\min_{\mathbf{x}_i\in\mathcal{D}} ||x_i - x||_2^2 \\ \text{s.t. } g_\sigma(\mathbf{x}) = 0. \tag{7}$$

Define the solution to the problem as $\mathbf{x}^* \in \mathcal{X}$ and $\mathbf{x}_m \in \mathcal{D}$ (i.e.)

$$\big(\mathbf{x}^*, \mathbf{x}_m\big) = \big(\underset{\mathbf{x}\in\mathcal{X}}{\text{argmax}}, \underset{\mathbf{x}_i\in\mathcal{D}}{\text{argmin}} ||\mathbf{x}_i - \mathbf{x}||_2^2\big) \\ \text{s.t. } g_\sigma(\mathbf{x}) = 0. \tag{8}$$

We restate the definition of $g_\sigma(\mathbf{x}) = \text{sgn}(\sum_i^n \alpha_i k(\mathbf{x}, \mathbf{x}_i) - \rho)$. We can increase the size of the set by noting that $0 \leq \alpha_i \leq \frac{1}{\nu n}$, which comes from the KKT conditions of the original optimization [7]. Thus our increased set's decision boundary becomes $\text{sgn}(\frac{1}{\nu n}\sum_i^n k(\mathbf{x}, \mathbf{x}_i) - \rho)$.

$$F \leq \max_{\mathbf{x}\in\mathcal{X}}\min_{\mathbf{x}_i\in\mathcal{D}} ||\mathbf{x}_i - \mathbf{x}||_2^2 \\ \text{s.t. } \frac{1}{\nu n}\sum_i^n k(\mathbf{x}, \mathbf{x}_i) - \rho \geq 0.$$

To further increase the size of the set around the point $\mathbf{x}_m$, we change the boundary to $\text{sgn}(\frac{1}{v}k(\mathbf{x}, \mathbf{x}_m) - \rho)$.

$$F \leq \max_{\mathbf{x}\in\mathcal{X}} ||\mathbf{x}_m - \mathbf{x}||_2^2 \\ \text{s.t. } \frac{1}{\nu}k(\mathbf{x}, \mathbf{x}_m) - \rho \geq 0.$$

Now we can determine the distance to the boundary of this larger set by setting

$$\frac{1}{\nu}k(\mathbf{x}, \mathbf{x}_m) = \rho \\ \exp(-||\mathbf{x} - \mathbf{x}_m||^2/\sigma^2) = \nu\rho \\ ||\mathbf{x} - \mathbf{x}_m|| = \sigma^2 \log(\frac{1}{\nu\rho})$$

This equality in the large set can then be used to upper-bound the original term $F \leq \sigma^2 \log(\frac{1}{\nu\rho})$.

■

Theorem 4.4: For SHIV if N is $O(T)$ there exists a policy $\hat{\pi} \in \hat{\pi}_{1:N}$ such that $E_{p(\mathbf{x}|\theta_N)}[l_N(\pi_\theta(\mathbf{x}))] \leq \epsilon_N + \eta_N + O(1/T) + (2L)\log(\frac{1}{\nu\rho})\sigma^2$

Proof:

$$\min_{\pi\in\pi_{1:N}} E_{p(\mathbf{x}|\theta_i)}[l_i(\pi_\theta(\mathbf{x}))] \\ \leq \frac{1}{N}\sum_{i=1}^N E_{p(\mathbf{x}|\theta_i)}(l_i(\mathbf{x})) \\ = \frac{1}{N}\sum_{i=1}^N \int_{g_\sigma(\mathbf{x})=0} l_i(\pi_\theta(\mathbf{x}))p(\mathbf{x}|\theta_i) + \int_{g_\sigma(\mathbf{x})=-1} l_i(\pi_\theta(\mathbf{x}))p(\mathbf{x}|\theta_i)d\mathbf{x} \\ \leq \frac{1}{N}\sum_{i=1}^N \int_{g_\sigma(\mathbf{x})=0} l_i(\pi_\theta(\mathbf{x}))p(\mathbf{x}|\theta_i)d\mathbf{x} + \frac{1}{N}\sum_{i=1}^N \int l_i(\pi_\theta(\mathbf{x}))p(\mathbf{x}|\theta_i)d\mathbf{x} \\ \leq \frac{1}{N}\sum_{i=1}^N \int_{g_\sigma(\mathbf{x})=0} l_i(\pi_\theta(\mathbf{x}))p(\mathbf{x}|\theta_i)d\mathbf{x} + \epsilon_N + O(1/T)$$

Line three in the proof separates the integral over states inside the decision function g_σ and outside the decision function. Line four bounds the expectation over the states outside the decision boundary by replacing the integral back to original density for the second term.

The last line bounds the second term by the original results of DAgger because these are on the distributions of states outside of the decision function g_σ, where SHIV is identical to DAgger. We note the no regret assumption holds for any unknown $p(\mathbf{x}|\theta)$, even ones chosen adversarially thus the fact θ_i is updated with states potentially different than DAgger will not change the result [6].

Using Lemma 4.2, we can bound the surrogate loss function as follows:

$$\leq \frac{1}{N}\sum_{i=1}^N \int_{g_\sigma(\mathbf{x})=0} (2L)L_w(\mathbf{x})p(\mathbf{x}|\theta_i)d_x + \eta_N + ...$$

Next by noting that the states evaluated under the worst case loss are within the decision boundary g_σ, we can apply Lemma 4.3.

$$\leq (2L)F + \eta_N + ... \\ \leq (2L)\log(\frac{1}{\rho\nu})\sigma^2 + \eta_N + \epsilon_N + O(1/T)$$

■

V. Discussion of Analysis

Theorem 4.4 shows the linear convergence guarantee of DAgger also holds for SHIV, however the hyperparameters σ and ν can lead to a higher surrogate loss after convergence. If the supervisor's policy has a high Lipschitz constant, L, then the risk sensitive parameter, σ needs to be smaller proportionally to achieve less surrogate loss as shown in Theorem 4.4.

We also recall the expected number of queries to a supervisor can be determined by $T \int_{\mathcal{X}} \mathbf{1}(g_\sigma(\mathbf{x}) = 0) p(\mathbf{x}|\theta) d\mathbf{x}$. Here we see that σ directly effects the size of the decision boundary, which in turn controls the expected number of queries. A trade off exists then between supervisor burden and the performance at convergence, but experimentally we show on several different examples one can achieve the same performance as DAgger in significantly less queries (up to 70%).

REFERENCES

[1] X. Guo, S. Singh, H. Lee, R. L. Lewis, and X. Wang, "Deep learning for real-time atari game play using offline monte-carlo tree search planning," in *NIPS*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 3338–3346.

[2] Intuitive Surgical, "Annual report 2014," 2014.

[3] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," 2015.

[4] H. Liu, J. D. Lafferty, and L. A. Wasserman, "Sparse nonparametric density estimation in high dimensions using the rodeo," in *International Conference on Artificial Intelligence and Statistics*, 2007, pp. 283–290.

[5] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," DTIC Document, Tech. Rep., 1989.

[6] S. Ross, G. J. Gordon, and J. A. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," 2010.

[7] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[8] B. Schölkopf and A. J. Smola, *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[9] S. T. Tokdar and R. E. Kass, "Importance sampling: a review," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 2, no. 1, pp. 54–60, 2010.

[10] R. Vert and J.-P. Vert, "Consistency and convergence rates of one-class svms and related algorithms," *The Journal of Machine Learning Research*, vol. 7, pp. 817–854, 2006.