Deep Reinforcement and InfoMax Learning

Bogdan Mazoure[*], Remi Tachet des Combes[*], Thang Doan, Philip Bachman, R Devon Hjelm

[*]Equal contribution

Introduction

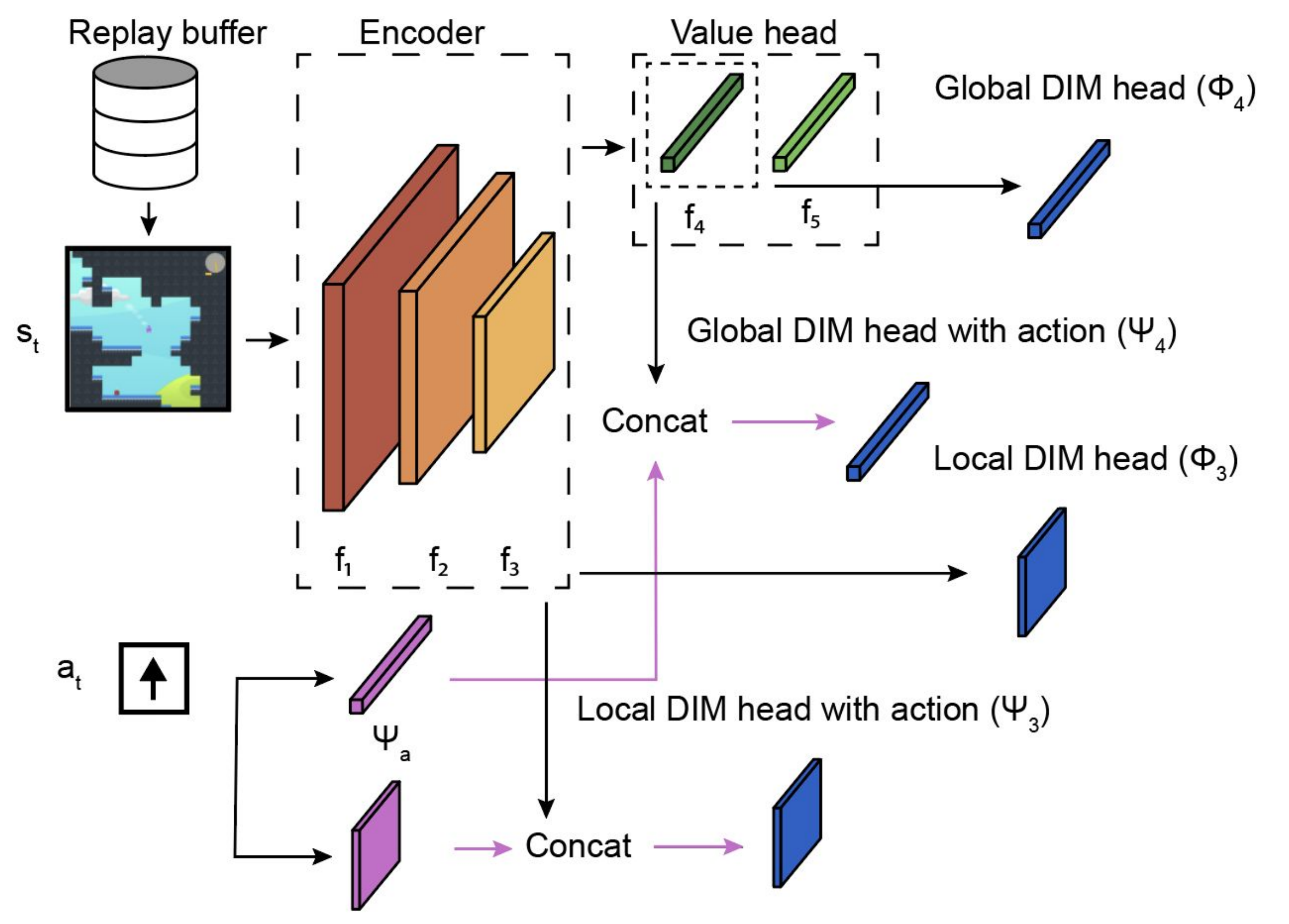
We strive for RL agents that are able to adapt quickly and reuse knowledge when presented a sequence of different tasks with variable reward functions. Viewing model-based agents from a representation learning perspective, a desired outcome is an agent that understands the underlying generative factors of the environment that determine the observed state/action sequences, leading to generalization to other environments built from the same generative factors.

In addition, learning a predictive model affords a richer learning signal than those provided by reward alone, which could reduce sample complexity compared to model-free methods.

