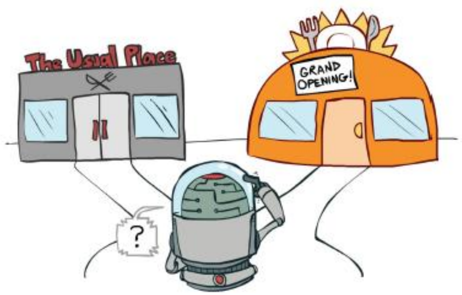
ECE7121 Learning-based control – 2025 Fall

Exploration

INHA UNIVERSITY

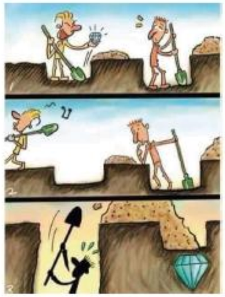
Recall: exploration

- Exploration vs Exploitation dilemma
  - The best long-term strategy may involve short-term sacrifices
  - This is not a problem unique to RL; it is a fundamental issue in the decision making of any intelligent agent.



Restaurant Selection

exploit:
go to your favorite restaurant
vs.
explore:
try something new



Oil Drilling

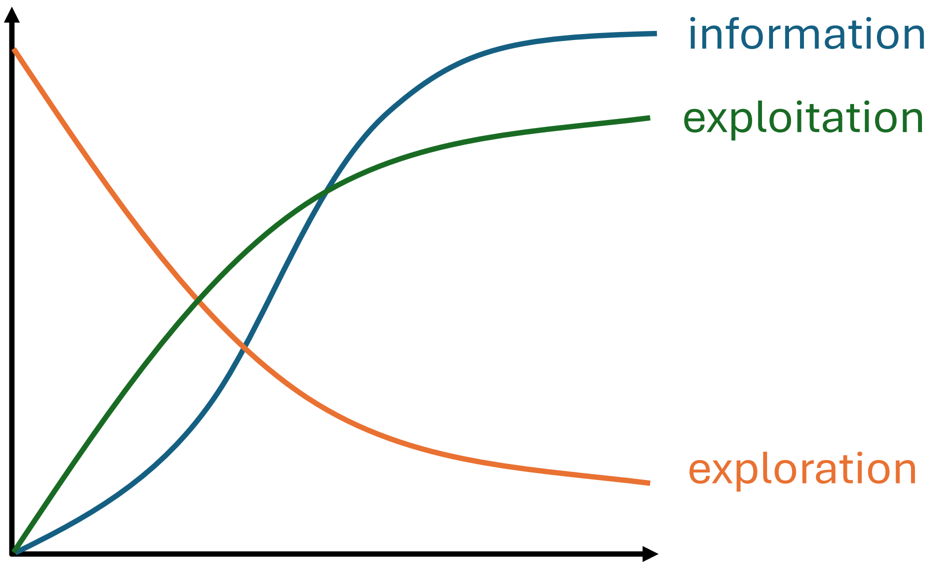
drill at the best-known location
vs.
drill at a new location

Online Ad Placement

show most successful ads
vs.
show a different random ad

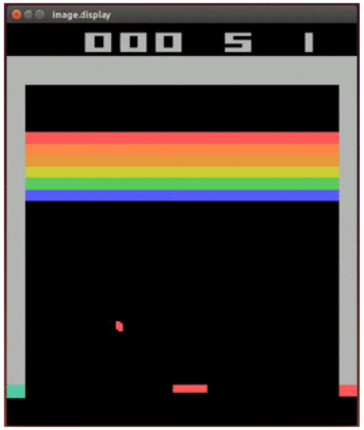
Recall: exploration

- $\epsilon - greedy$ algorithm
  - occasionally try something suboptimal (random)



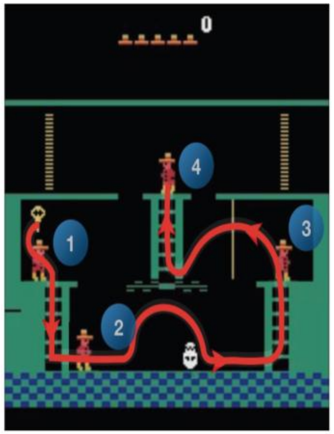
Motivation

- $\epsilon - greedy$ algorithm does not work well
  - there are many hard exploration tasks
  - put yourself in the algorithm's shoes
  - In Atari game
    - Breakout vs. Montezuma's revenge



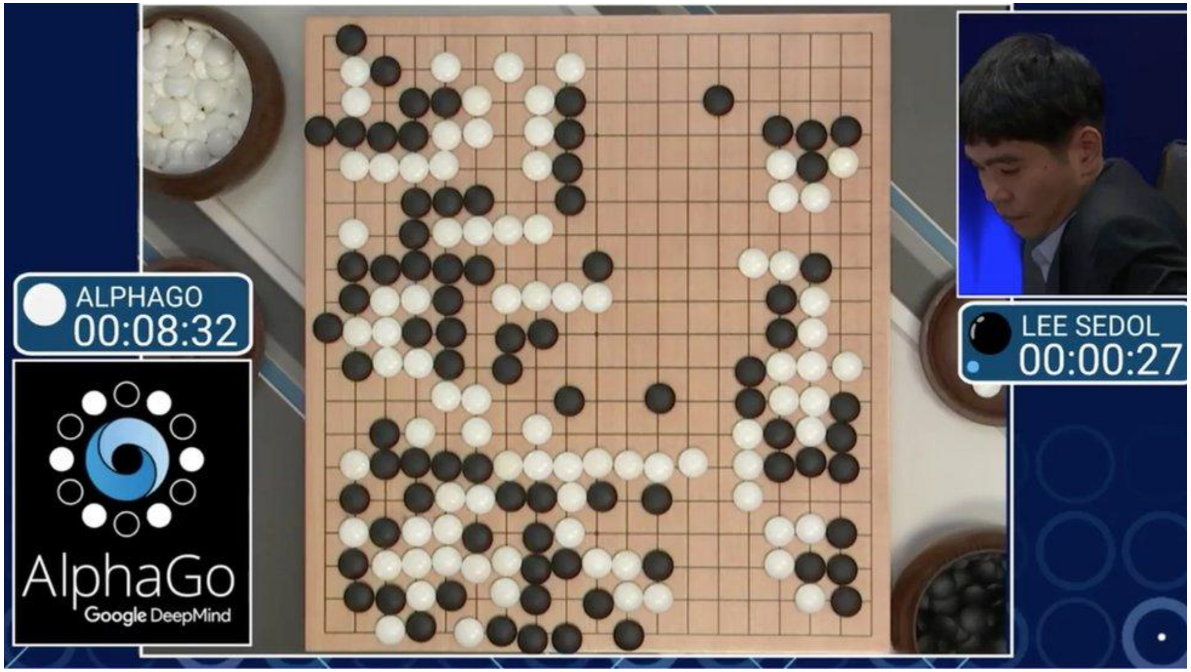
break a brick = +1

Get a key = +100
Open a door = +300
Find a treasure = +800



Motivation

> Go
  - state space is estimated be 10^{170}
  - we can't visit all the possible states



win = +1
lose = -1

Motivation

> How can an agent discover high-reward states?
  - this may require a long-term and complex behaviors, which could be not rewarding along the way

> How can an agent decide whether to explore or exploit?

