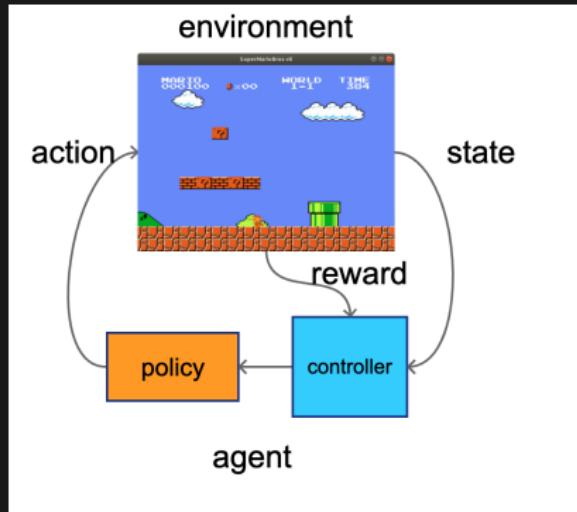
A painting depicting a traditional Aztec ceremony. In the center, a priest in elaborate feathered headdress and ceremonial attire stands on a platform, his arms raised in a gesture of offering or sacrifice. He holds a small object in one hand. Below him, a vast crowd of people, mostly men in traditional dress, looks up in awe. To the right, another figure in a detailed headdress and armor stands near a red banner with gold symbols. The background features a large, ornate pyramid with multiple levels and steps. The overall atmosphere is one of a solemn, historical event.

reinforcement learning current problems

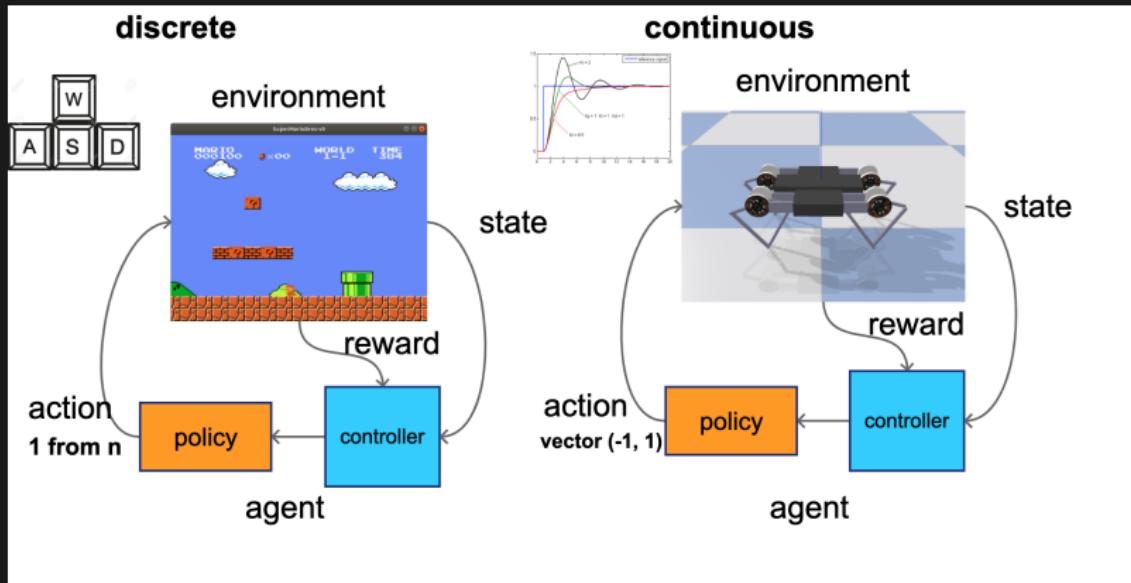
Michal CHOVANEC, PhD.

reinforcement learning



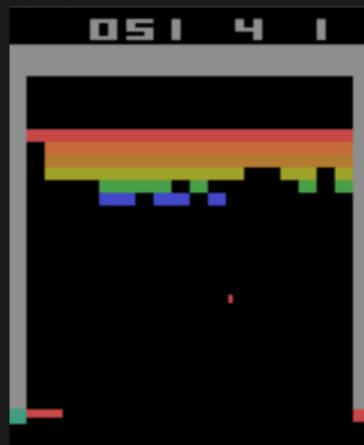
- ① **obtain state** - observation
- ② **choose action** - policy
- ③ **receive reward**
- ④ **learn from experiences**

action space



when it works ?

