Contrastive Reinforcement Learning

Density Estimation

Objective

Given samples $x \sim P_+$, learn a model $p_\theta(x)$ such that $p_\theta(x) \approx P_+(x)$.

Density Estimation

Objective

Given samples $x \sim P_+$, learn a model $p_\theta(x)$ such that $p_\theta(x) \approx P_+(x)$.

Maximum Likelihood Estimator (MLE)

Given i.i.d. samples $x_1, \cdots, x_n \sim P_+$, find parameterized density p_θ to maximize the likelihood

$$\max_\theta \prod_{i=1}^{n} p_\theta(x_i)$$

or

$$\max_\theta \sum_{i=1}^{n} \log p_\theta(x_i).$$

Therefore,

$$\textbf{MLE:} \quad \max_\theta \ \mathbb{E}_{x \sim P_+} \left[\log p_\theta(x)\right]$$

Density Estimation

Key Challenges

- High-dimensional data (images, states, trajectories)
- Normalization (computing partition function)
- Efficient sampling vs. likelihood evaluation

Core Idea of Contrastive Learning

Turn density estimation into classification:

- Data (positive): $x_+ \sim P_+$
- Noise (negative): $x_- \sim P_-$ (we choose this distribution)

Train classifier:

Is x data or noise?

Logistic regression for the binary classifier

Let $f_\theta(x)$ be the classifier score, then

$$P(x \in D) \approx \sigma(f_\theta(x)) = \frac{1}{1 + e^{-f_\theta(x)}}$$

From Density Estimation to Classification

Training a classifier is easier because

- It replaces full probabilistic modeling with a relative comparison task (is x more likely a data or noise?).
- It avoids normalization
- It can be solved using existing supervised learning methods.

Bayes Optimal Classifier

Binary Cross Entropy Loss

Given a sequence of training data with labels (x_i, y_i) where $y_i \in \{1, 0\}$, the likelihood of classifier is

$$\left(\prod_{i:\ y_i=1} \sigma(f_\theta(x_i)) \right) \left(\prod_{i:\ y_i=0} (1 - \sigma(f_\theta(x_i))) \right),$$

or the cross entropy loss

$$-\sum_i \left(y_i \log \sigma(f_\theta(x_i)) + (1 - y_i) \log(1 - \sigma(f_\theta(x_i))) \right).$$

Bayes Optimal Classifier

Binary Cross Entropy Loss

Given a sequence of training data with labels (x_i, y_i) where $y_i \in \{1, 0\}$, the likelihood of classifier is

$$\left(\prod_{i:\ y_i=1} \sigma(f_\theta(x_i)) \right) \left(\prod_{i:\ y_i=0} (1 - \sigma(f_\theta(x_i))) \right),$$

or the cross entropy loss

$$-\sum_i (y_i \log \sigma(f_\theta(x_i)) + (1 - y_i) \log(1 - \sigma(f_\theta(x_i)))).$$

Theorem

Assume x is sampled from data or noise equally likely, then the cross entropy loss is minimized when

$$f_\theta^*(x) = \log \left(\frac{P_+(x)}{P_-(x)} \right).$$

Proof

Recall the cross entropy loss

$$
\begin{aligned}
& -\mathbb{E}\left[y \log \sigma(f_\theta(x)) + (1-y)\log(1-\sigma(f_\theta(x)))\right] \\
=& -\frac{1}{2}\int \log \sigma(f_\theta(x_+))P_+(x_+)\,dx_+ - \frac{1}{2}\int \log(1-\sigma(f_\theta(x_-)))P_-(x_-)\,dx_- \\
=& -\frac{1}{2}\int \left(\log \sigma(f_\theta(x))P_+(x) + \log(1-\sigma(f_\theta(x)))P_-(x)\right)\,dx
\end{aligned}
$$

Assume the dimension of θ is large enough so that we can independently optimize $f_\theta(x)$ for each x, then

$$
\frac{\mathrm{d}}{\mathrm{d}f_\theta(x)}\Big[\log \sigma(f_\theta(x))P_+(x) + \log(1-\sigma(f_\theta(x)))P_-(x)\Big] = 0
$$

