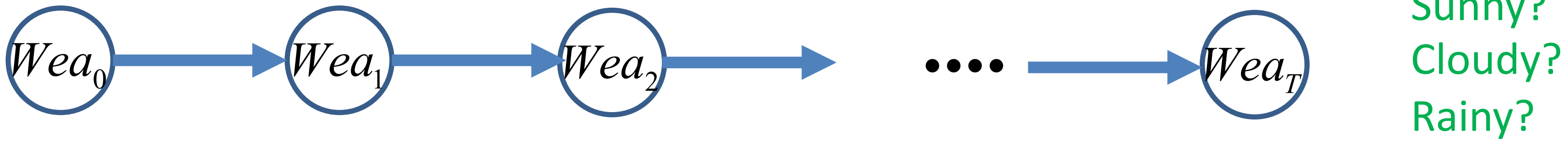
北京大學
PEKING UNIVERSITY

Thought Cloning: Learning to Think while Acting by Imitating Human Thinking

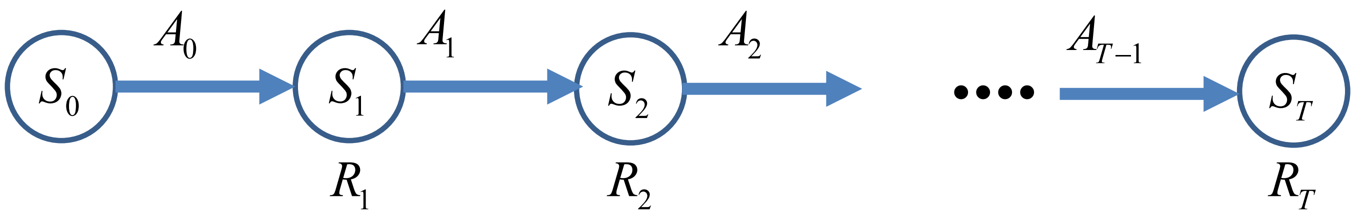
NIPS 2023

Kun-Peng Ning

Markov Property



Markov Property: $P(Wea_T \mid Wea_{T-1}, Wea_{T-2}, \cdots, Wea_0) = P(Wea_T \mid Wea_{T-1})$

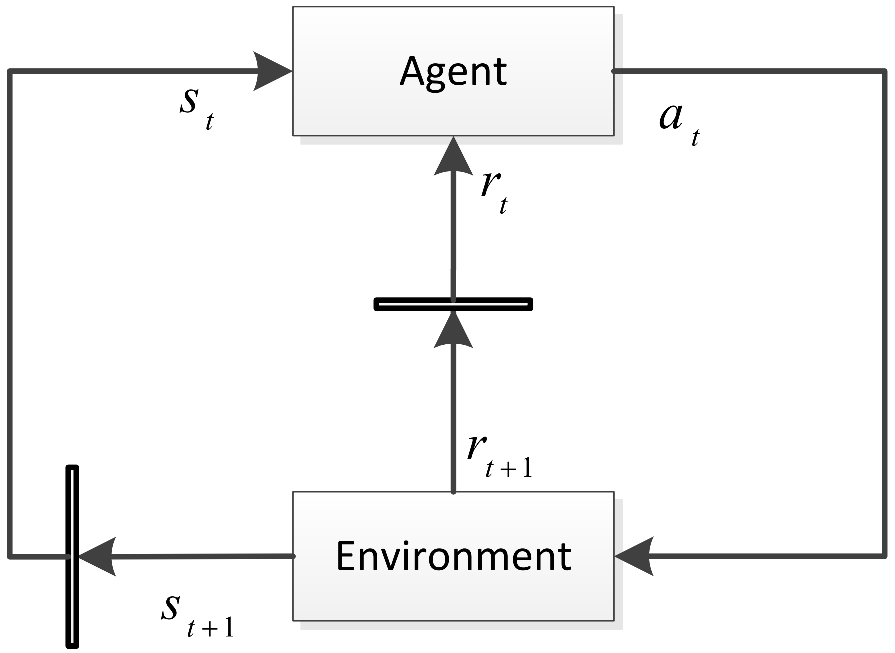


Markov Property in Decision Process:

$$P(r_T, s_T \mid s_0, a_0, r_1, s_1, a_1, \cdots s_{T-1}, a_{T-1}) = P(r_T, s_T \mid s_{T-1}, a_{T-1})$$

Formalization

Agent-Environment Interface



$< \mathrm{A}, \mathrm{S}, \mathrm{T}, R >$

State Space S

Action Space A

Transition Function T : $\mathrm{S} \times \mathrm{A} \rightarrow \mathrm{P}(\mathrm{S})$ $\qquad \mathrm{T}(s_{t+1} \mid s_t, a_t)$

Reward Function R: $\mathrm{S} \times \mathrm{A} \times \mathrm{S} \rightarrow \mathrm{P}(\quad)$ $\qquad P(r_t \mid s_t, a_t, s_{t+1})$

Policy $\pi : \mathrm{S} \rightarrow \mathrm{A}$ or $\mathrm{S} \rightarrow \mathrm{P}(\mathrm{A})$ $\qquad \pi(a_t \mid s_t)$

Goal

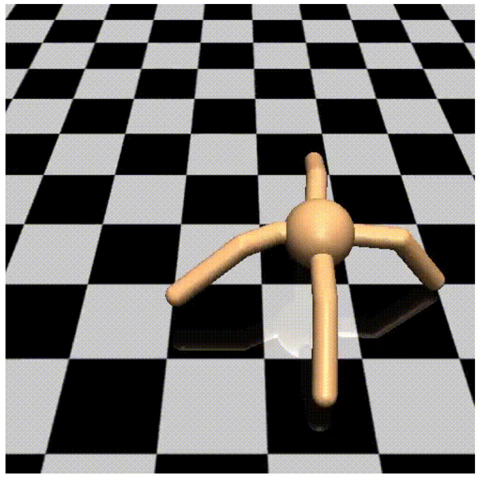
$$\begin{tikzcd} S_0 \arrow[r, "A_0"] & S_1 \arrow[r, "A_1"] & S_2 \arrow[r, "A_2"] & \cdots\cdots \arrow[r, "A_{T-1}"] & S_T \end{tikzcd}$$

