Generalization in Reinforcement Learning

Chengyang Ying
11 November 2022

Outline

· **Preliminary**
· Generalization in RL
· Generalize to Unseen MDPs
· Pretrained Large Models for All?
· Reference

Generalization

· A **generalization** is a form of abstraction whereby common properties of specific instances are formulated as general concepts or claims -- Wiki
(泛化是一种将具体实例的公共属性表述为一般概念或声明的抽象形式)

· The ability to categorize correctly new examples that differ from those used for training is known as **generalization** -- PRML

Generalization in Supervised Learning

Definition 2.1 (Generalization error) *Given a hypothesis $h \in \mathcal{H}$, a target concept $c \in \mathcal{C}$, and an underlying distribution $\mathcal{D}$, the* generalization error *or* risk *of h is defined by*

$$R(h) = \mathop{\mathbb{P}}_{x \sim \mathcal{D}}[h(x) \neq c(x)] = \mathop{\mathbb{E}}_{x \sim \mathcal{D}}\left[1_{h(x) \neq c(x)}\right], \tag{2.1}$$

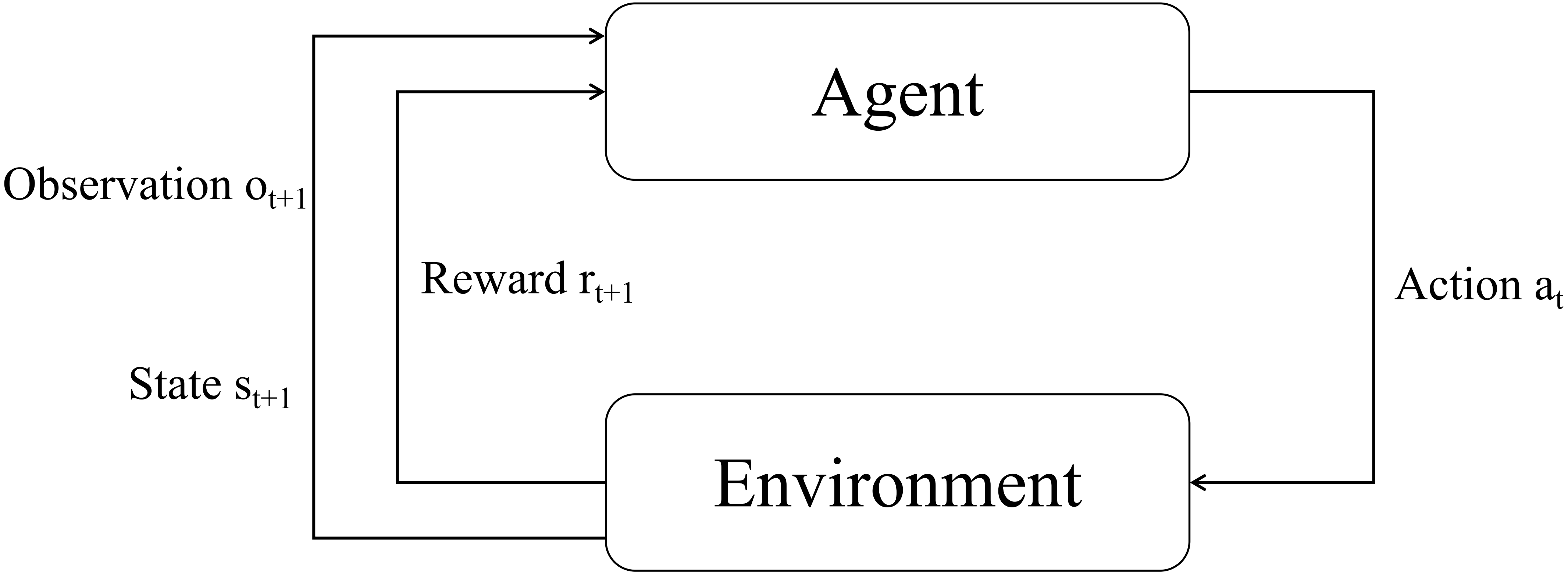
where 1_ω is the indicator function of the event ω.[2]

Definition 2.2 (Empirical error) *Given a hypothesis $h \in \mathcal{H}$, a target concept $c \in \mathcal{C}$, and a sample $S = (x_1, \ldots, x_m)$, the* empirical error *or* empirical risk *of h is defined by*

$$\widehat{R}_S(h) = \frac{1}{m} \sum_{i=1}^{m} 1_{h(x_i) \neq c(x_i)}. \tag{2.2}$$

A rough summary: train in finite samples and generalize to unseen samples or the distribution

Reinforcement Learning



In fully-observed MDP, o=s

Outline

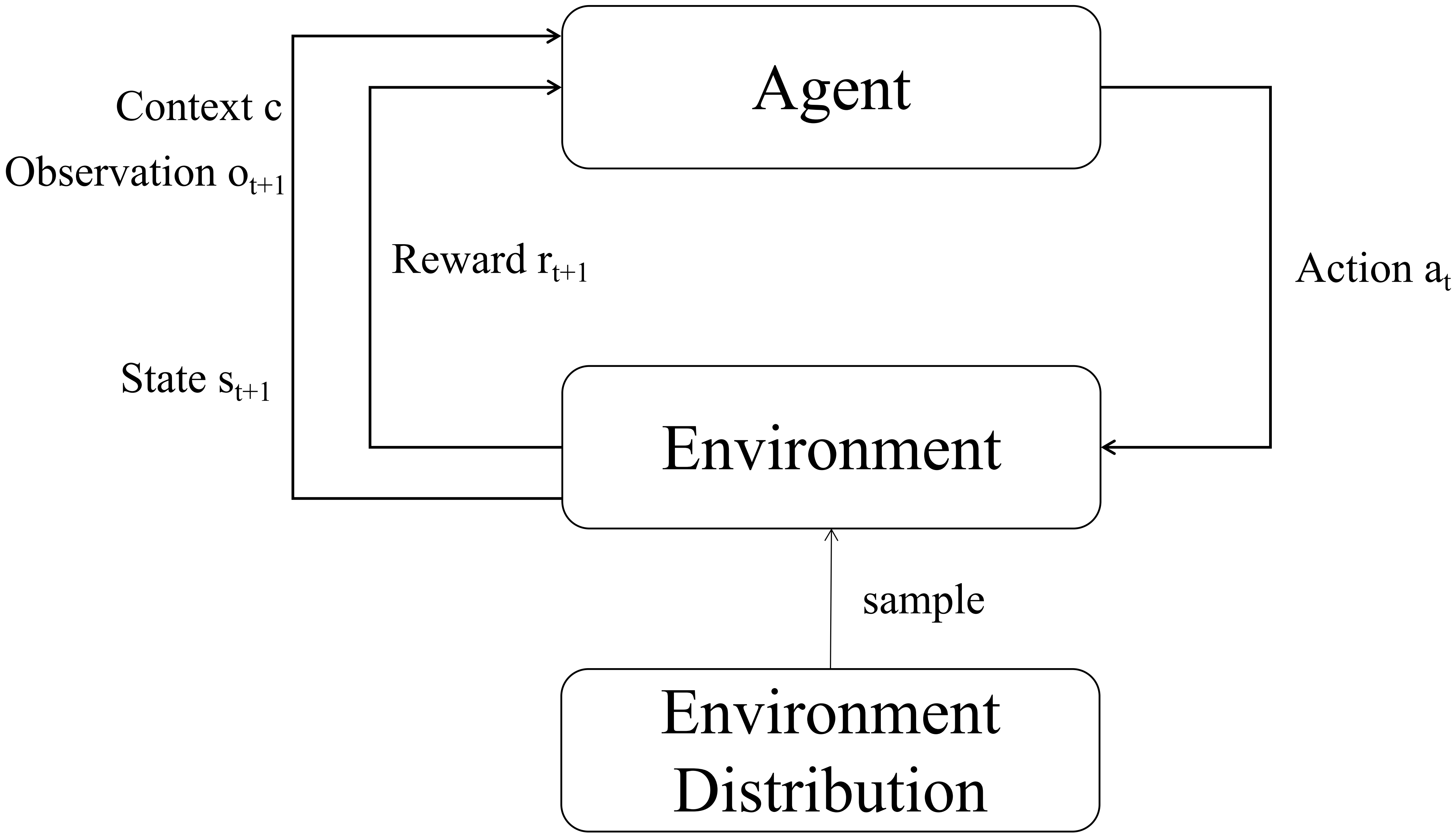
· Preliminary
· **Generalization in RL**
· Generalize to Unseen MDPs
· Pretrained Large Models for All?
· Reference

Generalization in RL

・There are many unknown distributions in RL...