Bayes Optimal Classifier

Solving,

$$P_+(x)\big(1-\sigma(f^*(x))\big)-P_-(x)\sigma(f^*(x))=0 \iff \sigma(f^*(x))=\frac{P_+(x)}{P_+(x)+P_-(x)}$$

$$\iff \boxed{f^*(x)=\log\left(\frac{P_+(x)}{P_-(x)}\right)}$$

Bayes Optimal Classifier

Solving,

$$P_+(x)\big(1-\sigma(f^*(x))\big) - P_-(x)\sigma(f^*(x)) = 0 \iff \sigma(f^*(x)) = \frac{P_+(x)}{P_+(x)+P_-(x)}$$

$$\iff \boxed{f^*(x) = \log\left(\frac{P_+(x)}{P_-(x)}\right)}$$

We then obtain

$$P_+(x) \approx e^{f_\theta^*(x)} P_-(x).$$

Goal-Conditioned Reinforcement Learning

From the environment,

- γ: discount factor.
- $p(s_{t+1}|s_t, a_t)$: transition kernel.
- **No reward!**

Goal-Conditioned Reinforcement Learning

From the environment,

- γ: discount factor.
- $p(s_{t+1}|s_t, a_t)$: transition kernel.
- **No reward!**

From us,

- $\rho(g)$: some prior distribution over goals. Chosen by us.
- Goal conditioned policy: $\pi(a \mid s, g)$: *policy / distribution over actions a in state s, trying to reach goal g*
- Random-goal policy (or marginalized policy): $\pi(a \mid s) = \sum_g \pi(a \mid s, g)\rho(g)$. Under this policy, a goal is sampled at each step under $\rho(g)$ and then action is chosen using $\pi(a \mid s, g)$.

We define

$$r_g(s_t, a_t) = (1-\gamma)p(s_{t+1} = g|s_t, a_t)$$

as the discounted probability of reaching goal g in the next step taking a_t in state s_t, and

$$Q^\pi(s, a, g) \triangleq \mathbb{E}_{\pi(\cdot|\cdot)}\left[\sum_{t=0}^{\infty} \gamma^t r_g(s_t, a_t) \mid s_0 = s, a_0 = a\right]$$

as the goal-conditioned Q function under the **random-goal policy** π.

Question

How to do efficient **policy evaluation** for all (s, a, g)?

Approach 1: Bootstrapping / TD-learning

Setting: Learn $Q(s, a, g)$ via TD updates

$$Q^{\pi}(s, a, g) \leftarrow r_g(s, a) + \gamma Q^{\pi}(s', a', g)$$

1. Tabular Complexity Scales with #Goals

- Tabular $Q(s, a, g)$ has size $|\mathcal{S}| \times |\mathcal{A}| \times |\mathcal{G}|$
- Policy evaluation requires solving Bellman equations for each goal
- Computational and sample complexity scale linearly in $|\mathcal{G}|$
- Does not exploit **shared structure**.

Preliminaries: Occupancy Measure

We have the following important objects

- $p_t^{\pi(\cdot|\cdot)}(s_t = s)$: *probability of being in s at time t under random goal policy π.*
- $p^{\pi(\cdot|\cdot)}(s_+ = s)$: *discounted probability of being in s at a future time (geometrically distributed) under random goal policy π*

$$p^{\pi(\cdot|\cdot)}(s_+ = s) = (1-\gamma)\sum_{t=1}^{\infty}\gamma^{t-1}p_t^{\pi(\cdot|\cdot)}(s_t = s).$$

Preliminaries: Occupancy Measure

We have the following important objects

- $p_t^{\pi(\cdot|\cdot)}(s_t = s)$: *probability of being in s at time t under random goal policy π.*
- $p^{\pi(\cdot|\cdot)}(s_+ = s)$: *discounted probability of being in s at a future time (geometrically distributed) under random goal policy π*

$$p^{\pi(\cdot|\cdot)}(s_+ = s) = (1-\gamma)\sum_{t=1}^{\infty}\gamma^{t-1}p_t^{\pi(\cdot|\cdot)}(s_t = s).$$

Q function as distribution

The goal-conditioned Q function is a distribution! Namely, the following conditional distribution:

$$Q^{\pi}(s, a, g) = p^{\pi(\cdot|\cdot)}(s_+ = g \mid s, a).$$

Putting it all together

Assume the initial (state, action) pair is sampled according to a given distribution $d_0(s, a)$. Then

$$\hat{Q}^\pi(s, a, g) = Q^\pi(s, a, g) d_0(s, a) = p^{\pi(\cdot|\cdot)}(s_+ = g \mid s, a) d_0(s, a)$$

is a distribution over support (s, a, g).

