Brown-UMBC Robot Learning Update

David Abel

May 2017

Michael and I have focused on three related topics recently.

1. **Compression and Sequential Decision Making**
   - *Big Picture*: What are the limits of compression in the context of sequential decision making?
   - *Current Technical Focus*: For a fixed but arbitrary distribution over MDPs, D, what is the fewest number of bits that can represent behavior that achieves a value loss of at most ε with probability $1 - \delta$?
2. **Properties of Good Abstractions**
   - *Big Picture*: What properties underly good abstractions for sequential decision making?
   - *Current Technical Focus*: Planning Complexity and Value Loss Bounds for state and action abstractions. Main goal is to understand what constitutes a good set of options.
3. **Computing & Learning Near Optimal Abstractions**
   - *Big Picture*: How can we compute near optimal abstractions?
   - *Current Technical Focus*: For a fixed but arbitrary distribution over MDPs, D, how can we compute or learn abstractions that possess the right value loss and planning complexity bounds?

I'll go into slightly more detail for each technical focus.

1 Compression and Sequential Decision Making

A hallmark result of information theory is the fundamental limit of compression. This limit inspired many formalisms and algorithms for communication and coding; our goal here is to arrive a similar result that (we hope) sheds light on algorithms for optimal abstraction in the context of sequential decision making.

1.1 Entropy

First, recall the definition of entropy:

Definition 1 (Entropy): *For any discrete random variable, X, taking on values in the alphabet, $\mathcal{X}$, with pmf $\Pr_{X \sim \mathcal{X}})(X = x) = p(x)$, the* **entropy** *of X is given by:*

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) \log_2 p(x) \tag{1}$$

Entropy is the ultimate governing quantity in information theory. It measures the degree of uncertainty in a random variable.

1.2 ...and Sequential Decision Making

We now introduce the multi task setting for RL and planning that we use across many of the introduced topics:

Definition 2 (Multi-Task Sequential Decision Making): *An agent is given a partial MDP: $\langle \mathcal{S}, \mathcal{A}, \gamma, s_{init} \in \mathcal{S} \rangle$, and an acting time, τ, and a fixed but arbitrary distribution over reward and transition function pairs, D. The following process repeats indefinitely:*

1. *The agent samples $\mathcal{R}_i, \mathcal{T}_i \sim D$.*
2. *The agent acts in $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}_i, \mathcal{T}_i, \gamma \rangle$ for τ time steps, starting in state s_{init}.*

We are interested in understanding the limits of the size of an agent's representation that can facilitate near optimal behavior in the multi-task setting. We take near-optimal behavior to mean that, for a given $\varepsilon \in [0, 1]$ and $\delta \in (0, 1]$, with probability $1 - \delta$ the agent acts ε-optimal. In the planning case, this is the bar for any solution returned by a planner. In the RL case, this is the bar for the policy the algorithm returns at the end of some finite learning budget.[1] Given this bar, how small can a representation be in principle for planning or RL?

We take two perspectives. First, we determine the compression limit of trajectories that achieve near-optimal behavior. Second, we look into the compression limit of policies.

[1] Naturally one could impose polynomial constraints here, as with efficient PAC-MDP.

1.2.1 MDP Compression View One: Trajectories

We first assume the traditional MDP setting: there is just one MDP the agent must plan with or interact with via RL. We later try to generalize to the multi task case.

Consider a sequence of random variables denoting a trajectory taken by some fixed policy, π:

$$\kappa_n = (s_1, \pi(s_1), r_1), (s_2, \pi(s_2), r_2), \ldots, (s_n, \pi(s_n), r_n) \tag{2}$$

We let K^n denote the random variable taking on values κ^n. Note that the entropy of K^n can be decomposed. First, consider the entropy of a single transition, $H(\kappa_1)$. So:

$$H(\kappa_1) = H(s, a, r) = \underbrace{H(r_1, \mid a_1, s_1)}_{\mathcal{R}(s_1, a_1)} + \underbrace{H(a_1 \mid s_1)}_{=H(\pi(s_1))} + \underbrace{H(s_1)}_{=0} \tag{3}$$

Extrapolating over time, we can decompose the entropy of a trajectory in terms of the major components of the MDP:

$$H(K^n) = \sum_{i=1}^{n-1} H(\pi(s_i)) + H(\mathcal{T}(s' \mid s_i, \pi(s_i)) + H(\mathcal{R}(s_i, \pi(s_i))) \tag{4}$$

Let $\ell_C(\kappa^n)$ denote the length of a code under some fixed coding scheme, C. A code just maps entities to a fixed alphabet, ideally in a way that reduces the overall number of bits needed to represent the original entity.

Now we can translate many central information theory results over to our analysis of trajectories, culminating in the following conjecture:

Conjecture 1.1. *Let K^n be the random variable denoting the length n experience trajectory of an agent acting in MDP M, starting in state s_1, according to policy π. Then the optimal one-to-one code, C^*, for all κ^n has expected code length bounded by the entropy of K^n:*

$$H(K^n) - \delta_n \leq \mathbb{E}\left[\frac{1}{n}\ell_{C^*}(\kappa^n)\right] \leq H(K^n) + \delta_n \tag{5}$$

The goal is then to translate over to the multi-task setting. One glaring weakness is that we're talking about one-to-one codes, which is not necessarily the right answer for RL/planning. For partial proofs and more detail, see Section 3.

