CS-566 Deep Reinforcement Learning

Improved Policy-Based Learning II: Trust Region Methods

Nazar Khan
Department of Computer Science
University of the Punjab

Information Theory

Information theory studies:

- how much information is present in distributions;
- how to compare different probability distributions.

We begin with the notion of *information* of an event.

Information of an Event

The information I of observing event x from distribution $p(X)$:

$$I(x) = -\log p(x).$$

Intuition:

- High probability $\Rightarrow$ low information gain
- Low probability $\Rightarrow$ high information gain

Extreme cases:

- $p(x) = 0 \; : \; I(x) = -\log 0 = \infty$
- $p(x) = 1 \; : \; I(x) = -\log 1 = 0$

Information and Uncertainty

Information can be interpreted as the reduction of uncertainty after observing x.

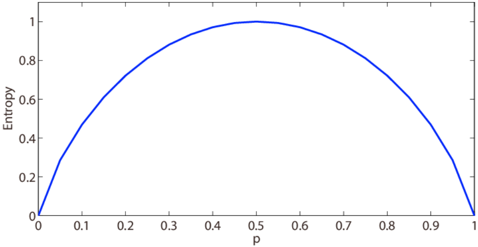


Figure: Entropy of a binary variable as a function of $p(x=1)$.

Entropy

Entropy $H[p]$ of a discrete distribution $p(X)$:

$$
\begin{aligned}
H[p] &= \mathbb{E}_{X\sim p}[I(X)] \\
&= \mathbb{E}_{X\sim p}[-\log p(X)] \\
&= -\sum_{x} p(x) \log p(x).
\end{aligned}
$$

Units:

- log base 2 $\Rightarrow$ entropy in *bits*
- log base e $\Rightarrow$ entropy in *nats*

Interpretation of Entropy

Entropy measures the **uncertainty** or **spread** of a distribution.
Example: binary variable $X \in \{0, 1\}$

- $p(x = 1) = 0$ or $1 \Rightarrow$ entropy $= 0$ (no uncertainty)
- $p(x = 1) = 0.5 \Rightarrow$ entropy is maximal

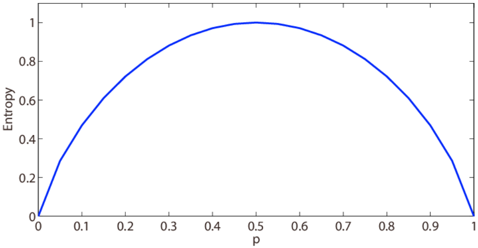


Figure: Entropy peaks at $p(1) = 0.5$, minimal at extremes.

Example: Computing Entropy

Example

For distribution $p = [0.2, 0.3, 0.5]$:

$$\begin{aligned} H[p] &= -0.2 \ln 0.2 - 0.3 \ln 0.3 - 0.5 \ln 0.5 \\ &= 1.03 \text{ nats} \end{aligned}$$

Cross-Entropy

Cross-entropy between distributions $p(X)$ and $q(X)$:

$$
\begin{aligned}
H[p, q] &= \mathbb{E}_{X \sim p}[-\log q(X)] \\
&= -\sum_{x} p(x) \log q(x).
\end{aligned}
$$

Connection to ML: Maximum likelihood training = minimizing cross-entropy between:

- data distribution p
- model distribution q

Kullback-Leibler Divergence

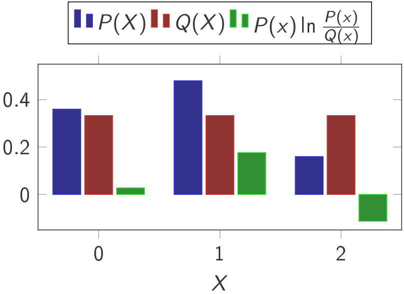
KL divergence (relative entropy) between p and q:

$$D_{\mathrm{KL}}[p\|q] = \mathbb{E}_{X\sim p}\left[-\log\frac{q(X)}{p(X)}\right]$$
$$= -\sum_x p(x)\log\frac{q(x)}{p(x)} = \sum_x p(x)\log\frac{p(x)}{q(x)}.$$