Key insight

If $\hat{Q}^\pi(s, a, g)$ is a distribution, then we can learn using contrastive learning!

Putting it all together

Assume the initial (state, action) pair is sampled according to a given distribution $d_0(s, a)$. Then

$$\hat{Q}^\pi(s, a, g) = Q^\pi(s, a, g) d_0(s, a) = p^{\pi(\cdot|\cdot)}(s_+ = g \mid s, a) d_0(s, a)$$

is a distribution over support (s, a, g).

Key insight

If $\hat{Q}^\pi(s, a, g)$ is a distribution, then we can learn using contrastive learning!

What do we need?

1. Some way to sample goals (s, a, g) following distribution $\hat{Q}^\pi(s, a, g)$.
2. Some noise distribution that is easy to sample from and close to $\hat{Q}^\pi(s, a, g)$.

How to sample from $P_+ \sim \hat{Q}^\pi(s, a, g)$?

Recall that $\hat{Q}^\pi(s, a, g) = p^{\pi(\cdot|\cdot)}(s_+ = g \mid s, a) d_0(s, a)$.

- Sample initial state and action pair (s, a). Use the random-goal policy to generate a sample trajectory $(s_0 = s, a_0 = a), (s_1, a_1), \ldots, (s_n, a_n)$
- Sample a $g^+ = s_{t_+}$ where time $t_+ \sim \mathrm{Geom}(1 - \gamma)$ and assign label $+$ to sample (s, a, g^+).

How to sample from P_- for some P_-?

- Sample the initial state, action pair (s, a) and the initial goal g^- and assign label $-$ to (s, a, g^-).

Contrastive RL

Recall the CL loss

$$\mathcal{L}(\theta) = -\mathbb{E}_{x_+ \sim P_+, x_- \sim P_-}\left[\log \sigma(f_\theta(x_+)) + \log(1 - \sigma(f_\theta(x_-)))\right] \implies f^*(x) = \log\left(\frac{P_+(x)}{P_-(x)}\right)$$

where

- $x = (s, a, g)$
- $P_+((s, a, g)) = p^{\pi(\cdot|\cdot)}(g \mid s, a) d_0(s, a)$
- $P_-((s, a, g)) = \rho(g) d_0(s, a)$

Contrastive RL

Recall that CL solution:

$$f^*(s,a,g) = \log \frac{P_+((s,a,g))}{P_-((s,a,g))} = \log \left(\frac{p^{\pi(\cdot|\cdot)}(g \mid s,a)}{\rho(g)} \right) = \log \left(\frac{Q^\pi(s,a,g)}{\rho(g)} \right).$$

So

$$Q^\pi(s,a,g) = e^{f^*(s,a,g)} \rho(g).$$

Contrastive RL

Recall that CL solution:

$$f^*(s,a,g) = \log \frac{P_+((s,a,g))}{P_-((s,a,g))} = \log \left(\frac{p^{\pi(\cdot|\cdot)}(g \mid s,a)}{\rho(g)} \right) = \log \left(\frac{Q^\pi(s,a,g)}{\rho(g)} \right).$$

So

$$Q^\pi(s,a,g) = e^{f^*(s,a,g)} \rho(g).$$

$$\arg\max_a f^*(s,a,g) \iff \arg\max_a Q^\pi(s,a,g)$$

Contrastive RL

Recall that CL solution:

$$f^*(s,a,g) = \log \frac{P_+((s,a,g))}{P_-((s,a,g))} = \log \left(\frac{p^{\pi(\cdot|\cdot)}(g \mid s,a)}{\rho(g)} \right) = \log \left(\frac{Q^\pi(s,a,g)}{\rho(g)} \right).$$

So

$$Q^\pi(s,a,g) = e^{f^*(s,a,g)} \rho(g).$$

$$\arg\max_a f^*(s,a,g) \iff \arg\max_a Q^\pi(s,a,g)$$

Note that since π is a random goal policy,

$$Q^\pi(s,a,g) \neq Q^*_g(s,a)$$

where $Q^*_g(s,a)$ is the Q-function under the policy for achieving goal g.

