String Theory meets Machine Learning - Exploring Standard Like Models

Robin Schneider

Uppsala University

November 2020

Reinforcement learning

What is reinforcement learning?

- Optimal control of (in)completely-known Markov decision processes.
- Training a learning agent to maximize a predefined goal by optimizing actions that manipulate its environment.
- Third branch of machine learning.
- Resembles most closely human learning.

Reinforcement learning

What is reinforcement learning?

- Optimal control of (in)completely-known Markov decision processes.
- Training a learning agent to maximize a predefined goal by optimizing actions that manipulate its environment.
- Third branch of machine learning.
- Resembles most closely human learning.

Main challenge of reinforcement learning:

- Balancing exploration and exploitation. A trade-off which has been studied for many years but remains unsolved.

Why Reinforcement learning?

Reinforcement Learning has a proven track record of beating us:

- AlphaGo Zero - Won against world champion in GO, a game with 10^{170} valid board positions.
- OpenAI Five - Won against world champions in DotA 2 [1912.06680].

Why Reinforcement learning?

Reinforcement Learning has a proven track record of beating us:

- AlphaGo Zero - Won against world champion in GO, a game with 10^{170} valid board positions.
- OpenAI Five - Won against world champions in DotA 2 [1912.06680].

Computations are plenty full:

- Up to 10^{428} topological inequivalent CY 3-folds [2008.01730]
- F-theory: $10^{272.000}$ flux vacua [1511.03209]

Why Reinforcement learning?

Reinforcement Learning has a proven track record of beating us:

- AlphaGo Zero - Won against world champion in GO, a game with 10^{170} valid board positions.
- OpenAI Five - Won against world champions in DotA 2 [1912.06680].

Computations are plenty full:

- Up to 10^{428} topological inequivalent CY 3-folds [2008.01730]
- F-theory: $10^{272.000}$ flux vacua [1511.03209]

Relevant applications for us:

- Branes with Brains [1903.11616]
- Explore and Exploit with Heterotic Line Bundle Models [2003.04817]
- (Learning to Unknot [2010.16263])