Our work is based on the hypothesis that a model-free agent whose representations are predictive of properties of future states (beyond expected rewards) will be more capable of solving and adapting to new RL problems and, in a way, incorporate aspects of model-based learning.

Algorithm

- DRIML optimizes an auxiliary objective together with the C51 loss
- The auxiliary objective is the InfoNCE loss, which lower bounds the mutual information between state-action pairs (s_t,a_t) at time *t* and state s_{t+k} for integer k > 0.


Replay buffer
Encoder
Value head
Global DIM head (Φ_4)
s_t
f_4
f_5
Global DIM head with action (Ψ_4)
Concat
Local DIM head (Φ_3)
f_1
f_2
f_3
a_t
Ψ_a
Local DIM head with action (Ψ_3)
Concat

The main components of the DRIML algorithm. Greek letters (purple and blue) denote DIM heads, green denotes the C51 value head, orange denotes the common encoder.

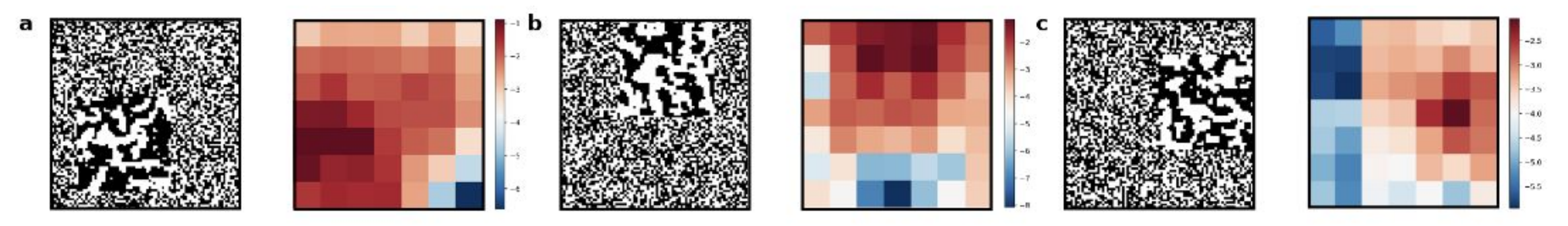
The auxiliary loss is described by the following equations, where *f* denote layers of the encoder trunk, and greek letters denote the DIM heads (either global or local).

$$\phi_{NtM}(s_t,a,s_{t+k}) := \Psi_N(f_N(s_t),\Psi_a(a_t))^\top \Phi_M(f_M(s_{t+k})),\ M,N\in\{3,4\}.$$

$$\mathcal{L}_{DIM}^{NtM} := -\mathbb{E}_{p_\pi(s_t,a_t,s_{t+k})}\mathbb{E}_{S^-}\left[\log\frac{\exp(\phi_{NtM}(s_t,a_t,s_{t+k}))}{\sum_{s'\in S^-\cup\{s_{t+k}\}}\exp(\phi_{NtM}(s_t,a_t,s'))}\right].$$

Results

Ising model



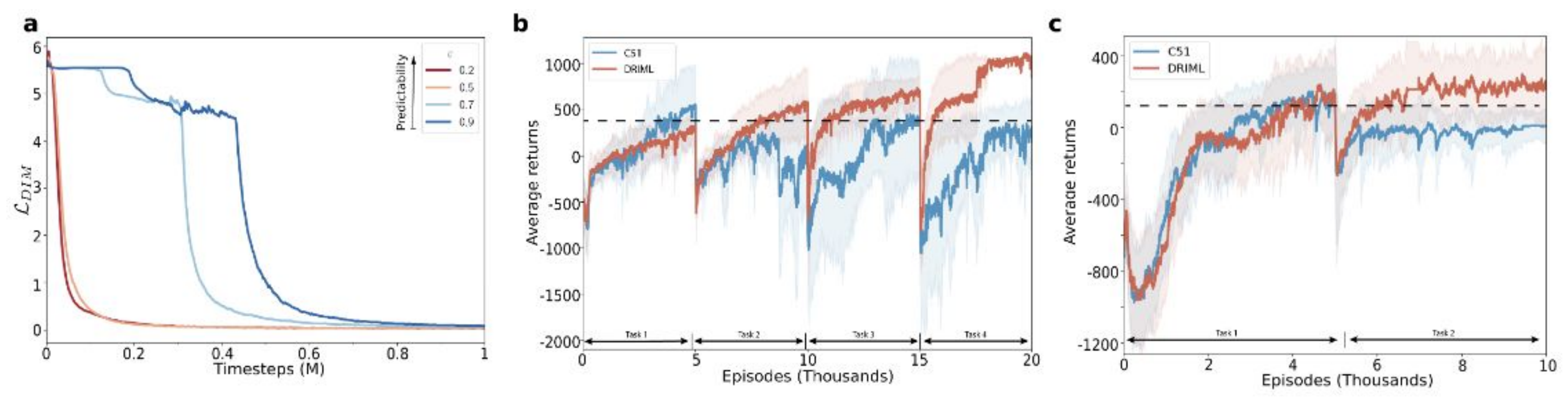
Ising models are perfect examples of quasi-deterministic structured systems. We made Ising models (temperature=1/0.4) evolve in a portion of a 84 x 84 screen, then fit the DRIML objective onto it.