- dense rewards
- small state space
- fully observable



when it works - AlphaGO, AlphaZero - DeepMind



Chess with Suren, AlphaZero's Most Astonishing "Zugzwang" Game

most famous algorithms

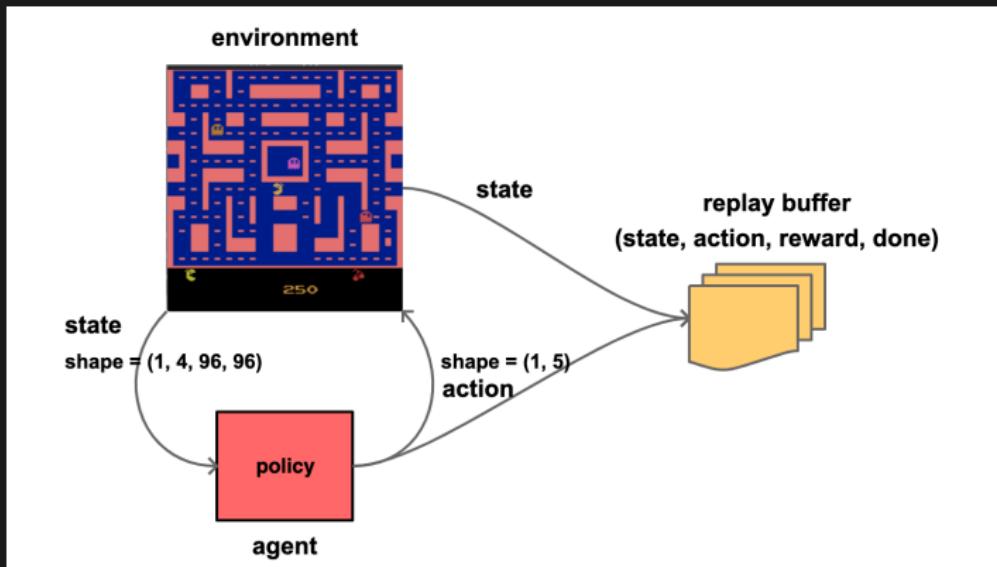
- deep Q network - **DQN**¹
- deep deterministic policy gradient - **DDPG**, D4PG²
- advantage actor critic - AC, **A2C**, A3C
- proximal policy optimization - **PPO**, TRPO³