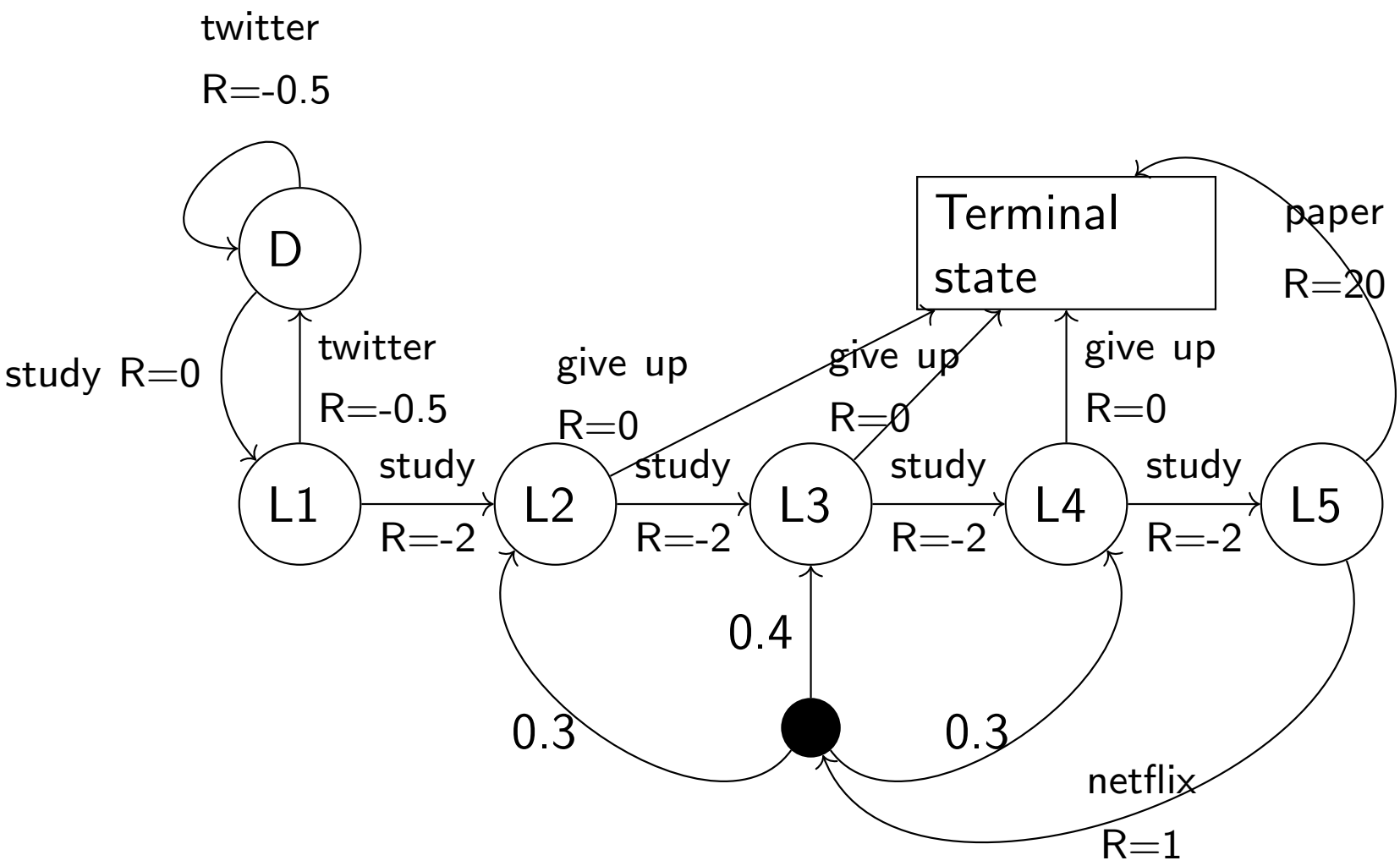
Markov Decision Process

Definition

A Markov decision process is a 4-tuple (S, A, P_a, R_a)*, where*

- S *is a set of states called the state space,*
- A *is a set of actions called the action space,*
- $P_a(s, s') = Pr(S_{t+1} = s' | S_t = s, A_t = a)$ *is the probability of transitioning into state* s' *given* s *and* a*,*
- $R_a(s, s')$ *is the immediate reward for state transition* $s \rightarrow s'$ *due to action* a*.*

A familiar MDP



Policy

How to pick actions?

Definition

A policy π is a distribution over actions given states,

$$\pi(a|s) = \mathbb{P}[A_t = a|S_t = s] \tag{1}$$

Policy

How to pick actions?

Definition

A policy π is a distribution over actions given states,

$$\pi(a|s) = \mathbb{P}[A_t = a|S_t = s] \tag{1}$$

Definition

The return G_t is the total discounted reward for time-step t.

$$G_t = R_{t+1} + \gamma R_{t+2} + ... = \sum_k \gamma^k R_{t+k+1}, \qquad \text{with } \gamma \in [0, 1] \tag{2}$$

State value function

Definition

The state value function $v_\pi(s)$ of an MDP is the expected return starting from state s and following policy π

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s] \tag{3}$$

State value function

Definition

The state value function $v_\pi(s)$ of an MDP is the expected return starting from state s and following policy π

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s] \tag{3}$$

Definition

The action value function $q_\pi(s, a)$ of an MDP is the expected return starting from state s, using action a, and then following policy π

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a] \tag{4}$$

Bellman equation

How to compute state value function?
Use **Bellman equations**:

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi[G_t|S_t = s] = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + ...|S_t = s] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma v(s_{t+1}) + ...|S_t = s] \end{aligned} \tag{5}$$

and

$$q_\pi(s, a) = \mathbb{E}_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1})|S_t = s, A_t = a] \tag{6}$$

Bellman equation

How to compute state value function?
Use **Bellman equations**:

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi[G_t|S_t = s] = \mathbb{E}_\pi[R_{t+1} + \gamma R_{t+2} + ...|S_t = s] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma v(s_{t+1}) + ...|S_t = s] \end{aligned} \tag{5}$$

