What is Reinforcement Learning for?

Reinforcement Learning

World

observation

reward

action
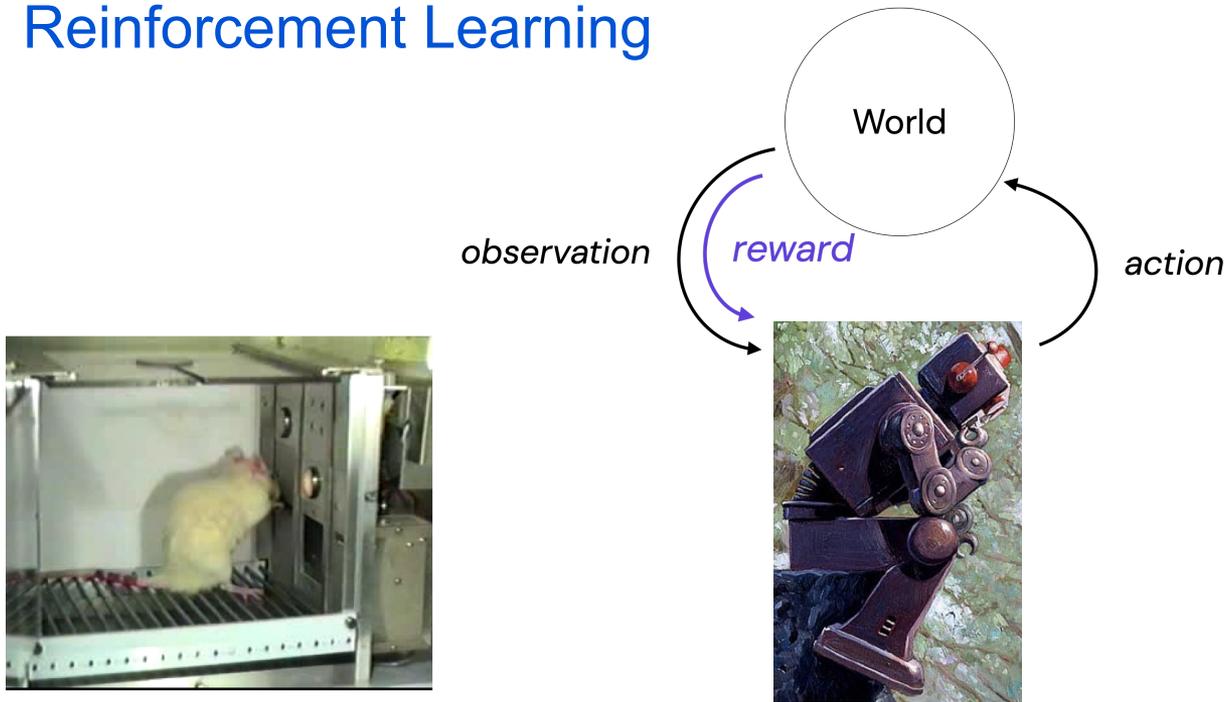
"*Part of the appeal of reinforcement learning is that it is in a sense the whole AI problem in a microcosm.*"
– Sutton, 1992

1. RL for understanding intelligence

- A way to model processes in the brain
- A way to model cognitive processes in animals and people

Learning Values: Temporal-Difference Error

- Value estimate at time step t: $v(S_t)$

- Value estimate at time step $t+1$: $r(S_t, A_t) + \gamma v(S_{t+1})$
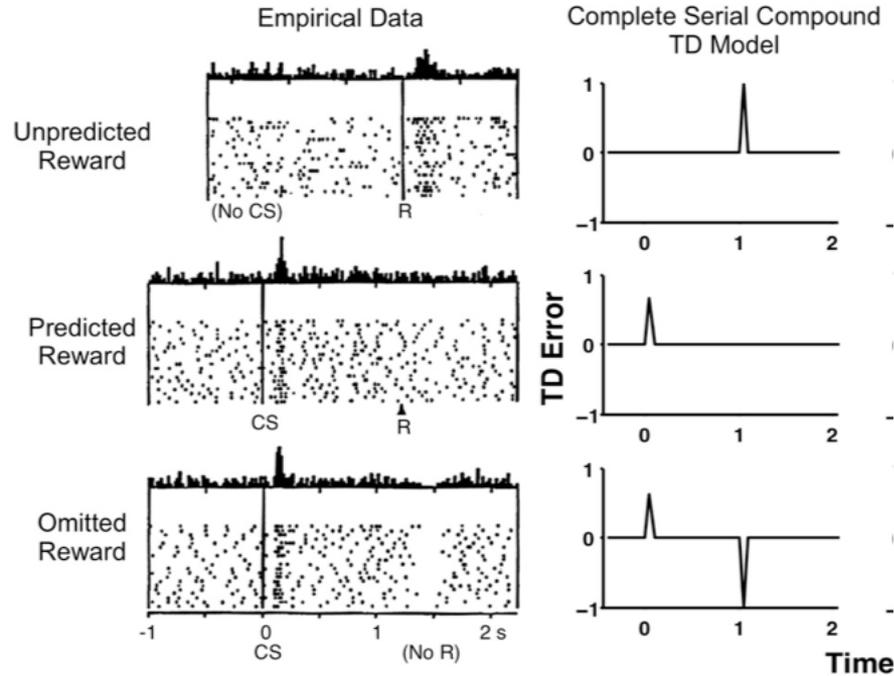
- *Temporal-difference error:*

$$\delta_t = r(S_t, A_t) + \gamma v(S_{t+1}) - v(S_t)$$

- If v is parameterized by w, change w so as to minimize the TD-error:

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha \delta_t \nabla_{\mathbf{w}} v_{\mathbf{w}}(S_t)$$
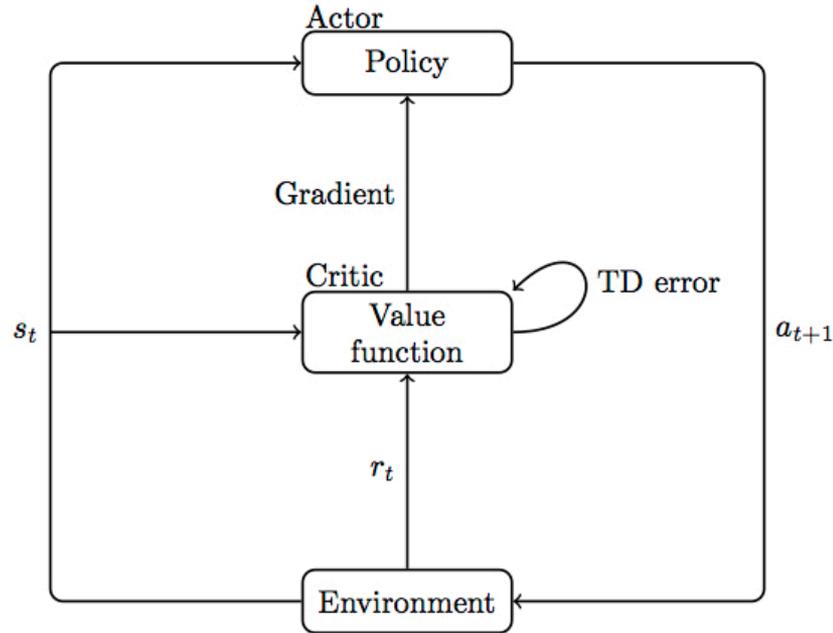
- Shultz, Dayan & Montague (1997): TD-errors model the activity of dopamine neurons

Dopamine neuron modelling

Cf. Shultz, Dayan et al, 1996; and lots of follow-up work including MNI, Psych.