Single MDP
・Model the transition distribution (model learning)
・When given a policy, calculate the return of the policy via finite trajectories (state-action pairs)
(When on-policy, the result is almost the same as the case in Supervised Learning)

Multiple MDPs (Main concerns today)
・There a distribution of MDPs, how to estimate them (transition, policy) via some sampled MDPs.


Agent
Context c
Observation o_{t+1}
Reward r_{t+1}
State s_{t+1}
Action a_t
Environment
sample
Environment Distribution

Outline

· Preliminary
· Generalization in RL
· **Generalize to Unseen MDPs**
· Pretrained Large Models for All?
· Reference

Generalization in unseen MDPs ≈ solving POMDP (NeurIPS21)

Main Result:

In standard supervised learning, ERM (empirical risk minimization) in the training set translates into good generalization performance (without distribution shift and with appropriate inductive biases)

In RL, similar “empirical risk minimization” approaches can be **sub-optimal** for generalizing to new environments

Ghosh D, Rahme J, Kumar A, et al. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability[J]. Advances in Neural Information Processing Systems, 2021, 34: 25502-25515.

Generalization in unseen MDPs ≈ solving POMDP (NeurIPS21)

A simple example in Classification and RL

Classifacation: Given an image from a dataset (like FashionMNIST), output its category. Correct: get reward 0. Wrong: get reward -1.

Reinforcement Learning: Repeat the above process until the result is correct.

Ghosh D, Rahme J, Kumar A, et al. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability[J]. Advances in Neural Information Processing Systems, 2021, 34: 25502-25515.

Generalization in unseen MDPs ≈ solving POMDP (NeurIPS21)

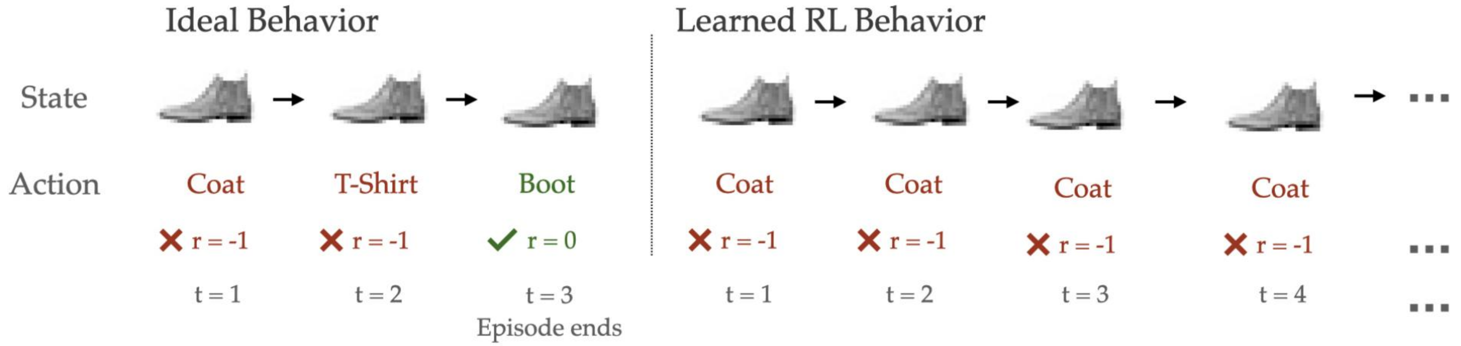


Figure 2: **Sequential Classification RL Problem.** In this task, an agent must keep guessing the label for an image until it gets it correct. To avoid low test return, policies should change actions if the label guessed was incorrect, but standard RL methods fail to do so, instead guessing the same incorrect label repeatedly.

In standard RL, there exists a deterministic policy is the optimal policy (choose the action with the maximal Q value)

In this case: first choosing the action it is most confident about, if incorrect, then the second, and so forth.

Ghosh D, Rahme J, Kumar A, et al. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability[J]. Advances in Neural Information Processing Systems, 2021, 34: 25502-25515.

Generalization in unseen MDPs ≈ solving POMDP (NeurIPS21)

Empirical results in FashionMNIST:

fix the length of the trajectory no more than 20

Adaptive: first choosing the action it is most confident about, if incorrect, then the second, and so forth

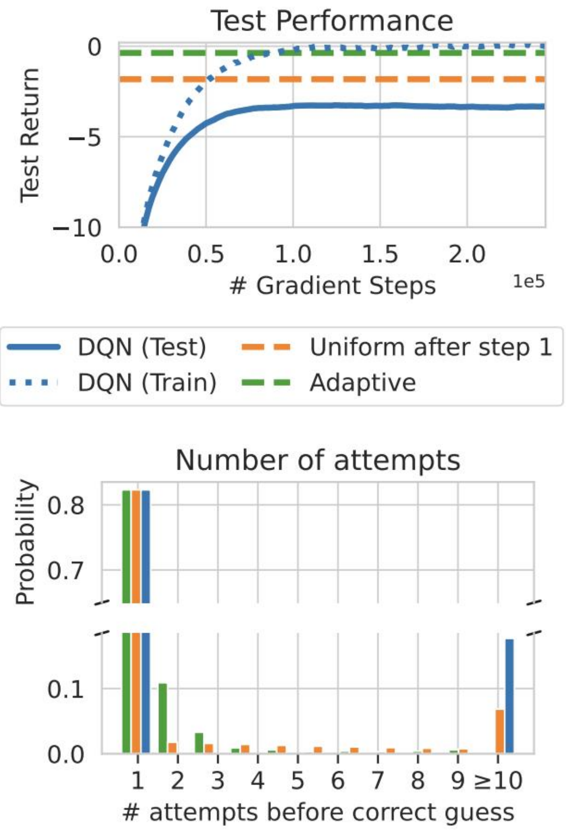
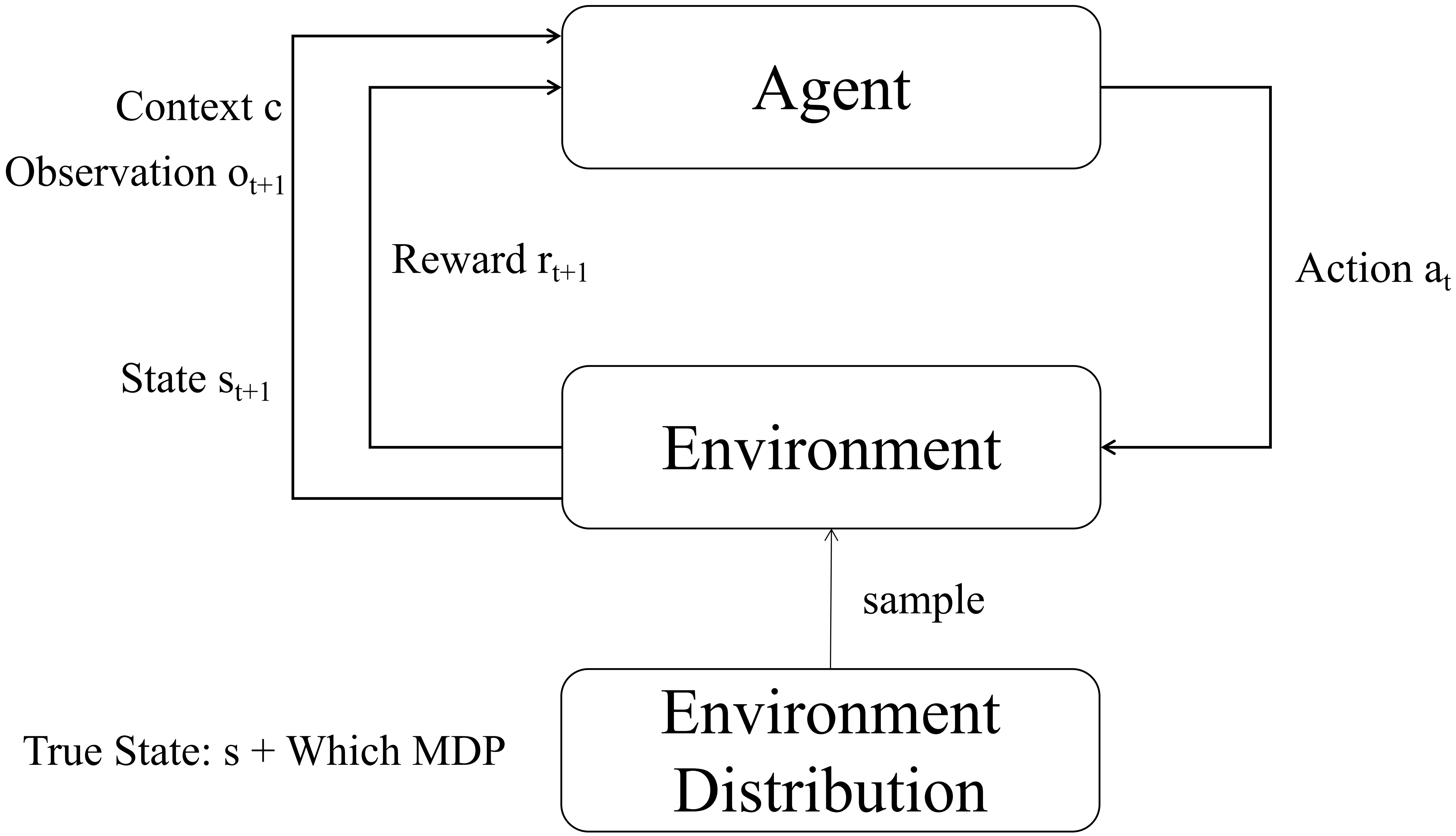


Figure 3: **DQN on RL FashionMNIST.** DQN achieves lower test performance than simple variants that leverage the structure of the RL problem.