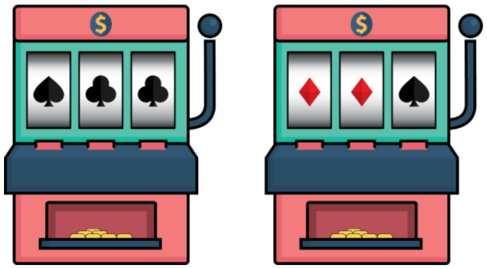
> How can we derive an optimal exploration strategy?

multi-armed bandits (1-step stateless RL problems)	contextual bandits (1-step RL problems)	small, finite MDPs (e.g., tractable planning, model-based RL setting)	large, infinite MDPs, continuous spaces

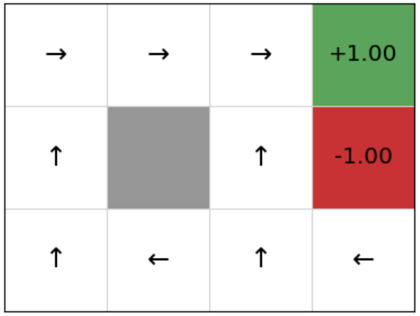
theoretically tractable ⟷ theoretically intractable

Exploration

> Multi-armed bandits
  - can be formalized as POMDP



> Small and finite MDPs
  - can be framed as a Bayesian model



> Large and infinite MDPs
  - optimal methods do not work here
  - take inspiration from the small problems

Classic exploration

> These approaches came from the multi-armed bandit
  - Epsilon-greedy: the agent does random exploration occasionally with probability ϵ

  - Boltzmann exploration: the agent draws actions from a Boltzmann distribution (softmax) over the learned $\hat{Q}$ values

  - Upper confidence bounds: the agent selects the greediest action to minimize the upper confidence bound $\hat{Q}(s, a) + U(s, a)$
    - U is reversely proportional to how many times action a has been taken

  - Thompson sampling: the agent keeps track of a belief over the probability of optimal actions and samples from this distribution

Classic exploration

> Epsilon-greedy: the agent does random exploration occasionally with probability ϵ
  - due to randomness, we end up exploring bad actions all over again
  - it is a binary decision to choose the best action $(1 - \epsilon)$ or random action (ϵ)
  - why not ranking the actions and choose the action accordingly?

> Boltzmann exploration: the agent draws actions from a Boltzmann distribution (softmax) over the learned $\hat{Q}$ values
  - gives higher probability to actions with higher estimated $\hat{Q}(s, a)$ values
  - $p(a|s) = \frac{exp\left(\frac{\hat{Q}(s,a)}{\tau}\right)}{\sum_b exp\left(\frac{\hat{Q}(s,b)}{\tau}\right)}$
  - $\tau > 0$: temperature
    - high τ $\rightarrow$ distribution is nearly uniform (more exploration)
    - low τ $\rightarrow$ distribution peaks sharply at the best action (more exploitation)

Classic exploration

> Boltzmann exploration: the agent draws actions from a Boltzmann distribution (softmax) over the learned $\hat{Q}$ values

- $p(a|s) = \frac{exp\left(\frac{\hat{Q}(s,a)}{\tau}\right)}{\sum_b exp\left(\frac{\hat{Q}(s,b)}{\tau}\right)}$
- it would be a good choice until we have explored enough
- but, if Q is fully converged, we don't need to choose suboptimal actions
- we may tune the τ during the process (e.g., decaying τ), but heuristic
- how can we be confident about $\hat{Q}$?

> Upper confidence bounds: the agent selects the greediest action to minimize the upper confidence bound $\hat{Q}(s,a) + U(s,a)$

- be optimistic with options of high uncertainty
- prefer actions for which you do not have a confident value estimation yet
  - because it has a great potential to be high-rewarding!

Classic exploration

- Upper confidence bounds (UCB): the agent selects the greediest action to minimize the upper confidence bound $\hat{Q}(s,a) + U(s,a)$
  - estimate an upper confidence $U_t(a)$ for each action value such that

$$Q(s,a) \leq \hat{Q}(s,a) + U(s,a)$$

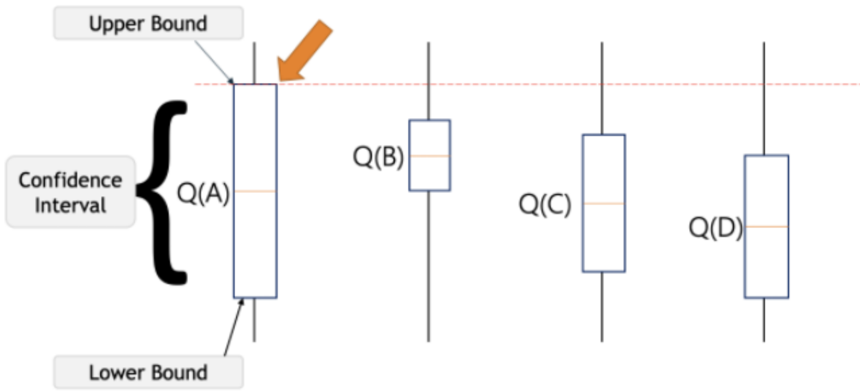
  - select the action that maximizes the upper confidence bound

$$a_t^{UCB} = \arg\max_{a \in \mathcal{A}} \hat{Q}(s,a) + U(s,a)$$

  - $U(s,a)$ is a function of the number of trials $N(s,a)$
    - small N $\rightarrow$ large bound U (estimated value is uncertain)
    - large N $\rightarrow$ small bound U (estimated value is certain/accurate)
    - central limit theorem: the uncertainty decreases as $\sqrt{N}$
    - $U(s,a) = c\sqrt{\frac{\ln \sum_a N(s,a)}{N(s,a)}}$

