Reinforcement Learning

Homework Assignment - Advanced Topics in RL

Alireza Gargoori Motlagh

June 23, 2023

---

1 Skill Discovery

1.1

- $I(S;Z)$: Mutual Information between states and skills. We want to maximize this as we want our states to be indicator of the skill and states be as discriminative as possible.
- $I(A;Z|S)$: Mutual Information between actions and skills, conditioned on states. This term is needed as want to avoid actions which make the skills distinguishable.
- $H(A|S)$: Entropy of actions, conditioned on states. We want to experience different actions on each state to ensure exploration.

1.2

$$F(\theta) = I(S;Z) + H(A|S) - I(A;Z|S)$$

The first two terms are maximized and we want to minimize the third term, due to the following reasons:

- By maximizing $I(S;Z)$ we want to encode the idea that the skill should control which states the agent visits. Conveniently, this mutual information dictates that we can infer the skill from the states visited.
- We want to ensure exploration in each state for our mixture of policies by maximizing $H(A|S)$; therefore an entropy term is added to the objective with the aim of having more exploration.
- As described in the objectives of optimization, we want our states to be a discriminator of our skill, and not the actions. In other words, as want to avoid actions which make the skills distinguishable, we aim to minimize $I(A;Z|S)$.

1.3

$$F(\theta) = I(S;Z) + H(A|S) - I(A;Z|S) =$$
$$[H(Z) - H(Z|S)] + H(A|S) - [H(A|S) - H(A|S,Z)] = H(Z) - H(Z|S) + H(A|S,Z)$$

- The first term $H(Z)$ encourages our prior distribution over $p(z)$ to have high entropy. In the original paper of DIAYN [1]they fix this distribution to be uniform to ensure the maximum entropy over skills and therefore learning different skills and not sampling skills only from a subset of more probable ones.

- The second term suggests that it should be easy to infer the skill z from the current state s. This was indeed one of the main required objectives described in the problem.
- The third term suggests that each skill should act as randomly as possible, achieved by using MaxEnt-policy.

1.4

As we cannot directly compute $p(z|s)$ since it is interactable due to the need of integrating over all states and/or skills, we approximate this using a parametric form $q_\phi(z|s)$ and find a variational lower bound as follows:

$$F(\theta) = H(Z) - H(Z|S) + H(A|S,Z)$$
$$= -\mathbb{E}_{z\sim p(z)}[\log(p(z))] + \mathbb{E}_{z\sim p(z),s\sim d^\pi(s)}[\log(p(z|s))] + H(A|S,Z)$$
$$\geq \mathbb{E}_{z\sim p(z),s\sim d^\pi(s)}[\log(q_\phi(z|s)) - \log(p(z))] + H(A|S,Z)$$

where we have used the fact that $D_{KL}(.,.) \geq 0$:

$$\mathbb{E}_{z\sim p(z),s\sim d^\pi(s)}[\log(p(z|s))] = \mathbb{E}_{z\sim p(z),s\sim d^\pi(s)} \log[\frac{p(z|s)q_\phi(z|s)}{q_\phi(z|s)}] =$$

$$\mathbb{E}_{z\sim p(z),s\sim d^\pi(s)} \log[\frac{p(z|s)}{q_\phi(z|s)}] + \mathbb{E}_{z\sim p(z),s\sim d^\pi(s)}[q_\phi(z|s)] = D_{KL}(p(z|s), q_\phi(z|s)) + \mathbb{E}_{z\sim p(z),s\sim d^\pi(s)}[q_\phi(z|s)]$$
$$\geq \mathbb{E}_{z\sim p(z),s\sim d^\pi(s)}[q_\phi(z|s)]$$

We can use this variational lower bound as the objective of our optimization and define the objective to be:

$$G(\theta,\phi) = \mathbb{E}_{z\sim p(z),s\sim d^\pi(s)}[log(q_\phi(z|s)) - log(p(z))] + H(A|S,Z)$$

1.5

The first term in the $G(\theta,\phi)$ is an indicator of exploration, as we want to have a high entropy of actions given the states and skills. However, the second term is in indicator of a pseudo-reward which we want to maximize: increase in the probability of discriminating the skill after knowing the state and therefore it is an indicator of exploitation.
So we have a trade-off between exploration and exploitation and we can settle this using a MaxEnt-policy to ensure exploration and using this with a factor α to scale between exploration and discriminability.