$R_1 \quad R_2 \quad R_T$

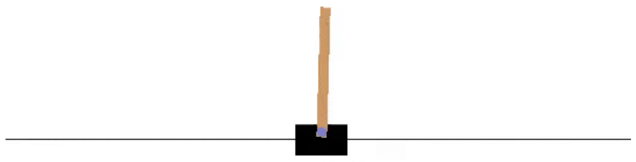
Long-term/Total/Accumulate Reward $G = R_1 + R_2 + \cdots + R_T$

To learn a polciy $\pi(A \mid S)$ to maximize $\mathrm{E}[G]$ in the environment

Score: XXX

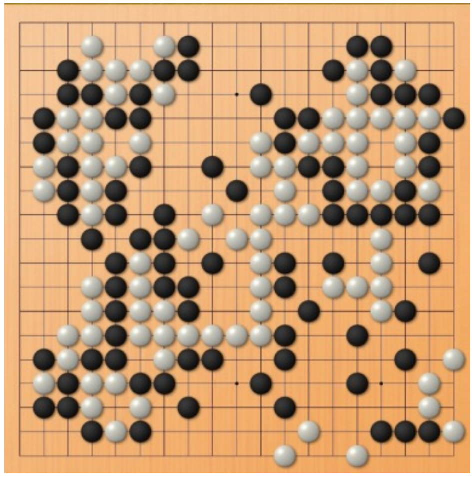


Run Distance



Time of Duration

Goal For Episodic Tasks



The Game will end in a specific time or a specific state!

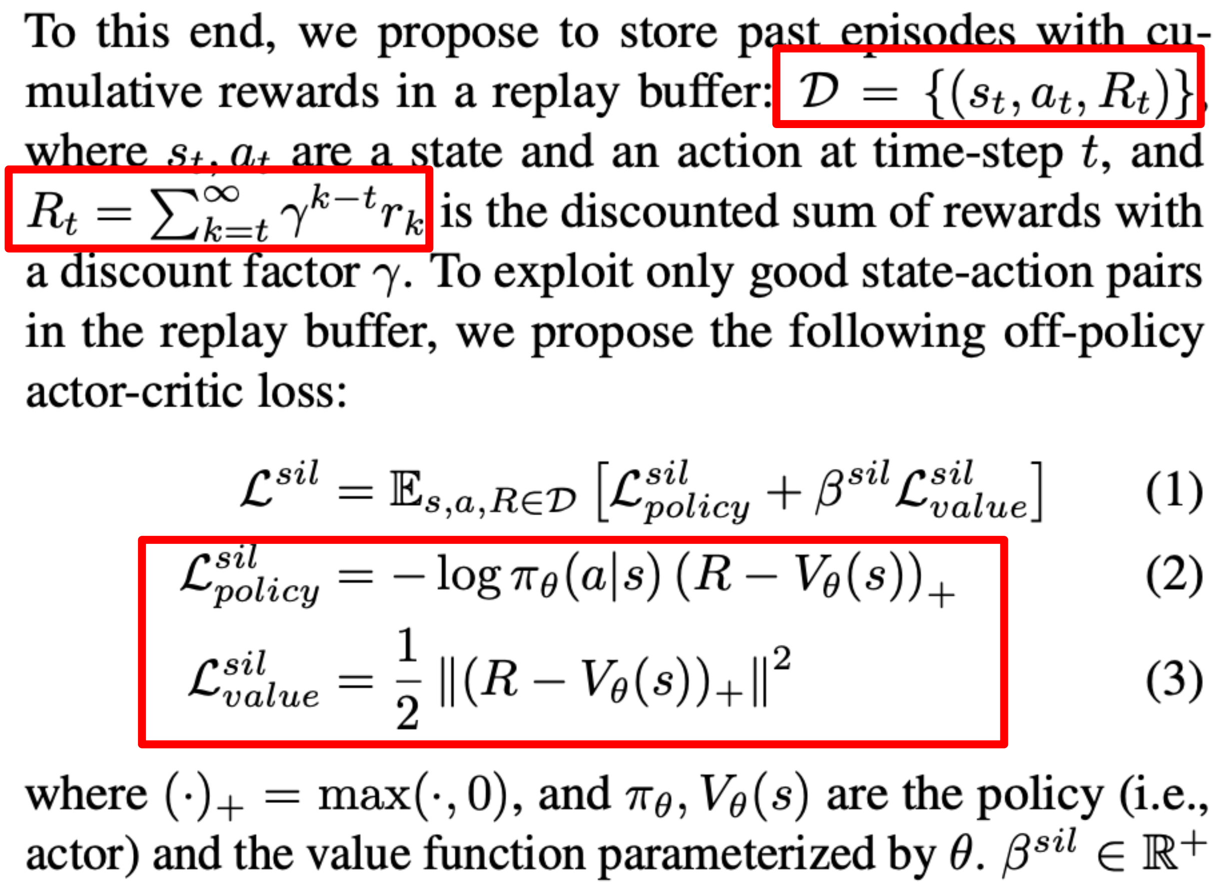
$$G = \sum_{t=0}^{T-1} R_{t+1}$$

Problem

- Challenges in Reinforcement Learning
  - Sample inefficient
  - Reward design (reward delay, sparse reward)
  - Offline RL
  - Imperfect demonstration
  - High-dimension (large search space, continual action space)
  - No test environment (Environment generalization)…

- Behavioral Cloning (Sample Inefficient)
  - Self-Imitation Learning
  - GAIL
  - Self-supervised Exploration
  - Learning from Human Preferences

Self-Imitation Learning



To this end, we propose to store past episodes with cumulative rewards in a replay buffer: $\boxed{\mathcal{D} = \{(s_t, a_t, R_t)\}}$, where s_t, a_t are a state and an action at time-step t, and $\boxed{R_t = \sum_{k=t}^{\infty} \gamma^{k-t} r_k}$ is the discounted sum of rewards with a discount factor γ. To exploit only good state-action pairs in the replay buffer, we propose the following off-policy actor-critic loss:

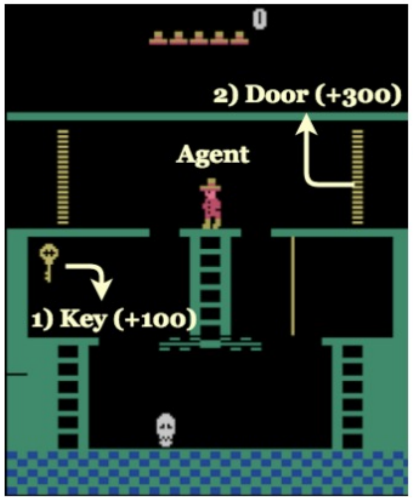
$$\mathcal{L}^{sil} = \mathbb{E}_{s,a,R\in\mathcal{D}}\left[\mathcal{L}^{sil}_{policy} + \beta^{sil}\mathcal{L}^{sil}_{value}\right] \qquad (1)$$