Classic exploration

> Upper confidence bounds (UCB): the agent selects the greediest action to minimize the upper confidence bound $\hat{Q}(s,a) + U(s,a)$

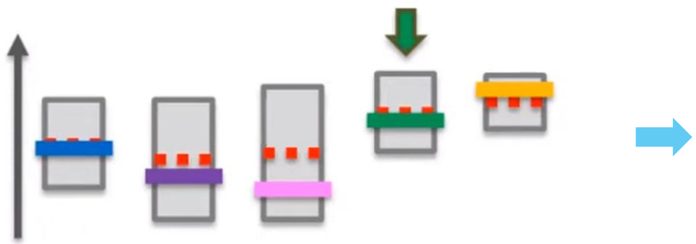


- still require a manually tuned exploration coefficient c
- assume a specific form for the confidence bound
  - in some states, dynamics could be highly stochastic, while in others not
- can be overly aggressive in exploring actions due to optimism
- difficult to extend directly to high-dimensional problems

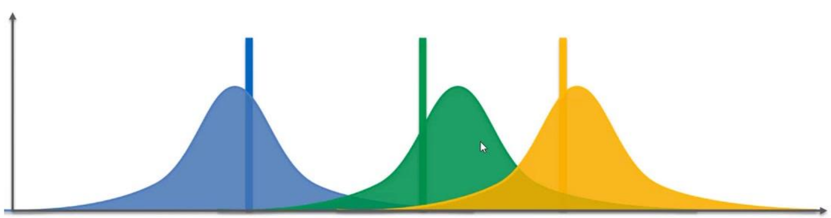
Classic exploration

> Thompson sampling: the agent keeps track of a belief over the probability of optimal actions and samples from this distribution

- estimate the distribution of $\hat{Q}$
- from each action, sample $\hat{Q}$ and compare
- execute the best action
- update $\hat{Q}$ distribution



UCB



Thompson sampling

- we assume that posterior distribution of $\hat{Q}$ follows a specific form (e.g., Gaussian)

Classic exploration

> Optimistic exploration
  - new state = good state
  - require estimating state visitation frequencies or novelty
  - typically realized by means of exploration bonuses

> Thompson sampling style
  - learn distribution over Q-functions or policies
  - sample and act according to sample

> What else?
  - information gain style: reason about information gain from new states
  - entropy loss & noise-based: implicit exploration

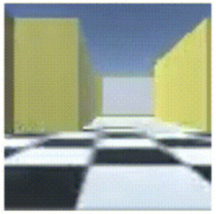
> These ideas can be extended to deep RL

Reward-based exploration (1)

> Hard-exploration problem
  - very sparse or even deceptive reward
  - random exploration can rarely discover successful states

> The noisy-TV problem
  - even if the RL agent is striving for the novel state, it could be daunting
  - noisy TV can attract the agent's attention forever

Agent in a maze with a noisy TV



Agent in a maze without a noisy TV

Exploration in deep RL (1)

> Revisit UCB (count-based exploration)
  - $U(s,a) = c\sqrt{\frac{\ln \sum_a N(s,a)}{N(s,a)}}$
    - we can use $N(s)$ instead of $N(s,a)$

  - in high-dimensional or continuous state spaces
    - many states we will never see at all
    - many states we will never see again
    - count become somehow useless

  - we need a non-zero count for most cases, even if we haven't seen them before
  - some states are more similar than others

Reward-based exploration (1)

> Density model (2016)
  - fit a density model $\rho(s;\theta)$ to approximate the frequency of visits
    - $\rho(s;\theta) \approx \rho_{data}(s)$
    - Negative Log-likelihood loss: $L(\theta) = -\frac{1}{n}\sum_i \log \rho_\phi(s_i)$
  - derive a pseudo count
    - present density model for s: $\rho(s;\theta) = \frac{N(s)}{n}$ (present state may not be s)
    - next step after observing s: $\rho'^{(s;\theta')} = \frac{N(s)+1}{n+1}$
    - from above two equations: $\widehat{N} = \frac{\rho(s;\theta)\left(1-\rho(s;\theta')\right)}{\rho(s;\theta')-\rho(s;\theta)}$ (estimation of count)
  - Algorithm
    - fit a model $\rho(s;\theta)$ with all states data seen so far
    - take a step and observe s
    - fit a new model with additional data $\rho'(s;\theta')$
    - estimate $\widehat{N}(s)$
    - set $r_i^+ = r_i + B(\widehat{N}(s))$ (not using Q as it is highly uncertain)

Reward-based exploration (1)

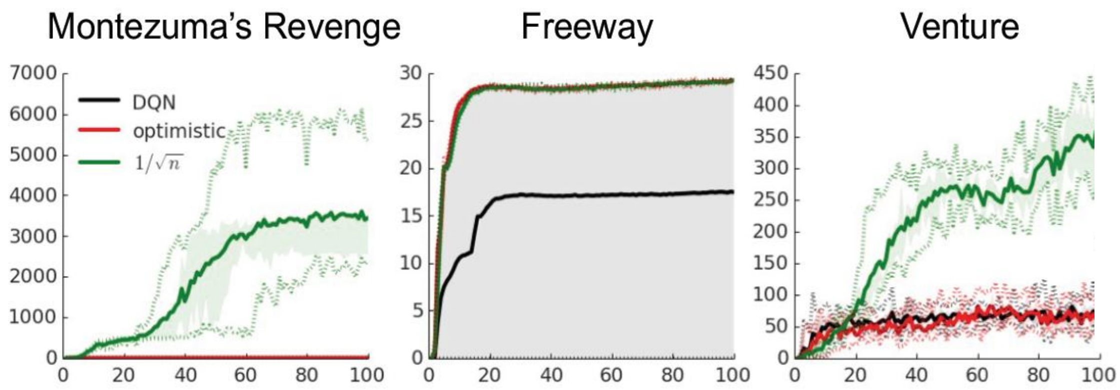
- Density model
  - added reward (intrinsic reward) can be seen as a bonus for exploration
  - common choice of $B(\widehat{N}(s))$ is
    - $\sqrt{1/\widehat{N}(s)}$ like we did in UCB
    - or $\sqrt{1/(\widehat{N}(s) + 0.01)}$