Ghosh D, Rahme J, Kumar A, et al. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability[J]. Advances in Neural Information Processing Systems, 2021, 34: 25502-25515.


Agent
Context c
Observation o_{t+1}
Reward r_{t+1}
Action a_t
State s_{t+1}
Environment
sample
True State: s + Which MDP
Environment Distribution

Generalization in unseen MDPs ≈ solving POMDP (NeurIPS21)

Formalization

MDP (epistemic POMDP): $\mathcal{M}^{\mathrm{po}} = (\mathcal{S}^{\mathrm{po}}, \mathcal{O}^{\mathrm{po}}, \mathcal{A}, T^{\mathrm{po}}, r^{\mathrm{po}}, \rho^{\mathrm{po}}, \gamma)$

State and Ovservation: $s_t^{\mathrm{po}} = (\mathcal{M}, s_t)$ $\quad o_t^{\mathrm{po}} = s_t$

Transition and Return:

$$T^{\mathrm{po}}((\mathcal{M}', s') \mid (\mathcal{M}, s), a) = \delta(\mathcal{M}' = \mathcal{M}) T_{\mathcal{M}}(s'|s, a) \qquad r^{\mathrm{po}}((\mathcal{M}, s), a) = r_{\mathcal{M}}(s, a) \, .$$

Ghosh D, Rahme J, Kumar A, et al. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability[J]. Advances in Neural Information Processing Systems, 2021, 34: 25502-25515.

Generalization in unseen MDPs ≈ solving POMDP (NeurIPS21)

Proposition 5.1. *If the true MDP $\mathcal{M}$ is sampled from $\mathcal{P}(\mathcal{M})$, and evidence $\mathcal{D}$ from $\mathcal{M}$ is provided to an algorithm during training, then the expected test-time return of π is equal to its performance in the epistemic POMDP $\mathcal{M}^{po}$.*

$$J_{\mathcal{M}^{po}}(\pi) = \mathbb{E}_{\mathcal{M}\sim\mathcal{P}(\mathcal{M})}[J_{\mathcal{M}}(\pi) \mid \mathcal{D}]. \tag{2}$$

In particular, the optimal policy in $\mathcal{M}^{po}$ is Bayes-optimal for generalization to the unknown MDP $\mathcal{M}$: it receives the highest expected test-time return amongst all possible policies.

Optimal policy is **non-Markovian** since the episode contains information about the identity of the MDP being acted in.

When restricted to Markovian policies, the optimal policy is in general **stochastic**.

Ghosh D, Rahme J, Kumar A, et al. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability[J]. Advances in Neural Information Processing Systems, 2021, 34: 25502-25515.

Generalization in unseen MDPs ≈ solving POMDP (NeurIPS21)

This result is not isolated.

Actually, when there are certain uncertainty in MDP, the optimal Markovian policy we found may not be deterministic.

For example, when the policy is disturbed by an adversary (SA-MDP)

Theorem 3. *There exists an SA-MDP and some stochastic policy $\pi \in \Pi_{MR}$ such that we cannot find a better deterministic policy $\pi' \in \Pi_{MD}$ satisfying $\tilde{V}_{\pi' \circ \nu^*(\pi')}(s) \geq \tilde{V}_{\pi \circ \nu^*(\pi)}(s)$ for all $s \in \mathcal{S}$.*

Zhang H, Chen H, Xiao C, et al. Robust deep reinforcement learning against adversarial perturbations on state observations[J]. Advances in Neural Information Processing Systems, 2020, 33: 21024-21037.
Ghosh D, Rahme J, Kumar A, et al. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability[J]. Advances in Neural Information Processing Systems, 2021, 34: 25502-25515.

Generalization in unseen MDPs ≈ solving POMDP (NeurIPS21)

How to find the optimal policy?

Use a sequential policy (like lstm, transformer)
Use a stochastic memoryless policy

Proposition 6.1. *Let $\pi, \pi_1, \cdots \pi_n$ be memoryless policies, and define $r_{\max} = \max_{i,s,a} |r_{\mathcal{M}_i}(s,a)|$. The expected return of π in $\hat{\mathcal{M}}^{po}$ is bounded below as:*

$$J_{\hat{\mathcal{M}}^{po}}(\pi) \geq \frac{1}{n}\sum_{i=1}^{n} J_{\mathcal{M}_i}(\pi_i) - \frac{\sqrt{2} r_{\max}}{(1-\gamma)^2 n}\sum_{i=1}^{n} \mathbb{E}_{s\sim d^{\pi_i}_{\mathcal{M}_i}}\left[\sqrt{D_{KL}\left(\pi_i(\cdot|s) \;||\; \pi(\cdot|s)\right)}\right], \qquad (3)$$

Proposition 6.2. *Let $f : \{\pi_i\}_{i\in[n]} \mapsto \pi$ be a function that maps n policies to a single policy satisfying $f(\pi, \cdots, \pi) = \pi$ for every policy π, and let α be a hyperparameter satisfying $\alpha \geq \frac{\sqrt{2} r_{max}}{(1-\gamma)^2 n}$. Then letting $\pi_1^*, \ldots \pi_n^*$ be the optimal solution to the following optimization problem:*

$$\{\pi_i^*\}_{i\in[n]} = \arg\max_{\pi_1,\cdots,\pi_n} \frac{1}{n}\sum_{i=1}^{n} J_{\mathcal{M}_i}(\pi_i) - \alpha \sum_{i=1}^{n} \mathbb{E}_{s\sim d^{\pi_i}_{\mathcal{M}_i}}\left[\sqrt{D_{KL}\left(\pi_i(\cdot|s) \;||\; f(\{\pi_i\})(\cdot|s)\right)}\right], \qquad (4)$$

the policy $\pi^* := f(\{\pi_i^*\}_{i\in[n]})$ is optimal for the empirical epistemic POMDP $\hat{\mathcal{M}}^{po}$.

Ghosh D, Rahme J, Kumar A, et al. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability[J]. Advances in Neural Information Processing Systems, 2021, 34: 25502-25515.

Generalization in unseen MDPs ≈ solving POMDP (NeurIPS21)

Algorithm 1 Linked Ensembles for the Epistemic POMDP (LEEP)

1: Receive training contexts $\mathcal{C}_{\text{train}}$, number of ensemble members n
2: Bootstrap sample training contexts to create $\mathcal{C}^1_{\text{train}}, \ldots \mathcal{C}^n_{\text{train}}$, where $\mathcal{C}^i_{\text{train}} \subset \mathcal{C}_{\text{train}}$.
3: Initialize n policies: $\pi_1, \ldots \pi_n$
4: **for** iteration $k = 1, 2, 3, \ldots$ **do**
5:     **for** policy $i = 1, \ldots, n$ **do**
6:         Collect environment samples in training contexts $\mathcal{C}^i_{\text{train}}$ using policy π_i
7:         Take gradient steps wrt π_i on these samples with augmented RL loss:

$$\pi_i \leftarrow \pi_i - \eta \nabla_i (\mathcal{L}^{RL}(\pi_i) + \alpha \mathbb{E}_{s \sim \pi_i, \mathcal{C}^i_{\text{train}}} [D_{KL}(\pi_i(a|s) \| \max_j \pi_j(a|s))])$$

8: Return $\pi = \max_i \pi$: $\pi(a|s) = \frac{\max_i \pi_i(a|s)}{\sum_{a'} \max_i \pi_i(a'|s)}$.

Everytime train in contextual MDP with many contexts rather than a single MDP

Ghosh D, Rahme J, Kumar A, et al. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability[J]. Advances in Neural Information Processing Systems, 2021, 34: 25502-25515.