¹Mnih et al. 2013

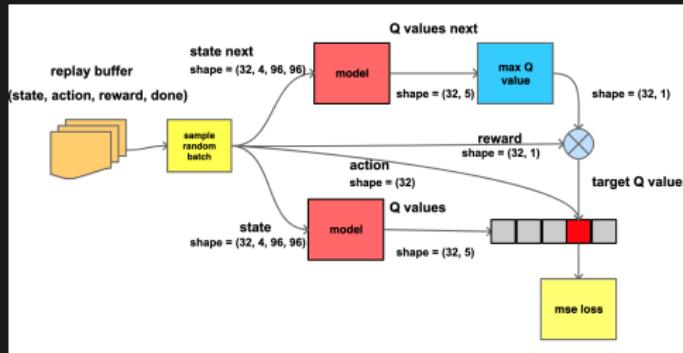
²Lillicrap, Hunt et al. 2016

³Schulman, et al. 2017

DQN - Playing Atari with Deep Reinforcement Learning



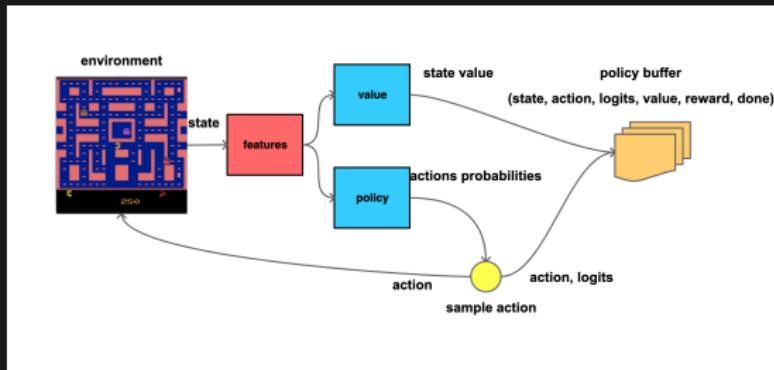
DQN - Playing Atari with Deep Reinforcement Learning



$$Q(s_n, a_n; \hat{\theta}) = R_n + \max_{\alpha \in \mathcal{A}} \hat{Q}(s_{n+1}, \alpha; \hat{\theta})$$

$$\mathcal{L}_{\theta} = (Q(s, a; \hat{\theta}) - Q(s, a; \theta))^2$$

PPO - Proximal Policy Optimization Algorithms



$$\mathcal{L} = \log \pi(a_n | s_n) A_n^\pi$$

$$\mathcal{L} = \frac{1}{N} \sum_n^N \frac{\pi^{now}(a_n | s_n)}{\pi^{prev}(a_n | s_n)} A_n^{\pi^{old}}$$

- naive policy gradient - unstable
- minimize policy divergence
- clipping or KL-divergence

problems

- hard exploration tasks (sparse rewards)
- generalisation
- sample efficiency

Montezuma's revenge - 10 years of tears?

source : <https://paperswithcode.com/sota/atari-games-on-atari-2600-montezumas-revenge>

year	name	score
2013	Playing Atari with Deep Reinforcement Learning	0
2015	Deep Reinforcement Learning with Double Q-learning	0
2017	Curiosity-driven Exploration by Self-supervised Prediction ^a	0
2021	MuZero	2500
2018	Count-Based Exploration with Neural Density Models ^b	3705
2019	Exploration by Random Network Distillation ^c	8152
2021	GoExplore* ^d	43 000

* requires environment state saving/loading

^a<https://arxiv.org/abs/1705.05363>

^b<https://arxiv.org/abs/1703.01310>

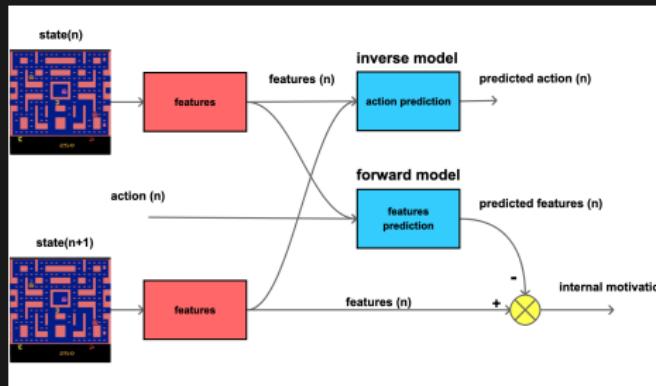
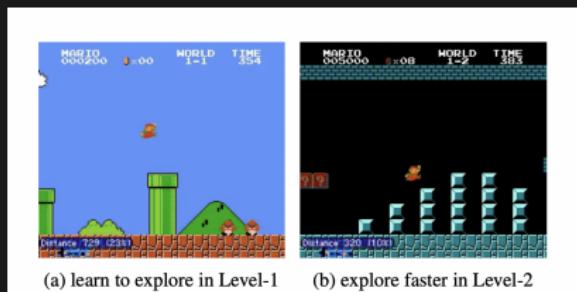
^c<https://arxiv.org/abs/1810.12894>