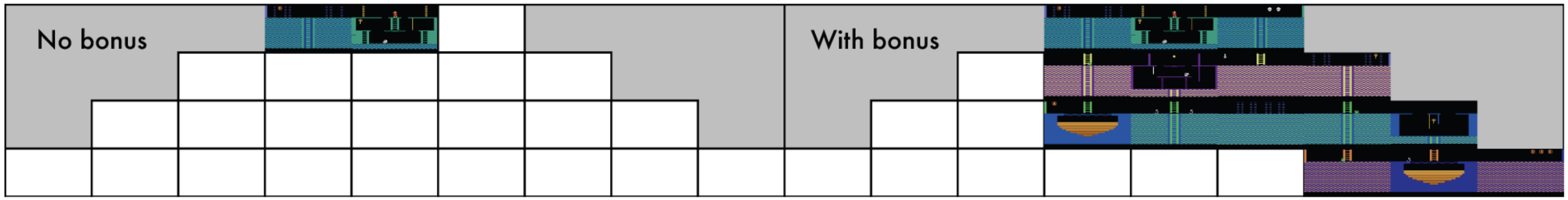
  - if we use a large neural network, density update is infinitesimal
    - $\rho(s;\theta') - \rho(s;\theta) \approx 0, \ \rightarrow \ \widehat{N} = \frac{\rho(s;\theta)\left(1-\rho(s;\theta')\right)}{\rho(s;\theta') - \rho(s;\theta)}$ diverges

  - there are other options to estimate the density function
    - use the prediction gain
    - context tree switching
    - PixelCNN
    - Gaussian mixture model

Reward-based exploration (1)

> Density model
  - does it work?



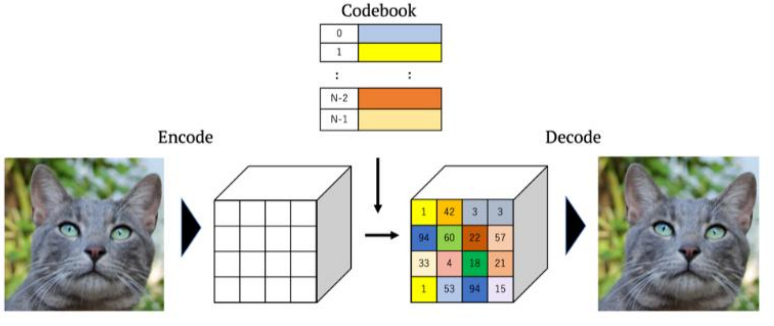


Reward-based exploration (1)

> Additional idea: map high-dimensional states into a discrete hash code via $\phi(s)$ and count $N(\phi(s))$ instead of $N(s)$
  - this makes count-based exploration feasible
    - the hash space is much smaller than the raw state space
    - counting becomes simple

  - how can we put similar states into same or similar hash codes?
  - classical hashing method works poorly on complex data (e.g. image)

  - we can learn a compression using an autoencoder (e.g. VQ-VAE)



Reward-based exploration (2)

> So far,
  - bonus came from the novelty of states we encounter
  - we encourage the agent to look for states it did not see that often
  - simple assumption: many visits → more information
fewer visits → less information
  - now, we quantify the amount of information about the environment

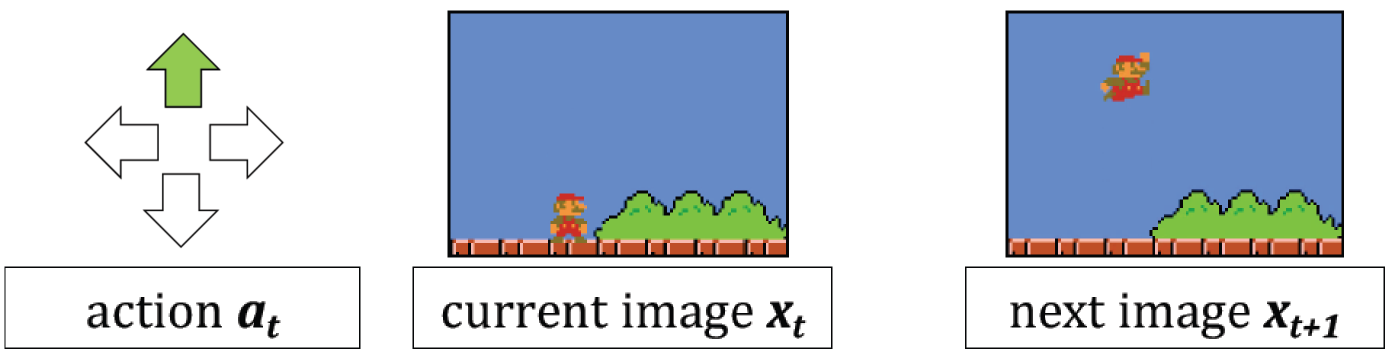
> Prediction-based exploration
  - if we have an enough knowledge, then we can predict accurately
    - forward dynamics prediction model is a great way to approximate how much knowledge our model has obtained about the environment

Reward-based exploration (2)

- Prediction-based exploration
  - forward dynamics prediction model and error

$$f_\theta : (s_t, a_t) \rightarrow s_{t+1}, \qquad e(s_t, a_t) = \|f_\theta(s_t, a_t) - s_{t+1}\|^2$$

  - curiosity = prediction error, chase for the curiosity
  - large prediction error: high bonus
  - low prediction error: low bonus



Reward-based exploration (2)

> Deep predictive models (2015)
  - predicting high-dimensional state spaces (images) can be very difficult
  - train a forward dynamics model in an encoding space ϕ

$$f_\theta : (\phi(s_t), a_t) \rightarrow \phi(s_{t+1}), \qquad e_t = \|f_\theta(\phi(s_t), a_t) - \phi(s_{t+1})\|^2$$