1.6

As described in the previous part, we can use a MaxEnt-policy such as SAC to achieve the desired exploration and take care of that. To have enough discriminability we define a pseudo-reward as the increase in the probability of discriminating the skill after knowing the state:

$$r_z(s,a) = \log(q_\phi(z|s)) - \log(p(z))$$

By adopting this approach, the agent is encouraged to explore states that can be easily distinguished, thereby receiving rewards. Simultaneously, the discriminator is improved to more accurately infer the skill variable based on the states encountered.

1.7

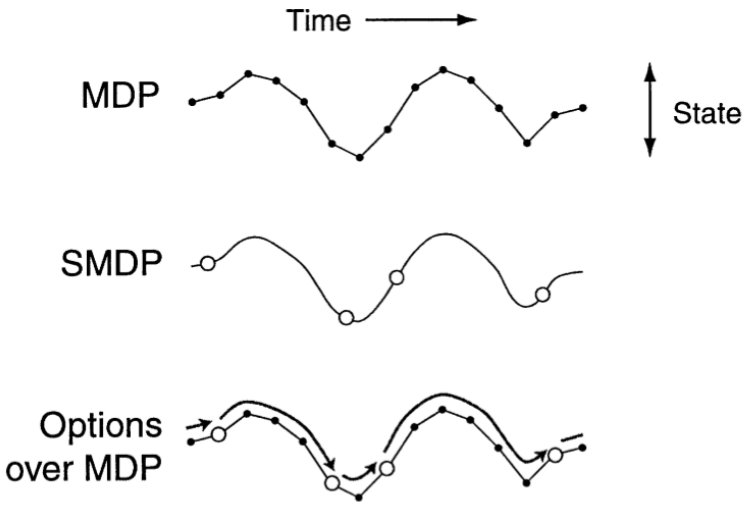
$$I(s';z|s) = H(s'|s) - H(s'|s,z) = H(z|s) - H(z|s,s')$$

In this algorithm, we are encouraged to have a model which reduces its uncertainty after knowing the task in hand. In other words, the first term indicates that future is hard to predict for different skills, but after knowing the skill, we have a predictable future. Maximizing this objective corresponds to maximizing the diversity of transitions produced in the environment, while making z informative about the next state s' by minimizing the second term.
Also, as mutual information is symmetric, an equal notion of discrimination between tasks after knowing the next state can be derived from the second equation.

2 Hierarchical RL

2.1

As we know, there is no notion of a course of action over a variable period of time in MDPs and therefore MDPs do not include temporal abstraction. However, the actions in SMDPs take different amounts of time and are devised to model temporally-extended actions in different time courses.
This can be justified with the following figure. While the trajectory of an MDP is made up of small, discrete-time transitions between states, we have continus-time transitions and interval-dependent discounts:



2.2

To find the bellman equations for an option o with a time span of k through a policy μ we can write:

$$V^{\mu}(s) = \mathbb{E}_{\mu}[r_s^o + \gamma^k V^{\mu}(s_{t+k})|\epsilon(\mu;s;t)] = \sum_{o \in O_s} \mu(o,s)[r_s^o + \sum_{s'} p_{s's}^o V^{\mu}(s')]$$

This is analogous to the bellman equation in MDPs, just by replacing current reward with r_s^o as we have a discounted time course reward, and replacing the transition probability of $p(s'|s,a)$ by $p^o_{s's}$ as this describes the probability of state-prediction using option o.

Now to derive the Q-value bellman equations we can write:

$$Q^\mu(s,o) = \mathbb{E}_\mu[r_s^o + \gamma^k V^\mu(s_{t+k})|\epsilon(o;s;t)]$$

$$= \mathbb{E}_\mu[r_s^o + \gamma^k \sum_{o' \in O_{s_{t+k}}} \mu(s_{t+k},o')Q^\mu(s_{t+k},o')]|\epsilon(o;s;t)]$$

$$= r_s^o + \sum_{s'} p^o_{s's} \sum_{o' \in O_{s_{t+k}}} \mu(s_{t+k},o')Q^\mu(s_{t+k},o')$$

The second equation is derived based on the definition of value of next state as the expectation of Q-values on the next state over the current policy. The third equation is the result of definition of $p^o_{s's}$ as the probability of reaching s' after k steps from s, using option o.