How to Parametrize f?

Log linear parametrization [5, 2]:

$$f(s, a, g) = \phi(s, a)^\top \psi(g).$$

Implicitly learns representations with nice properties:

- Emergent exploration [3].
- Planning via interpolation [1].
- Triangle inequalities [4].

Note: $\phi(s, a)$ and $\psi(g)$ are generally their own neural networks and highly non-linear.

Representation learning ✓

Q-function as distribution

Q function as occupancy measure

The goal-conditioned Q function is a distribution! Namely,

$$Q^\pi(s, a, g) = p^{\pi(\cdot|\cdot)}(s_+ = g \mid s, a).$$

Proof.

$$\begin{aligned} p_t^{\pi(\cdot|\cdot)}(s_t = g) &= \mathbb{E}_{p_{t-1}^{\pi(\cdot|\cdot)}(s_{t-1})\pi(a_{t-1}|s_{t-1})}\left[p(s_t = g \mid s_{t-1}, a_{t-1})\right] \quad \text{(Markov)} \\ &= \mathbb{E}\left[p(s_t = g \mid s_{t-1}, a_{t-1})\right]. \end{aligned}$$

□

Q-function as distribution

Q function as occupancy measure

The goal-conditioned Q function is a distribution! Namely,

$$Q^{\pi}(s, a, g) = p^{\pi(\cdot|\cdot)}(s_+ = g \mid s, a).$$

Proof.

$$\begin{aligned} p_t^{\pi(\cdot|\cdot)}(s_t = g) &= \mathbb{E}_{p_{t-1}^{\pi(\cdot|\cdot)}(s_{t-1})\pi(a_{t-1}|s_{t-1})}\left[p(s_t = g \mid s_{t-1}, a_{t-1})\right] \quad \text{(Markov)} \\ &= \mathbb{E}\left[p(s_t = g \mid s_{t-1}, a_{t-1})\right]. \end{aligned}$$

Substituting,

$$p^{\pi(\cdot|\cdot)}(s_+ = g) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t \mathbb{E}\left[p(s_{t+1} = g \mid s_t, a_t)\right].$$

□

Q-function as distribution

Proof.

$$
\begin{aligned}
p^{\pi(\cdot|\cdot)}(s_+ = g) &= (1-\gamma)\sum_{t=0}^{\infty}\gamma^t \mathbb{E}\left[p(s_{t+1} = g \mid s_t, a_t)\right] \\
&= \mathbb{E}\left[(1-\gamma)\sum_{t=0}^{\infty}\gamma^t p(s_{t+1} = g \mid s_t, a_t)\right] \\
&= \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t r_g(s_t, a_t)\right].
\end{aligned}
$$

Q-function as distribution

Proof.

Thus,

$$p^{\pi(\cdot|\cdot)}(s_+ = g \mid s, a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_g(s_t, a_t) \mid s_0 = s, a_0 = a\right]$$
$$= Q^{\pi}(s, a, g)$$

□

References

Benjamin Eysenbach, Vivek Myers, Ruslan Salakhutdinov, and Sergey Levine.
Inference via interpolation: Contrastive representations provably enable planning and inference.
Advances in Neural Information Processing Systems, 37:58901–58928, 2024.

Benjamin Eysenbach, Tianjun Zhang, Sergey Levine, and Russ R Salakhutdinov.
Contrastive learning as goal-conditioned reinforcement learning.
Advances in Neural Information Processing Systems, 35:35603–35620, 2022.

Grace Liu, Michael Tang, and Benjamin Eysenbach.
A single goal is all you need: Skills and exploration emerge from contrastive rl without rewards, demonstrations, or subgoals.
arXiv preprint arXiv:2408.05804, 2024.

Vivek Myers, Chongyi Zheng, Anca Dragan, Sergey Levine, and Benjamin Eysenbach.
Learning temporal distances: Contrastive successor features can provide a metric structure for decision-making.
arXiv preprint arXiv:2406.17098, 2024.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals.
Representation learning with contrastive predictive coding.
arXiv preprint arXiv:1807.03748, 2018.