  - normalize the prediction error by the maximum error so far

$$\bar{e}_t = e_t / \max e_i$$

  - define the intrinsic reward accordingly

$$r_t = \frac{e_t(s_t, a_t)}{tC} \quad (C \text{ is a decay parameter})$$

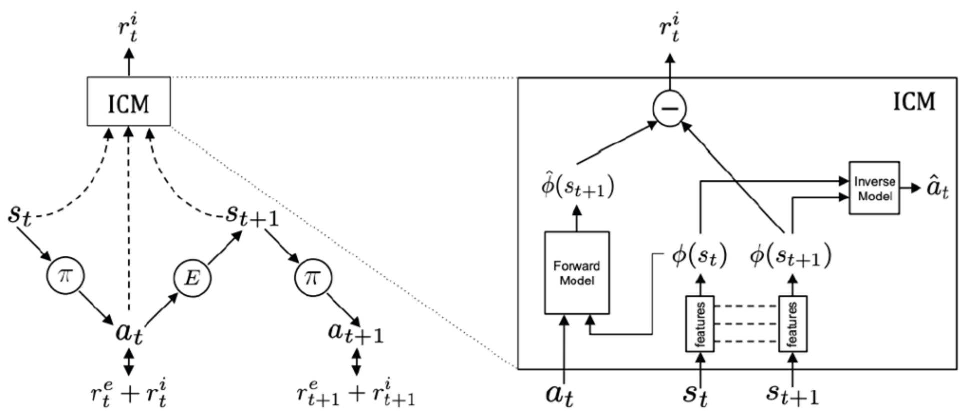
  - experiments in the paper have shown that a dynamics model without embedding has very poor behavior

Reward-based exploration (2)

> Intrinsic curiosity module (ICM, 2017)
  - ICM trains the state space encoding $\phi(s_t)$ with an inverse dynamics model

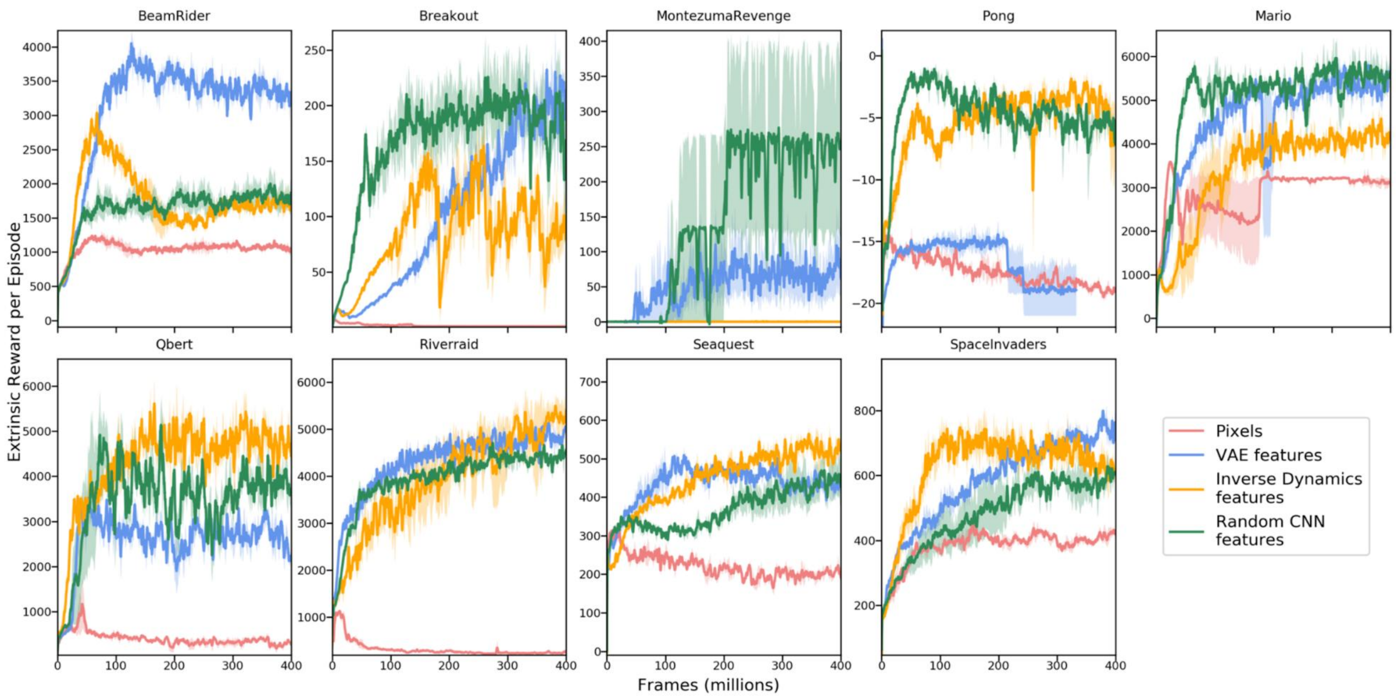
$$g: \left(\phi(s_t), \phi(s_{t+1})\right) \rightarrow a_t$$

  - predicting forward dynamics model is difficult as many factors in the environment cannot be controlled by the agent
  - the feature space should capture changes related to the agent's actions
  - by learning an inverse model together, ϕ focuses on action-related state change (still intrinsic reward is only dependent to forward error dynamics error)



Reward-based exploration (2)

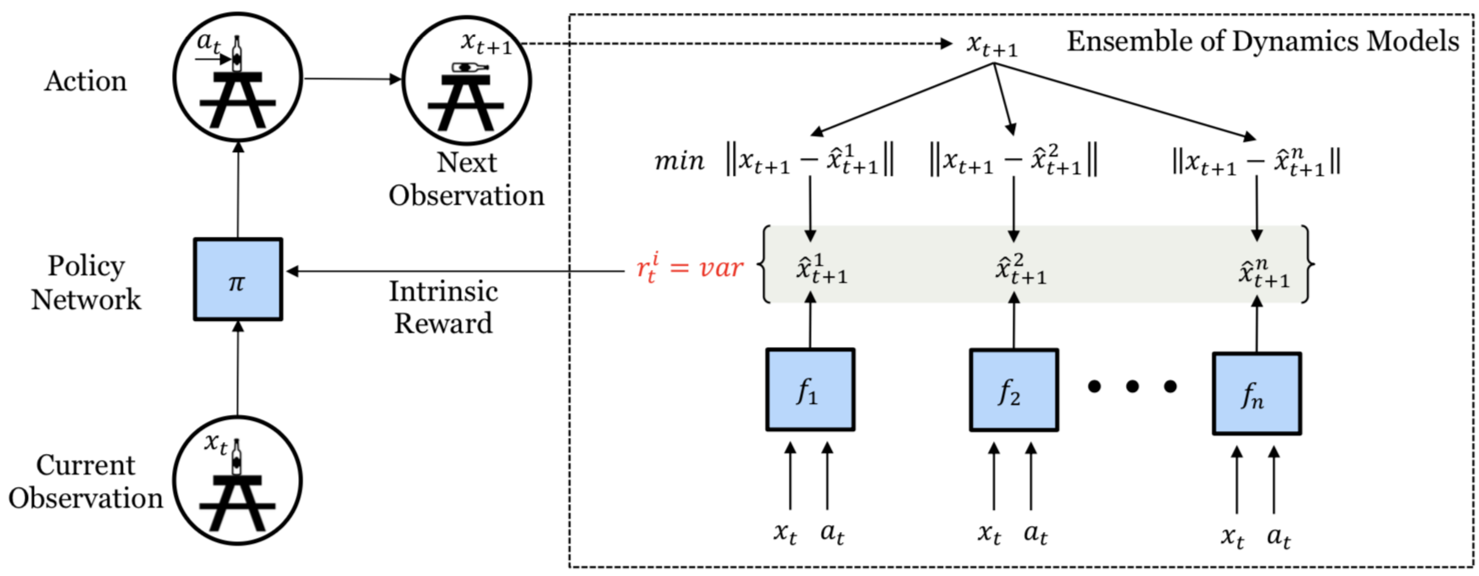
> Performance comparison depending on $\phi(s_t)$
  - Raw image pixels / VAE / IDF (inverse dynamic feature) / Random