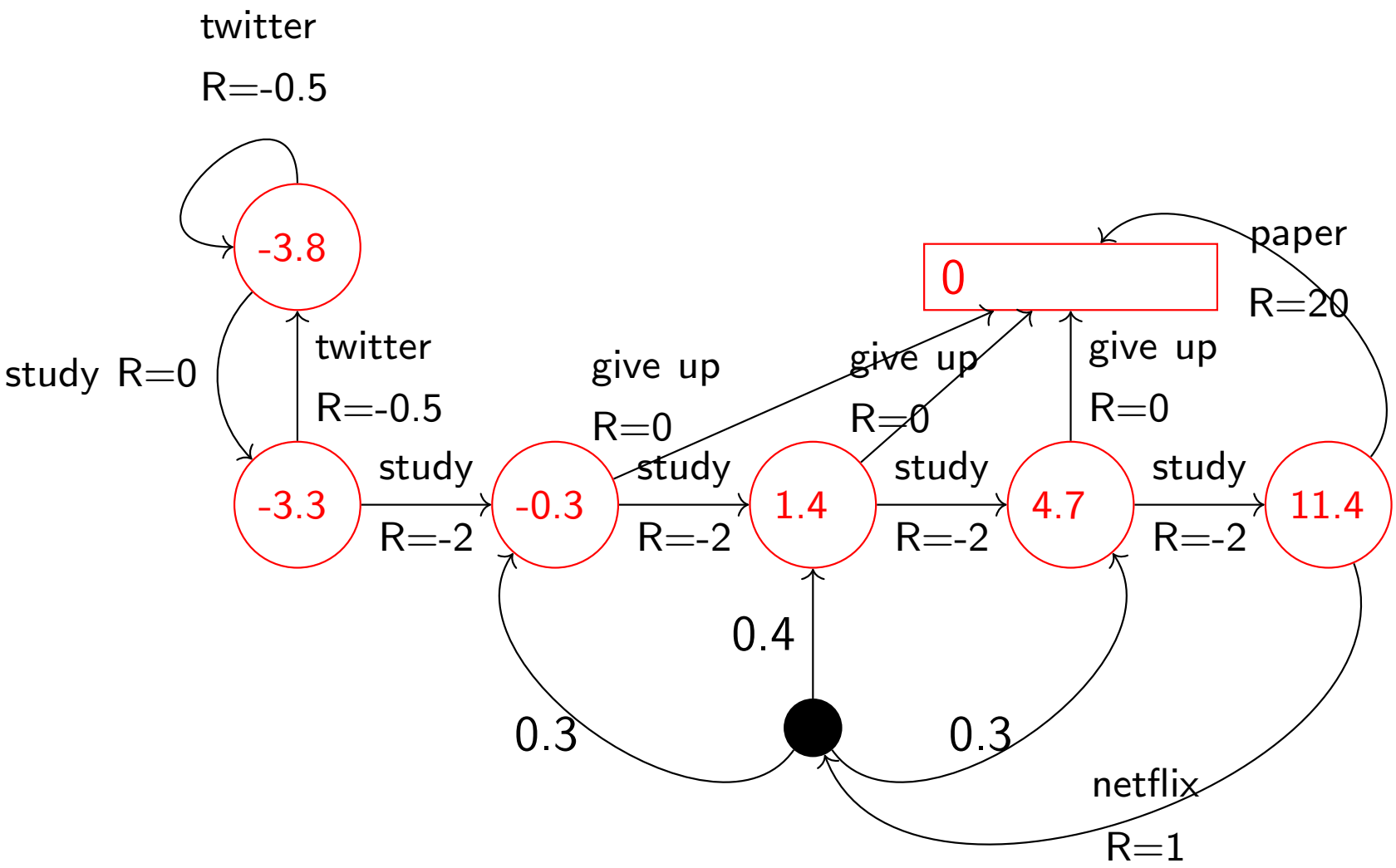
and

$$q_\pi(s, a) = \mathbb{E}_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1})|S_t = s, A_t = a] \tag{6}$$

Example **STmML**. Take a uniform policy and $\gamma = 1$:

$$v(L_5) = \frac{1}{2} \cdot 20 + \frac{1}{2}\left[1 + \frac{3}{10}v(L_2) + \frac{4}{10}v(L_3) + \frac{3}{10}v(L_4)\right] \tag{7}$$

MDP with $v(s)$


twitter
R=-0.5
-3.8
study R=0
twitter
R=-0.5
-3.3
study
R=-2
-0.3
give up
R=0
study
R=-2
1.4
give up
R=0
study
R=-2
4.7
0
give up
R=0
study
R=-2
11.4
paper
R=20
0.4
0.3
0.3
netflix
R=1