Control: Actor-critic architecture

- Parameters of the policy move to make more likely action a that has positive advantage:

$$A(s, a) = r(s, a) + \gamma \mathbf{E}(v(s') \,|\, s, a) - v(s)$$

- O'Doherty et al (2004): fMRI evidence that dorsolateral striatum implements an actor and ventral striatum a critic

Generalizing Actions: Options Framework

- An *option* is a defined by a tuple $\langle I_\omega(s), \pi_\omega(a \,|\, s), \beta_\omega(s) \rangle$

  - An *initiation function*   (precondition)

  - An *internal policy*   (behavior)

  - A *termination function* (post-condition)

- Eg robot navigation: if no obstacle in front (initiation) go forward (policy) until something is too close (termination)

Cf. Sutton, Precup & Singh, 1998; Precup, 2000

Possible Neural Correlates of Options
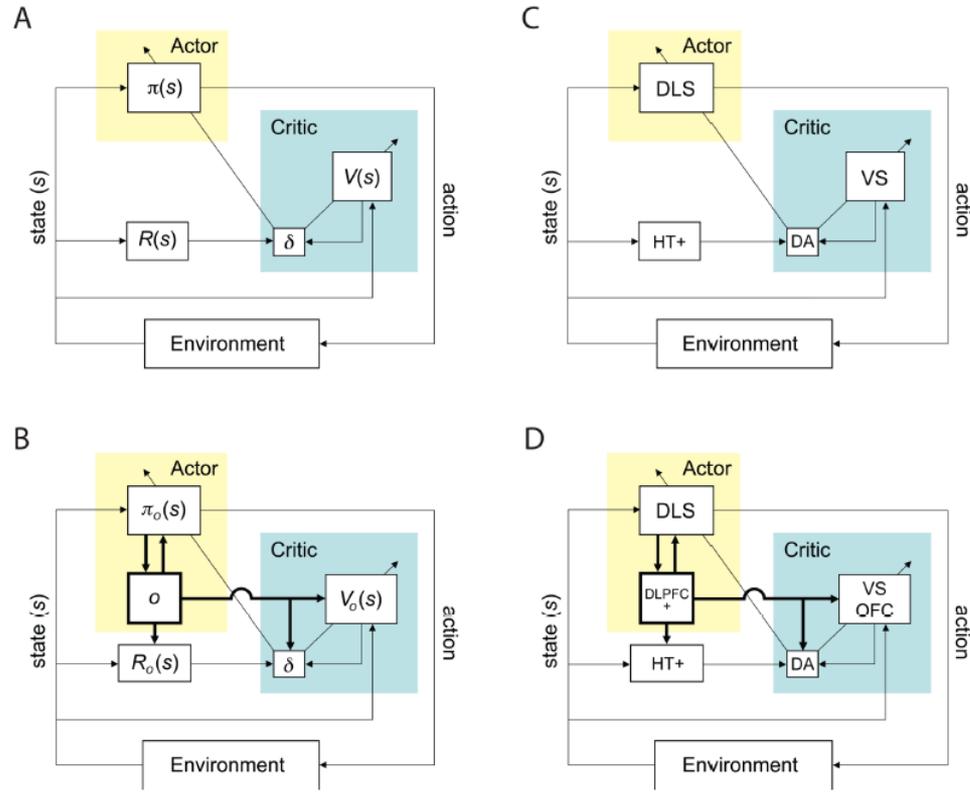
From Botvinick, Niv & Barto, 2009

> **"**
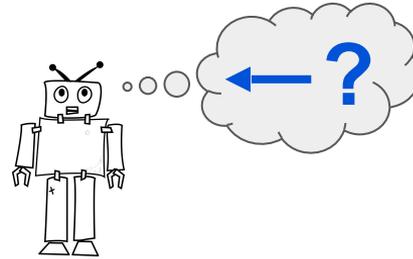> Affordances […] relations between abilities of organisms and features of their environment.
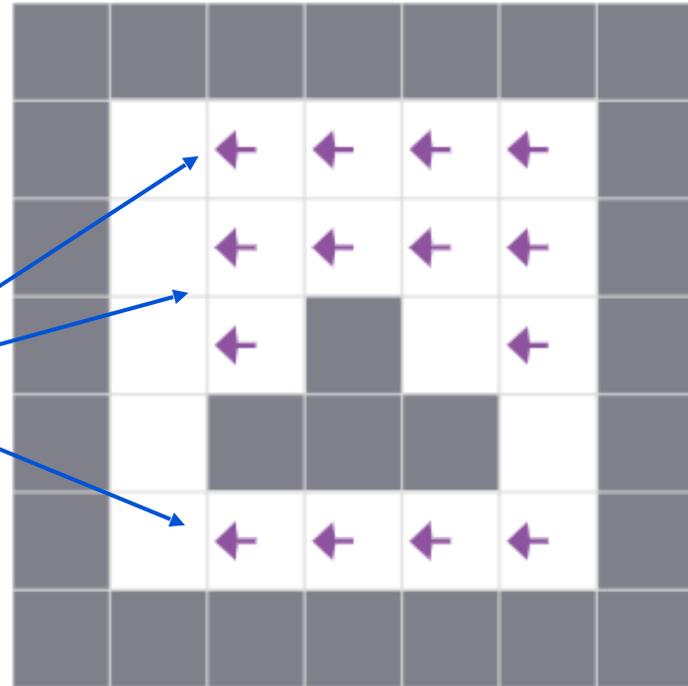>
> Gibson, 1977