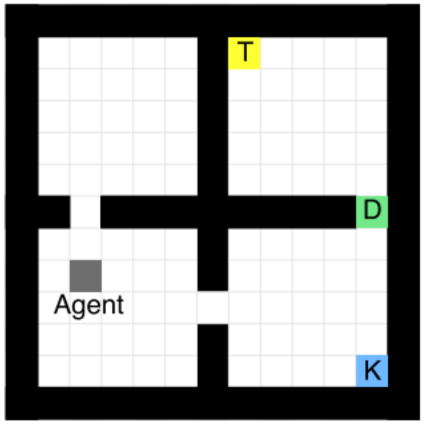
$$\boxed{\begin{aligned} \mathcal{L}^{sil}_{policy} &= -\log \pi_\theta(a|s)\left(R - V_\theta(s)\right)_+ && (2) \\ \mathcal{L}^{sil}_{value} &= \frac{1}{2}\left\|\left(R - V_\theta(s)\right)_+\right\|^2 && (3) \end{aligned}}$$

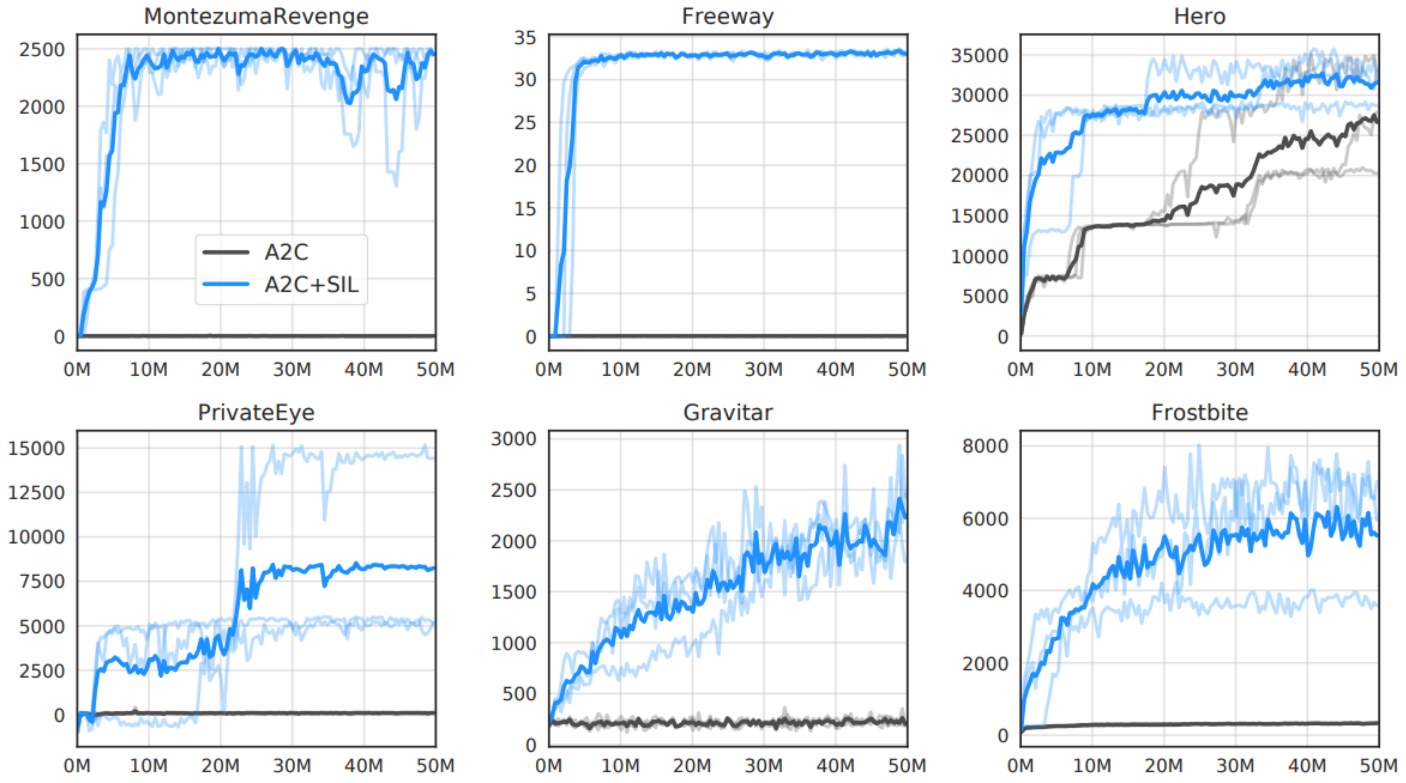
where $(\cdot)_+ = \max(\cdot, 0)$, and $\pi_\theta, V_\theta(s)$ are the policy (i.e., actor) and the value function parameterized by θ. $\beta^{sil} \in \mathbb{R}^+$