^d<https://arxiv.org/abs/2004.12919>

Curiosity-driven Exploration by Self-supervised Prediction ^a

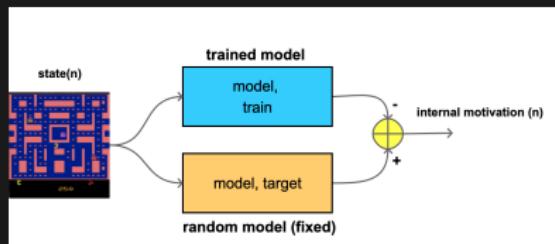
^aPathak et al. 2017

- **predict next state**, forward model
 - **motivation == prediction error**
 - not working ...



Exploration by Random Network Distillation ^a

^aBurda et al. 2018

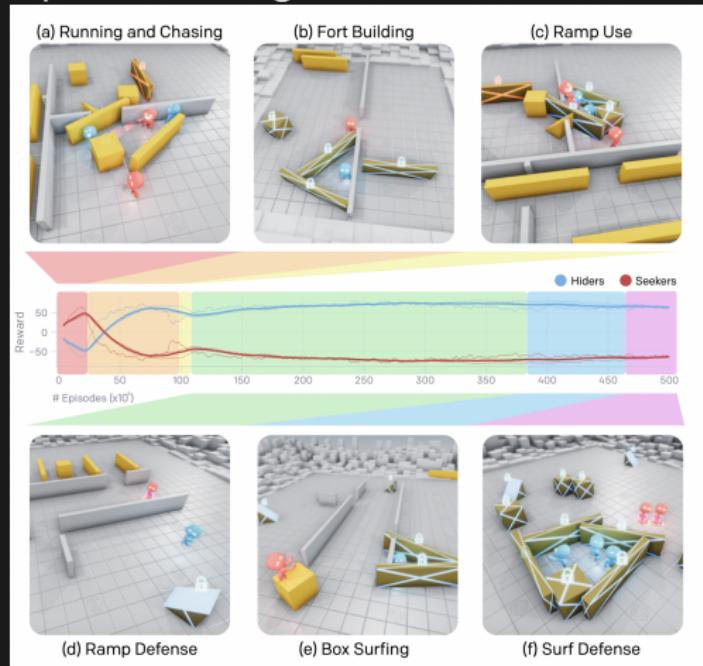


	Gravitar	Montezuma's Revenge	Pitfall!	PrivateEye	Solaris	Venture
RND	3,906	8,152	-3	8,666	3,282	1,859
PPO	3,426	2,497	0	105	3,387	0
Dynamics	3,371	400	0	33	3,246	1,712
SOTA	2,209 ¹	3,700 ²	0	15,806²	12,380¹	1,813³
Avg. Human	3,351	4,753	6,464	69,571	12,327	1,188

hide and seek ^a

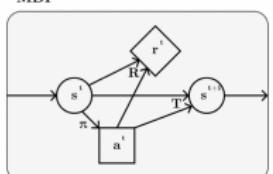
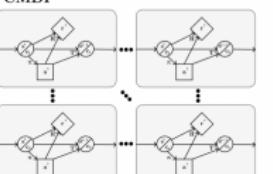
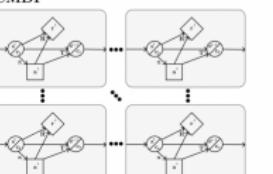
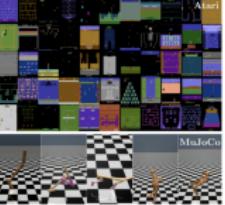
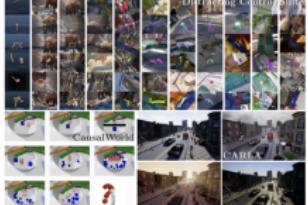
^aBaker et al. 2020

