Advanced Topics in Deep Reinforcement Learning

Exploration

Why Explore?

Exploration: (Strategically) deviating from the policy to improve learning later on

Why Explore?

Exploration: (Strategically) deviating from the policy to improve learning later on

- Find important states (e.g. goals)

Why Explore?

Exploration: (Strategically) deviating from the policy to improve learning later on

- Find important states (e.g. goals)
- Check alternative paths

Why Explore?

Exploration: (Strategically) deviating from the policy to improve learning later on

- Find important states (e.g. goals)
  Increase state coverage
- Check alternative paths

Why Explore?

Exploration: (Strategically) deviating from the policy to improve learning later on

- Find important states (e.g. goals)
  Increase state coverage
- Check alternative paths
  Avoid local minima

Why Explore?

Exploration: (Strategically) deviating from the policy to improve learning later on

- Find important states (e.g. goals)
  Increase state coverage
- Check alternative paths
  Avoid local minima

=> Exploration increases data quality

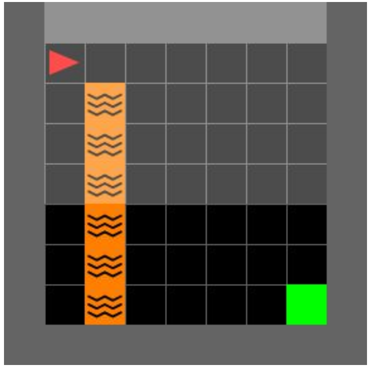
What Is High Data Quality?

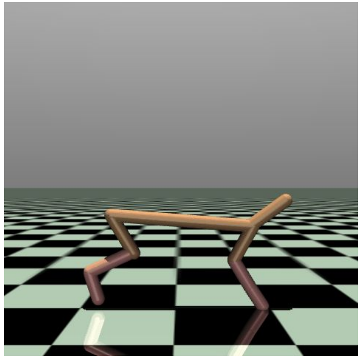
- Data quality is problem dependent
  - State coverage is important when learning policies with many steps
  - Avoiding local minima is relevant in most reward structures
  - The degree of both varies (tradeoff with efficiency)

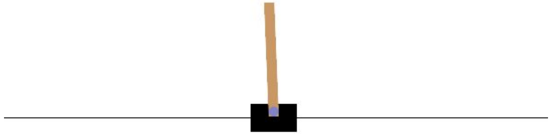
What Is High Data Quality?

- Data quality is problem dependent
  - State coverage is important when learning policies with many steps
  - Avoiding local minima is relevant in most reward structures
  - The degree of both varies (tradeoff with efficiency)

Examples:







Strategies for Exploration

- Randomness (e.g. action noise, e-greedy)

Strategies for Exploration

- Randomness (e.g. action noise, e-greedy)
  Easy to implement, inefficient

Strategies for Exploration

- Randomness (e.g. action noise, e-greedy)
  Easy to implement, inefficient
- Random sequences (e.g. ez-greedy)

Strategies for Exploration

- Randomness (e.g. action noise, e-greedy)
  Easy to implement, inefficient
- Random sequences (e.g. [ez-greedy])
  Easy to implement, diverse sequences, inefficient, broad state coverage only

Strategies for Exploration

- Randomness (e.g. action noise, e-greedy)
  Easy to implement, inefficient
- Random sequences (e.g. ez-greedy)
  Easy to implement, diverse sequences, inefficient, broad state coverage only
- State novelty (e.g. RND, NovelD, E3B, count-based methods)

Strategies for Exploration

- Randomness (e.g. action noise, e-greedy)
  Easy to implement, inefficient
- Random sequences (e.g. ez-greedy)
  Easy to implement, diverse sequences, inefficient, broad state coverage only
- State novelty (e.g. RND, NovelD, E3B, count-based methods)
  Maximizes state coverage, hard to measure, computational overhead

Strategies for Exploration

- Randomness (e.g. action noise, e-greedy)
  Easy to implement, inefficient
- Random sequences (e.g. ez-greedy)
  Easy to implement, diverse sequences, inefficient, broad state coverage only
- State novelty (e.g. RND, NovelD, E3B, count-based methods)
  Maximizes state coverage, hard to measure, computational overhead
- Intrinsic rewards (e.g. ICM, RIDE, meta-learned rewards)