Optimal value function and policy

Definition

The optimal state value function $v_*(s)$ is the maximum value function over all policies

$$v_*(s) = \max_\pi v_\pi(s). \tag{8}$$

Theorem

Define a partial ordering over policies

$$\pi \geq \pi' \text{ if } v_\pi(s) \geq v_{\pi'}(s) \; \forall \; s. \tag{9}$$

Then for any Markov decision process

- *exists an optimal policy π_* that is better than or equal to all other policies, $\pi_* \geq \pi \; \forall \; \pi$,*
- *all optimal policies achieve the optimal value function $v_{\pi_*}(s) = v_*(s)$.*

Literature

How do we find the optimal policy or state value function? By consulting the literature :)

- Youtube Lecture series: David Silver - Introduction to Reinforcement Learning
- Book: Sutton and Barto - Introduction to Reinforcement Learning
- (Optional) Interdisciplinary course at Uppsala University

Actor Critic models

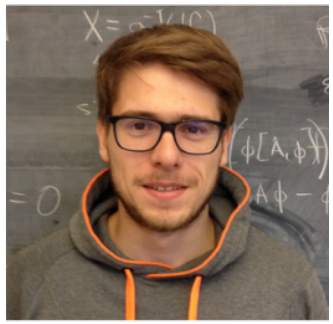
What happens if our state space is too vast to store in memory or too large to explore at all? Neural networks come to our rescue, we can approximate π or $v_\pi(s)$ (deep reinforcement learning).

Actor Critic models

What happens if our state space is too vast to store in memory or too large to explore at all? Neural networks come to our rescue, we can approximate π or $v_\pi(s)$ (deep reinforcement learning).
Asynchronous Advantage Actor Critic (A3C) surpassed state of the art performance in many benchmarks [1602.01783]. They

- use a NN as Actor to update π,
- use a NN as Critic to update $q_\pi(s, a)$,
- can be trained on a single CPU by updating global parameters asynchronously from local agents,
- are stable and robust (on considered benchmarks).

A game



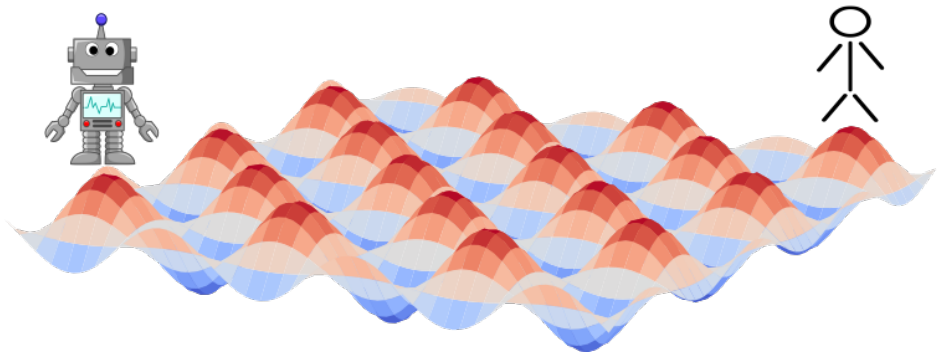


Figure: *Who finds the most realistic string vacua?*

Standard like models

Heterotic string compactification with three ingredients [1106.4804,1202.1757,1307.4787].

1. Calabi Yau manifold M.
2. Line bundle sum $V = \otimes_{a=1}^{5} L_a$.
3. Freely acting discrete symmetry Γ for Wilson line.

For example:

$$\mathcal{M}_{5302} = \left[\begin{array}{c||ccc} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \end{array}\right]^{6,30}_{-48} \quad \text{and} \quad V = \begin{bmatrix} -1 & 0 & 0 & 0 & 1 \\ 4 & -3 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 \\ 1 & 1 & 0 & -2 & 0 \end{bmatrix}$$

and $|\Gamma| = 2$. There are a total of 6294 such models.