  - random features are simple yet strong
  - IDF generalizes better

Reward-based exploration (2)

> Exploration via disagreement (2019)
  - use uncertainty of a forward dynamics model as an intrinsic reward
  - uncertainty can be measured with an ensemble of prediction
  - high disagreement → low confidence → needs more exploration
  - intrinsic reward is differentiable, which enable it to be directly optimized



Reward-based exploration (2)

- Going back to the original question
  - we started prediction-based exploration because
    - state visitation count is difficult for high-dimensional state
    - we may want to add more information on the intrinsic reward
      (beyond the state novelty, forward dynamics model gives us how do we know the environment well)
    - but sometimes learning a dynamics model can be very difficult, too!
      (e.g., noisy TV problem)

  - In fact, we can use any kind of predicting function for the exploration

$$f_\theta : (s_t, a_t) \rightarrow x_t, \qquad e = \|f_\theta(s_t, a_t) - x_t\|^2$$

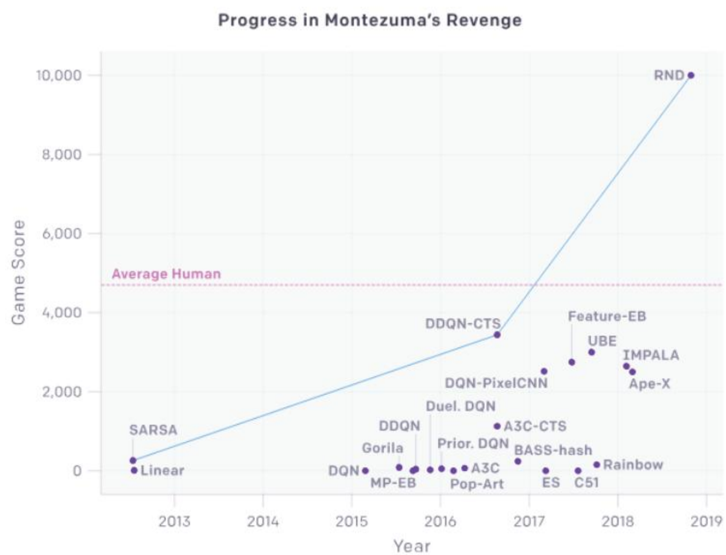
    - because if we have collected enough data for (s_t, a_t) (=experienced enough), e will become small

Reward-based exploration (2)

- Random network distillation (RND) (2018)
  - idea: predict something that is independent from the main task
  - here, we predict the random feature embedding f_ϕ
    (randomly initialized but fixed embedding neural network)
  - a network f_θ is trained to predict f_θ
  - intrinsic reward: $r(s_t) = \left\| f_\theta(s_t) - f_\phi(s_t) \right\|^2$
  - it can be seen as a generalized method for count-based exploration in high-dimensional state spaces
    - a random network tends to embed similar states into a similar latent space

Reward-based exploration (2)

> Random network distillation (RND) (2018)



- works better for non-episodic setting (can't discriminate episodic novelty)
- target is deterministic (while forward dynamics can be stochastic)
- it is inside the class of functions that the predictor can represent
- normalization is important, the scale of rewards is tricky (random target)
  - normalize by a running estimate of std. of intrinsic return

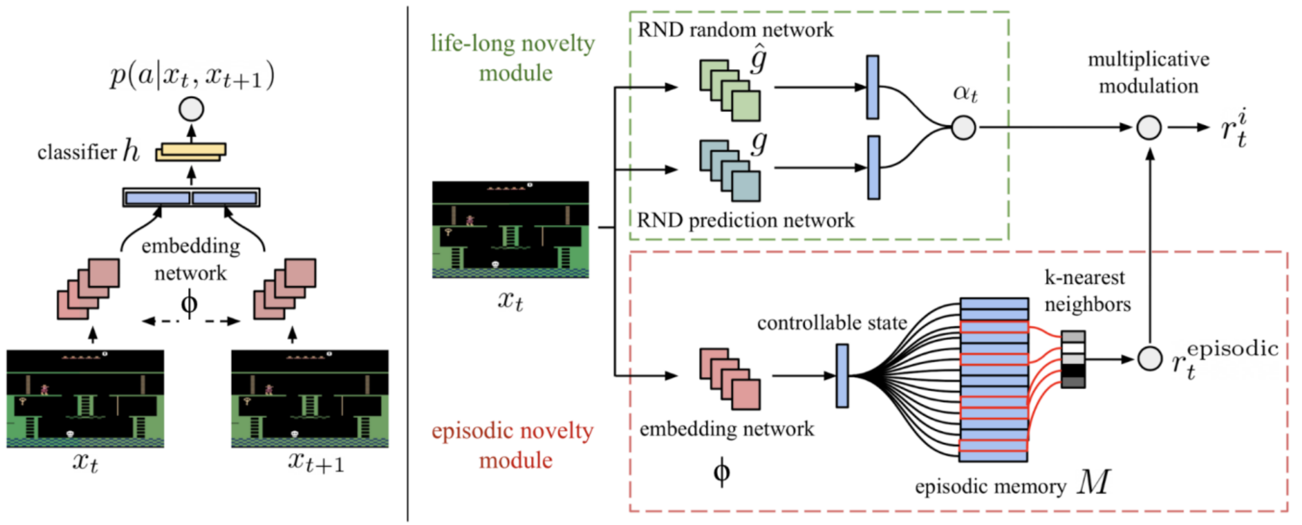
Memory-based exploration

> Reward-based exploration has some disadvantages
  - function approximation is slow
  - exploration bonus is non-stationary
  - knowledge fading: states are no longer novel and do no longer provide intrinsic reward signals