Affordances

Captures states and actions that complete an intent
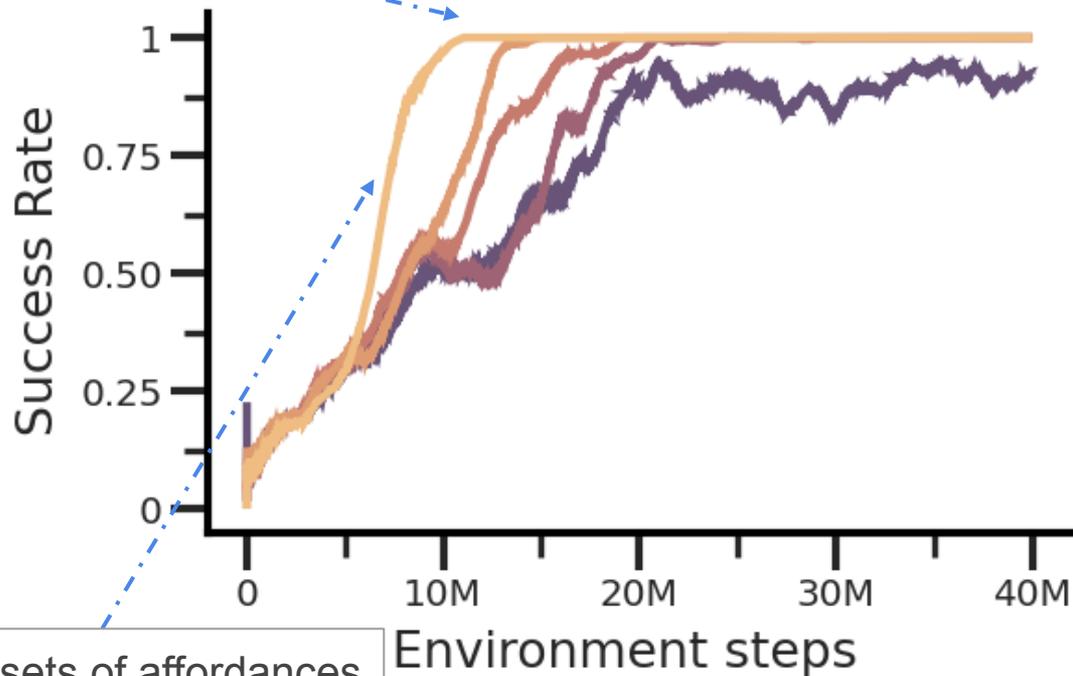
$$\mathcal{AF}_{\mathcal{I}} \subseteq \mathcal{S} \times \mathcal{A}$$

Affordances are the subset of states and actions which complete the intent to go left.

What is the impact of the affordance set size on performance ?

2. RL for applications

- Build super-human agents
- Tackle very complex control tasks
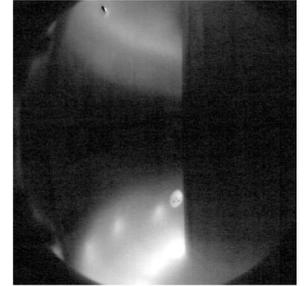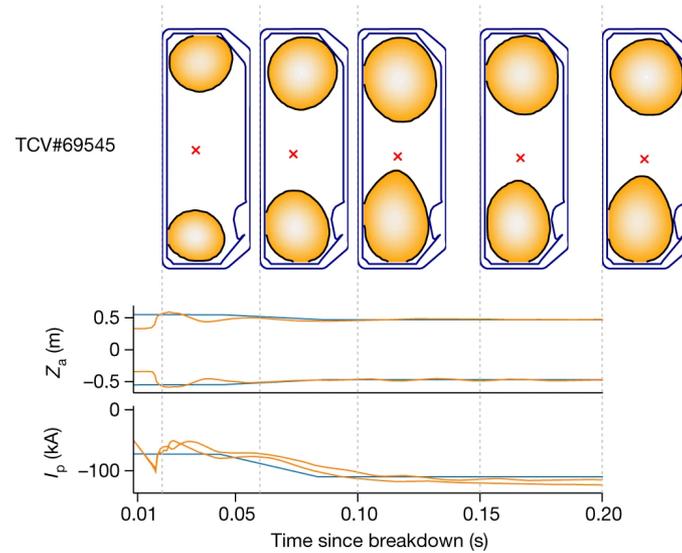- Learn to search and explore

Super-human performance: Games

Complex Control Tasks

TCV#69545

Bellemare et al, Nature, 2020

Degrave et al, Nature, 2022

Small Molecule Drug Discovery

Protein

Drug

Initial data
{(x,y),..}

occasionally query "oracle"

generate molecule(s) → predict reward cheaply (DNN)

compute "real" reward

growing dataset

train generator

train reward predictor

This part is expensive and/ or noisy! (w/ biased noise)
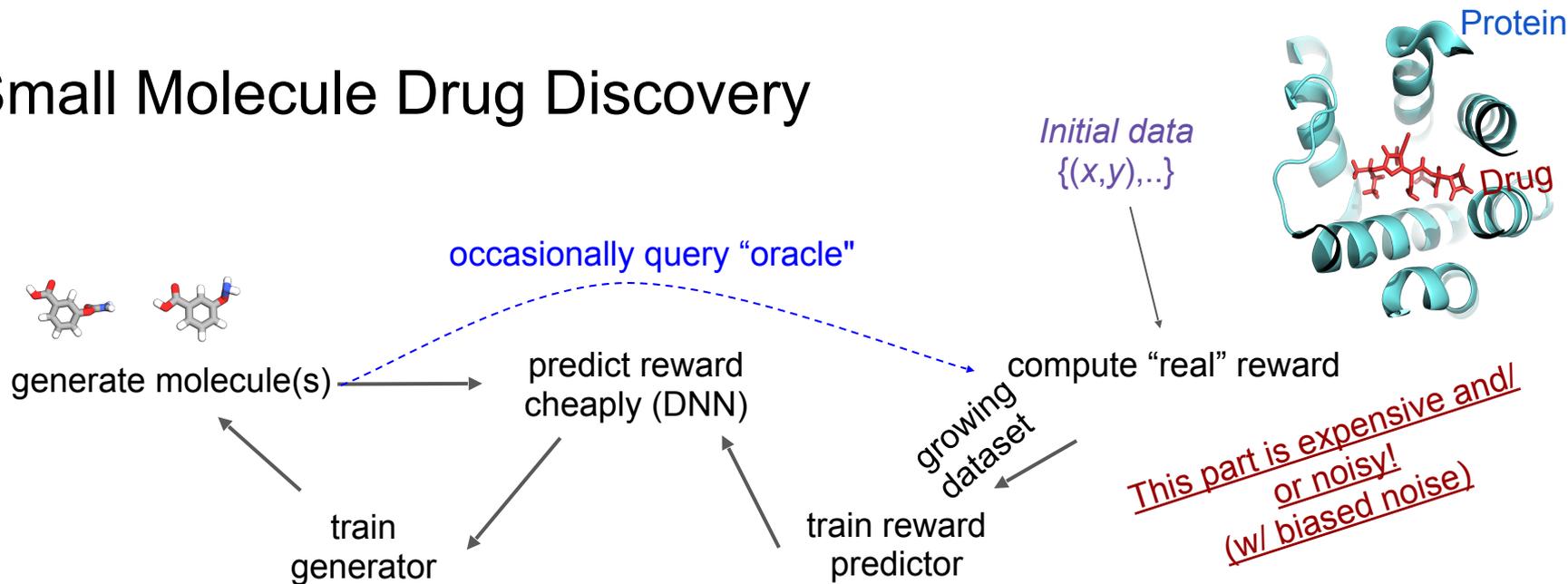
Oracle?