1.2.2 MDP Compression View Two: Policies

A similar analysis takes place for policies. The main result here isn't quite complete, so for brevity, I'll leave it at that.

1.3 Measuring MDP Hardness

As a final note, I'm also diving into a project to evaluate MDP (and bandit) difficulty via entropy (both for planning and learning). That is, I conjecture that increasing the entropy of some relevant constituent of an MDP (while keeping everything else fixed) monotonically raises planning/sample complexity of the MDP.

.....

2 Properties of Good Abstractions

Our goal here is to understand what underlies a useful abstraction for sequential decision making. In particular, we have little sense of which features to use, which options to use, which hierarchical architecture to use, and what motivation might exist to prefer some over any other. We argue for and investigate three properties:

1. *Low Planning/Sample Complexity*: Using the abstraction, an agent can efficiently generate it's best guess as to what good behavior is.

   Formalism: the usual computational complexity and sample bound metrics.

2. *Low Value Loss*: The agent's best guess about good behavior is pretty accurate.

   Formalism: Defined with respect to a policy as follows:

   Definition 3 (Value Loss): *The* **value loss** *of a policy, π_A, relative to an MDP M is defined by an $\varepsilon \in \left[0, \frac{\text{RMAX}}{1-\gamma}\right]$ such that:*

   $$\max_s |V^*(s) - V^{\pi_A}| \leq \varepsilon \tag{6}$$

3. *Easy Computability/Learnability of Abstractions*: Abstractions should be efficiently computable.

   Formalism: Demands slightly different things depending on the setting. In planning, the constraint is that the agent can efficiently compute an abstraction. In RL, we're after a sample bound for determining an abstraction.

Our goal is to understand the landscape of possible abstractions as they relate to these properties. Here we'll focus primarily on state abstraction. As a reminder, state abstraction *type* is defined relative to an indicator function on state pairs:

Definition 4 (State Abstraction Type): *A state abstraction type is a collection of functions $\phi : \mathcal{S}_G \mapsto \mathcal{S}_A$ associated with a fixed indicator function on state pairs:*

$$\mathbb{1} : \mathcal{S} \times \mathcal{S} \mapsto \{0, 1\}, \tag{7}$$

such that when any function ϕ cluster state pairs, the indicator must be true for that state pair:

$$\phi(s_1) = \phi(s_2) \rightarrow \mathbb{1}\{s_1, s_2\}. \tag{8}$$

2.1 Planning/Sample Complexity

Results still pending. Primarily investigating how different option types impact these quantities.

2.2 Value Loss Results

Given the level of background detail required I'm going to skip the details of these results. To summarize, there are many types of state and action abstraction, and we've proven value loss bounds for them.

2.3 Easy Computability/Learnability

Here, we have two types of results. In the *planning* multi-task setting, where the agent gets to observe and solve each MDP, we offer the following Theorem:

Theorem 2.1. *For any state abstraction type such that:*

1. *The state abstraction type's indicator function is transitive.*
2. *There exists a polynomial time procedure, F, that can compute a state abstraction of that type for any one MDP.*

For any given $\delta \in (0, 1]$ D, at most:

$$m = O\left(\frac{\ln\left(\frac{1}{1-\delta}\right)}{\delta}\right) \tag{9}$$

samples from D (and subsequent queries to F) are needed to compute an abstraction that holds with probability at least $1 - 2\delta$.

See the Appendix for the proof sketch. If the indicator function is *not* transitive, then we need to solve an NP-Complete problem that is also hard to approximate.

We're now focusing on a learning result of the same form, in which the agent has to perform RL in each sampled MDP instead of calling a procedure F.

2.3.1 State and Action Abstraction Interplay

As a final note, one of the major things we're working up to is to understand the role of each type of abstraction, and prove where delicate interplay between them is necessary. In particular, we'd like to know under what conditions using state and action abstraction is strictly better than just using one or the other? Is there a limit on the improvement achievable? And finally: when are multiple levels of abstraction beneficial?

3 Proofs for Compression

Recall our overarching goal is to prove a limit on the expected code size for compression of trajectories. First we introduce a sample entropy limit result based on the Asymptotic Equipartition property:

Theorem 3.1 (Asymptotic Equipartition Property for MDP Trajectories)**.** *For $\kappa_1, \kappa_2, \ldots, \kappa_n$ sampled according to the stochastic process $M^\pi(s_1)$, assuming that there exists a moment of the random variable K_i with a uniform upper bound for all i:*

$$-\frac{1}{n}\log p(\kappa_1, \kappa_2, \ldots, \kappa_n) \xrightarrow[\Pr]{} \frac{1}{n} H(\kappa_1, \ldots, \kappa_n) \tag{10}$$

Proof.