Reward structure

1. V has to be an $S(U(1)^5)$ bundle, thus $c_1(V) = 0$. R_{max}=5.
2. The *weak* stability constraint; each line bundle has slope zero somewhere in the Kähler cone; $\mu(L_a) = 0, a = 1, ..., 5$. R_{max}=2.
3. There are three fermionic matter generations; the index of each line bundle is in the range $-3|\Gamma| \leq \text{index}(L_a) \leq 0$, where $|\Gamma|$ is the rank of the freely acting symmetry. R_{max}= 10.
4. There are three fermionic matter generations; the index of V is determined by $\text{ind}(V) \stackrel{!}{=} -3|\Gamma|$. R=$10^2$.
5. The Bianchi identity and Bogomolov bounds impose $0 < c_2(V) \leq c_2(\mathcal{M})$. R=$10^4$.
6. Exclude exotic Higgs triplets; $\text{ind}(L_a \otimes L_b) < 0$. R=$10^4$.
7. Require at least one Higgs doublet; $h^2(\wedge^2 V) \geq 0$. R=10^6.
8. There are no antigenerations, $h^2(V) \stackrel{!}{=} 0$. R=10^7.
9. V needs to be slope stable somewhere in the Kähler cone.

Stacking I

Environment: Stacking of line bundles L_{1-4} on top of each other.
Observation space: The line bundle sum V. Hence $S_t \sim \mathbb{Z}^{(5,\text{nProj})}$.
Action space: Pick a slope stable line bundle with $h^1(L) \leq 3|\Gamma|$ from a precompiled list of n_{line} line bundles.
# of configurations: $n_{\text{conf}} = n_{\text{line}}^4$. Take e.g. 5302 with max charge $q_{\text{max}} = 2$ and $|\Gamma| = 2$, then $n_{\text{line}} = 2890$ and $n_{\text{conf}} \approx 7 \cdot 10^{13}$.
Example:

$$\begin{bmatrix} -1 & 0 & 0 & 0 & 1 \\ 2 & 0 & -1 & -1 & 0 \\ -2 & -1 & 0 & -2 & 5 \\ 0 & -1 & 0 & 2 & -1 \\ 0 & 2 & -2 & 2 & -2 \\ 2 & 2 & 1 & 0 & -5 \end{bmatrix} \overset{A_t=7}{\longrightarrow} \begin{bmatrix} -1 & -2 & 0 & 0 & 3 \\ 2 & -2 & -1 & -1 & 2 \\ -2 & 0 & 0 & -2 & 4 \\ 0 & 2 & 0 & 2 & -4 \\ 0 & 0 & -2 & 2 & 0 \\ 2 & 2 & 1 & 0 & -5 \end{bmatrix} \tag{10}$$