ICML'18
Google Brain
Citation 308

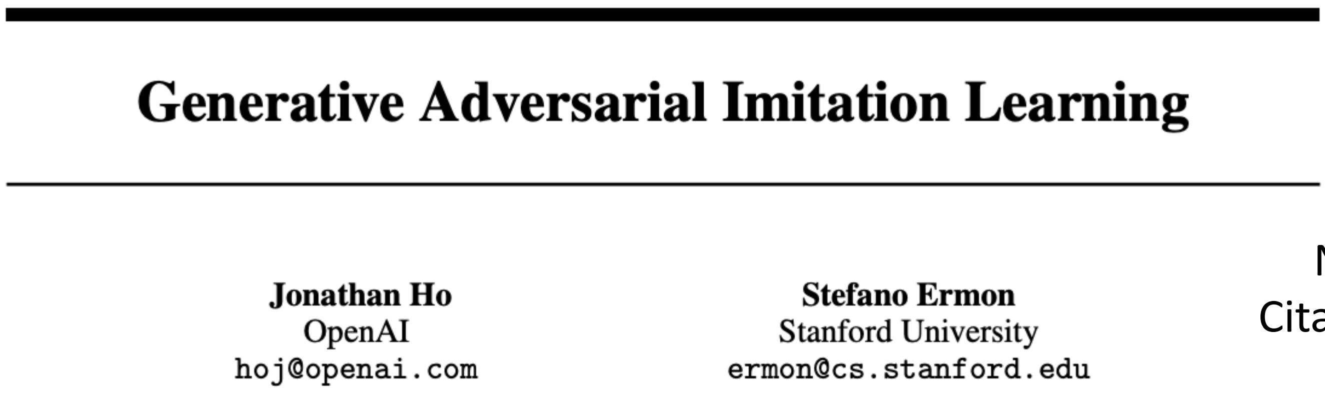
Self-Imitation Learning







GAIL

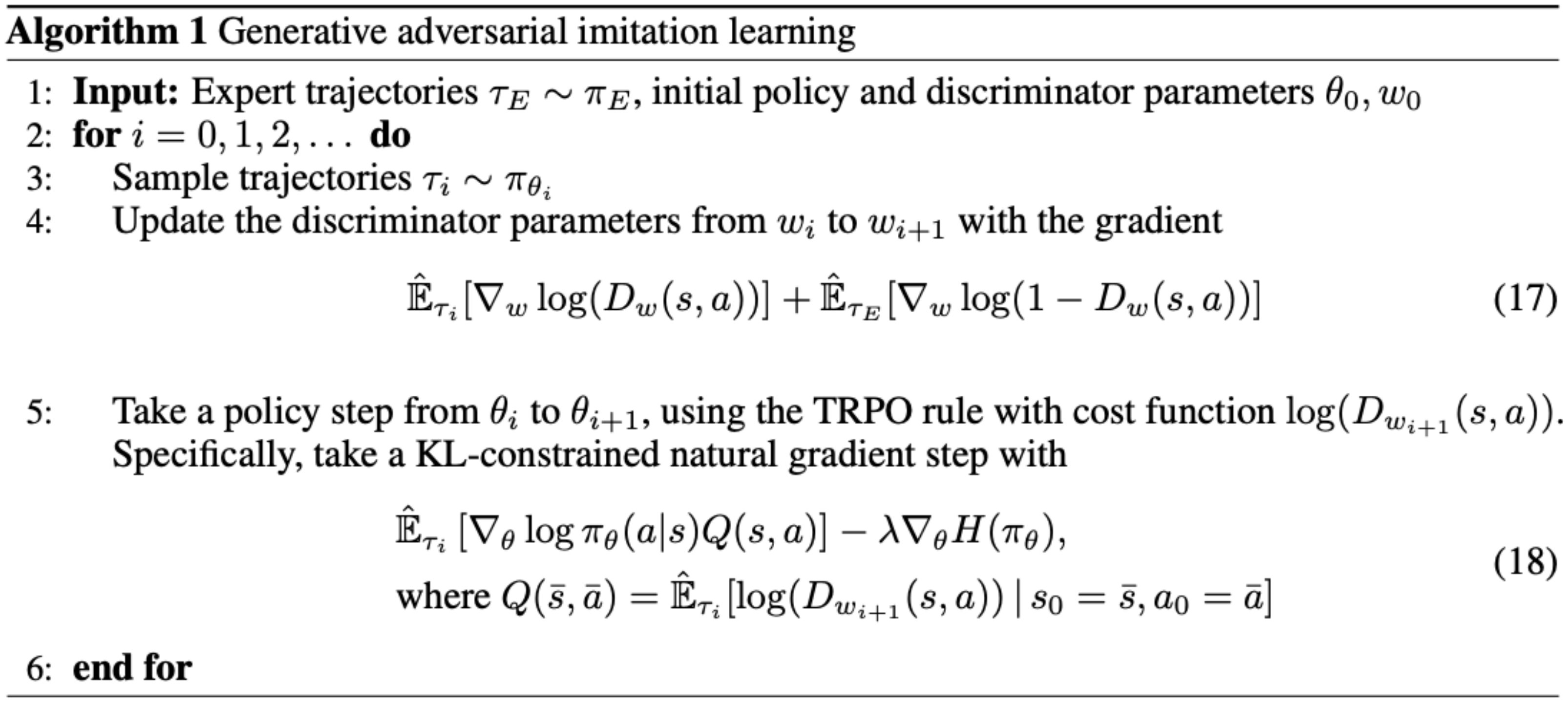


Generative Adversarial Imitation Learning

Jonathan Ho
OpenAI
hoj@openai.com

Stefano Ermon
Stanford University
ermon@cs.stanford.edu

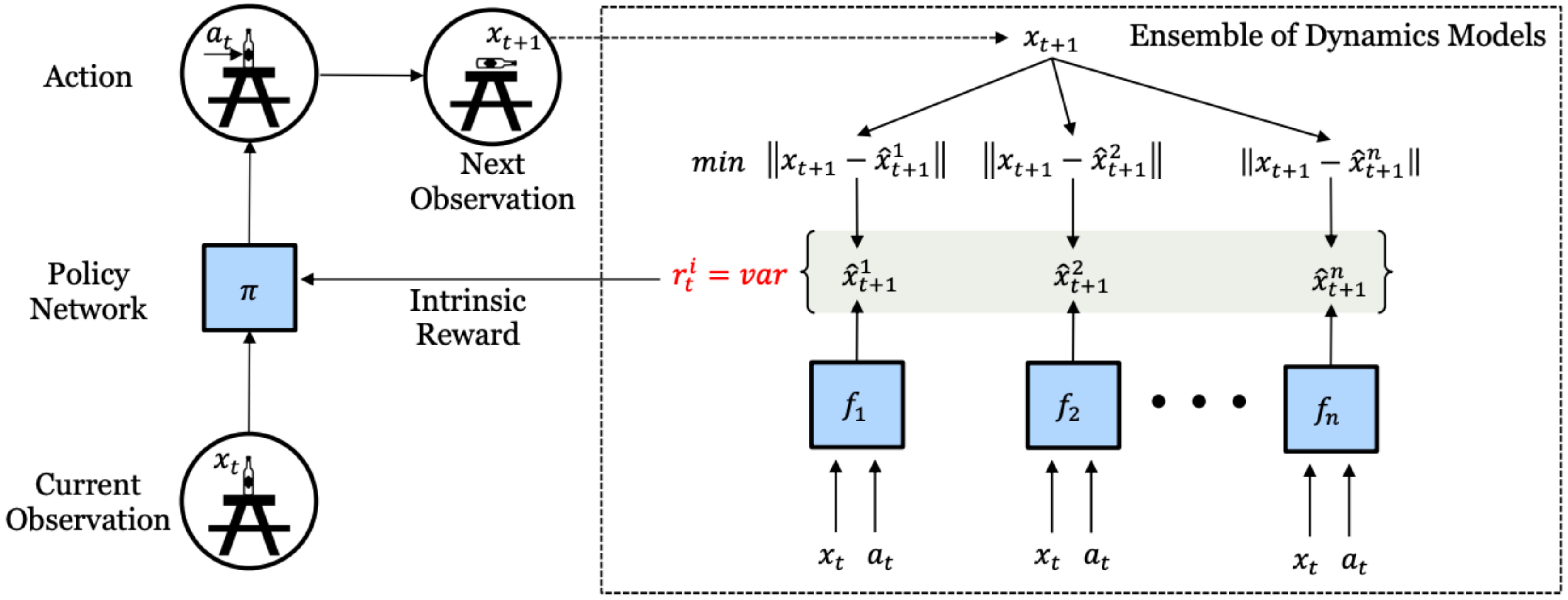
NIPS'16
Citation 3176

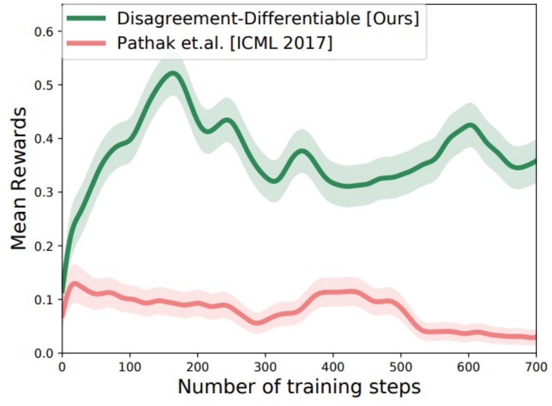


Algorithm 1 Generative adversarial imitation learning

```
1: Input: Expert trajectories $\tau_E \sim \pi_E$, initial policy and discriminator parameters $\theta_0, w_0$
2: for $i = 0, 1, 2, \dots$ do
3:     Sample trajectories $\tau_i \sim \pi_{\theta_i}$
4:     Update the discriminator parameters from $w_i$ to $w_{i+1}$ with the gradient
           $$\hat{\mathbb{E}}_{\tau_i}[\nabla_w \log(D_w(s,a))] + \hat{\mathbb{E}}_{\tau_E}[\nabla_w \log(1 - D_w(s,a))] \quad (17)$$
5:     Take a policy step from $\theta_i$ to $\theta_{i+1}$, using the TRPO rule with cost function $\log(D_{w_{i+1}}(s,a))$.
       Specifically, take a KL-constrained natural gradient step with
           $$\hat{\mathbb{E}}_{\tau_i}\left[\nabla_\theta \log \pi_\theta(a|s) Q(s,a)\right] - \lambda \nabla_\theta H(\pi_\theta),$$