OpenAI - Emergent Tool Use from Multi-Agent Interaction



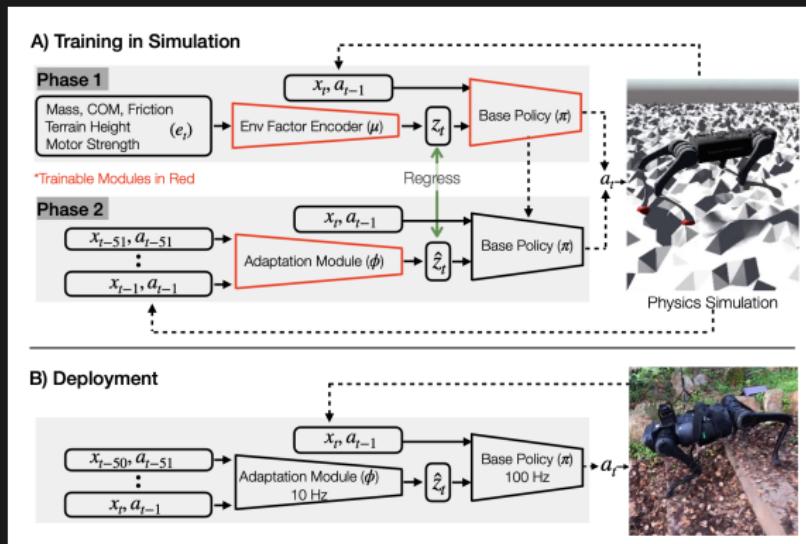
generalisation ^a

^aKirk et al. 2021

	Singleton Environments	IID Generalisation Environments	OOD Generalisation Environments
Graphical Models	 A diagram of a Markov Decision Process (MDP). It shows a state node s^i with an arrow to a reward node r^i . From r^i , an arrow points to a transition node T^i . From T^i , an arrow points to the next state node s^{i+1} . A policy node π has an arrow pointing to action node a^i . Action node a^i has arrows pointing to both s^{i+1} and r^i .	 A diagram of a Causal Markov Decision Process (CMDP). It shows two parallel causal paths from state s^i to s^{i+1} . The top path goes through a reward node r^i and a transition node T^i . The bottom path goes through an action node a^i . Both paths have policy nodes π preceding them.	 A diagram of a Causal Markov Decision Process (CMDP) under Out-of-Distribution (OOD) generalisation. It shows two parallel causal paths from state s^i to s^{i+1} . The top path goes through a reward node r^i and a transition node T^i . The bottom path goes through an action node a^i . Both paths have policy nodes π preceding them.
Train and Test Distribution	 A blue dot equals a red dot. Train = Test	 A scatter plot showing a uniform distribution of blue dots within a red rectangular boundary. $p_{\text{train}}(c) = p_{\text{test}}(c)$ Train Distribution = Test Distribution	 A scatter plot showing a uniform distribution of blue dots on the left and a red rectangular boundary on the right. $p_{\text{train}}(c) \neq p_{\text{test}}(c)$ Train Distribution \neq Test Distribution
Example Benchmarks	 Screenshots from the Alari and MuJoCo benchmarks.	 Screenshots from the OpenAI Gym and NoFluff Learning Environment benchmarks.	 Screenshots from the DivingBench, CasualWorld, and CARLA benchmarks.

RMA: Rapid Motor Adaptation for Legged Robots ^a

^aKumar et al. 2021



sample efficiency

- intuitive unit **one Montezuma experiment**
 - 128M samples runs **65hours**
 - eats less than **4G memory**
 - fits **3 experiments** into single GPU (12G)
-
- RND ⁴ $4.5 * 10^9$ samples, score 8152 on MR
 - Never give up ⁵ $3.5 * 10^{10}$ samples, score 10 000 on MR
 - SND $1.28 * 10^8$ score 10 000 on MR