DRIML successfully captures the predictable portion of the screen, even at the very beginning of the Ising evolution.

Ms.PacMan continual learning

We used a simplified version of Ms.Pacman[1] to generate
(**a**) 4 tasks where P(ghost takes a random action)=ϵ and plot the DRIML loss over training steps,
(**b**) domains where only one of 4 ghosts can kill the agent, and tasks switch after 5k trajectories, and
(**c**) a similar setup to b, except that an Ising model is evolving in the walled regions of the screen.

DRIML is able to predict deterministic portions of the screen, which help it adapt in a domain adaptation scenario, and achieve better returns.



[1]https://github.com/bmazoure/ms_pacman_gym

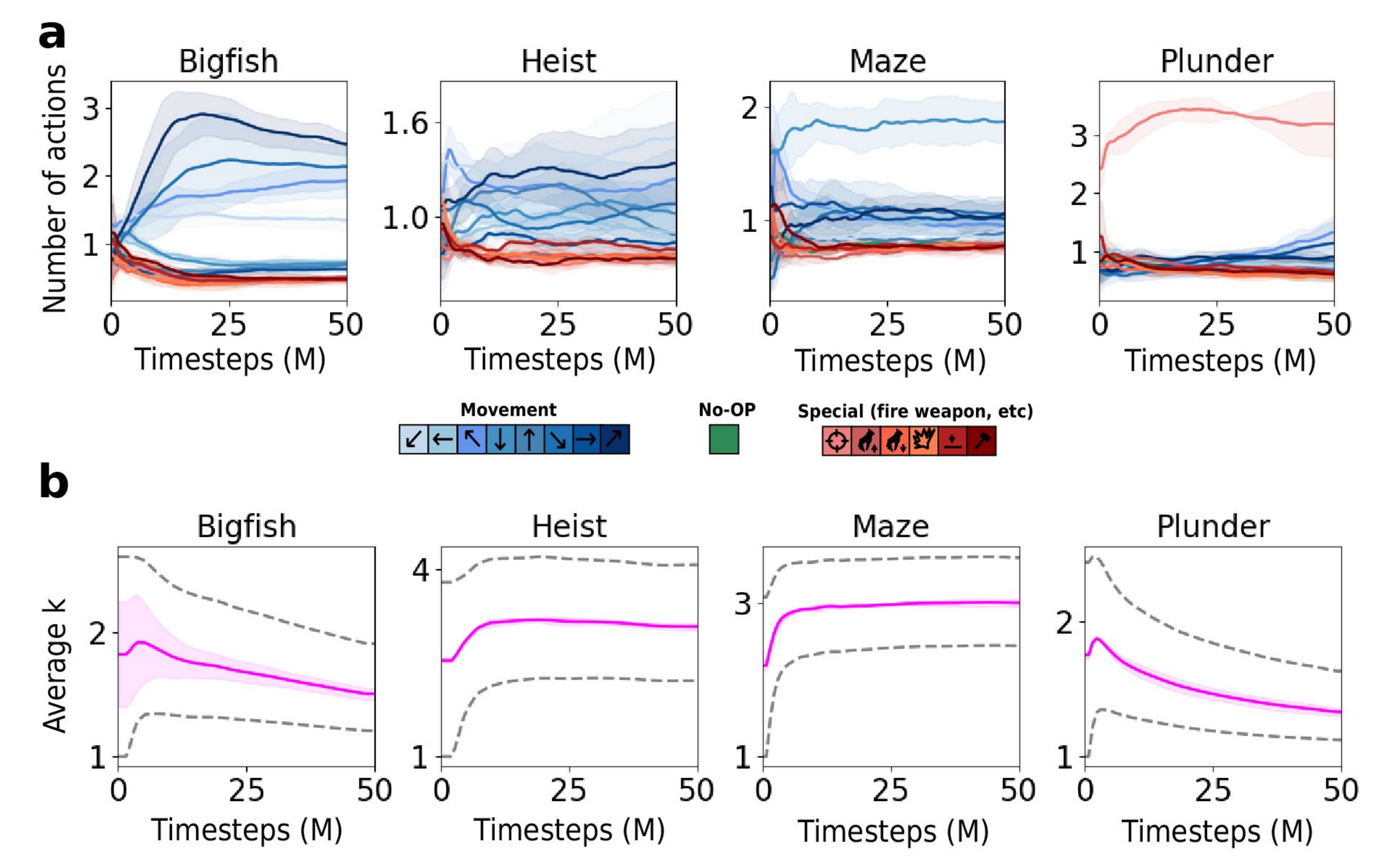
Procgen

Env	C51	CPC-1→ 5	CURL	DRIML-noact	DRIML-randk	DRIML-fix	DRIML-ada
bigfish	1.33±0.12	1.17±0.16	2.7±1.3	1.19±0.04	1.12±1.03	2.02±0.18	**4.45±0.71**
bossfight	0.57±0.05	0.52±0.07	0.6±0.06	0.47±0.01	0.56±0.03	0.67±0.02	**1.05±0.19**
caveflyer	9.19±0.29	6.4±0.56	6.94±0.25	8.26±0.26	7.92±0.15	**10.18±0.41**	6.77±0.04
chaser	0.22±0.04	0.21±0.02	0.35±0.04	0.23±0.02	0.26±0.01	0.29±0.02	**0.38±0.04**
climber	1.68±0.1	1.71±0.11	1.75±0.09	1.57±0.01	2.21±0.48	**2.26±0.05**	2.2±0.08
coinrun	**29.7±5.44**	11.4±1.55	21.17±1.94	13.15±1.21	21.55±1.97	27.24±1.92	22.88±0.4
dodgeball	1.2±0.08	1.05±0.04	1.09±0.04	1.22±0.04	1.19±0.03	1.28±0.02	**1.44±0.06**
fruitbot	3.86±0.96	4.56±0.93	4.89±0.71	5.42±1.33	6.84±0.24	5.4±1.02	**9.53±0.29**
heist	1.54±0.1	0.93±0.08	1.06±0.05	1.04±0.02	1.0±0.05	1.3±0.05	**1.89±0.02**