- Ideal: send **diverse** batches (10-100k) of candidates to a lab, O(weeks)
- For now: use noisy physics simulator, O(15 CPUs)/molecule

McGill

Mila

Drug Discovery as Reinforcement Learning Problem

"empty molecule"

Build molecules block by block

Episodes end with **terminal** R > 0
(no intermediate rewards)

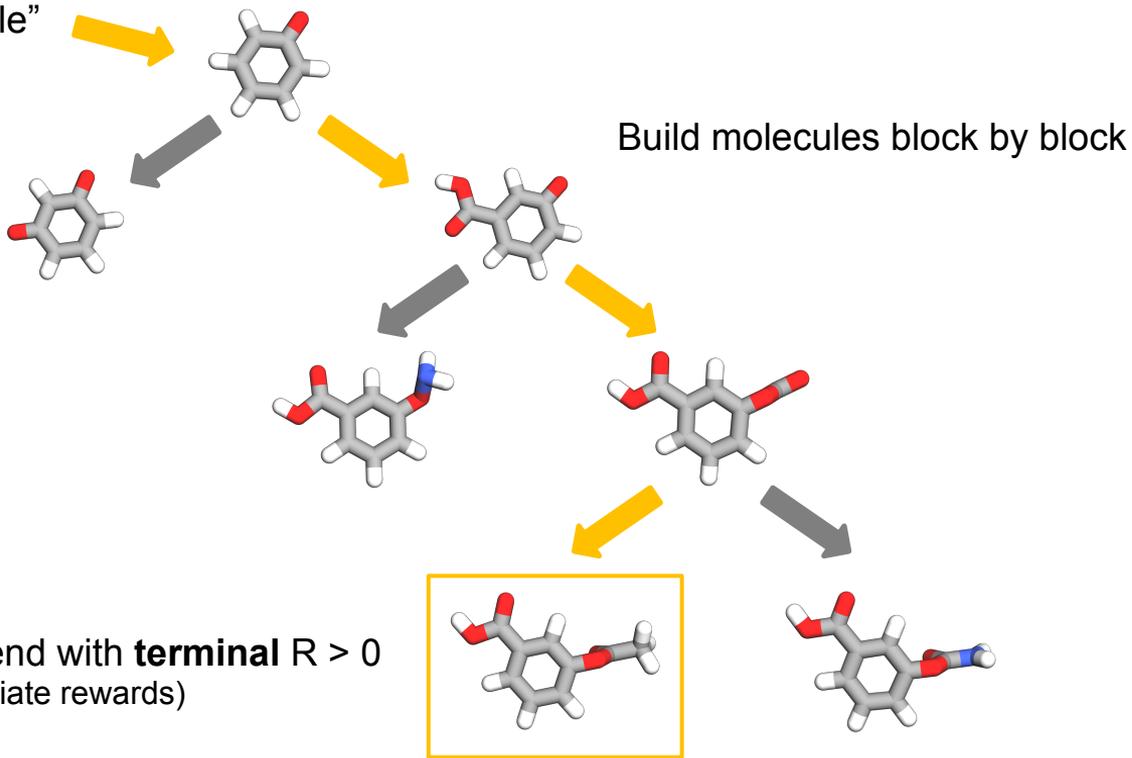# Just apply Reinforcement Learning?

- We have an environment (actions = build molecule)
- We have a (noisy, learned) reward
- RL! (Segler et al., 2017; De Cao & Kipf, 2018; Popova et al., 2019; Gottipati et al., 2020; Angermueller et al., 2020)

But RL greedily looks for one mode, even when we encourage entropy!
Not great for *diverse* batch oracle queries

Just apply MCMC?

- We have a Markov Chain (actions = edit molecule)
- We have a reward = unnormalized probability, want to sample from it
- MCMC! (Seff et al., 2019; Xie et al., 2020)

But MCMC is slow, gets stuck in modes easily (lack of diversity!), requires mode-mixing for any new sample

Not great for *diverse* batch oracle queries!

What about the usual generative models?

- Trained from positive samples only (e.g. existing drugs)
  But we have a more informative (non-binary) signal! (reward)
- We don't just want high reward, we want to avoid low reward (and have the data)
- Still possible to do well: Jin et al., 2018; Shi et al., 2020; Luo et al., 2021
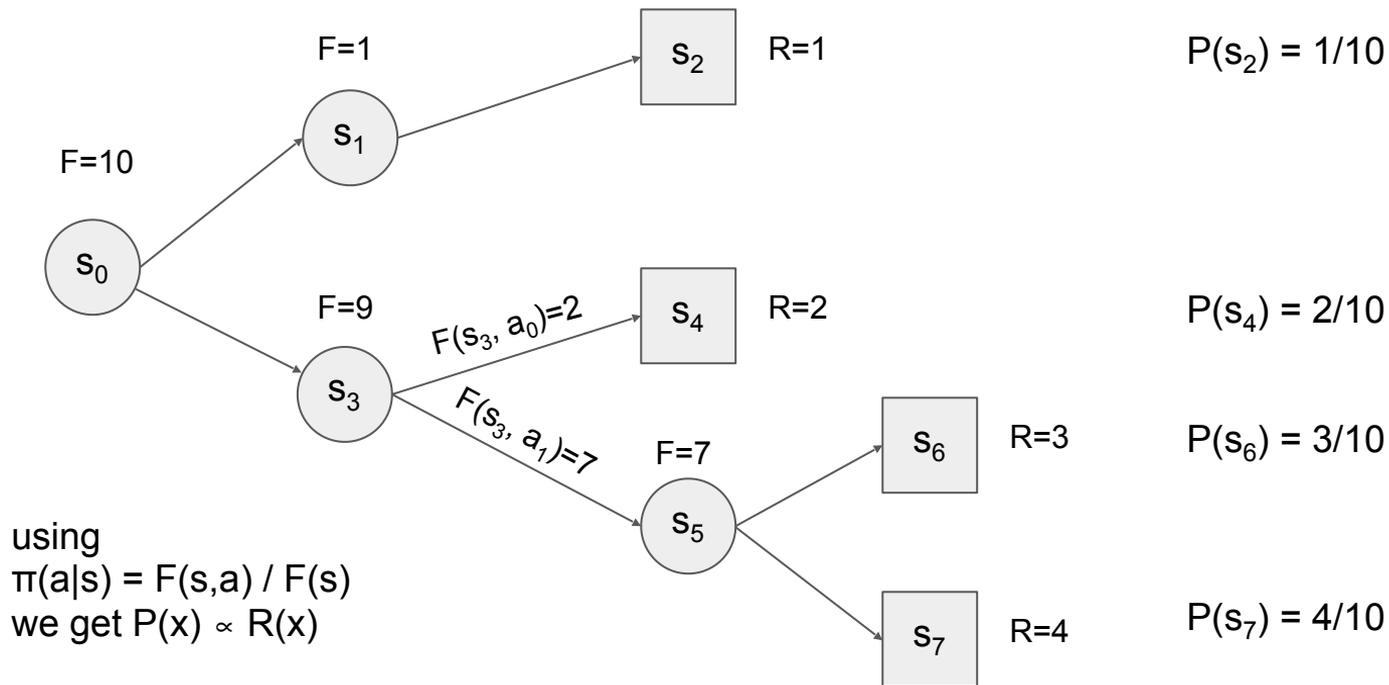
McGill

Mila

GFlowNet

Generative framework for discrete objects which have a reward (or energy).

