ECE433/COS435 Reinforcement Learning

Assignment 3: Reward Design

Spring 2026

Fill me in

Your name here.

March 16, 2026

Collaborators

Fill me in

Please fill in the names and NetIDs of your collaborators in this section. Make sure you each submit separately, but feel free to work together and submit the same answers as long as you put your names together here.

Instructions

Writeups should be typesetted in Latex and submitted as PDF. **Please submit the asked-for snippet and answer in the solutions box as part of your writeup. Attach code as a .zip file *with the exact structure as we've distributed the zip*, we will run the code against standardized unit tests.**

Grading. The problem set will be graded out of 50 points, and the coding assignment will also be graded out of 50 points, making the total score 100 points.

Question 0 (0 points). Feedback

How many hours did you spend on this homework? Please fill in the solution after you've done all the questions.

Solution

Your solution here...

Problem Set (50 points)

Question 1: Potential-Based Reward Shaping (15 points)

Ng, Harada, and Russell (1999) proved that *potential-based reward shaping* preserves the set of optimal policies. Consider an MDP $M = (S, A, T, R, \gamma)$ and a shaped MDP $M' = (S, A, T, R', \gamma)$ where

$$R'(s, a, s') = R(s, a, s') + F(s, s'), \qquad F(s, s') = \gamma\,\Phi(s') - \Phi(s)$$

for some **potential function** $\Phi : S \to \mathbb{R}$.

(a) **(10 points)** Prove that any optimal policy π^* under R is also optimal under R'.

Solution

Your solution here...

Question 2: Reward Engineering vs. Exploration (15 points)

Sparse reward problems can be addressed in (at least) two fundamentally different ways:

(i) **Reward engineering**: Designing better reward functions (shaping, curriculum, learned reward models).

(ii) **Exploration methods**: Improving how the agent explores (curiosity-driven exploration, count-based bonuses, Go-Explore, hindsight experience replay).

(a) **(5 points)** Which approach do you think is more promising as a general research direction? Make an argument for your chosen side, considering:

- Scalability to complex tasks
- Robustness (how likely is each to introduce new failure modes?)
- Practicality (which requires less domain expertise?)

There is no single right answer—we are looking for thoughtful reasoning.

Solution

Your solution here...

(b) **(5 points)** Can the two approaches be combined? Describe a concrete scenario where using both reward shaping *and* improved exploration together would be more effective than either alone.

Solution

Your solution here...

Question 3: What Can Go Wrong with Human Feedback? (15 points)

Reinforcement Learning from Human Feedback (RLHF) trains a reward model $\hat{R}_\phi$ from human preference comparisons and then optimizes a policy π_θ against $\hat{R}_\phi$ (typically using PPO with a KL penalty against a reference policy).

(a) **(5 points)** Gao et al. (2023) showed that as you optimize more aggressively against $\hat{R}_\phi$, the *proxy* reward keeps increasing but the *true* reward (as judged by humans) eventually *decreases*. This is sometimes called "Goodhart's Law" applied to RL.

- Explain why this happens. How is this analogous to overfitting in supervised learning?
- How might the KL penalty $\beta \cdot D_{\mathrm{KL}}(\pi_\theta \| \pi_{\mathrm{ref}})$ help mitigate this?

Solution

Your solution here...

(b) **(5 points)** Give a **concrete, realistic** example of reward hacking in an RLHF-trained language model. Your example should describe:

- A plausible pattern in the human preference data
- How a reward model might learn an incorrect generalization from this pattern
- How the policy could exploit this incorrect generalization
- Why the resulting behavior would be undesirable

Solution

Your solution here...

(c) **(5 points)** Beyond reward overoptimization, what are two other fundamental challenges with using human feedback to train AI systems? For each, explain the problem and sketch a potential mitigation.

Solution

Your solution here...

Coding Assignment (50 points)

In this assignment you will use the `stable_baselines3` implementation of PPO to train an agent on `MountainCarContinuous-v0` with different reward functions. You will experience firsthand how reward design affects learning—and how naive reward shaping can backfire.

See `hw3.py`. Install dependencies: `pip install stable-baselines3 gymnasium`.

Background: MountainCarContinuous-v0

The car starts in a valley at $x \approx -0.5$ and must reach the flag at $x \geq 0.45$. The physics use gravity $g = 0.0025$ and the action is a force in $[-1, 1]$.

Part 1: Sparse Reward Baseline (10 points)

We provide `SparseMountainCarWrapper`, which strips all default rewards and only gives +1 when the car reaches the flag. Run PPO with this wrapper and confirm that it fails to learn. Where does the reward plateau? Implement a modified reward, RewardShaping1, that is just adding a bonus for distance to the goal state (going right). Why does distance to goal not help in this case? How might this violate reward shaping assumptions?

Solution

Discuss here briefly.

Part 2: Your Reward Shaping (40 points)

Implement 3 other versions of `RewardShapingWrapper` with your own reward functions, try to come up wtih a better reward function that will get you the most true rewards (steps to the top of the hill).

Solution

Include a comparison plot of all four reward shaping functions you came up with and a brief (3–5 sentence) analysis of the best true reward. You should have 4 lines with confidence intervals, one for each reward modification you made measuring the true reward against the sparse baseline.