Generalization in unseen MDPs ≈ solving POMDP (NeurIPS21)

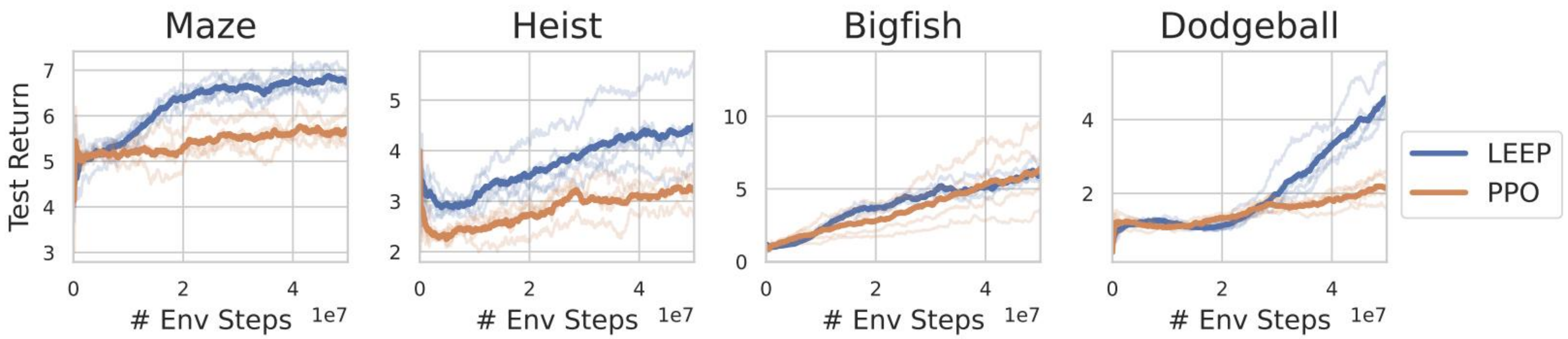


Figure 4: Test set return for LEEP and PPO throughout training in four Procgen environments (averaged across 5 random seeds). LEEP achieves higher test returns than PPO on three tasks (Maze, Heist and Dodgeball) and matches test return on Bigfish while having less variance across seeds.

Ghosh D, Rahme J, Kumar A, et al. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability[J]. Advances in Neural Information Processing Systems, 2021, 34: 25502-25515.

CaDM (ICML20)

Conder generalization in test environments with unseen contexts c

goal: learn the environment, i.e., use current K past transition $\tau^{\mathrm{P}}_{t,K} = \{(s_{t-K}, a_{t-K}) \cdots (s_{t-1}, a_{t-1})\}$ to encode the context c

Forward model: use c, current state and current action to predice the next state

Lee K, Seo Y, Lee S, et al. Context-aware dynamics model for generalization in model-based reinforcement learning[C]//International Conference on Machine Learning. PMLR, 2020: 5757-5766.

CaDM (ICML20)

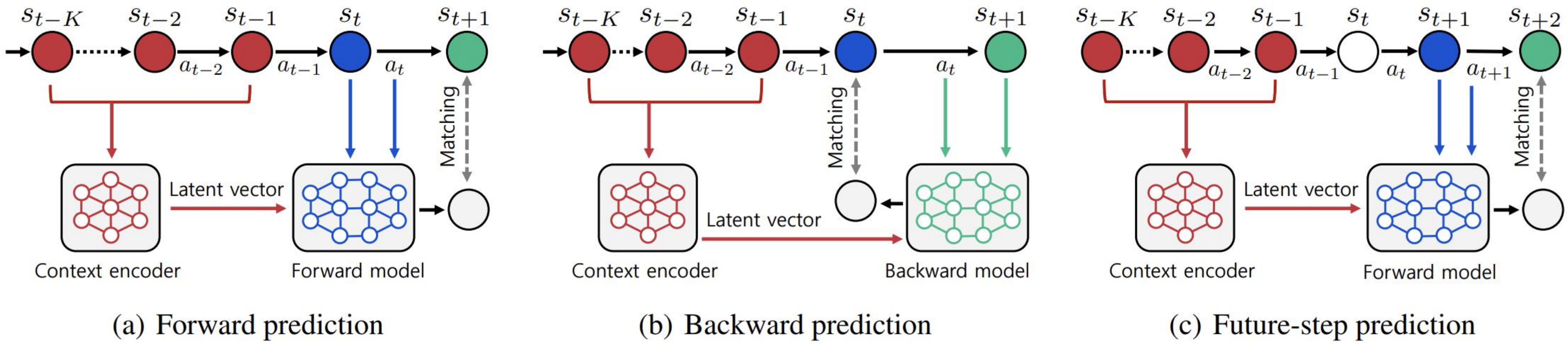


Figure 2. Illustrations of our framework. We decompose the task of learning a global dynamics model into context encoding and transition inference. (a) Our dynamics model predicts the next state conditioned on the latent vector. (b) We introduce a backward dynamics model that predicts a previous state by utilizing a context latent vector. (c) We force the context latent vector to be temporally consistent by utilizing it for predictions in the future timesteps.

Lee K, Seo Y, Lee S, et al. Context-aware dynamics model for generalization in model-based reinforcement learning[C]//International Conference on Machine Learning. PMLR, 2020: 5757-5766.

CaDM (ICML20)

Backward model: capture contextual information while mitigating the risk of overly focusing on predicting only the ”seen” forward dynamic (intuition: context should be useful for predicting both forward and backward transition)

$$\mathcal{L}^{\texttt{pred}} = \mathbb{E}_{(\tau^{\texttt{F}}_{t,M}, \tau^{\texttt{P}}_{t,K}) \sim \mathcal{B}} \left[\mathcal{L}^{\texttt{pred}}_{\texttt{forward}} + \beta \mathcal{L}^{\texttt{pred}}_{\texttt{backward}} \right], \quad (1)$$

$$\mathcal{L}^{\texttt{pred}}_{\texttt{forward}} = -\frac{1}{M} \sum_{i=t}^{t+M-1} \log f\left(s_{i+1} | s_i, a_i, g\left(\tau^{\texttt{P}}_{t,K}; \phi\right); \theta\right),$$

$$\mathcal{L}^{\texttt{pred}}_{\texttt{backward}} = -\frac{1}{M} \sum_{i=t}^{t+M-1} \log b\left(s_i | s_{i+1}, a_i, g\left(\tau^{\texttt{P}}_{t,K}; \phi\right); \psi\right),$$

Action choose:

use MPC methods (like CEM) to take an action (assume the reward function is known)

combine with model free methods $\pi\left(a_t | s_t, g\left(\tau^{\texttt{P}}_{t,K}; \phi\right)\right)$

Lee K, Seo Y, Lee S, et al. Context-aware dynamics model for generalization in model-based reinforcement learning[C]//International Conference on Machine Learning. PMLR, 2020: 5757-5766.

Results for Model-Based

	Half-cheetah			Ant		
	Training	Test (moderate)	Test (extreme)	Training	Test (moderate)	Test (extreme)
Vanilla DM	1560.7± 453.1	1026.7± 164.7	686.7± 189.4	646.4± 89.0	520.0± 97.6	385.8± 85.2
Stacked DM	1301.4± 310.5	761.1± 236.6	661.5± 220.5	492.3± 68.7	417.1± 46.8	338.9± 51.5
GrBAL	117.0± 88.7	-43.7± 106.9	-94.5± 141.3	55.0± 10.0	46.5± 6.5	42.9± 3.8
ReBAL	1086.7± 90.0	657.5± 184.9	396.6± 188.5	100.1± 12.3	73.1± 15.5	53.0± 17.2
PE-TS	4347.1± 300.9	2019.6± 274.8	1422.3± 162.8	1183.3± 51.1	1075.1± 103.6	856.6± 66.5
Vanilla + CaDM	3536.5± 641.7	1556.1± 260.6	1264.5± 228.7	1851.0± 113.7	1315.7± 45.5	821.4± 113.5
PE-TS + CaDM	**8264.0**± 1374.0	**7087.2**± 1495.6	**4661.8**± 783.9	**2848.4**± 61.9	**2121.0**± 60.4	**1200.7**± 21.8

	CrippledHalfCheetah			SlimHumanoid		
	Training	Test (moderate)	Test (extreme)	Training	Test (moderate)	Test (extreme)
Vanilla DM	1005.1± 429.0	870.0± 308.0	577.3± 76.5	1119.8± 317.6	1004.4± 798.2	1155.5± 556.9
Stacked DM	630.6± 211.3	545.1± 289.8	417.9± 145.8	1057.4± 547.5	876.2± 1005.2	651.8± 449.9
GrBAL	151.9± 122.7	-9.2± 17.1	16.6± 23.0	-62.6± 233.1	-562.8± 253.5	-398.6± 177.2
ReBAL	701.7± 119.7	904.5± 90.7	833.0± 118.0	1205.8± 546.8	85.8± 388.9	108.7± 357.6
PE-TS	1846.8± 380.7	1916.5± 328.2	1227.6± 35.2	1339.6± 524.0	758.6± 528.8	810.4± 363.4
Vanilla + CaDM	2435.1± 880.4	1375.3± 290.6	966.9± 89.4	**1758.2**± 459.1	**1228.9**± 374.0	**1487.9**± 339.0
PE-TS + CaDM	**3294.9**± 733.9	**2618.7**± 647.1	**1294.2**± 214.9	1371.9± 400.0	903.7± 343.9	814.5± 274.8

Table 1. The performance (average returns) of trained dynamics models on various control tasks. The transition dynamics of environments are changing in both training and test environments. The results show the mean and standard deviation of returns averaged over five runs.