Reward-proportional sampling:
without MCMC!

$$\pi(x) \approx \frac{R(x)}{Z} = \frac{R(x)}{\sum_{x' \in \mathcal{X}} R(x')}$$

Background: SumTrees (& control as inference: SoftAC/SoftQL)

$\pi(a|s) = Q(s,a) / V(s) = F(s,a) / F(s)$

F=1

s_1

s_2 R=1 $P(s_2) = 1/10$

F=10

s_0

F=9

s_3

$F(s_3, a_0)=2$

s_4 R=2 $P(s_4) = 2/10$

$F(s_3, a_1)=7$

F=7

s_5

s_6 R=3 $P(s_6) = 3/10$

s_7 R=4 $P(s_7) = 4/10$

using
$\pi(a|s) = F(s,a) / F(s)$
we get $P(x) \propto R(x)$

What if it's a DAG?

Naively applying SoftQL/SumTree yields the wrong solution

F=3

s_2 R=1

$P(s_2) = 1/12$

F=12

s_1

s_0

F=9

$F(s_3, a_0)=2$

s_4 R=2

$P(s_4) = 4/12$

s_3

$F(s_3, a_1)=7$

F=7

s_6 R=3

$P(s_6) = 3/12$

s_5

using
$\pi(a|s) = F(s,a) / F(s)$
we get $P(\tau) \propto R(\tau)$
        !=
        $P(x) \propto R(x)$

s_7 R=4

$P(s_7) = 4/12$

McGill

Mila

What if it's a DAG?

Naively applying SoftQL/SumTree yields the wrong solution

F=3 → s_2, R=1, $P(s_2) = 1/12$

F=12, s_0

s_1

F=9, s_3

$F(s_3, a_0)=2$ → s_4, R=2, $P(s_4) = 4/12$

$F(s_3, a_1)=7$

F=7, s_5

s_6, R=3, $P(s_6) = 3/12$

s_7, R=4, $P(s_7) = 4/12$
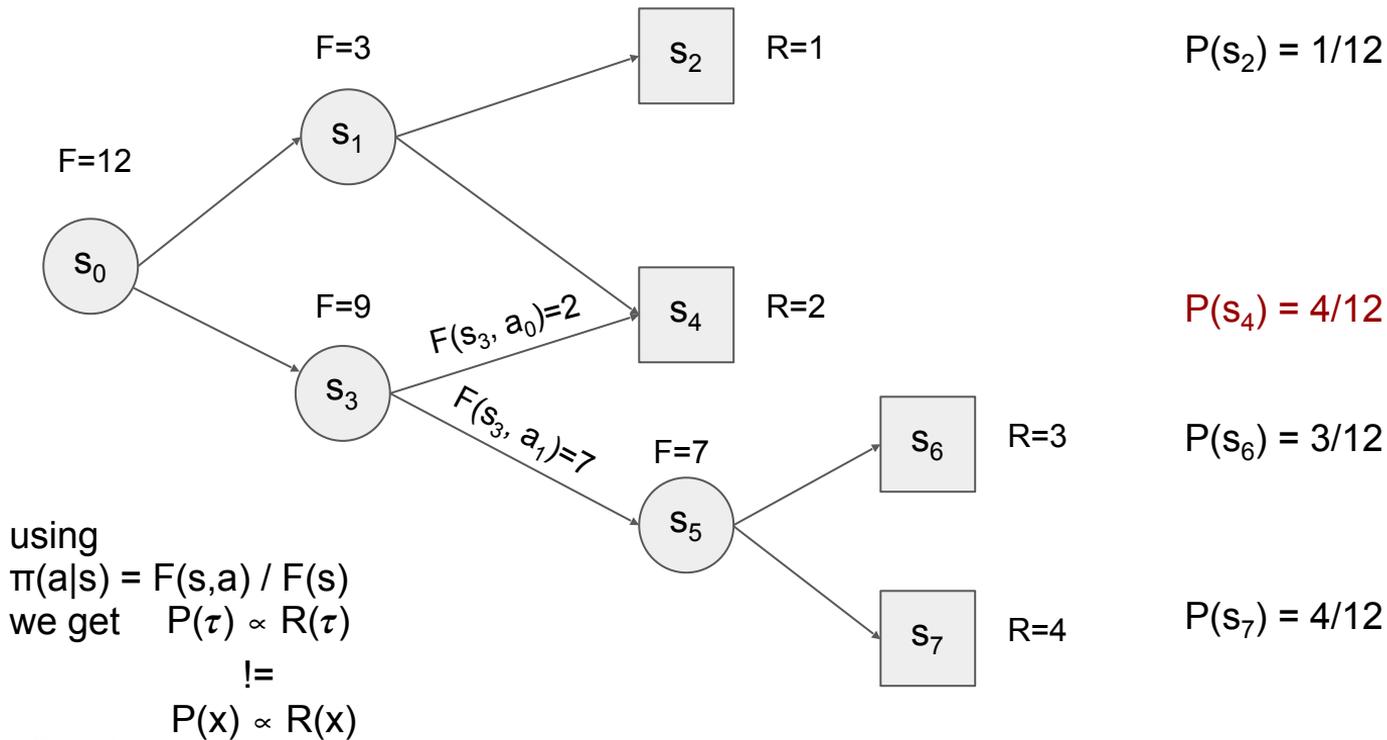
using
$\pi(a|s) = F(s,a) / F(s)$
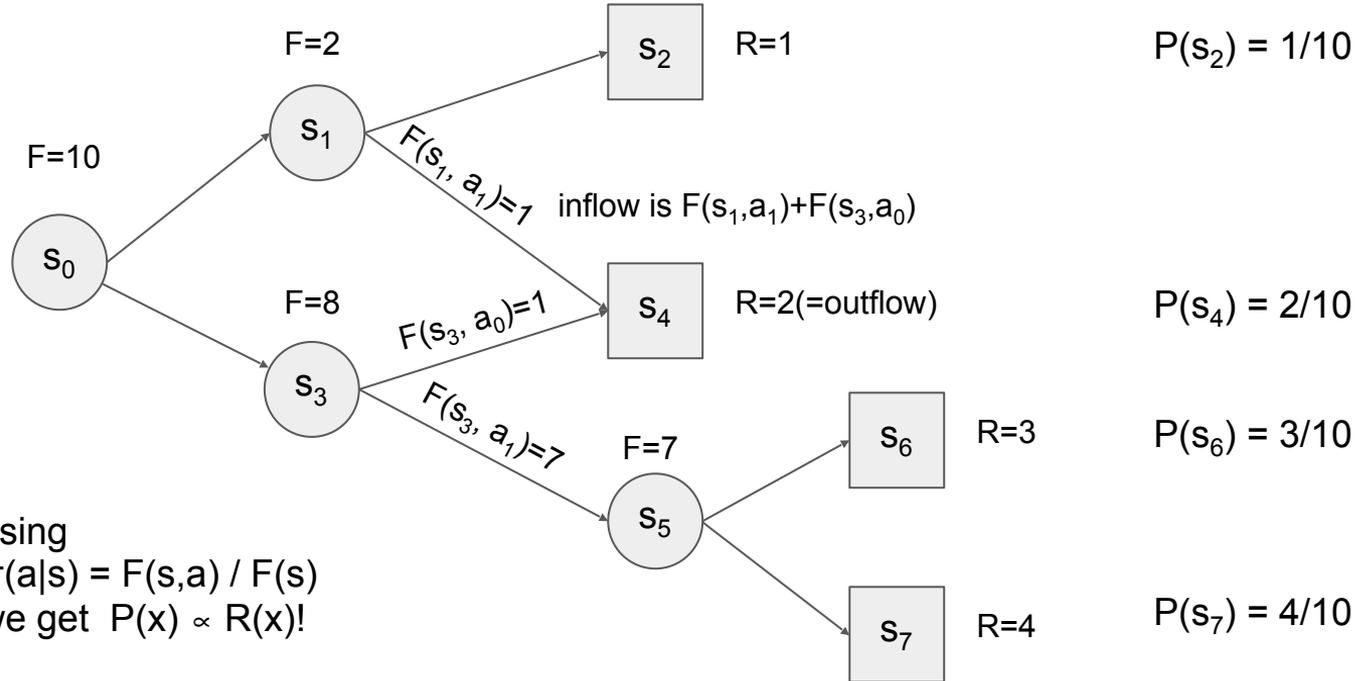we get $P(\tau) \propto R(\tau)$
           !=
        $P(x) \propto R(x)$
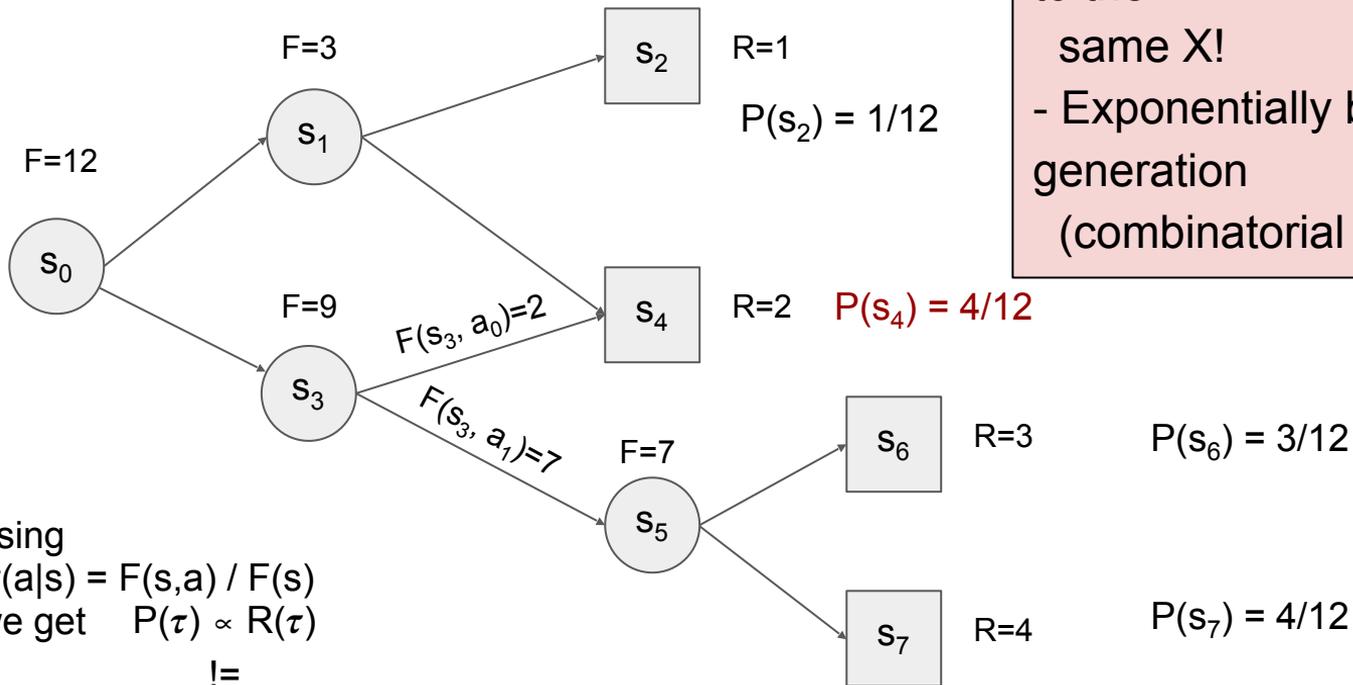
- $P(\tau) \propto R(\tau)$ is bad if many τ lead to the
  same X!
- Exponentially bad in graph generation
  (combinatorial # of paths)