Strategies for Exploration

- Randomness (e.g. action noise, e-greedy)
  Easy to implement, inefficient
- Random sequences (e.g. ez-greedy)
  Easy to implement, diverse sequences, inefficient, broad state coverage only
- State novelty (e.g. RND, NovelD, E3B, count-based methods)
  Maximizes state coverage, hard to measure, computational overhead
- Intrinsic rewards (e.g. ICM, RIDE, meta-learned rewards)
  Can eliminate local minima, hard to construct, often domain specific

Strategies for Exploration

- Randomness (e.g. action noise, e-greedy)
  Easy to implement, inefficient
- Random sequences (e.g. ez-greedy)
  Easy to implement, diverse sequences, inefficient, broad state coverage only
- State novelty (e.g. RND, NovelD, E3B, count-based methods)
  Maximizes state coverage, hard to measure, computational overhead
- Intrinsic rewards (e.g. ICM, RIDE, meta-learned rewards)
  Can eliminate local minima, hard to construct, often domain specific
- Model/ensemble disagreement (e.g. for DQN and PPO)

Strategies for Exploration

- Randomness (e.g. action noise, e-greedy)
  Easy to implement, inefficient
- Random sequences (e.g. ez-greedy)
  Easy to implement, diverse sequences, inefficient, broad state coverage only
- State novelty (e.g. RND, NovelD, E3B, count-based methods)
  Maximizes state coverage, hard to measure, computational overhead
- Intrinsic rewards (e.g. ICM, RIDE, meta-learned rewards)
  Can eliminate local minima, hard to construct, often domain specific
- Model/ensemble disagreement (e.g. for DQN and PPO)
  Exploration around current policy, high computational overhead, not ideal for state coverage

Exploration via Randomness

- Option 1: take random action with a certain probability
- Option 2: add randomly sampled noise to prediction

Exploration via Randomness

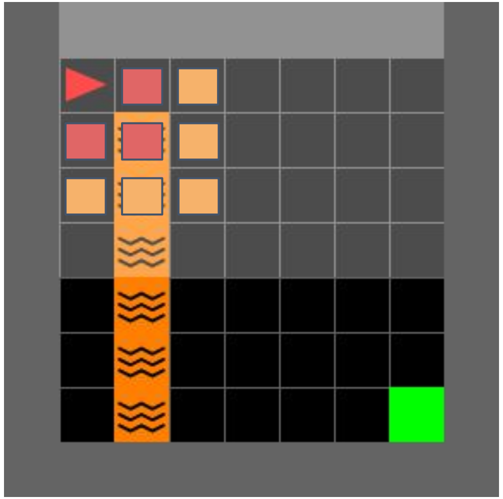
- Option 1: take random action with a certain probability
- Option 2: add randomly sampled noise to prediction

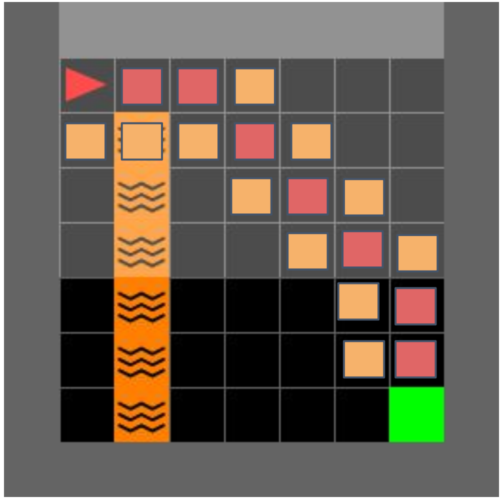
Either way, noise should usually decrease during training

Exploration via Randomness

- Option 1: take random action with a certain probability
- Option 2: add randomly sampled noise to prediction