- **Interpretation:** A measure of how different two distributions are.
- $D_{\mathrm{KL}}[p\|q] \geq 0$
- Not symmetric: $D_{\mathrm{KL}}[p\|q] \neq D_{\mathrm{KL}}[q\|p]$

KL-Divergence
Discrete Example

	$X=0$	$X=1$	$X=2$
$P(x)$	$\frac{9}{25}$	$\frac{12}{25}$	$\frac{4}{25}$
$Q(x)$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$
$\frac{P(x)}{Q(x)}$	1.08	1.44	0.48
$\ln \frac{P(x)}{Q(x)}$	0.077	0.365	−0.734
$P(x) \ln \frac{P(x)}{Q(x)}$	0.028	0.175	−0.117



$$D_{\mathrm{KL}}(P \| Q) = \sum_x P(x) \ln \frac{P(x)}{Q(x)}$$

$$= \frac{9}{25} \ln \frac{9/25}{1/3} + \frac{12}{25} \ln \frac{12/25}{1/3} + \frac{4}{25} \ln \frac{4/25}{1/3} \quad = 0.0853$$

KL as Entropy + Cross-Entropy

We can rewrite KL-divergence using entropy and cross-entropy:

$$D_{\mathrm{KL}}[p\|q] = \underbrace{\sum_x p(x)\log p(x)}_{-H[p]} - \underbrace{\sum_x p(x)\log q(x)}_{-H[p,q]}$$
$$= H[p,q] - H[p].$$

Summary:

- Entropy: uncertainty of a single distribution
- Cross-entropy: expected code length using incorrect model
- KL-divergence: difference between distributions

These appear throughout machine learning and RL loss functions.

Trust Region Optimization

Goal: Reduce variance and instability in policy gradient updates.

Problem:

- Simply increasing learning rate or step size $\rightarrow$ instability
- Large updates may collapse performance
- Need large improvements without deviating too far from the old policy

Trust Region idea:

- Constrain the update size during optimization
- Adaptively expand/shrink allowed region based on update quality

Trust Region Policy Optimization (TRPO)

Concept:

- Maximize improvement in policy
- Prevent policy from moving too far from previous one
- Reduce update variance and prevent collapse

Objective:

$$J(\theta) = \mathbb{E}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \cdot A_t \right]$$

Constraint (trust region):

$$\mathbb{E}_t \left[\text{KL}(\pi_{\theta_{\text{old}}} || \pi_\theta) \right] \leq \delta$$

Intuition: Take the *largest safe policy step*.

TRPO Notes and Impact

Key points:

- Uses KL divergence to limit policy change
- Uses second-order optimization (complex)
- Stable and reliable, good for large problems

Applications:

- Robotic control (swimming, hopping, walking)
- Atari games

Downside: Algorithmically complex (requires second-order methods)

Implementations:

- OpenAI Spinning Up
- Stable Baselines

Proximal Policy Optimization (PPO)

Motivation:

- Keep benefits of TRPO
- Remove complexity (no second-order derivatives)
- Make training faster and easier

PPO Variants:

- **PPO-Penalty:** Penalizes KL divergence in objective
- **PPO-Clip:** Clipping mechanism to limit policy change

Concept: Take a large step, but **clip** if too far from old policy

PPO-Clip Objective

Clipped surrogate objective:

$$J(\theta) = \mathbb{E}_t\Big[\min(r_t(\theta)A_t,\ \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)A_t)\Big]$$

where:

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$$

Effect:

- If update too large $\rightarrow$ clipped
- Controls destructive updates without explicit trust-region compute

TRPO vs PPO

Feature	TRPO	PPO
Stability	High	High
Complexity	High (2nd order)	Low
Compute cost	High	Moderate
Constraint	Hard KL bound	Clipping / soft penalty
Use case	Research stability	Practical training default

Both are on-policy methods.

Outcome: PPO became standard baseline for deep RL tasks