Lee K, Seo Y, Lee S, et al. Context-aware dynamics model for generalization in model-based reinforcement learning[C]//International Conference on Machine Learning. PMLR, 2020: 5757-5766.

Results for Model-Free

	HalfCheetah			Ant		
	Training	Test (moderate)	Test (extreme)	Training	Test (moderate)	Test (extreme)
Vanilla PPO	2043.4± 802.9	807.7± 553.6	574.0± 645.6	211.9± 44.5	149.4± 27.0	117.3± 23.1
Stacked PPO	1125.4± 85.5	361.1± 141.7	5.7± 208.1	90.6± 16.3	53.2± 10.6	46.0± 10.9
PPO + PC	1584.9± 404.3	642.1± 488.3	462.1± 534.5	249.9± 85.0	207.0± 33.8	163.5± 30.4
PPO + EP	1620.9± 491.5	895.3± 445.1	674.2± 686.8	138.8± 34.9	107.8± 19.9	93.5± 32.4
PPO + CaDM	**2652.0**± 1133.6	**1224.2**± 630.0	**1021.1**± 676.6	**268.6**± 77.0	**228.8**± 48.4	**199.2**± 52.1

	CrippledHalfCheetah			SlimHumanoid		
	Training	Test (moderate)	Test (extreme)	Training	Test (moderate)	Test (extreme)
Vanilla PPO	2059.6± 658.3	1223.6± 559.9	781.7± 270.3	7685.5± 2599.4	3761.3± 1582.4	2751.6± 869.4
Stacked PPO	1238.1± 102.5	967.1± 146.6	904.4± 146.5	4831.0± 688.1	2443.0± 535.6	1577.8± 573.5
PPO + PC	**2920.7**± 771.7	1162.2± 456.5	546.3± 215.9	7130.1± 3378.0	3928.5± 1848.7	2362.6± 781.9
PPO + EP	1494.2± 311.7	1017.0± 201.1	719.0± 438.5	4824.7± 1508.7	2224.7± 882.9	1293.4± 729.0
PPO + CaDM	2356.6± 624.3	**1454.0**± 462.6	**1025.0**± 296.2	**10455.0**± 1004.9	**4975.7**± 1305.7	**3015.1**± 1508.3

Table 2. The performance (average return) of trained agents on various control tasks. The transition dynamics of environments are changing in both training and test environments. The results show the mean and standard deviation of returns averaged over five runs.

Lee K, Seo Y, Lee S, et al. Context-aware dynamics model for generalization in model-based reinforcement learning[C]//International Conference on Machine Learning. PMLR, 2020: 5757-5766.

Outline

· Preliminary
· Generalization in RL
· Generalize to Unseen MDPs
· **Pretrained Large Models for All?**
· Reference

A Generalist Agent (Gato) Deepmind 2022.5

A Generalist Agent (Gato) Deepmind 2022.5

· *"Inspired by progress in large-scale language modeling, we apply a similar approach towards building a single generalist agent beyond the realm of text outputs."*

· Use a single agent with the same parameters to handle multi-modal tasks (including RL, CV, NLP)

· Parameters: 34M ~ 1.18B   1B = 1000,000,000

(As a comparison: GPT-2 ~ 1.5B, GPT-3 ~ 100B, Switch Transformer ~ 1600B, WuDao ~1750B)

· In the part of RL, Gato only focuses on **supervised learning**

Reed S, Zolna K, Parisotto E, et al. A generalist agent[J]. arXiv preprint arXiv:2205.06175, 2022.

A Generalist Agent

Goal:
use one NN with the same parameters for 604 tasks, including:

- Control Tasks
  - Atari Games
  - DM Control
  - Meta World
- Vision and Language
  - MassiveText (text)
  - ALIGN (image-text)
- Robotics
  - RGB Stacking (real and sim)

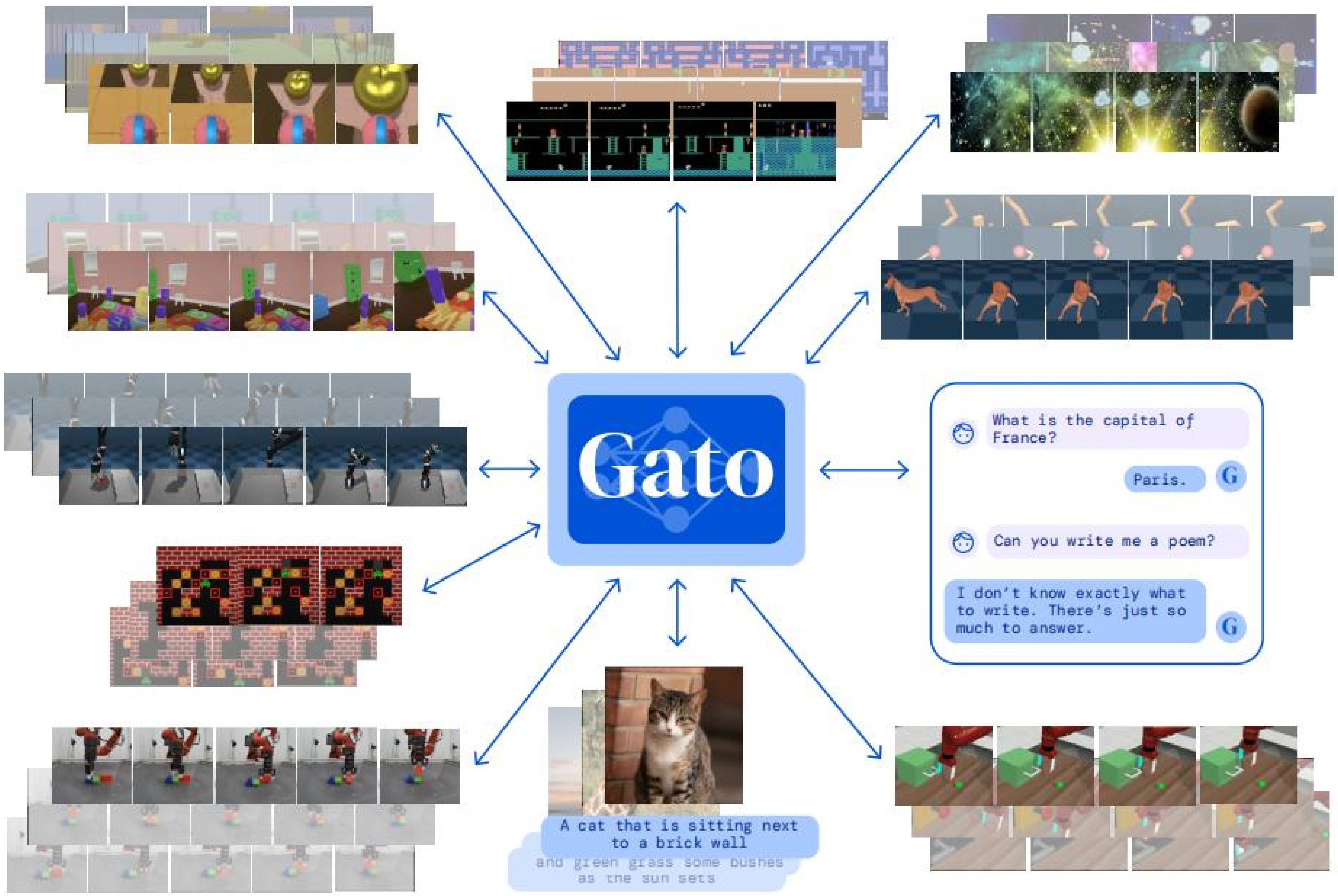


Figure 1 | **A generalist agent.** Gato can sense and act with different embodiments across a wide range of environments using a single neural network with the same set of weights. Gato was trained on 604 distinct tasks with varying modalities, observations and action specifications.