> Idea of memory-based exploration
  - use separate external memories
  - by maintaining separate memories, one can distinguish between episodic novelty and long-term novelty

Memory-based exploration

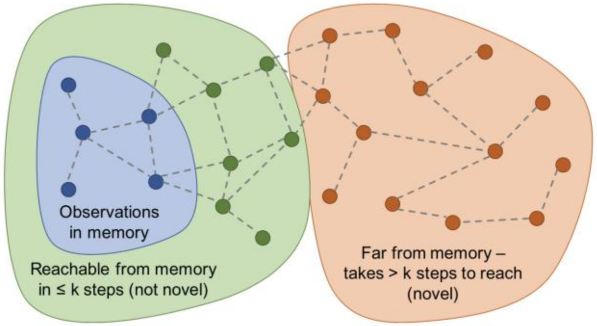
> Never give up (2020)
  - combines episodic novelty and long-term novelty bonuses



  - Rapidly discourages revisiting the same state within the same episode
  - Slowly discourages revisiting states that have been visited many times across episodes

Memory-based exploration

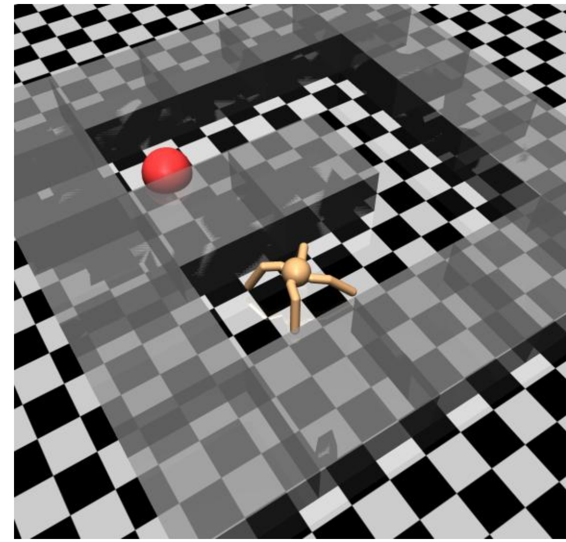
> Agent57 (2020)
  - the first RL agent who beats Atari57 consistently
  - use population of policies
    - each policy has its own pair of exploration parameters
    - a meta-controller is trained to select from policies
  - re-parameterization of Q (separating Q according to reward types $\mathrm{r}^{\mathrm{e}}, r^{i}$)
  - episodic curiosity
    - not calculating the distance between two states using Euclidean distance
    - measure the number of steps needed to transit between to states
    - the novelty depends on the reachability
    - train a Siamese neural network that predicts how far two states are apart



Skill-based exploration

> Ant-maze: more controller side RL task
  - to reach the goal point, first the agent needs to learn how to walk
  - skills: distinct behavior patterns
    - can be represented by a latent variable

> skill-based exploration
  - train multiple skills (latent-conditioned sub-policies)
  - learned skills can be reused and fine-tuned for downstream continuous control tasks



Ant-maze
reach the goal = +1

Skill-based exploration

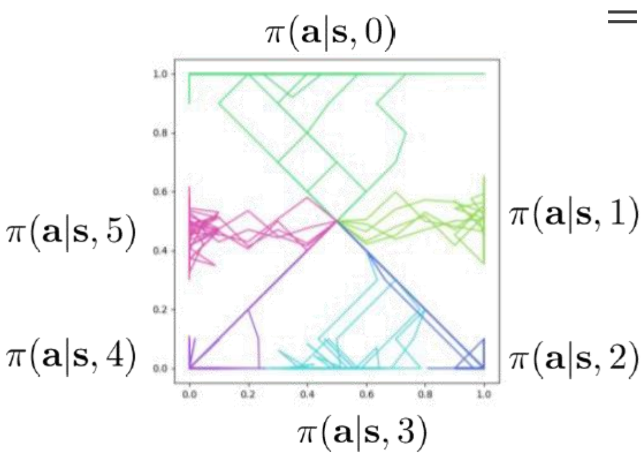
- Diversity is all you need (DIAYN, 2018)
  - Skill-based policy: $\pi(a|s,z)$
  - $Z \sim p(z)$ is a latent variable, the policy conditioned on a fixed z is a skill
  - maximize mutual information (MI) between skills and states $I(S;Z)$
    - skill should control which states the agent visits
  - minimize MI between skills and actions given the state $I(A;Z|S)$
    - skills are identified through the states they visited, not through action patterns
  - maximize the entropy $H(A|S)$, promoting diverse exploration
  - Objective is $F(\theta) = I(S;Z) + H[A|S] - I(A;Z|S)$

$$
\begin{aligned}
&= (H[Z] - H[Z|S]) + H[A|S] - (H[A|S] - H[A|S,Z]) \\
&= H[Z] - H[Z|S] + H[A|S,Z]
\end{aligned}
$$