McGill

Mila

Interpreting the DAG as a flow network

F(s) such that inflow = outflow

F=2 s_2 R=1 $P(s_2) = 1/10$

s_1

$F(s_1, a_1)=1$ inflow is $F(s_1,a_1)+F(s_3,a_0)$

F=10

s_0

F=8 $F(s_3, a_0)=1$ s_4 R=2(=outflow) $P(s_4) = 2/10$

s_3

$F(s_3, a_1)=7$ F=7 s_6 R=3 $P(s_6) = 3/10$

s_5

using
$\pi(a|s) = F(s,a) / F(s)$
we get $P(x) \propto R(x)$!

s_7 R=4 $P(s_7) = 4/10$

McGill

Mila

Flow consistency

Satisfy flow conditions, for all s'

$$\sum_{s,a:T(s,a)=s'} F(s,a) = R(s') + \sum_{a' \in \mathcal{A}(s')} F(s',a')$$

in flow of s'   out flow of s'

This is very similar to a Bellman Equation, the bread and butter of RL!

Satisfying the flow equations yields the right sampling proportions

How to train GFlowNet

Take inspiration from RL to learn *F:*

$$\tilde{\mathcal{L}}_\theta(\tau) = \sum_{s' \in \tau \neq s_0} \left(\sum_{s,a:T(s,a)=s'} F_\theta(s,a) - R(s') - \sum_{a' \in \mathcal{A}(s')} F_\theta(s',a') \right)^2$$

Dangerous objective, $F(s_0,.)$ is going to be huge! $F(s_0) = Z$

How to train GFlowNet

Instead, learn the log, **and** match flows in log-space

$$\mathcal{L}_{\theta,\epsilon}(\tau) = \sum_{s' \in \tau \neq s_0} \left(\log \left[\epsilon + \sum_{s,a:T(s,a)=s'} \exp F_\theta^{\log}(s,a) \right] - \log \left[\epsilon + R(s') + \sum_{a' \in \mathcal{A}(s')} \exp F_\theta^{\log}(s',a') \right] \right)^2$$

with an epsilon (care less about tiny flows)