2.3

$$V^*(s) = \max_\mu V^\mu(s)$$

$$= \max_\mu \mathbb{E}_\mu[r_s^o + \gamma^k V^*(s_{t+k})|\epsilon(\mu;s;t)]$$

This is equivalent to the maximum of options over the possible set of options in the state and hence, the objective could be rewritten as:

$$V^*(s) = \max_{o \in O_s} \mathbb{E}[r_s^o + \sum_{s'} p^o_{s's} V^*(s')|\epsilon(o;s;t)]$$

$$= \max_{o \in O_s} r_s^o + \sum_{s'} p^o_{s's} V^*(s') = \max_{o \in O_s} \mathbb{E}[r + \gamma^k V^*(s')|\epsilon(s,o)]$$

Finding the optimality Bellman equation for Q-values is easily done just by using the fact that $V^*(s) = \max_{o' \in O_s} Q^*(s,o')$:

$$Q^*(s,o) = \max_\mu Q^\mu(s,o)$$

$$= \max_\mu \mathbb{E}_\mu[r_s^o + \gamma^k V^*(s_{t+k})|\epsilon(o;s;t)]$$

$$= \max_\mu \mathbb{E}_\mu[r_s^o + \gamma^k \max_{o' \in O_{s_{t+k}}} Q^*(s_{t+k},o')|\epsilon(o;s;t)]$$

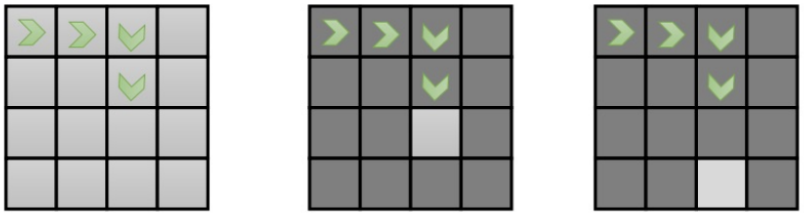
$$= r_s^o + \sum_{s'} p^o_{s's} \max_{o' \in O_{s_{t+k}}} Q^*(s_{t+k},o')$$

$$= \mathbb{E}[r + \gamma^k \max_{o' \in O_{s_{t+k}}} Q^*(s',o')|\epsilon(s,o)]$$

3 Inverse RL

3.1

Inverse RL allows us to learn a reasonable reward function instead of simply imitating the expert; this allows us to have a generalization of policy to unseen states while the expert has not encountered those states or observations. This is done by learning a parametric reward function through the expert state action pairs, which is an anology to learning the reason behind what expert used to make those decisions.
However, this problem is by itself very ill-posed. In other words, this is an underspecified problem and many reward functions can explain the same behaviour of expert. A simple example would be learning the reward function in a grid world, where many rewards can reason about the trajectories of expert. The following figure from the lecture slides shows this underspecificity. While the trajectory of expert is available, we cannot infer what was the true goal and many possibilities could be the true reason:



3.2

We have defined a linear function of features for the reward function:

$$r_\psi(s,a) = \sum_{i=1}^{n} \psi_i f_i(s,a) = \psi^T \mathbf{f}(s,a)$$

$$\mathbb{E}_{\pi^{r_\psi}}[f(s,a)] = \mathbb{E}_{\pi^*}[f(s,a)]$$

3.2.1

This choice of reward function is feature matching and matches the expectations of features between the expert behaviour and the optimal(desired) policy based on the learned reward function π^{r_ψ}. This is a very reasonable choice since we expect our policy to have in average the same features from the expert's policy.

3.2.2

However, this is still very ambiguous as the problem of underspecificity still remains. In other words, multiple ψ vectors can result in the same expected feature vectors. An example for that would be the above grid world where different features and hence ψ vectors could have the same result as the expert's.

3.2.3

Maximum margin principle tries to find the weights where we want to expected reward under the expert's policy be better than any other possible (and hence the maximum utility) policy by a margin m. Basically we are trying to pick the weights which make the rewards under the expert's policy better than other policies.
The problem with that formulation is that if the space of policies is large and continuous, there are likely other policies that are similar to the expert's policy (and even identical) and just maximizing the distance between all other policies may not be a reasonable idea. A better formulation of the problem could be to set weights by a similarity measure such as D_{KL} between policies and rewrite the objective as follows:

$$\begin{aligned} \min_{\psi} \quad & \frac{1}{2}\|\psi\|^2 \\ \text{s.t.} \quad & \psi^T \mathbb{E}_{\pi^*}[f(s,a)] \geq \max_{\pi \in \Pi} \psi^T \mathbb{E}_{\pi}[f(s,a)] + D(\pi, \pi^*) \end{aligned}$$

In this way, we are maximizing the distance more for the policies that are more distinct from the expert's, whereas similar policies have a less distance and get less margin.
However this definition is still problematic as this is a hard constrained problem which is not optimal for deep learning models. Also, there is no notion of suboptimality in this definition.