A Generalist Agent

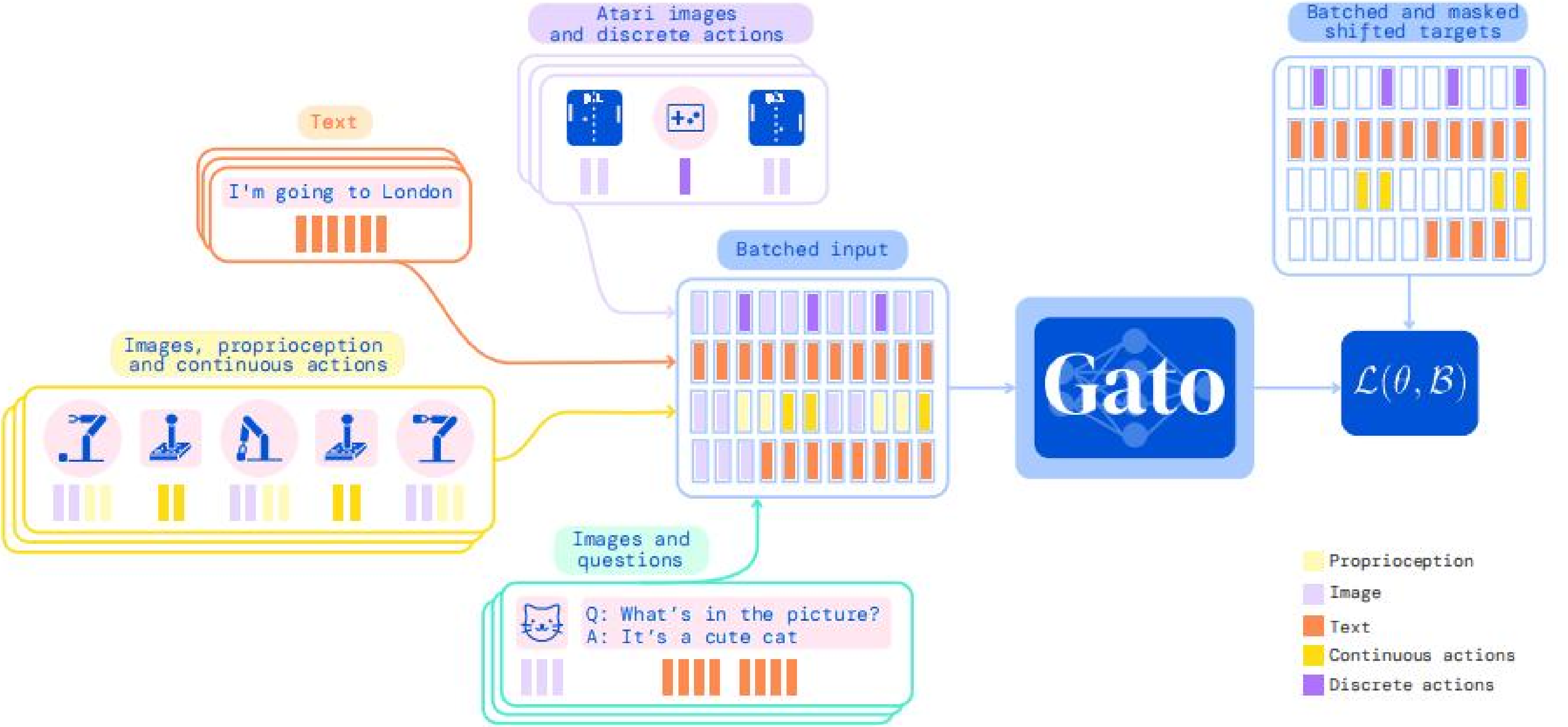


Figure 2 | **Training phase of Gato**. Data from different tasks and modalities is serialized into a flat sequence of tokens, batched, and processed by a transformer neural network akin to a large language model. Masking is used such that the loss function is applied only to target outputs, i.e. text and various actions.

Reed S, Zolna K, Parisotto E, et al. A generalist agent[J]. arXiv preprint arXiv:2205.06175, 2022.

Method

Serialize all data into a flat sequence of tokens

· Text: SentencePiece
· Images: ViT

· Discrete Values, e.g. Atari button presses: flattened into sequences of integers
· Continuous Values, e.g. proprioceptive inputs or joint torques: original value -> [-1, 1] -> discretized to 1024 uniform bins.

Train: Supervised Learning $\mathcal{L}(\theta, \mathcal{B}) = -\sum_{b=1}^{|\mathcal{B}|} \sum_{l=1}^{L} m(b, l) \log p_\theta \left(s_l^{(b)} | s_1^{(b)}, \ldots, s_{l-1}^{(b)}\right)$

m(b,t) = 1 iff s_l^{(b)} is output

Results: Simulated control tasks > 450 for 50%

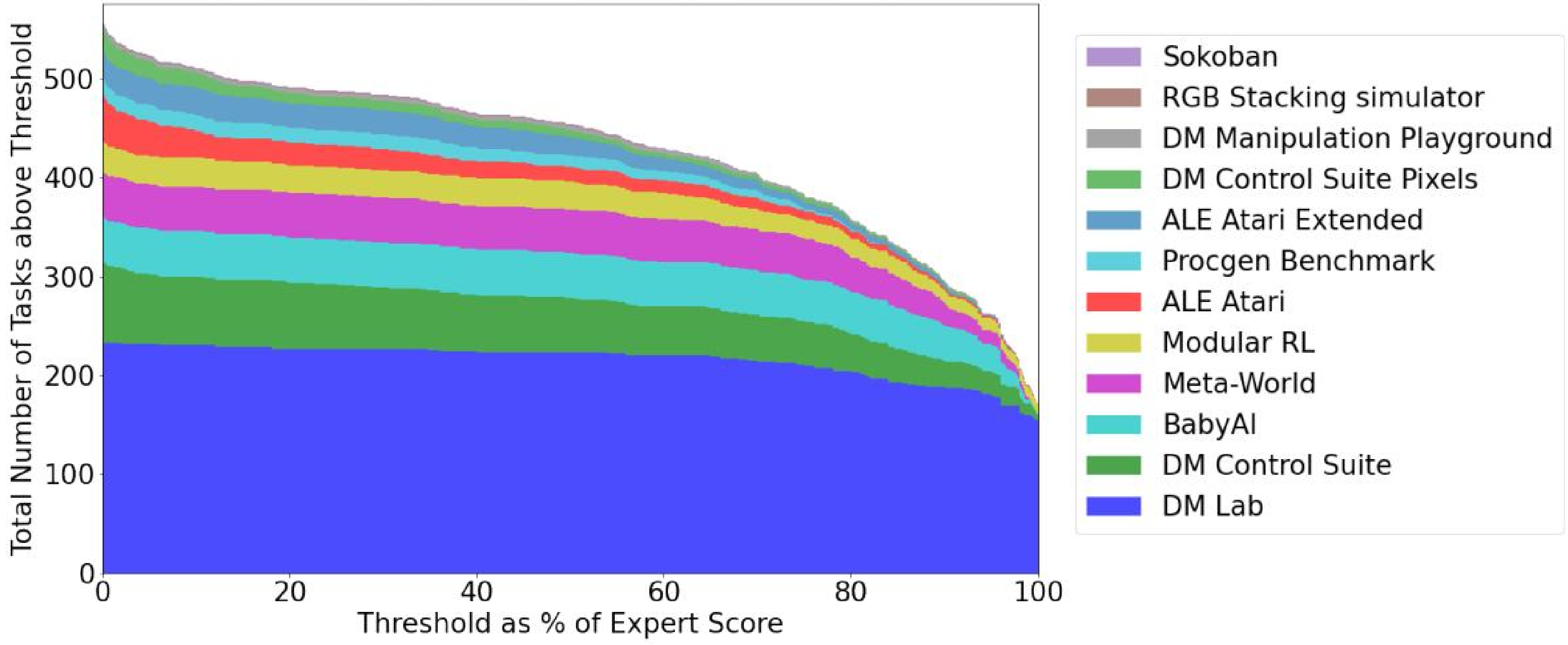


Figure 5 | **Gato's performance on simulated control tasks.** Number of tasks where the performance of the pretrained model is above a percentage of expert score, grouped by domain. Here values on the x-axis represent a specific percentage of expert score, where 0 corresponds to random agent performance. The y-axis is the number of tasks where the pretrained model's mean performance is equal to or above that percentage. That is, the width of each colour band indicates the number of tasks where Gato's mean performance is above a percentage of the maximum score obtained by a task-specific expert.

Reed S, Zolna K, Parisotto E, et al. A generalist agent[J]. arXiv preprint arXiv:2205.06175, 2022.

Results: Robotics

Table 2 | **Gato real robot Skill Generalization results.** In addition to performing hundreds of other tasks, Gato also stacks competitively with the comparable published baseline.

Agent	Group 1	Group 2	Group 3	Group 4	Group 5	Average
Gato	**24.5%**	33%	**50.5%**	76.5%	**66.5%**	**50.2%**
BC-IMP (Lee et al., 2021)	23%	**39.3%**	39.3%	**77.5%**	66%	49%

Skill generalization:

Test in five triplets of object shapes are not included in the training data

Reed S, Zolna K, Parisotto E, et al. A generalist agent[J]. arXiv preprint arXiv:2205.06175, 2022.