           $$\text{where } Q(\bar{s},\bar{a}) = \hat{\mathbb{E}}_{\tau_i}[\log(D_{w_{i+1}}(s,a)) \mid s_0 = \bar{s}, a_0 = \bar{a}] \quad (18)$$
6: end for
```

Self-supervised Exploration


Action
a_t
x_{t+1}
Next Observation
Policy Network
π
Intrinsic Reward
Current Observation
x_t
Ensemble of Dynamics Models
x_{t+1}
$min \; \|x_{t+1} - \hat{x}^1_{t+1}\| \quad \|x_{t+1} - \hat{x}^2_{t+1}\| \quad \|x_{t+1} - \hat{x}^n_{t+1}\|$
$r^i_t = var \{ \hat{x}^1_{t+1} \; \hat{x}^2_{t+1} \; \hat{x}^n_{t+1} \}$
f_1
f_2
f_n
$x_t \; a_t$

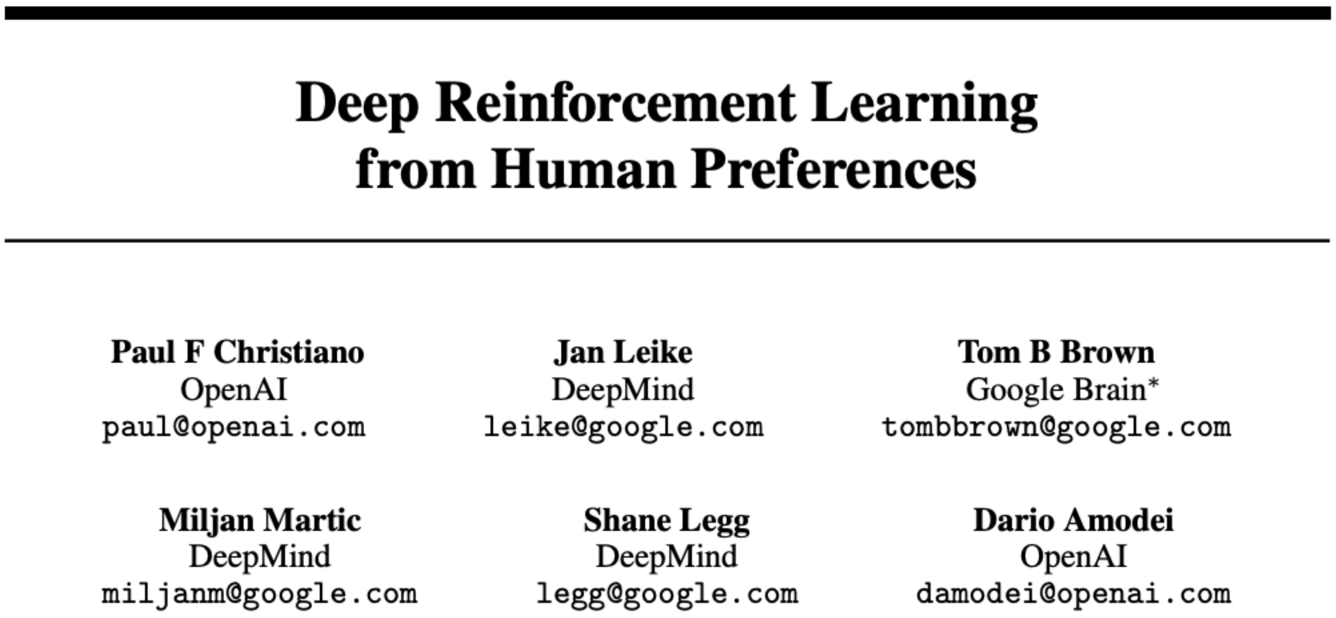


(b) Real Robot

(c) Real Robot Setup

ICML’19
UC Berkelely
Citation 333

Learning from Human Preferences



Deep Reinforcement Learning from Human Preferences

Paul F Christiano
OpenAI
paul@openai.com

Jan Leike
DeepMind
leike@google.com

Tom B Brown
Google Brain*
tombbrown@google.com

Miljan Martic
DeepMind
miljanm@google.com

Shane Legg
DeepMind
legg@google.com

Dario Amodei
OpenAI
damodei@openai.com

NIPS’17
OpenAI & DeepMind
Citation 1461

Learning from Human Preferences

- At each time t:
  - Observation $o_t \in O$
  - Action $a_t \in A$
  - ~~Reward $r_t \in R$~~