3.3

$$p(O_t|s_t, a_t) = \exp(r_\psi(s_t, a_t))$$

To find the probability of a trajectory under optimality variables we have:

$$p(\tau|O_{1:T}) = \frac{p(\tau, O_{1:T})}{p(O_{1:T})}$$

$$\propto p(\tau)p(O_{1:T}|\tau) = p(\tau)\Pi_{t=1}^{T} p(O_t|\tau)$$

$$= p(\tau)\Pi_{t=1}^{T} p(O_t|s_t, a_t) = p(\tau)\Pi_{t=1}^{T} \exp(r_\psi(s_t, a_t))$$

where we have used the fact that optimality variables are independent given state action pairs of the corresponding step. We can simplify the above equation to get:

$$p(\tau|O_{1:T}) \propto p(\tau) \exp\big(\sum_{t=1}^{T} r_\psi(s_t, a_t)\big)$$

To state the above equation precisely, we can plug in the partition function to scale this probability between 0 and 1:

$$p(\tau|O_{1:T}) = \frac{1}{Z_\psi} p(\tau) \exp\big(\sum_{t=1}^{T} r_\psi(s_t, a_t)\big)$$

where Z_ψ is the integral of the numerator over the trajectories:

$$Z_\psi = \int p(\tau) \exp\big(\sum_{t=1}^{T} r_\psi(s_t, a_t)\big) d\tau$$

3.3.1

$$L_\psi = \frac{1}{N}\sum_{i=1}^{N}\log(p(\tau_i|O_{1:T},\psi))$$

Substituting what we found for $p(\tau|O_{1:T})$ in the above equation yields:

$$L_\psi =$$

$$\frac{1}{N}\sum_{i=1}^{N}[\log(p(\tau_i))+\log(\exp(\sum_{t=1}^{T}r_\psi(s_t,a_t)))-\log(Z_\psi)] = \frac{1}{N}\sum_{i=1}^{N}[\log(p(\tau_i))+(\sum_{t=1}^{T}r_\psi(s_{t,i},a_{t,i}))]-\log(Z_\psi)$$

As we want to maximize the log-likelihood function, we can ignore the first term as it is not a function of ψ parameters; therefore we have:

$$\max_\psi L_\psi = \max_\psi \frac{1}{N}\sum_{i=1}^{N}(\sum_{t=1}^{T}r_\psi(s_{t,i},a_{t,i})) - log(Z_\psi) = \max_\psi \frac{1}{N}\sum_{i=1}^{N}r_\psi(\tau_i) - \log(Z_\psi)$$

where we have:

$$r_\psi(\tau_i) = \sum_{t=1}^{T}r_\psi(s_{t,i},a_{t,i}) \quad Z_\psi = \int p(\tau)\exp(r_\psi(\tau_i))d\tau$$

3.3.2

If we just ignore the log-normalizer we are just maximizing the sum of rewards along a trajectory and we would assign high rewards to these without considering that other trajectories would probably get also high trajectory although they're not in the expert's trajectories. It would make the trajectories that the expert's has seen more likely than other trajectories which are not observed.

3.3.3

$$\nabla_\psi L_\psi = \frac{1}{N}\sum_{i=1}^{N}\nabla_\psi r_\psi(\tau_i) - \nabla_\psi \log(Z_\psi)$$

The first term is simply the expectation of gradient of sum of rewards of trajectories under the expert's policy; that is:

$$\frac{1}{N}\sum_{i=1}^{N}\nabla_\psi r_\psi(\tau_i) = \mathbb{E}_{\tau\sim\pi^*(\tau)}[\nabla_\psi r_\psi(\tau_i)]$$

The second term could be simplified as follows:

$$\nabla_\psi \log(Z_\psi) = \nabla_\psi \frac{\nabla_\psi(Z_\psi)}{Z_\psi} = \frac{1}{Z_\psi}\int p(\tau)\exp(r_\psi(\tau_i))\nabla_\psi r_\psi(\tau_i)d\tau$$