Stacking II

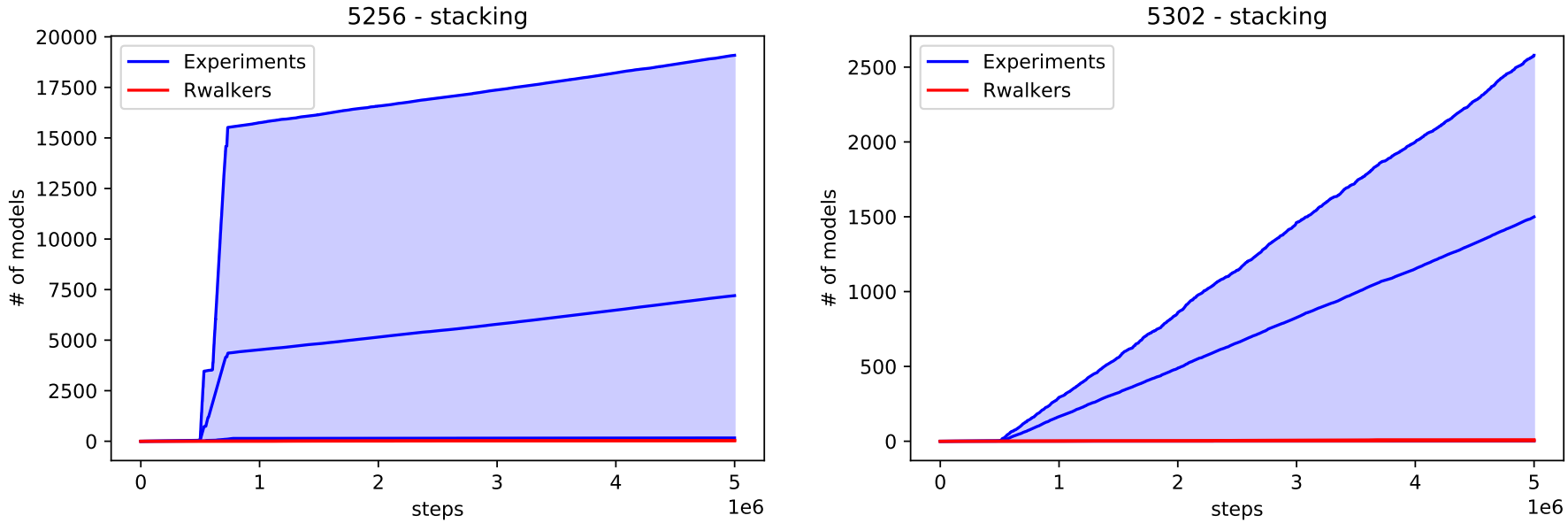
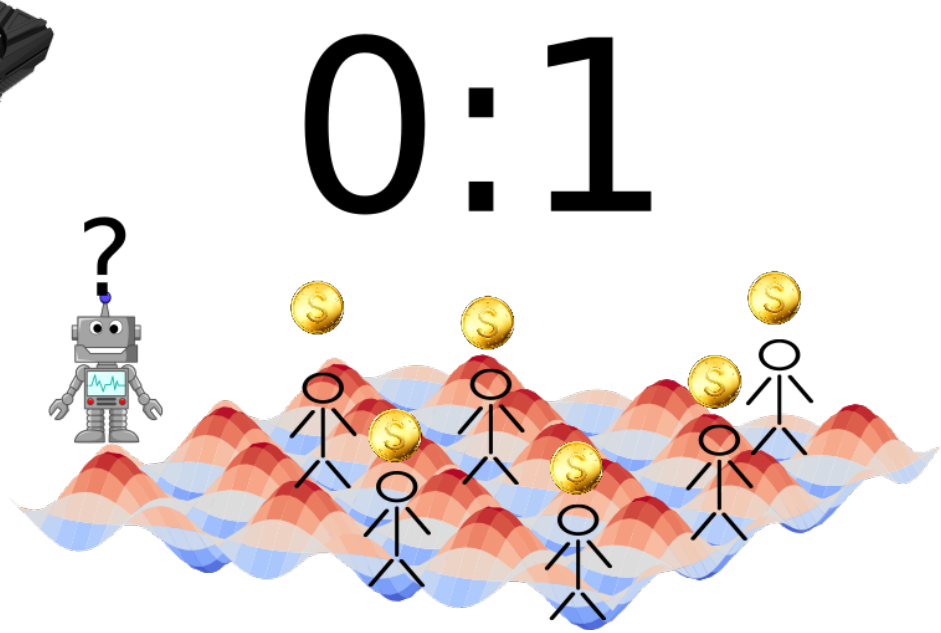


Figure: *Number of models found in two selected sets of stacking experiments (in blue) for the manifolds 5256 and 5302 plotted for comparison with each five random walkers (in red).*

Half time



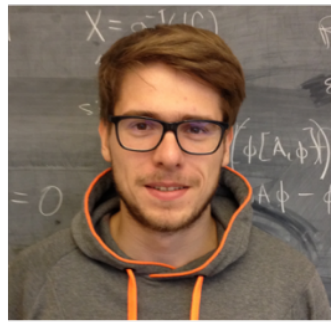


Figure: *An early lead.*

Flipping I

Environment: Flipping of charges q_i^j in L_{1-4}.
Observation space: The line bundle sum V. Hence $S_t \sim \mathbb{Z}^{(5,\text{nProj})}$.
Action space: Pick a charge q_i^j and add ± 1. Thus there are $A_t \in \{1, ..., 4 \cdot 2 \cdot \text{nProj}\}$ actions.
# of configurations: $n_{\text{conf}} = (2 \cdot q_{\text{max}} + 1)^{4 \cdot h^{1,1}}$. Take e.g. 5302 with max charge $q_{\text{max}} = 2$, then $n_{\text{conf}} \approx 5 \cdot 10^{16}$.
Example:

$$\begin{bmatrix} 1 & 1 & -1 & 0 & -1 \\ -1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & -1 & -1 \\ 1 & 1 & -1 & -1 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \end{bmatrix} \overset{A_t=0}{\longrightarrow} \begin{bmatrix} 2 & 1 & -1 & 0 & -2 \\ -1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & -1 & -1 \\ 1 & 1 & -1 & -1 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{11}$$