- Trajectory segment :
  - $\sigma = ((o_0, a_0), (o_1, a_1), \dots (o_{k-1}, a_{k-1})) \in (O \times A)^k$
  - $\sigma^1 \succ \sigma^2$: The human preferred trajectory segment σ^1 to trajectory segment σ^2.

- **Quantitative**: We say that preferences $\succ$ are generated by a reward function $r: O \times A \to R$ if

$$((o_0^1, a_0^1), \dots, (o_{k-1}^1, a_{k-1}^1)) \succ ((o_0^2, a_0^2), \dots, (o_{k-1}^2, a_{k-1}^2))$$

where

$$r(o_0^1, a_0^1) + \dots + r(o_{k-1}^1, a_{k-1}^1) > r(o_0^2, a_0^2) + \dots + r(o_{k-1}^2, a_{k-1}^2)$$

- **Qualitative**: Sometimes we have no reward function. In these cases, all we can do is qualitatively evaluate how well the agent satisfies to the human's preferences.

- Parameters:
  - Policy $\pi: O \to A$
  - Reward function estimate $\hat{r}: O \times A \to R$

- Three processes:

  1. The policy π interacts with the environment to produce a set of trajectories $\{\tau^1, \ldots, \tau^i\}$. The parameters of π are updated by a traditional reinforcement learning algorithm, in order to maximize the sum of the predicted rewards $r_t = \hat{r}(o_t, a_t)$.
  2. We select pairs of segments (σ^1, σ^2) from the trajectories $\{\tau^1, \ldots, \tau^i\}$ produced in step 1, and send them to a human for comparison.
  3. The parameters of the mapping $\hat{r}$ are optimized via supervised learning to fit the comparisons collected from the human so far.

- Trajectory segment σ^1 and trajectory segment σ^2:
  - σ^1 is better than σ^2 or σ^2 is better than σ^1.
  - Two segments are equally good.
  - Two segments can not be compared.

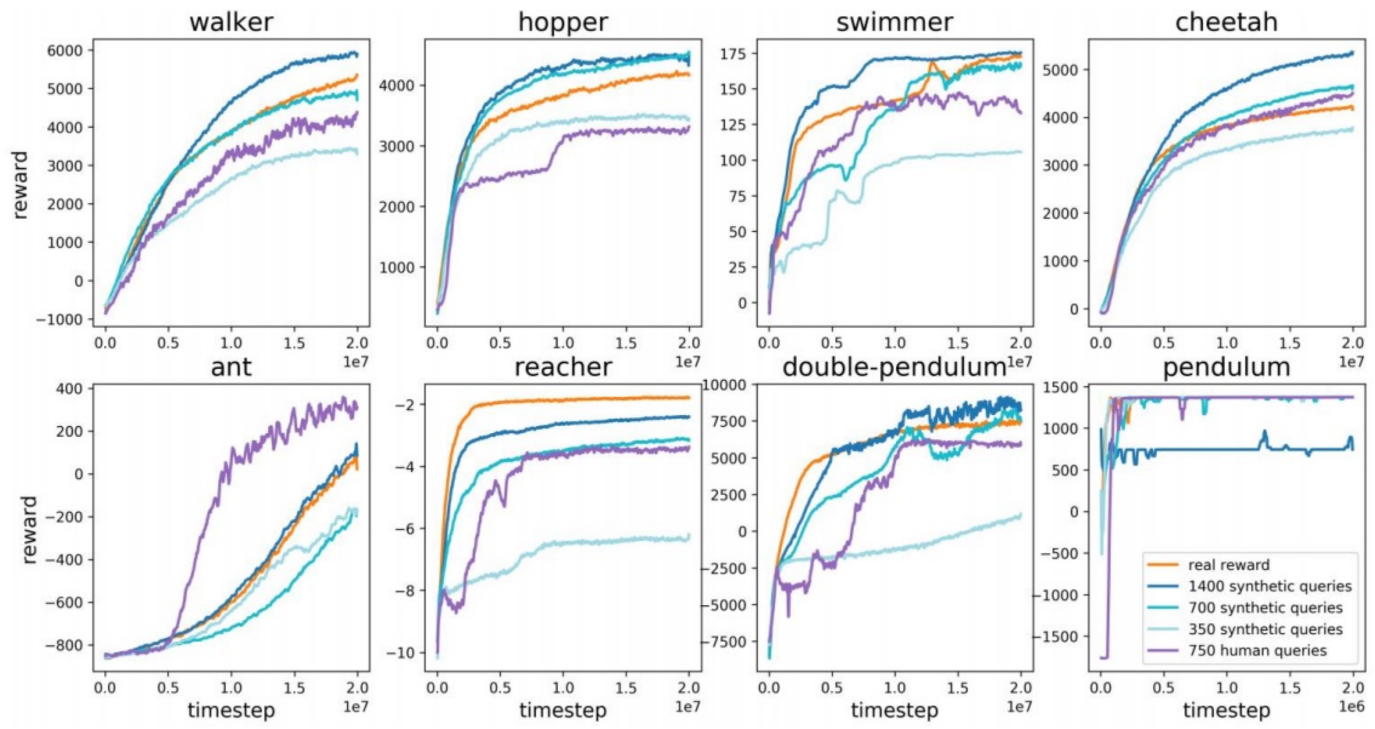
- Store in database :
  - $(\sigma^1, \sigma^2, \mu)$
  - σ^1 and σ^2 are the two segments.
  - μ is a distribution over {1, 2} indicating which segment the user preferred.
  - If two segments are incomparable, the comparison is not included in the database.