Either way, noise should usually decrease during training

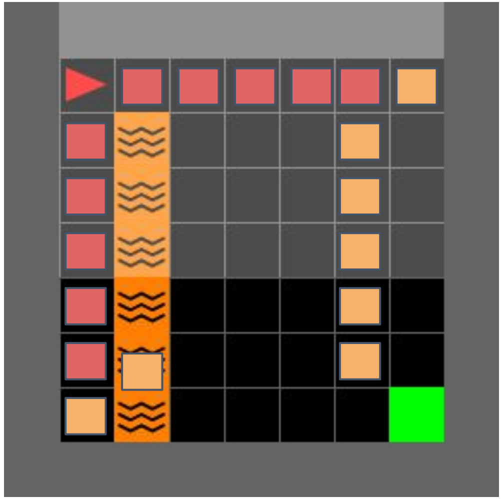


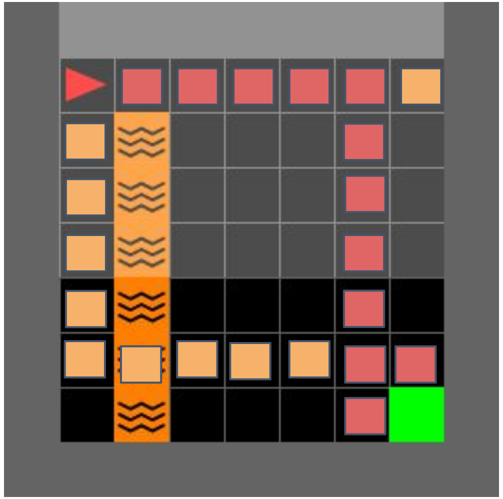


Exploration via Random Sequences

Instead of sampling a step sample a full sequence of steps

Example assuming we sample the same action for 5 steps:





Exploration via State Novelty

Why look for novelty?

- Exploring the whole state space
- Either focus on good performance or states we haven’t seen very often

Exploration via State Novelty

Why look for novelty?

- Exploring the whole state space
- Either focus on good performance or states we haven't seen very often

Why does it work?

- Avoids spending time in mediocre states
- Still helps cover the state space

Measuring State Novelty

- Counting state visitation (limited to discrete state spaces)

Measuring State Novelty

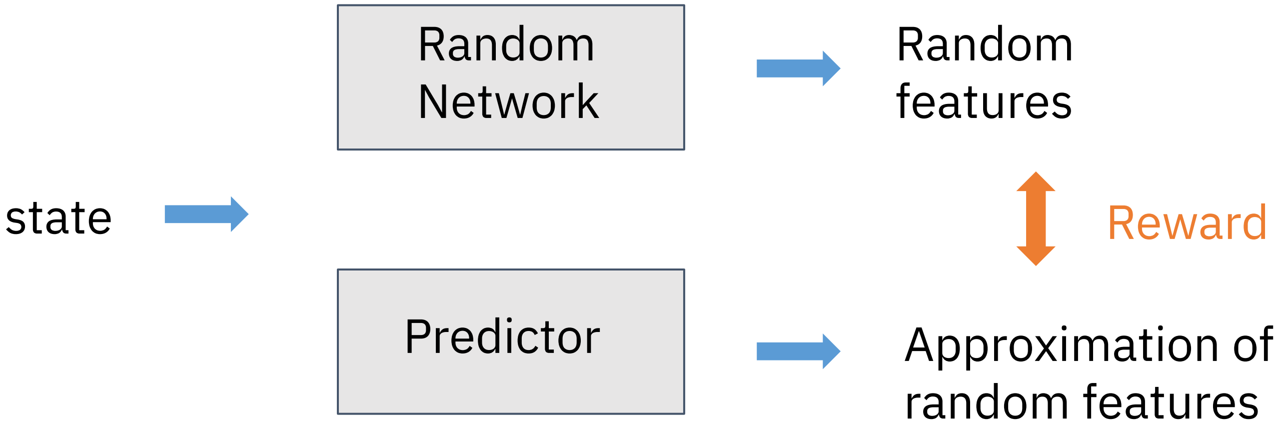
- Counting state visitation (limited to discrete state spaces)
- Approximating counts (in continuous state spaces)

Measuring State Novelty

- Counting state visitation (limited to discrete state spaces)
- Approximating counts (in continuous state spaces)
- Prediction-error based novelty

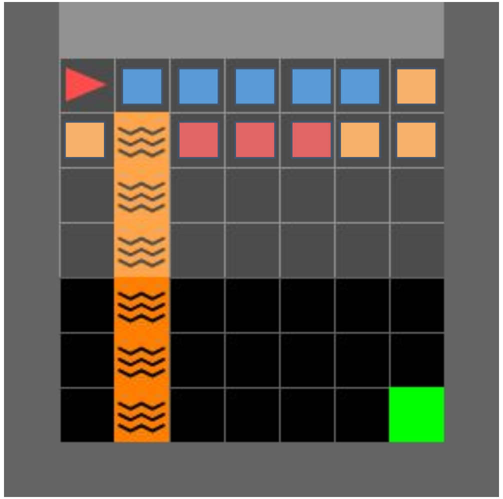
Measuring State Novelty

- Counting state visitation (limited to discrete state spaces)
- Approximating counts (in continuous state spaces)
- Prediction-error based novelty