Since we had:

$$p(\tau|O_{1:T},\psi) = \frac{1}{Z_\psi} p(\tau) \exp r_\psi(\tau)$$

the second term could be rewritten as the expectation of gradient of sum of rewards under the optimal policy:

$$\nabla_\psi \log(Z_\psi) = \mathbb{E}_{\tau\sim p(\tau|O_{1:T},\psi)}[\nabla_\psi r_\psi(\tau_i)]$$

Therefore we have:

$$\nabla_\psi L_\psi = \mathbb{E}_{\tau\sim\pi^*(\tau)}[\nabla_\psi r_\psi(\tau_i)] - \mathbb{E}_{\tau\sim p(\tau|O_{1:T},\psi)}[\nabla_\psi r_\psi(\tau_i)]$$

We can further simplify the second term by expanding the reward of a trajectory as the sum of rewards along the path to get:

$$\mathbb{E}_{\tau\sim p(\tau|O_{1:T},\psi)}[\nabla_\psi r_\psi(\tau_i)] = \sum_{t=1}^{T} \mathbb{E}_{(s_t,a_t)\sim p(s_t,a_t|O_{1:T},\psi)}[\nabla_\psi r_\psi(s_t,a_t)]$$

The expectation is over $p(s_t, a_t|O_{1:T},\psi)$ which could be rewritten as:

$$p(s_t,a_t|O_{1:T},\psi) = p(a_t|s_t,O_{1:T},\psi)p(s_t|O_{1:T},\psi)$$

Also from the Bayesian RL we had:

$$p(a_t|s_t,O_{1:T},\psi) = \frac{\beta(a_t,s_t)}{\beta(s_t)} \quad p(s_t|O_{1:T},\psi) \propto \alpha(s_t)\beta(s_t)$$

where $\beta(s_t,a_t)$ is the backward message and $\alpha(s_t)$ is the forward message.
By multiplying the above equations we get:

$$p(s_t,a_t|O_{1:T},\psi) = p(a_t|s_t,O_{1:T},\psi)p(s_t|O_{1:T},\psi) \propto \beta_t(s_t,a_t)\alpha_t(s_t) = \mu_t(s_t,a_t)$$

Hence the second term of the $\nabla_\psi L_\psi$ would be equal to:

$$\mathbb{E}_{\tau\sim p(\tau|O_{1:T},\psi)}[\nabla_\psi r_\psi(\tau_i)] = \sum_{t=1}^{T} \mathbb{E}_{(s_t,a_t)\sim\mu_t(s_t,a_t)}[\nabla_\psi r_\psi(s_t,a_t)] = \sum_{t=1}^{T} \int_A \int_S \mu_t(s_t,a_t)\nabla_\psi r_\psi((s_t,a_t))ds_t da_t$$

Putting all together we have:

$$\nabla_\psi L_\psi = \mathbb{E}_{\tau\sim\pi^*(\tau)}[\nabla_\psi r_\psi(\tau_i)] - \sum_{t=1}^{T} \int_A \int_S \mu_t(s_t,a_t)\nabla_\psi r_\psi((s_t,a_t))ds_t da_t$$

3.3.4

In the new equation we can find the forward and backward messages whereas the partition function was interactable to compute as it needed to integrate over all trajectories. It allows us to sample trajectories to calculate the second expectation term while it was not possible in the first equation for $\log(Z_\psi)$. Therefore we have a formulation which gives us a practical way to update ψ weights.
However, this also needs to know the dynamics and typically works for discrete and small action-state spaces.

3.3.5

As the second term in the gradient of log-likelihood required a bayesian approach to solve for sub-optimal policies, we can instead sample trajectories by running a MaxEnt-RL to ensure the sub-optimality and imporving this MaxEnt-policy. Therefore we have:

$$\nabla_\psi L_\psi \approx \frac{1}{N}\sum_{i=1}^{N}\nabla_\psi r_\psi(\tau_i) - \frac{1}{M}\sum_{j=1}^{M}\nabla_\psi r_\psi(\tau_j)$$

where the first term is derived from the the expert's trajectories and the second term from the rollouts of the current policy.
Since this would introduce bias to the solution as the previous trajectories of the MaxEnt-policy are from a different policy (off-policy methods), we must use importance sampling to overcome this challenge, by assigning weights to the policy trajectories:

$$\nabla_\psi L_\psi \approx \frac{1}{N}\sum_{i=1}^{N}\nabla_\psi r_\psi(\tau_i) - \frac{1}{\sum_j w_j}\sum_{j=1}^{M} w_j \nabla_\psi r_\psi(\tau_j)$$