jumper	**13.22±0.83**	2.28±0.44	10.27±0.61	4.31±0.64	5.62±0.27	12.64±0.64	12.16±0.42
leaper	5.03±0.14	4.01±0.71	3.94±0.46	5.4±0.09	4.24±1.17	6.17±0.29	**6.35±0.46**
maze	2.36±0.09	1.14±0.08	0.82±0.2	1.44±0.26	1.18±0.03	1.38±0.08	**2.62±0.1**
miner	0.13±0.01	0.13±0.02	0.1±0.0	0.12±0.01	0.15±0.01	0.14±0.01	**0.19±0.02**
ninja	**9.36±0.01**	6.23±0.82	5.84±1.21	6.44±0.22	8.13±0.26	9.21±0.25	8.74±0.28
plunder	2.99±0.07	3.0±0.06	2.77±0.14	3.2±0.05	3.34±0.09	3.37±0.17	**3.58±0.04**
starpilot	2.44±0.12	2.87±0.05	2.68±0.09	3.7±0.3	3.93±0.04	**4.56±0.21**	2.63±0.16
Norm.score	1.0	0.23	0.52	0.59	0.92	1.48	1.9

Average train-time rewards on 500 fixed levels of Procgen. Normalization is performed with respect to the lowest score in each row.

Adaptive selection of predictive timescale

DRIML-ada performs the best on most Procgen tasks; the predictive timescale k is sampled from a non-homogeneous geometric distributions, with probabilities of $P(A_{t+1}|A_t)$ learned at training time. This avoids DRIML to learn “redundant” action sequences, and focuses on timescales where the agent’s behavior switches drastically.



References

Hjelm et al. (2019). “Learning deep representations by mutual information estimation and maximization.” In: ICLR.

Bachman et al. (2019). “Learning Representations by Maximizing Mutual Information Across Views.”. In: NeurIPS.

Contact: bogdan.mazoure@mail.mcgill.ca

Paper: https://arxiv.org/abs/2006.07217

Blog: https://bmazoure.github.io/posts/deep-rl-infomax-learning/

SCAN ME
Paper

SCAN ME
Blog