Exploration via State Novelty

Assume we learned to walk along the border to the goal:



Exploration via Intrinsic Rewards

We alter the rewards instead of the policy:

- The agent will learn exploration through the rewards
- No need to choose an explicit exploration strategy
- Algorithm-agnostic
- Can encode very specific exploration behavior

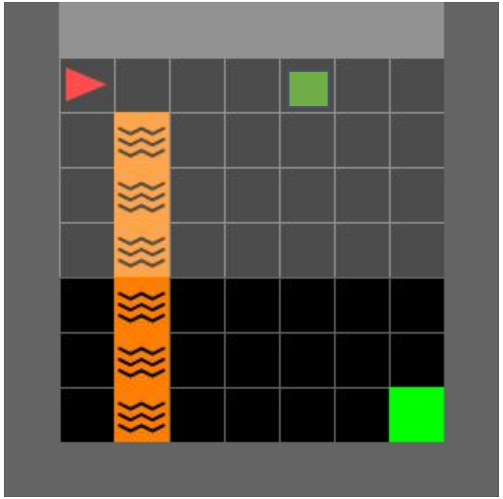
Designing Intrinsic Rewards

- Novelty-based methods often use rewards

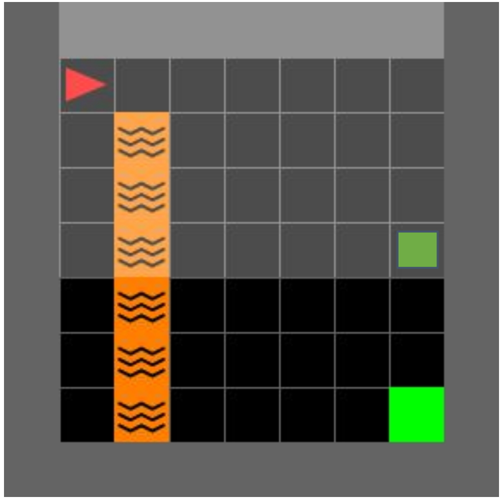
Designing Intrinsic Rewards

- Novelty-based methods often use rewards
- Rewards via sub-goals (e.g. AMIGo)

Teacher proposes sub-goal

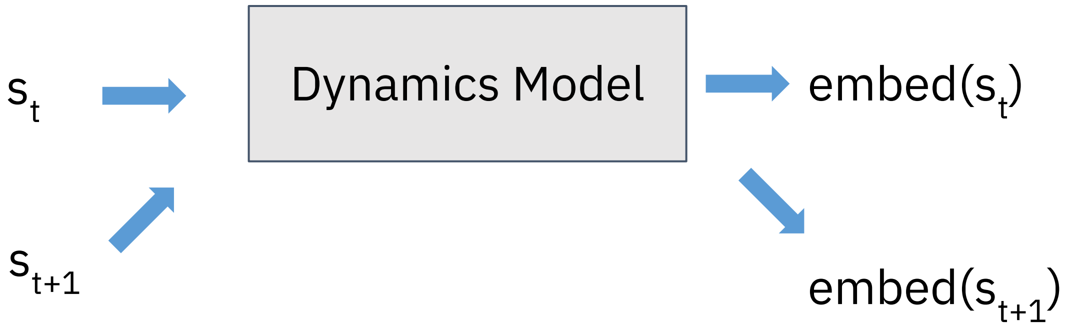


Difficulty increases over time



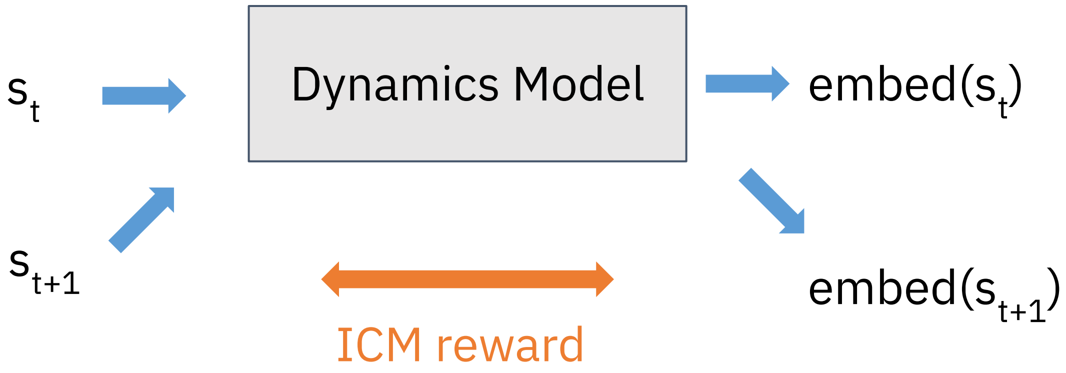
Designing Intrinsic Rewards

- Novelty-based methods often use rewards
- Rewards via sub-goals (e.g. AMIGo)
- Impact on learning (e.g. ICM, RIDE)