Flipping II

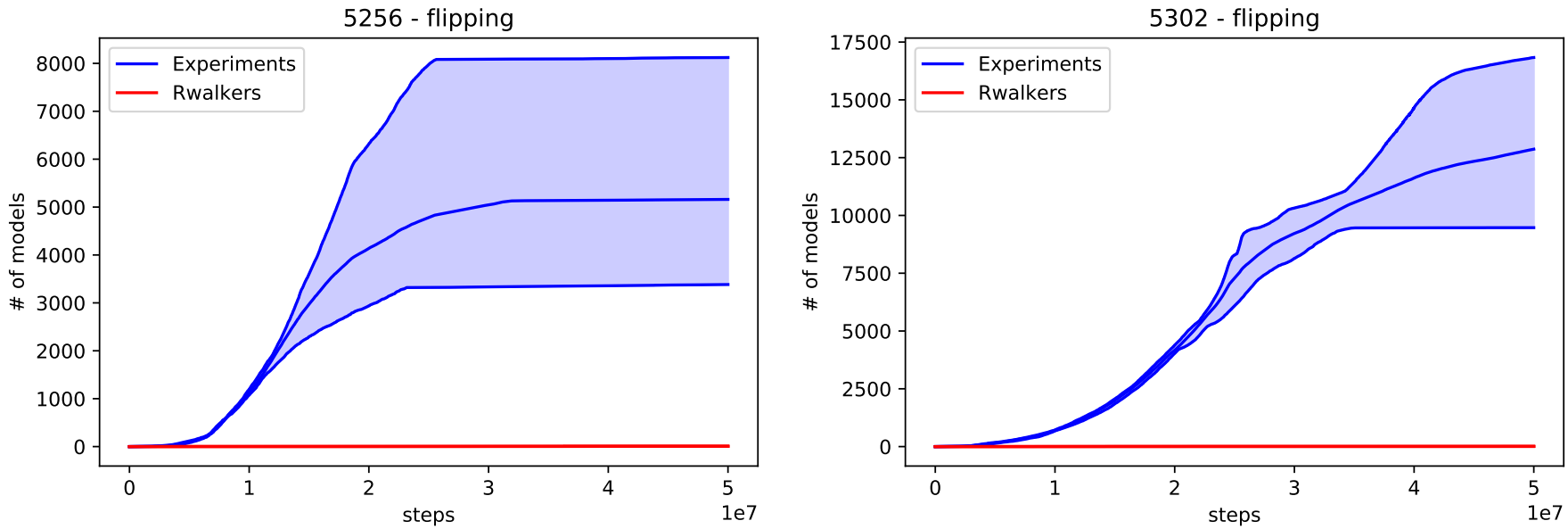


Figure: *Number of models found in two selected sets of flipping experiments (in blue) for the manifolds 5256 and 5302 plotted for comparison with each five random walkers (in red).*