Analysis: Pretrain + Finetune

1. A model pretrained only on data from the same domain as the task to be fine-tuned on, *same domain only data*.
2. A model pretrained only on non-control data, *no control data*.
3. A model fine-tuned from scratch, i.e. no pretraining at all, *scratch*.

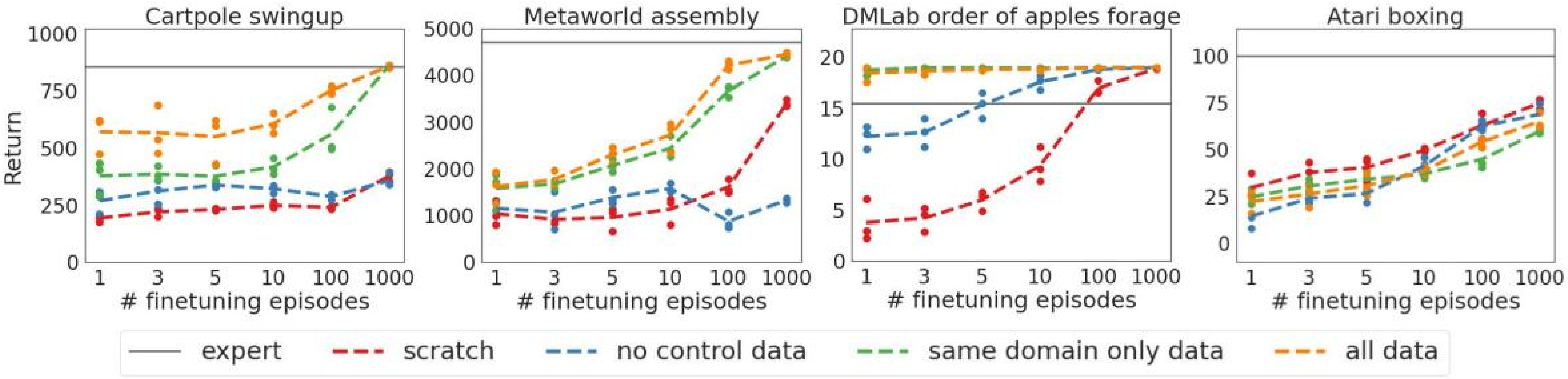


Figure 9 | **Few-shot performance, ablating over various pretraining settings.** Orange corresponds to the base Gato pretrained on all data. Red is trained from scratch only on the few-shot data. `364M` parameter variants of Gato were used for this experiment to save compute.

Reed S, Zolna K, Parisotto E, et al. A generalist agent[J]. arXiv preprint arXiv:2205.06175, 2022.

Algorithm Distillation (AD) Deepmind (Sumitted to ICLR23)

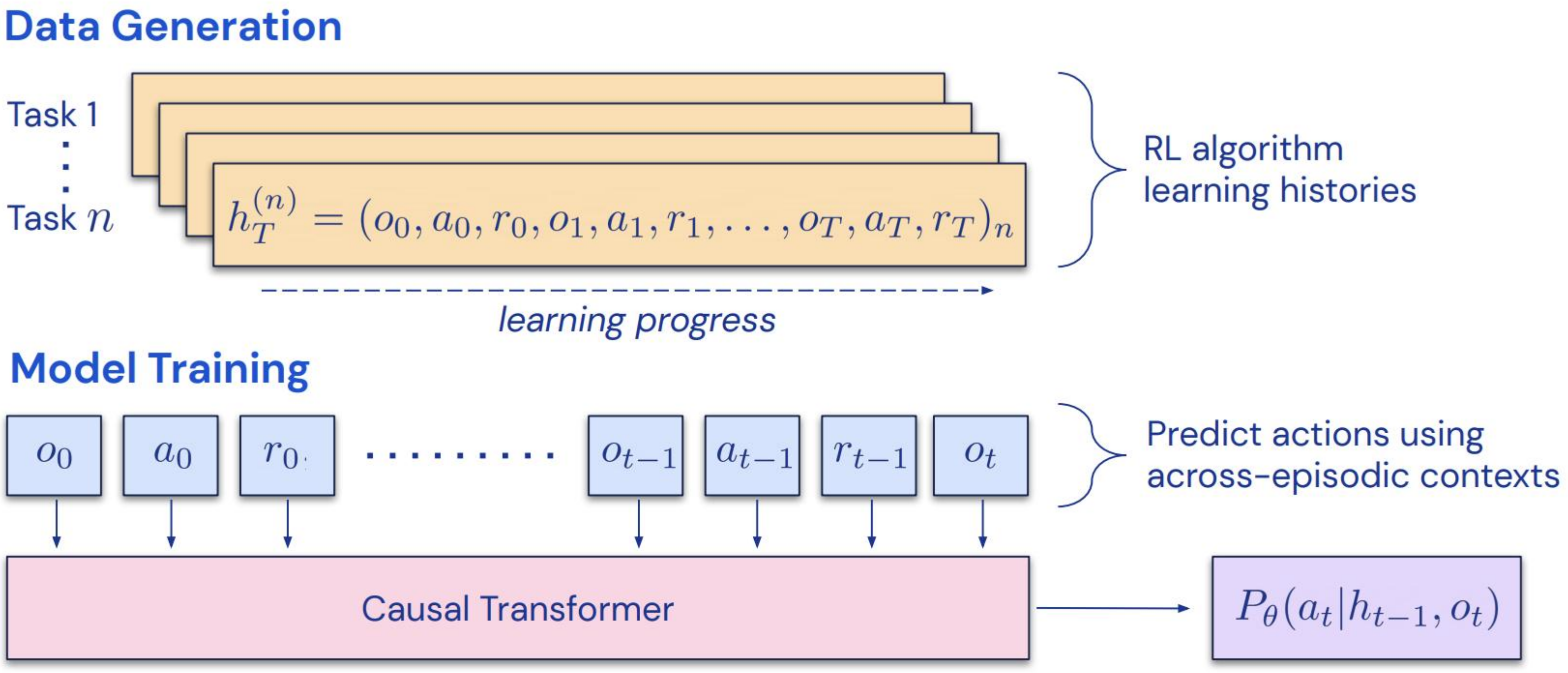


Figure 1: Algorithm Distillation (AD) has two steps – (i) a dataset of learning histories is collected from individual single-task RL algorithms solving different tasks; (ii) a causal transformer predicts actions from these histories using across-episodic contexts. Since the RL policy improves throughout the learning histories, by predicting actions accurately AD learns to output an improved policy relative to the one seen in its context. AD models state-action-reward tokens, and does not condition on returns.

Laskin M, Wang L, Oh J, et al. In-context Reinforcement Learning with Algorithm Distillation[J]. arXiv preprint arXiv:2210.14215, 2022.

Algorithm Distillation (AD)

Policy Distillation (PD): learns policies from offline RL data via imitation learning, not Reinforcement Learning algorithms (E.g. Gato)

Algorithm Distillation (AD): learns an in-context policy improvement operator by optimizing a causal sequence prediction loss on the learning histories of an RL algorithm

if a transformer’s context is long enough to include policy improvement due to learning updates it should be able to represent not only a fixed policy but a policy improvement operator by attending to states, actions and rewards from previous episodes.

Laskin M, Wang L, Oh J, et al. In-context Reinforcement Learning with Algorithm Distillation[J]. arXiv preprint arXiv:2210.14215, 2022.

Algorithm Distillation (AD)

Algorithm 1 `Algorithm Distillation`

Require: Train $\{\mathcal{M}^{\text{train}}\}$ and test $\{\mathcal{M}^{\text{test}}\}$ tasks, observations $o \in \mathcal{O}$, actions $a \in \mathcal{A}$, and rewards $r \in \mathcal{R}$.
Require: Network parameters ϕ_i for $i = 1, \ldots, N$ source RL algorithms.
Require: Network parameters θ for a causal transformer P_θ that predicts actions.
Require: An empty buffer to store data $\mathcal{D}$.

for $i = 1 \ldots N$ **do** ▷ Part 1: Dataset Generation
    Sample a task $\mathcal{M}_i^{\text{train}}$ randomly from the train task distribution.
    Train the source RL algorithm ϕ_i until it converges to the optimal policy.
    Save the learning history $h_T^{(i)} = (o_0, a_0, r_0, \ldots, o_T, a_T, r_T)_i$ to the dataset $\mathcal{D} \leftarrow \mathcal{D} \cup h_T^{(i)}$.
end for
while P_θ not converged **do** ▷ Part 2: Algorithm Distillation
    Randomly sample a multi-episodic subsequence $\bar{h}_j^{(i)} = (o_j, a_j, r_j, \ldots, o_{j+c}, a_{j+c}, r_{j+c})_i$ of length c.
    Autoregressively predict the actions with P_θ and compute the NLL loss in Eq. 6.
    Backpropagate to update the transformer parameters.
end while
for $k = 1 \ldots M_{\text{seeds}}$ **do** ▷ Part 3: Autoregressive Evaluation
    Sample a task $\mathcal{M}_k^{\text{test}}$ randomly from the test task distribution. Initialize empty context queue C.
    Unroll the transformer $P_\theta(\cdot|C)$ in the environment storing sequential transitions (*i.e.* histories) in C.
    Measure the return accumulated by the agent for each episode of evaluation.
end for

Laskin M, Wang L, Oh J, et al. In-context Reinforcement Learning with Algorithm Distillation[J]. arXiv preprint arXiv:2210.14215, 2022.