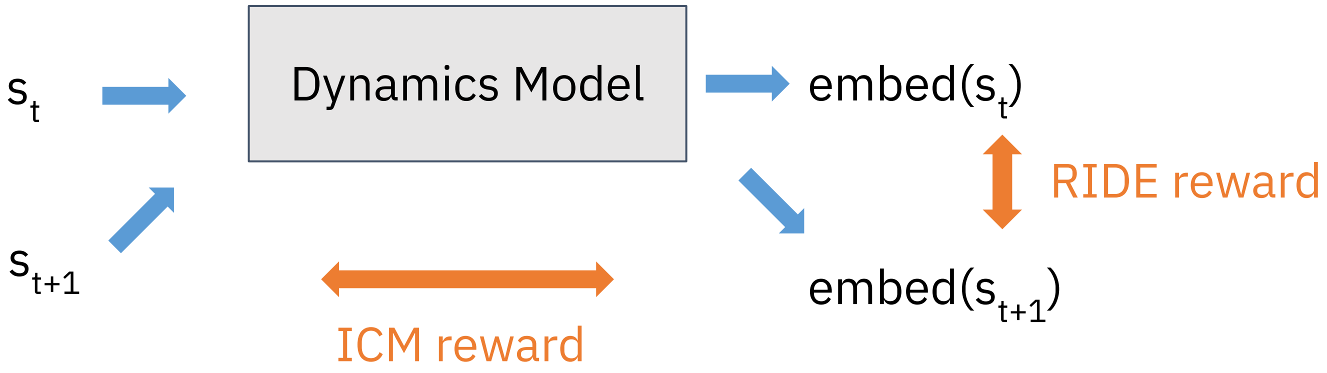
Designing Intrinsic Rewards

- Novelty-based methods often use rewards
- Rewards via sub-goals (e.g. AMIGo)
- Impact on learning (e.g. ICM, RIDE)



Designing Intrinsic Rewards

- Novelty-based methods often use rewards
- Rewards via sub-goals (e.g. AMIGo)
- Impact on learning (e.g. ICM, RIDE)



Designing Intrinsic Rewards

- Novelty-based methods often use rewards
- Rewards via sub-goals (e.g. AMIGo)
- Impact on learning (e.g. ICM, RIDE)
- Meta-learning reward functions

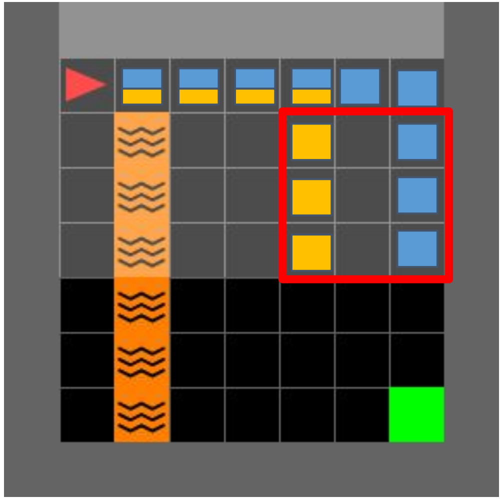
Designing Intrinsic Rewards

- Novelty-based methods often use rewards
- Rewards via sub-goals (e.g. AMIGo)
- Impact on learning (e.g. ICM, RIDE)
- Meta-learning reward functions

Designing intrinsic rewards is very hard and often domain-specific!

Exploration via Ensembles

Idea: if we train multiple policies on the same data, disagreement between the predictions can be a useful signal that data is insufficient



Region of Disagreement

My Understanding of RL Algorithms

- ❑ I understand what exploration accomplishes
- ❑ I can describe at least 3 ways of exploring
- ❑ I could implement at least one exploration algorithm

- ❑ I can describe each exploration idea
- ❑ I know the pros and cons of different exploration methods

- ❑ I can describe one algorithm per idea
- ❑ I can compare exploration algorithms
- ❑ I can explain which idea would work for a given problem