Learning from Human Preferences

- Fitting the reward function

$$\hat{P}[\sigma^1 \succ \sigma^2] = \frac{\exp \sum \hat{r}(o_t^1, a_t^1)}{\exp \sum \hat{r}(o_t^1, a_t^1) + \exp \sum \hat{r}(o_t^2, a_t^2)}.$$

$$\operatorname{loss}(\hat{r}) = - \sum_{(\sigma^1,\sigma^2,\mu)\in\mathcal{D}} \mu(1) \log \hat{P}[\sigma^1 \succ \sigma^2] + \mu(2) \log \hat{P}[\sigma^2 \succ \sigma^1].$$



Method

- Behavioral Cloning
  - Learning from demonstration (LfD)

$$\sigma = ((o_0, a_0), (o_1, a_1), \dots (o_{k-1}, a_{k-1})) \quad \rightarrow \pi(\cdot)$$

- Thought Cloning
  - Training AI agents to **think like humans do.**
  - Not just clone the behaviors of human demonstrations, **but also the thoughts humans have as the perform these behaviors.**

Method

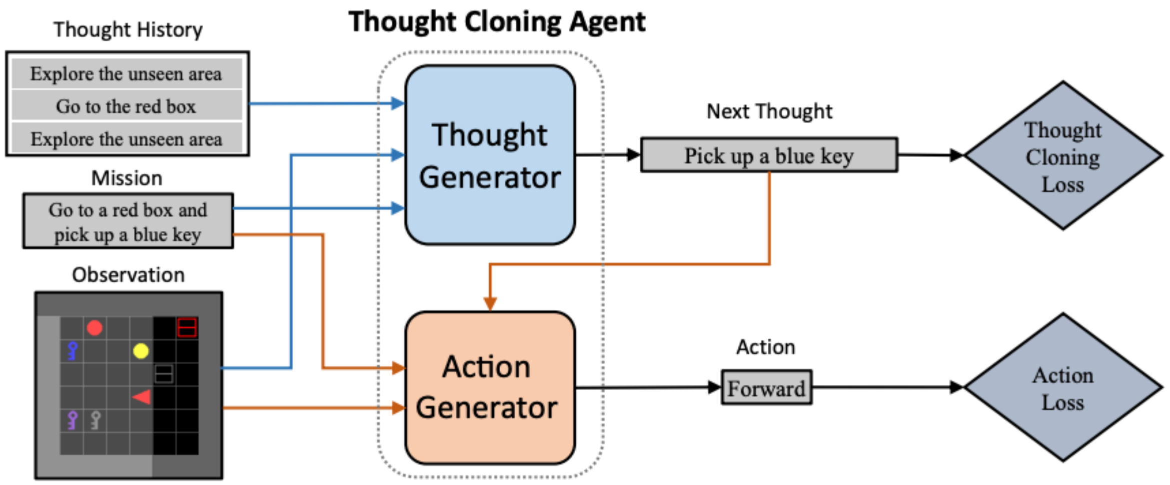


Figure 1: Overall framework for Thought Cloning (TC). The TC agent has two components: the Thought Generator and Action Generator. At each timestep, the TC agent receives an observation, a mission, and a history of thoughts as inputs. The Thought Generator generates thoughts, and the Action Generator generates actions conditioned on these thoughts. Generated thoughts and actions are compared to the ground truth from the demonstration dataset to calculate the loss.

Thought Dataset $\mathcal{D} = \{D_i\}_{i=1}^{N}$ $D_i = (m, \{(o_t, th_t, a_t)\}_{t=1}^{T})$

$$\min_{\theta_u, \theta_l} \sum_{t=1}^{T} -\alpha \log \pi_{\theta_u}(th_t | m, \{o_\tau\}_{\tau=1}^{t}, \{th_\tau\}_{\tau=1}^{t-1}) - \log \pi_{\theta_l}(a_t | m, \{o_\tau\}_{\tau=1}^{t}, th_t)$$