Algorithm Distillation (AD)

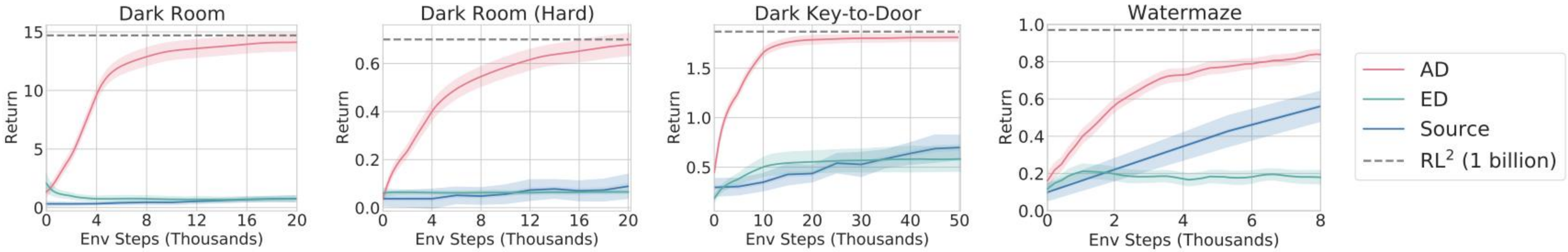


Figure 4: *Main results:* we evaluate AD, RL^2, ED, and the source algorithm on environments that require memory and exploration. In these environments, an agent must reach a goal location that can only be inferred through a binary reward. AD is consistently able to in-context reinforcement learn across all environments and is more data-efficient than the A3C ("Dark" environments) (Mnih et al., 2016) or DQN (Watermaze) (Mnih et al., 2013) source algorithm it distilled. We report the mean return $\pm$ 1 standard deviation over 5 training seeds with 20 test seeds each.

ED: Expert Distillation (similar to Gato), RL^2: a meta rl algorithm, source: basic algorithm for collecting data

Laskin M, Wang L, Oh J, et al. In-context Reinforcement Learning with Algorithm Distillation[J]. arXiv preprint arXiv:2210.14215, 2022.

Algorithm Distillation (AD)

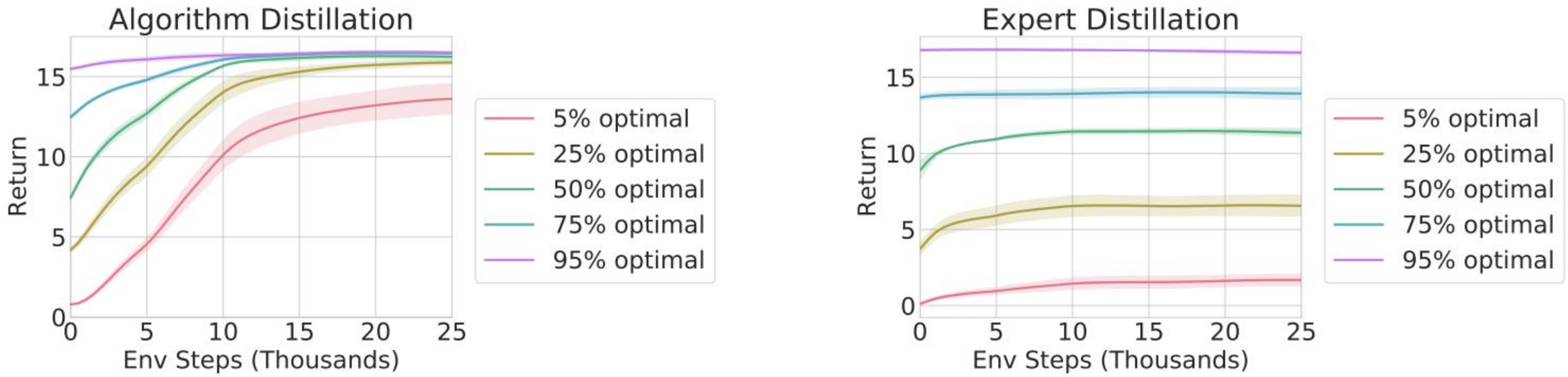


Figure 5: *AD and ED conditioned on partial demonstrations:* We compare the performance of AD and ED when prompted with a demonstration from the source algorithm's training history on Dark Room (semi-dense). While ED slightly improves and then maintains performance from the input policy, AD is able to improve it in-context until the policy is optimal or nearly optimal.

sample policies from the hold-out test-set data along different points of the source algorithm history - from a near-random policy to a near-expert policy to pre-fill the context for both AD and ED
AD improves every policy in-context until it is near-optimal

Laskin M, Wang L, Oh J, et al. In-context Reinforcement Learning with Algorithm Distillation[J]. arXiv preprint arXiv:2210.14215, 2022.

LSTM vs Transformer

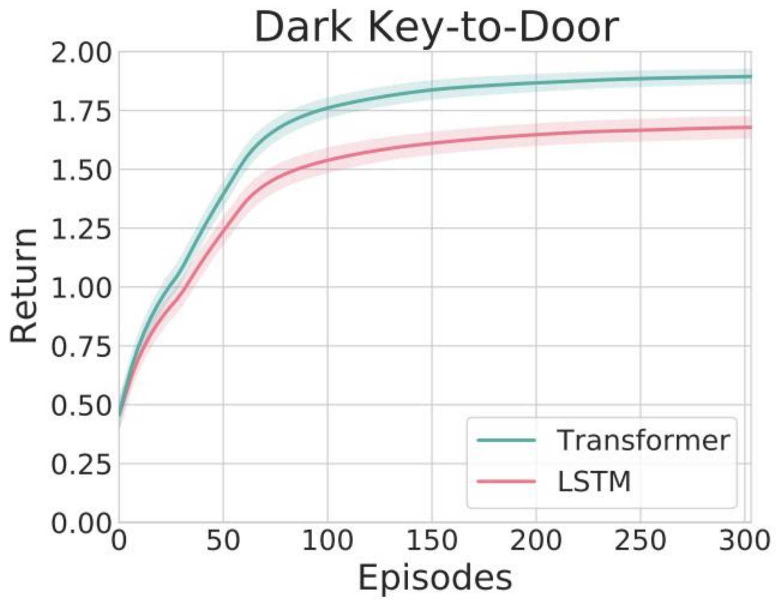


Figure 14: Comparison between algorithm distillation with a Transformer and LSTM architecture on Dark Key-to-Door. Mean $\pm$ 1 standard deviation over 5 training seeds and 20 evaluation seeds. 300 episodes corresponds to 15k environment steps.

Laskin M, Wang L, Oh J, et al. In-context Reinforcement Learning with Algorithm Distillation[J]. arXiv preprint arXiv:2210.14215, 2022.

Conclusion

· Generalize to unseen MDPs somehow makes the fully observable environment partially observable, which is similar to uncertainty RL.

· To handle this issue, there are some types of methods: use stochastic policies, data augmentation, use sequential model.

· The capacity of Current NNs is enough to capture the gap between different RL tasks (Attention is all you need).

Outline

· Preliminary
· Generalization in RL
· Generalize to Unseen MDPs
· Pretrained Large Models for All?
· **Reference**

Reference

· Ghosh D, Rahme J, Kumar A, et al. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability[J]. Advances in Neural Information Processing Systems, 2021, 34: 25502-25515.

· Lee K, Seo Y, Lee S, et al. Context-aware dynamics model for generalization in model-based reinforcement learning[C]//International Conference on Machine Learning. PMLR, 2020: 5757-5766.

· Reed S, Zolna K, Parisotto E, et al. A generalist agent[J]. arXiv preprint arXiv:2205.06175, 2022.

· Laskin M, Wang L, Oh J, et al. In-context Reinforcement Learning with Algorithm Distillation[J]. arXiv preprint arXiv:2210.14215, 2022.

Thanks for Listening

Questions?