Transfer learning

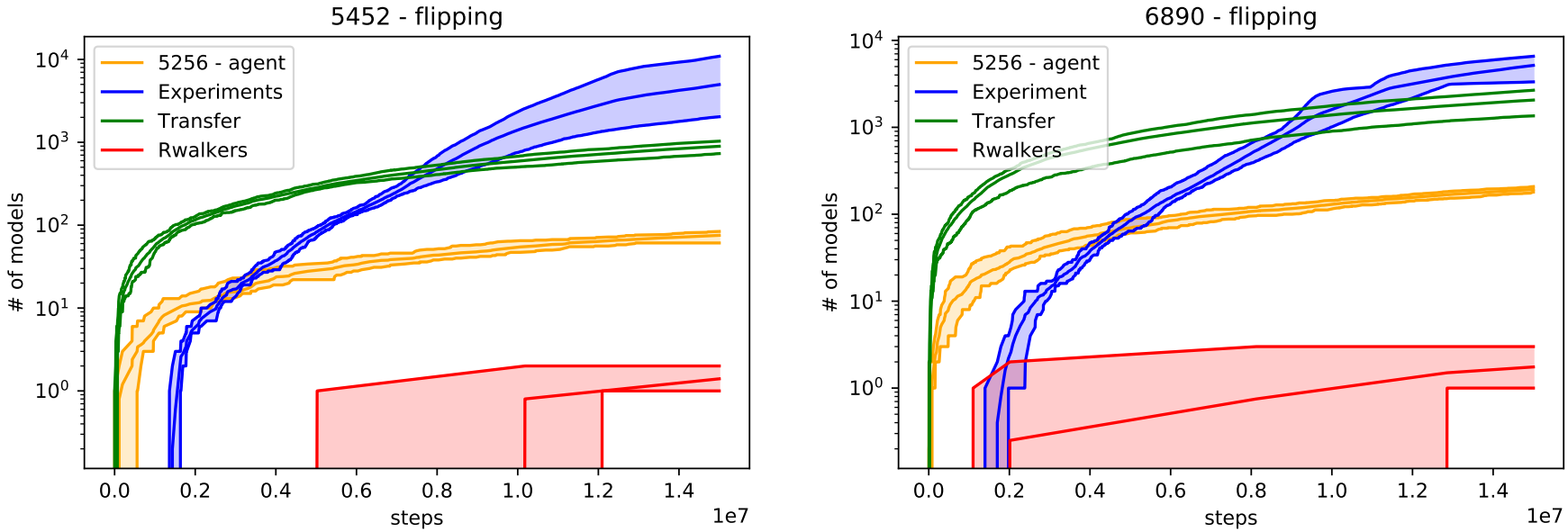
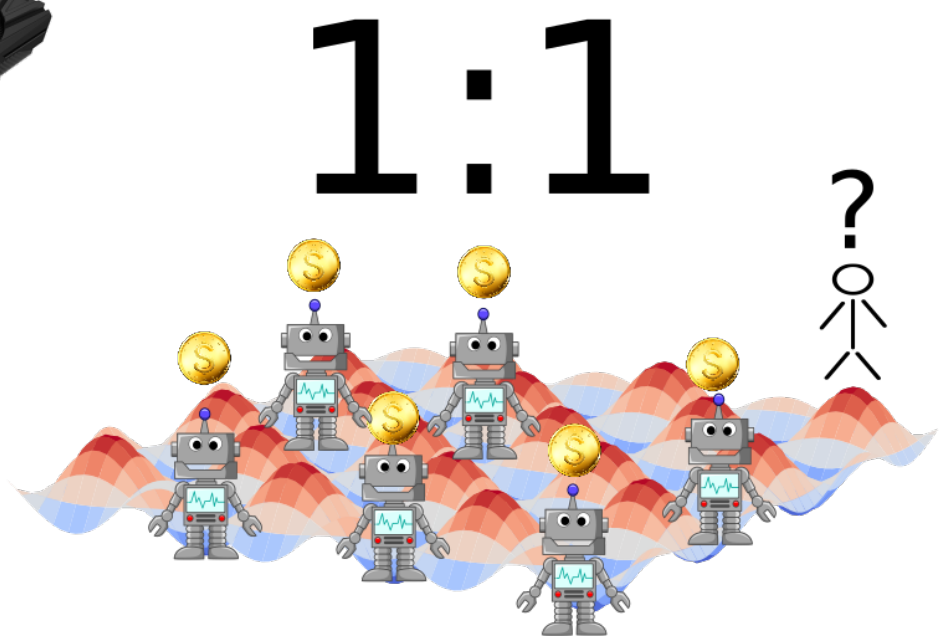


Figure: *Number of models found for selected sets of flipping experiments (in blue), random walkers (in red), pretrained agents (in yellow), and transfer agents (in green) on the manifolds 5452 and 6890. Note the logarithmic scale on the y-axis.*

Final Score



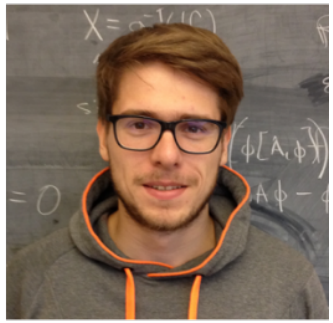


Figure: *Looks like a draw to me.*