Method

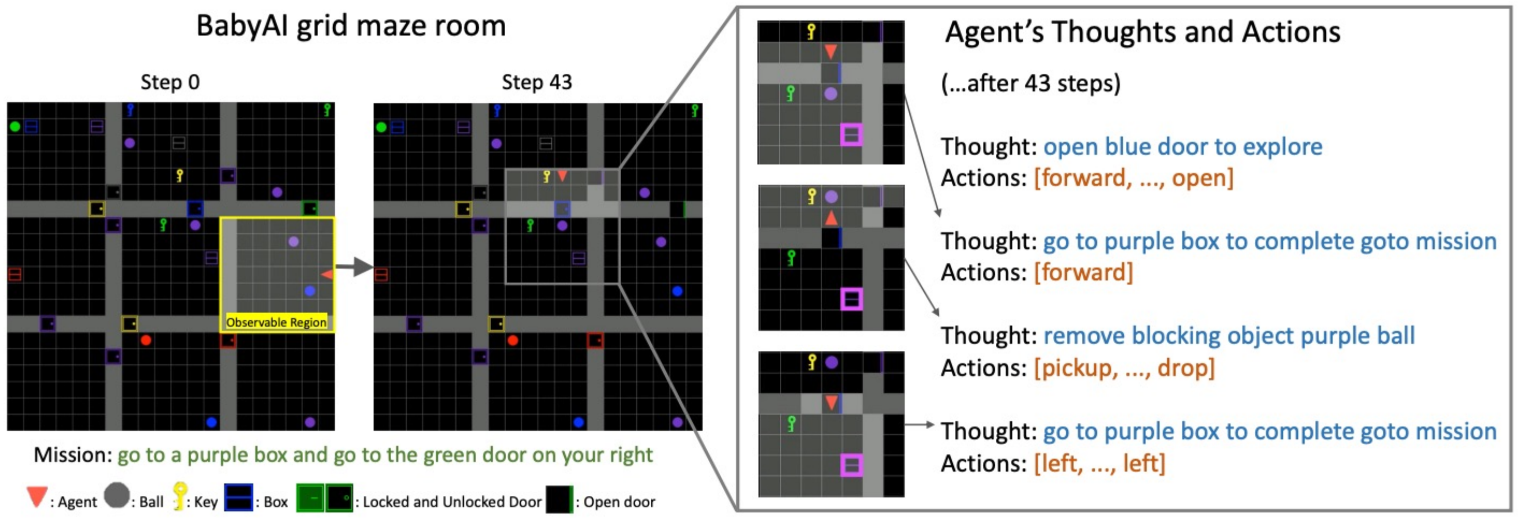
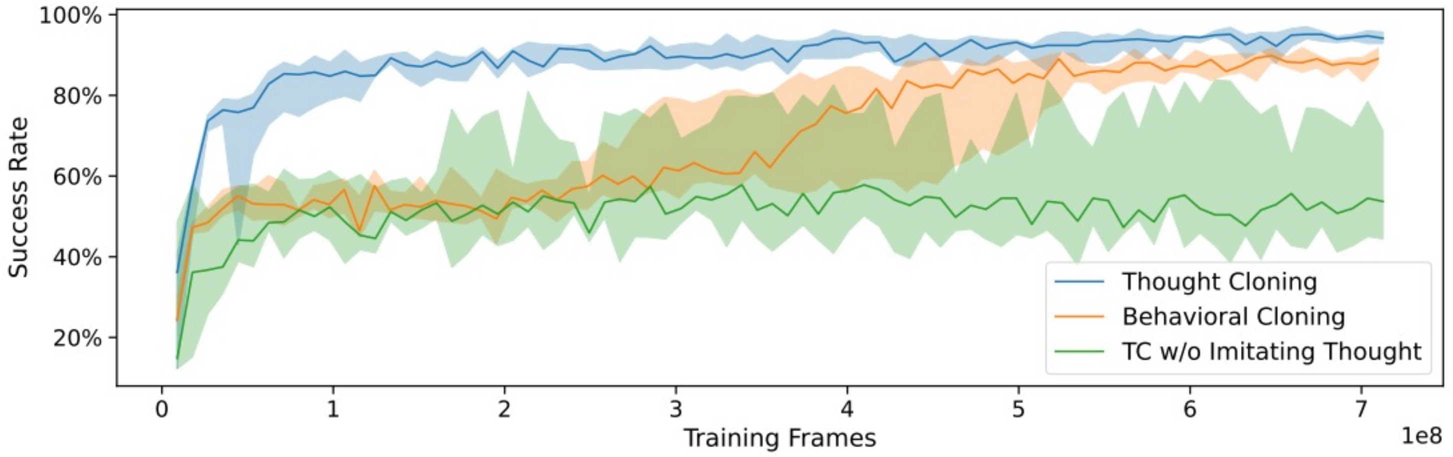
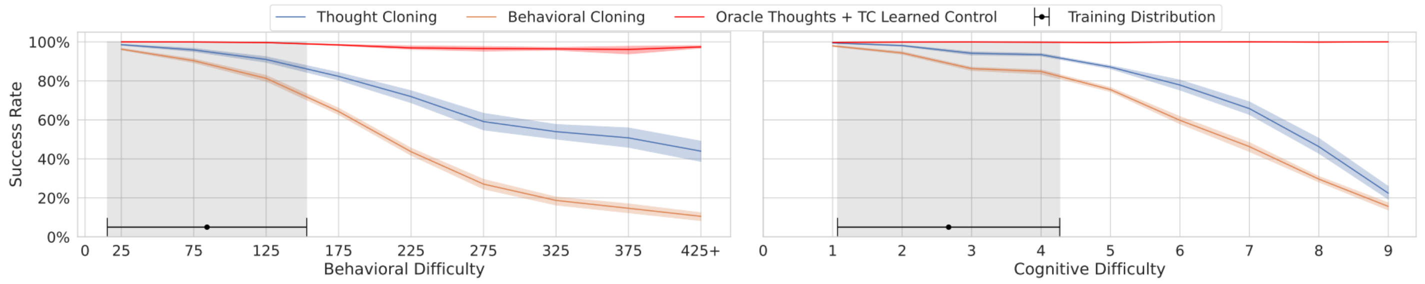

BabyAI grid maze room
Step 0
Step 43
Observable Region
Mission: go to a purple box and go to the green door on your right
: Agent
: Ball
: Key
: Box
: Locked and Unlocked Door
: Open door
Agent's Thoughts and Actions
(...after 43 steps)
Thought: open blue door to explore
Actions: [forward, ..., open]
Thought: go to purple box to complete goto mission
Actions: [forward]
Thought: remove blocking object purple ball
Actions: [pickup, ..., drop]
Thought: go to purple box to complete goto mission
Actions: [left, ..., left]

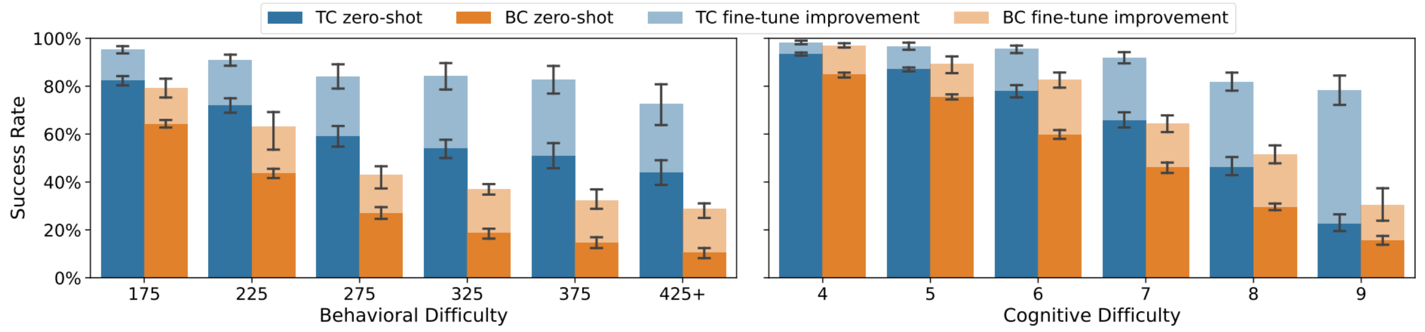
Figure 2: **Left**: A BabyAI [26] environment example. The environment contains various colored items (*ball, key, box, door*). The agent can pick up, drop, and move objects or open and close doors, while locked doors can only be unlocked with color-matched keys. The agent can observe the 7×7 grid cells in front of it, which can be blocked by walls and closed doors. **Right**: An example from a trained Thought Cloning agent planning and replanning. The mission requires reaching the purple box (highlighted), but a purple ball blocks the way. The agent's thoughts and actions show replanning when encountering the obstacle, removing it, and resuming the previous goal.

Experiments





(a) Zero-shot success rates of agents on environments that are increasingly out of distribution.



(b) Success rates after fine-tuning agents on out-of-distribution environments

Thanks