Can Evolution Strategies Solve Sparse-Reward Tasks?

Evaluating Gradient-Free Optimization in Reinforcement Learning

Evolution Strategies for Sparse-Reward GridWorld Navigation (Problem Overview)

• **Goal**: Train a neural network agent to navigate an 8×8 GridWorld with obstacles

• **Task setup:**

Start position → bottom-left corner

Goal position → top-right corner

- Environment contains 8 randomly placed obstacles
- State representation: One-hot encoded grid position
- Action space: Up, down, left, right

- **Reward structure**

+1 → reach goal

−0.1 → hit obstacle

0 → otherwise (sparse rewards)

- **Policy model**

2-layer MLP with 64 hidden units

8,580 parameters

Outputs probability distribution over actions

- **Key challenge**

Learning effective navigation under sparse reward signals

Sparse reward environments are difficult to learn:

Learning signal is mostly zero:

- Agent must discover goal through random exploration
- Standard policy gradients struggle

Limitations of traditional methods:

- REINFORCE / PPO rely on informative reward signals
- High variance and slow convergence in sparse settings

Evolution Strategies (ES) as an alternative:

- Gradient-free optimization
- Uses parameter perturbations instead of backpropagation through environment
- Evaluates full episode performance (fitness)
- Effective in non-differentiable or sparse-signal problems

Motivation:

- Test ES effectiveness vs PPO in sparse reward navigation
- Inspired by Salimans et al. (2017)

Implementation

- Evolutionary Strategies Setup:
  - Vanilla ES (parameter perturbation method)
  - 50 perturbations per iteration
  - Noise scale (sigma) = 0.1
  - Learning rate = 0.05
  - 80 training iterations
  - Max 50 steps per episode
  - 5 evaluation episodes per perturbation (to reduce noise from random obstacles)
- Stability and variance reduction
  - Standardized fitness values each iteration
  - Averaged multiple rollouts per perturbation
  - Observed that single-rollout estimates were too noisy

- PPO Baseline:
  - PPO-Clip with GAE
  - 128 rollout steps per iteration, 200 iterations
  - Adam optimizers (policy: 3e-4, value: 1e-3)
  - Clip $\epsilon = 0.2$, $\gamma = 0.99$, $\lambda = 0.95$
  - Advantage normalization + entropy bonus (0.01)
  - Gradient clipping (max norm = 0.5)
  - Evaluation: deterministic policy, 10 episodes

Results and Observations

- Results (8×8 grid, 8 obstacles)
  - With reward shaping, both ES and PPO reached 100% success rate
    Training was lightweight on CPU (fast iterations, low memory)
  - Learned policies still succeeded when evaluated with sparse rewards only (goal reward)
- What mattered for ES
  - Noise scale was sensitive (sigma choice strongly affected stability)
  - Single-rollout fitness per perturbation was too noisy
  - Averaging multiple rollouts per perturbation was necessary
  - Fitness standardization was important for stable updates
- What mattered for PPO (given our implementation)
  - PPO learns from fixed-length rollouts (128 steps/iter), not full-episode returns
  - Advantage normalization + entropy bonus helped exploration
  - Gradient clipping kept updates stable
- Limitations / why ES vs PPO didn't separate
  - Reward shaping likely made the task too easy
  - 8×8 grid is small enough that both methods can quickly find a working strategy
  - Need harder settings (pure sparse training, larger grids, more obstacles, multi-stage tasks) to meaningfully differentiate performance