The weights could be derived as follows:

$$w_j = \frac{p(\tau_j)p(O_{1:T}|\tau_j,\psi)}{p(\pi(\tau_j))} = \frac{p(s_1)\Pi_t p(s_{t+1}|s_t,a_t)\exp(r_\psi(s_t,a_t))}{p(s_1)\Pi_t p(s_{t+1}|s_t,a_t)\pi(a_t|s_t)}$$
$$= \frac{\Pi_t \exp(r_\psi(s_t,a_t))}{\Pi_t \pi(a_t|s_t)} = \frac{\exp(\sum_t r_\psi(s_t,a_t))}{\Pi_t \pi(a_t|s_t)} = \frac{\exp(r_\psi(\tau_j))}{\Pi_t \pi(a_t^j|s_t^j)}$$

4 Offline RL

4.1

$$l(\pi) = \mathbb{E}_{p_\pi(\tau)}[\sum_{t=0}^{H}\delta(a_t,a_t^*)], \quad p_\pi(a_t \neq a_t^*|s) \leq \epsilon$$

Note that this loss is expected over the distribution of states which expert has visited. We can write the probability that π makes a mistake under the expert's distribution of states, denoted by ϵ_t as follows:

$$\epsilon_t = \mathbb{E}_{s\sim d_{\pi^*}^t}[\mathbb{E}_{a\sim\pi}[\delta(a_t,\pi^*(s_t))]]$$

Define p_t as the probability that π has not made a mistake until time step t. Also consider d_t as the distribution of state π is in at time t conditioned on the fact it has not made a mistake so far and d_t' represents the distribution of states at time t obtained by following π^* but conditioned on the fact that π made at least one mistake in the first $t-1$ visited states. Therefore we can write:

$$\epsilon_t^{\pi^*} = p_{t-1}\epsilon_t + (1-p_{t-1})\epsilon_t^{'}$$

The expected loss of π is at most one if it has made a mistake or the expected loss if it has not made any mistake; therefore:

$$l(\pi) \leq \sum_{t=1}^{H} p_{t-1}\mathbb{E}_{s\sim d_t} p_{t-1} l(\pi(s)) + (1-p_{t-1})\times 1$$

$$\leq l(\pi^*) + \sum_{t=1}^{H} \sum_{i=1}^{t} \epsilon_i$$

$$\leq \text{Const.} + H \sum_{i=1}^{t} \epsilon_i$$

$$= \text{Const.} + H^2 p_\pi(a_t \neq a_t^* | s)$$

$$\leq \text{Const.} + H^2 \epsilon$$

This is intuitevly correct; as the current policy makes a mistake at any time step, it might reaches a new state where the expert has not been and always incur maximal loss of 1 at each step from then on. [2]We know that ERM makes no guarantees about error when encountering OOD inputs different from training stage. Therefore having two sums where the upper bound of error in each time step is ϵ and therefore the order of the expected error of the learned policy being $\mathcal{O}(H^2\epsilon)$.

4.2

In contast with the offline case, in the online case as the agent can interact with the environment, there is no problem with OOD states and the upper bound of error would get linear as follows:

$$\epsilon_t = \mathbb{E}_{s \sim d_\pi^t}[\mathbb{E}_{a \sim \pi}[\delta(a_t, \pi^*(s_t))]]$$

$$l(\pi) \leq l(\pi^*) + \sum_{t=1}^{H} \epsilon_t$$

$$\leq l(\pi^*) + \sum_{t=1}^{H} \epsilon$$

$$\leq \text{Const.} + H\epsilon$$

Therefore the order of the upper bound for the training error would be $\mathcal{O}(H\epsilon)$. We can think of this as the reason of getting the expectation over the agent's distribution of states and not being under the expert's states' distribution.

References

[1] B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine, "Diversity is all you need: Learning skills without a reward function," 2018.

[2] S. Ross and D. Bagnell, "Efficient reductions for imitation learning," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, Y. W. Teh and M. Titterington, Eds., vol. 9. Chia Laguna Resort, Sardinia, Italy: PMLR, 13–15 May 2010, pp. 661–668. [Online]. Available: https://proceedings.mlr.press/v9/ross10a.html