The proof follows from the Markov Inequality, applied to the r-th moment of $p(\kappa_i)$, for $r > 1$:

$$\Pr\left\{\left|-\frac{1}{n}p(\kappa_1, \kappa_2, \ldots, \kappa_n) - \frac{1}{n}H(\kappa^n)\right| > \delta\right\} \leq \frac{\mathbb{E}\left[|\log p(\kappa_i)|^r\right]}{n^2\delta^2} \longrightarrow 0 \text{ as } n \to \infty$$

□

Therefore, we get a notion of the typical set of trajectories:

Definition 5 (Typical Trajectory Set): *The* **set of typical trajectories**, *denoted $A^n_{\pi,\delta}$, is the set of trajectories $(\kappa_1, \kappa_2, \ldots, \kappa_n) \in \mathcal{K}^n$ with the property:*

$$2^{-(H(\kappa^n)+\delta)} \leq p(\kappa^n) \leq 2^{-(H(\kappa^n)-\delta)} \tag{11}$$

Theorem 3.2. *As a result of the AEP, we can show that $A^n_{\pi,\delta}$ has several properties:*

1. *If $\kappa^n \in A^n_{\pi,\delta}$, then:*

$$H(\kappa^n) - \delta \leq -\frac{1}{n}p(\kappa^n) \leq H(\kappa^n) + \delta \tag{12}$$

2. $\Pr(\kappa^n \in A^n_{\pi,\delta}) > 1 - \delta$

Theorem 3.3. *Let K^n be the random variable denoting the length n experience trajectory of an agent acting in MDP M, starting in state s_1, according to policy π. Then there exists a one-to-one code for all κ^n such that the expected code length is bounded by the entropy of K^n:*

$$\mathbb{E}\left[\frac{1}{n}\ell(\kappa^n)\right] \leq H(\kappa^n) + \delta(1 + \log|\mathcal{K}|) + \frac{2}{n} \tag{13}$$

Proof.

| Follows from the above properties of the AEP.

□

Theorem 3.4 (Kraft Inequality)**.** *Assume $\mathcal{C}$ is a prefix code, and that K is the size of the coding alphabet. Let $\ell(x)$ denote the length of the code for x. Then:*

$$\sum_{x \in \mathcal{X}} K^{-\ell(x)} \leq 1 \tag{14}$$

From the Kraft Inequality, we get a limit on compression of the desired form. I just need to clean up some of the details.

4 Proofs for Computing Abstraction

I don't go into every last detail, but here's the general idea:

Proof.

Let Φ refer to some state abstraction function type associated with indicator $\mathbb{1} : \mathcal{S} \times \mathcal{S} \mapsto \{0, 1\}$. Given $\delta \in [0, 1)$ a fixed $\mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma, s_1 \in \mathcal{S}$, and a fixed but unknown distribution on reward functions D.

For each sampled reward function $\mathcal{R}_i \sim D$, consider a graph G_i where each state s_i of the MDP is mapped to a vertex v_i. An edge between v_1 and v_2 indicates $\mathbb{1}\{s_1, s_2\}$.
Let $e_i(v_1, v_2)$ denote the boolean expressing whether there is an edge between v_1 and v_2 for reward function $\mathcal{R}_i \sim D$. After m sampled reward functions, we compute the empirical mean of each edge:

$$\hat{e}(v_1, v_2) = \frac{1}{m} \sum_{i=1}^{m} e_i(v_1, v_2) \tag{15}$$

By the Hoeffding inequality, we can bound the deviation of $\hat{e}$ from its expectation with respect to the distribution. That is, for all $\varepsilon \in (0, 1]$:

$$\begin{aligned} \Pr\left(|\hat{e}(a, b) - \mathbb{E}\left[e(a, b)\right]| \geq \varepsilon\right) &\leq 2 \exp\left(-\frac{2m^2\varepsilon^2}{m}\right) \\ &= 2 \exp\left(-2m\varepsilon^2\right) \end{aligned}$$

Thus:

$$\Pr\left(|\hat{e}(a, b) - \mathbb{E}\left[e(a, b)\right]| \leq \varepsilon\right) \geq 2 \exp\left(-2m\varepsilon^2\right) \tag{16}$$

Let $1 - \delta = 2 \exp\left(-2m\varepsilon^2\right)$.
Consider the threshold graph, $G_{\hat{e}}$, formed by including those edges for which $\hat{e}(v_1, v_2) - \varepsilon \geq 1 - \delta$:

$$\forall_{v_1, v_2 \in V_G} : \hat{e}_G(v_1, v_2) = \mathbb{1}\left\{\hat{e}_i(v_1, v_2) - \varepsilon \geq 1 - \delta\right\}$$

So, the edges we keep in $G_{\hat{e}}$ are those that occur with probability at least $1-\delta$. Thus, with probability $(1-\delta)^2$, the edges we return are true edges. Note that $(1-\delta)^2 = 1-2\delta+\delta^2 \geq 1-\delta$. Thus, since $\varepsilon > 1-\delta$, we conclude that for

$$\therefore m \geq \frac{\ln\left(\frac{2}{1-\delta}\right)}{2\delta}$$

sampled and solved MDPs, we can compute an abstraction ϕ based on the type Φ associated with $\mathbb{1}$ that holds with probability. $\square$