skill diversity

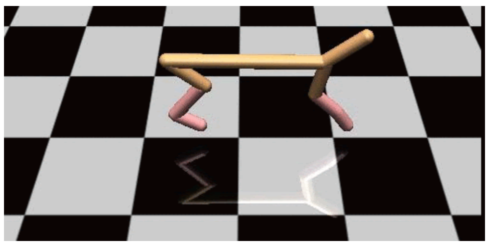
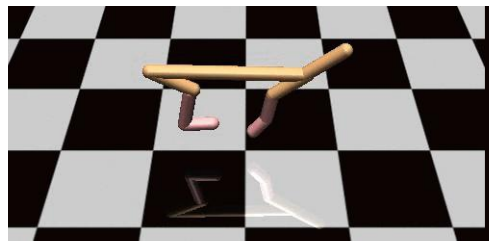
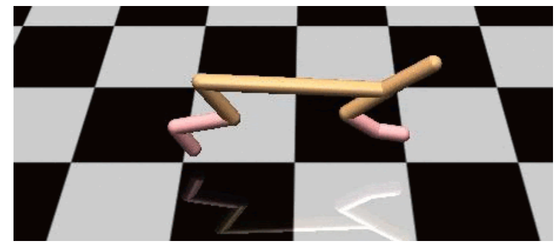
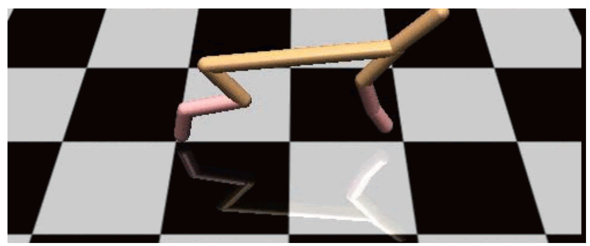
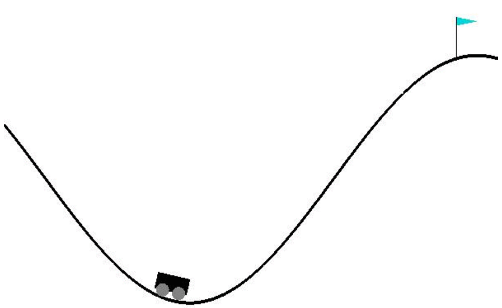
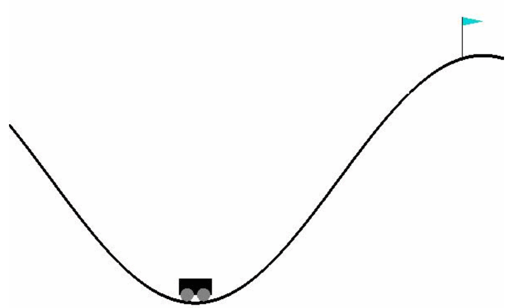
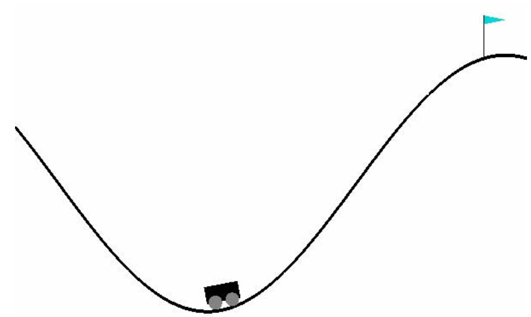
skills to be distinguishable

each skill maintains action entropy (not deterministic A)



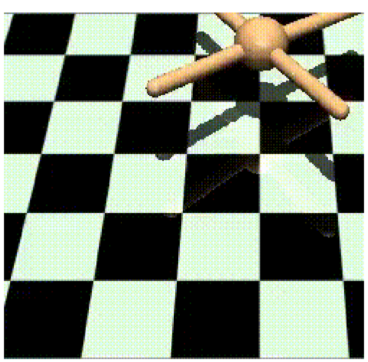
Skill-based exploration

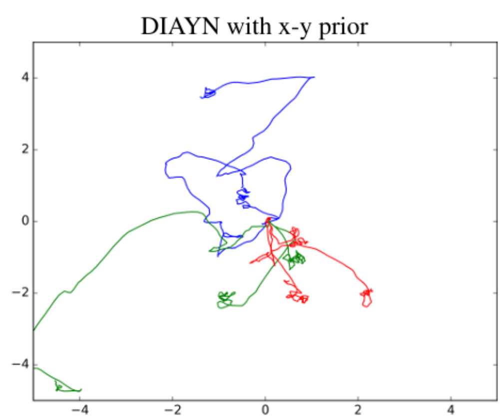
> Diversity is all you need (DIAYN, 2018)
  - learned skills

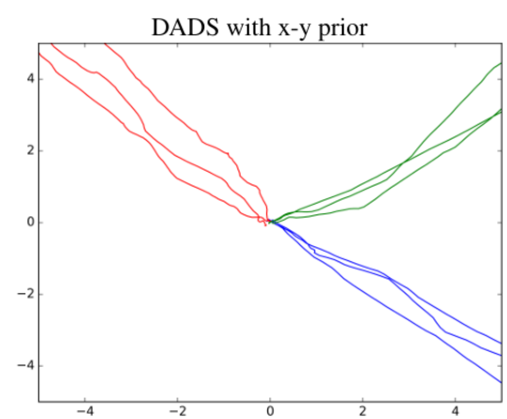








Skill-based exploration

> Dynamics-aware unsupervised discovery of skills (DADS, 2019)
  - model-based RL version of skill discovery
  - train a skill conditioned dynamics model $q_{\phi}(s'|s,z)$
    - only learn skills that are predictable
      (random walks are unpredictable, but have high state entropy)
    - predictable skills can serve as reusable behavioral primitives
    - learning the dynamics of the environment only for certain skills is much easier than learning the whole environment dynamics

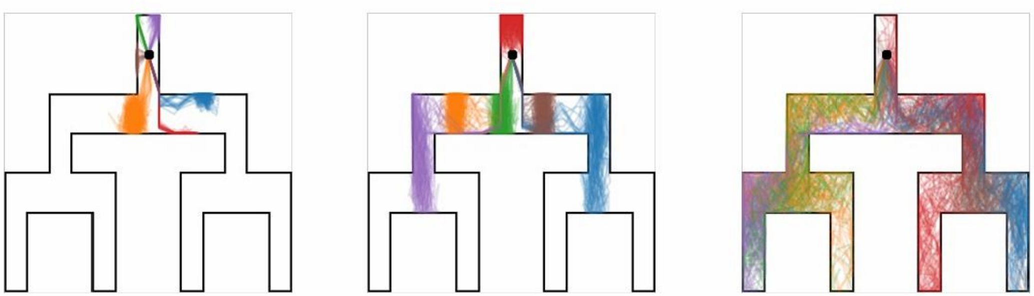






Skill-based exploration

> Contrastive intrinsic control (CIC, 2022)
  - maximize MI using state transition (trajectory) instead of state $I(\tau; Z)$
    - ensuring skills correspond to dynamic behaviors rather than static poses
  - mutual information and entropy-based exploration often fail in complex environments due to weak skill discriminators
    - discriminator: classifier or regressors that differentiates skills
    - it requires exponentially many samples to train when skill space is large
  - Key Idea: replace weak discriminators with a contrastive loss, enabling stronger and more scalable skill discrimination.



DIAYN DADS CIC

DIAYN

CIC