Relationship to MaxEntRL

- Quite similar in spirit but different mechanism (recent papers establish formal relationship)
- *Sampling of trajectories is always proportional to the reward at the end*
- If multiple policies are optimal their paths continue to be generated
- In fact, all paths continue to be generated
- Ongoing work: extensions to non-DAG, rewards at all states
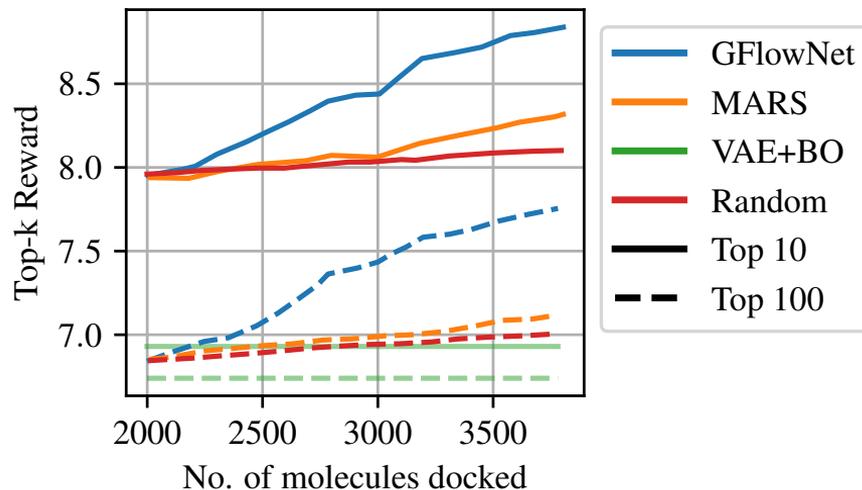
Works well! Molecule results

Pre-train reward function once on 300k molecules (computed on CPU simulator)
Modes are found faster, with better rewards
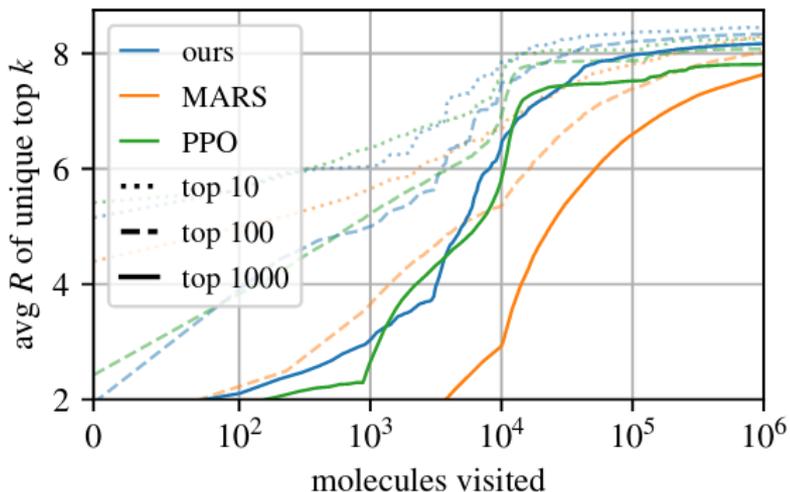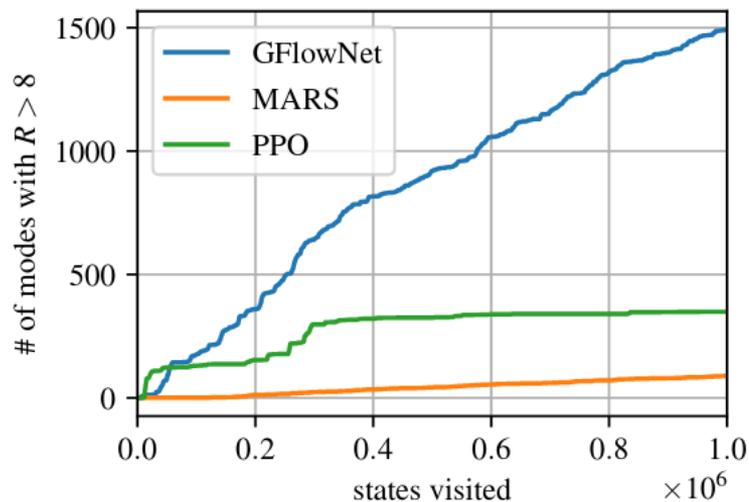
modes = Bemis-Murcko scaffolds

Works for Batch Active Learning too!

Average return over 3 runs of the top-k candidates in an iterative batch generation approach

Reinforcement Learning for therapeutics

- Very big, largely untapped potential!
- **Reward design is crucial**
- **Need to consider specifics of the problem**

- Great opportunity to improve existing algorithms!
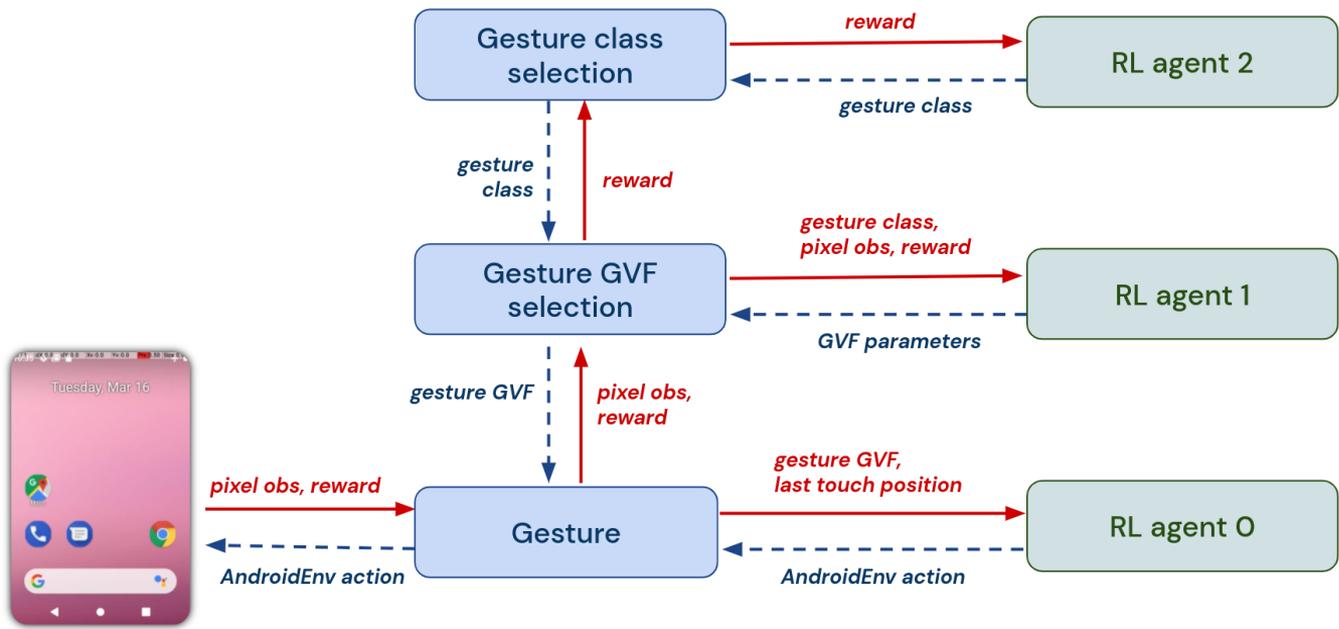- **Sample efficiency of RL needs to be improved**