NGU on my machine means **740 days !!!**

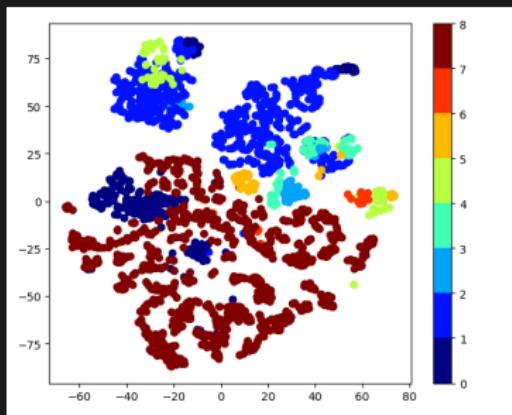
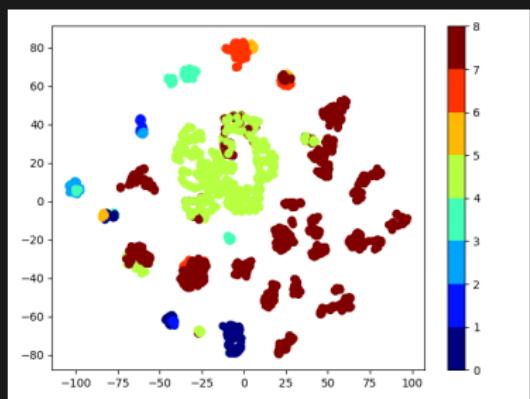
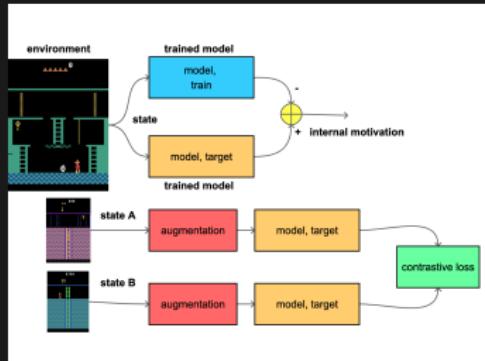
⁴Burda et al. 2018

⁵Badia et al. 2020

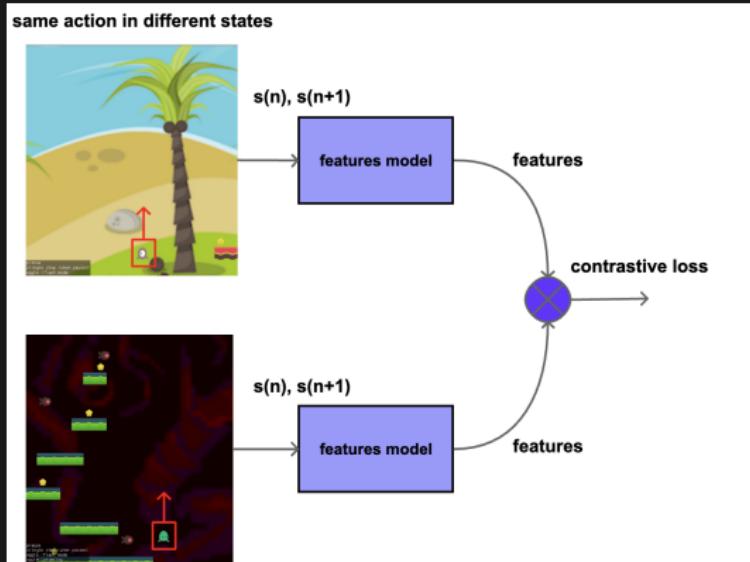
my current research

- siamese network distillation - reached SOTA score with 1/100 samples
- symmetry driven generalisation, Noether's theorem in RL

siamese network distillation



symmetry driven generalisation



$$\forall s(n), s(n+1) | a \in \mathbb{D} : f(s(n), s(n+1); \theta) = const_a$$

recommended sources

- book : Maxim Lapan, 2020, Deep Reinforcement Learning Hands-On second edition
- book : Enes Bilgin, 2020, Mastering Reinforcement Learning with Python
- youtuber : Yannic Kilcher, link
- youtuber : Two Minute Papers, link
- web : Paper With Code, link
- web : Intellabs, link

Q&A



- <https://github.com/michalnand/>
- michal.nand@gmail.com