3. RL for building general agents

- General computational tools
- Ability to incorporate many sources of data and powerful architectures
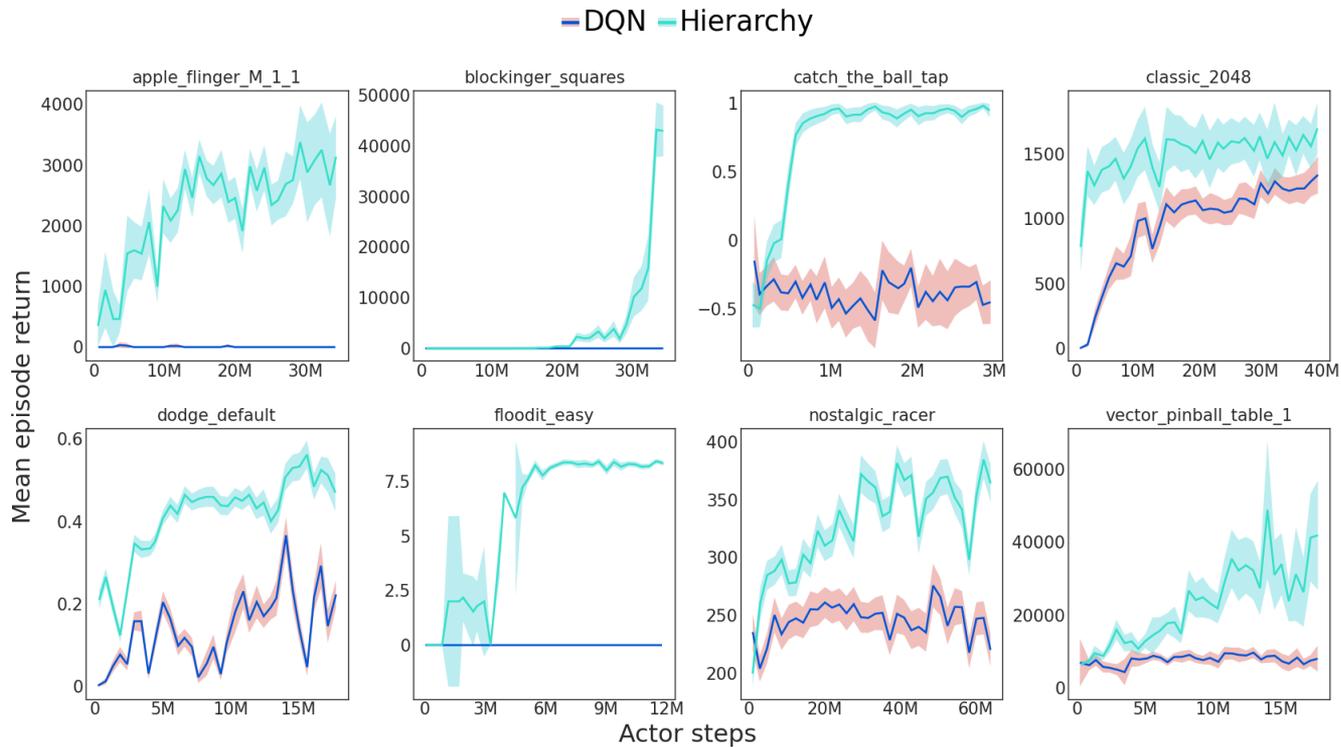- But training them from interaction!

Example: Training to interact with all Android apps
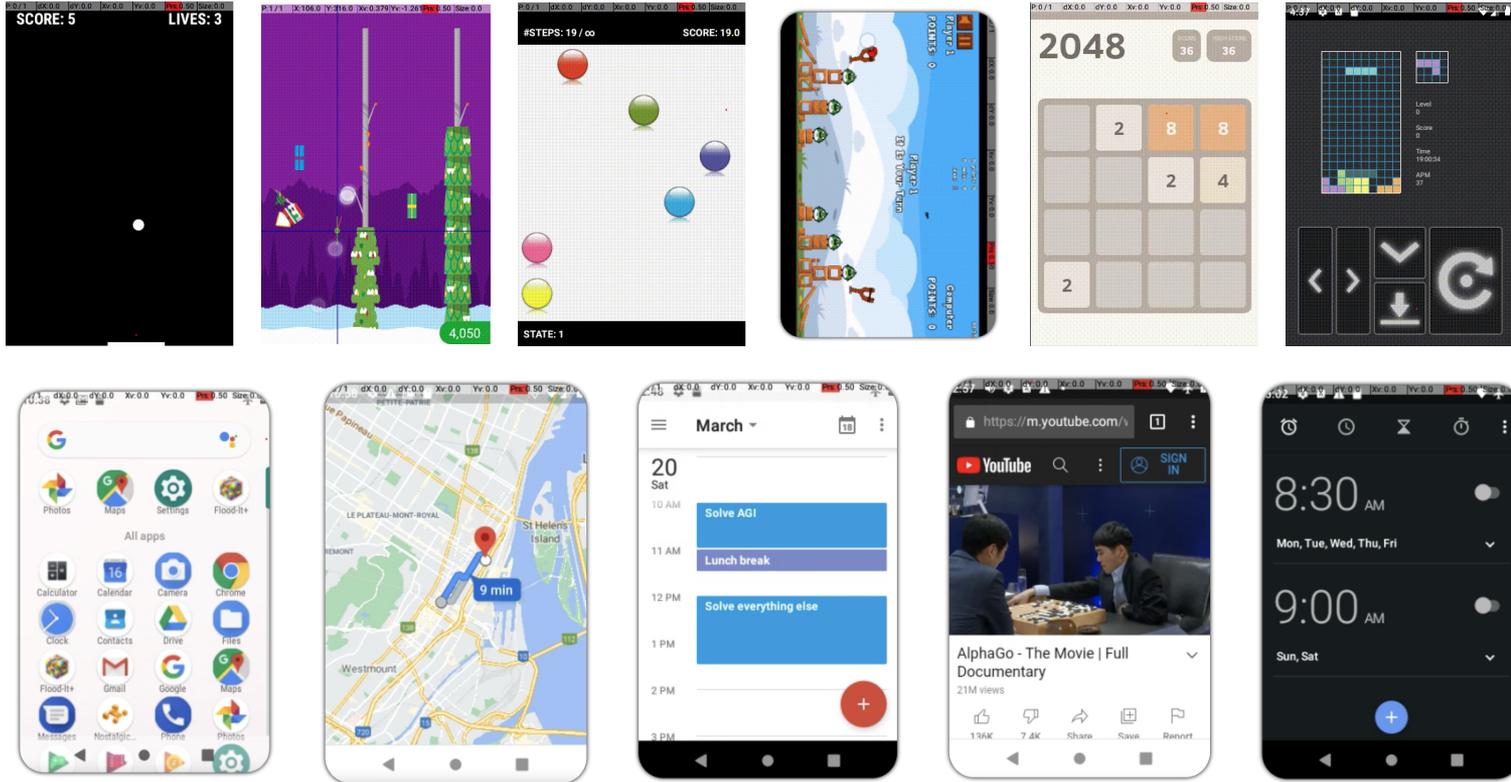
How: Hierarchical RL (and now maybe large models)

Some results, still lots to do

What does Reinforcement Learning Bring to AI?

Ability to develop new skills and exceed humans!

A clear, unified algorithmic approach

Many open questions ! My favourites: continual representation learning, model-based RL, planning, uncertainty-driven/ purposeful exploration