Understanding Self-Predictive RL

Tianwei Ni, Benjamin Eysenbach, Erfan Seyedsalehi, Michel Ma,
Clement Gehring, Aditya Mahajan, Pierre-Luc Bacon

Donghu Kim

Representation Learning

<u>Example: Visual tasks (Image classification, Segmentation, Depth estimation ⋯)</u>

A classic problem, curse of dimensionality.

We need to embed the input into a low-dimensional space, but that causes information loss.

Then what information should we keep, and what should we discard?

Obviously, the ones necessary to solve our task!

Representation Learning = Learning to encode large inputs into smaller embeddings, *while preserving the information required for the task(s).*

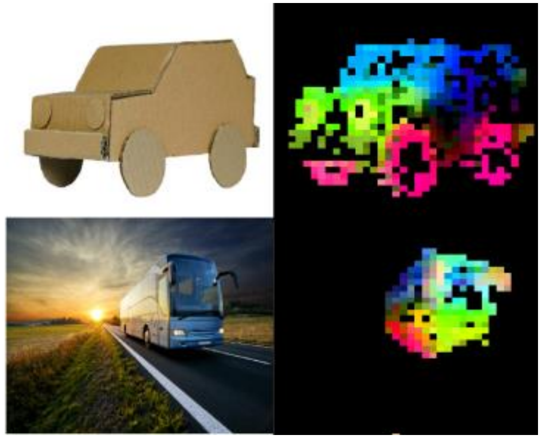
Representation Learning

End-to-end deep-learning is one form of representation learning.

e.g., CNN-encoder + MLP-predictor

Self-supervised / Unsupervised methods get representations useful for downstream tasks.

e.g., DINOv2



Representation Learning for Reinforcement Learning

Similarly, representation learning is necessary in reinforcement learning as well.

Again, these could be learned by end-to-end deep learning,

- e.g., Any model-free algorithm with deep networks (DQN, SAC, DDPG, $\cdots$)

or by self-supervised / unsupervised objectives.

- e.g., Dreamer, TD-MPC, CURL, SPR, $\cdots$

Introduction

Representation learning has become a hot topic in RL since around 2020.

What should we choose, out of all these?

ALM

LSFM

TD-MPC

Dreamer

OFENet

MuZero

EfficientZero

TCRL
(TD-MPC w/o planning)

VPN

PlaNet

PSR

TD7

DeepMDP

SPR

SAC-AE
(SAC + AutoEncoder)

CRAR

CURL

DRIML

DBC

Introduction

Representation learning has become a hot topic in RL since around 2020.

What if I told you that most of these boil down to latent self-prediction vs observation reconstruction?

ALM

LSFM

TD-MPC

EfficientZero

TCRL
(TD-MPC w/o planning)

DeepMDP

SPR

CRAR

DRIML

DBC

ϕ_L

Dreamer

OFENet

PlaNet

PSR

SAC-AE
(SAC + AutoEncoder)

ϕ_O

MuZero

VPN

TD7

CURL

ignore these guys :)

Basic Notations

Notation will be focused on POMDPs than MDPs.

- MDP: Full access to the world/environment via state s_t.
- POMDP: Partial access to the world/environment via observation o_t.
- Unlike MDP, POMDP agents benefit from using history $h_t = (h_{t-1}, a_{t-1}, o_t) = (o_1, a_1, o_2, a_2, \dots o_t)$

History Representations

- Encoder ϕ
- History representation $z = \phi(h)$

Abstraction Theory

"Abstraction can be thought of as a process that maps the ground representation, the original description of a problem, to an abstract representation, a much more compact and easier one to work with." (Li et al., 2006)

⋯so just embedding to a latent space.

Abstraction theory defines what kind of abstraction we want.

X-abstraction : The encoder provides an X-abstraction if it preserves all necessary information required for X.

Representation Learning = Learning to encode large inputs into smaller embeddings, *while preserving the information required for the task(s).*

[1] Towards a Unified Theory of State Abstraction for MDPs., Li et al., AIandM 2022.

3 Key Abstractions for MDPs

Representation Learning = Learning to encode large inputs into smaller embeddings, *while preserving the information required for the task(s).*

What kind of information should be preserved in RL?

1. Information for predicting the return → ϕ_{Q^*} (Return prediction)
2. Information for predicting environment dynamics → ϕ_L (Latent self-prediction)
3. Information for predicting environment dynamics & observations → ϕ_O (Observation prediction)

3 Key Abstractions for MDPs

1. Q*-irrelevance abstraction (ϕ_{Q^*})

= ϕ_{Q^*} preserves necessary information for predicting the true return Q*.

= Formally, $\phi_{Q^*}(h_i) = \phi_{Q^*}(h_j) \rightarrow Q^*(h_i, a) = Q^*(h_j, a), \forall a$

Learned by default in end-to-end model-free algorithms (DQN, SAC, ...)

Also learned by default in classic model-based algorithms (Dyna, Dreamer?)

3 Key Abstractions for MDPs

2. Self-predictive abstraction / Model-irrelevance abstraction (ϕ_L = RP + ZP)

= ϕ_L preserves necessary information for predicting environment dynamics (reward + transition dynamics).

= ϕ_L preserves necessary information for expected reward prediction (RP)

AND for next latent (z) distribution prediction (ZP)

A weaker version of ZP is EZP – preserving information for expected next latent prediction (EZP)

$$\exists R_z : \mathcal{Z} \times \mathcal{A} \to \mathbb{R}, \quad s.t. \quad \mathbb{E}[r \mid h, a] = R_z(\phi_L(h), a), \quad \forall h, a, \tag{RP}$$

$$\exists P_z : \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{Z}), \quad s.t. \quad P(z' \mid h, a) = P_z(z' \mid \phi_L(h), a), \quad \forall h, a, z', \tag{ZP}$$

$$\mathbb{E}[z' \mid h, a] = \mathbb{E}[z' \mid \phi_L(h), a], \quad \forall h, a. \tag{EZP}$$

*RP: There exists a function R_z capable of predicting the rewards from the representation $\phi_L(h)$ = The representation have all information for reward prediction

3 Key Abstractions for MDPs

3. Observation-predictive abstraction / Belief abstraction (ϕ_O = RP + OP + Rec)

= ϕ_O preserves necessary information for predicting environment dynamics and its observations.

= ϕ_O preserves necessary information for expected reward prediction (RP)

AND for next observation (o) prediction (OP)

AND is a recurrent encoder (Rec)

Rec condition is satisfied by regular feedforward or recurrent networks, *but not by Transformers*.

(assume to be always satisfied)

Observation reconstruction (OR) is a condition closely related to OP.

$$\exists \psi_z : \mathcal{Z} \times \mathcal{A} \times \mathcal{O} \to \mathcal{Z}, \quad s.t. \quad \phi(h') = \psi_z(\phi_O(h), a, o'), \quad \forall h, a, o', \tag{Rec}$$

$$\exists P_o : \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{O}), \quad s.t. \quad P(o' \mid h, a) = P_o(o' \mid \phi_O(h), a), \quad \forall h, a, o', \tag{OP}$$

$$\exists \psi_o : \mathcal{Z} \to \mathcal{O}, \quad s.t. \quad o = \psi_o(\phi_O(h)), \quad \forall h. \tag{OR}$$

[1] Approximate information state for approximate planning and reinforcement learning in partially observed systems., Subramanian et al., J. Mach. Learn. Res., 2022.

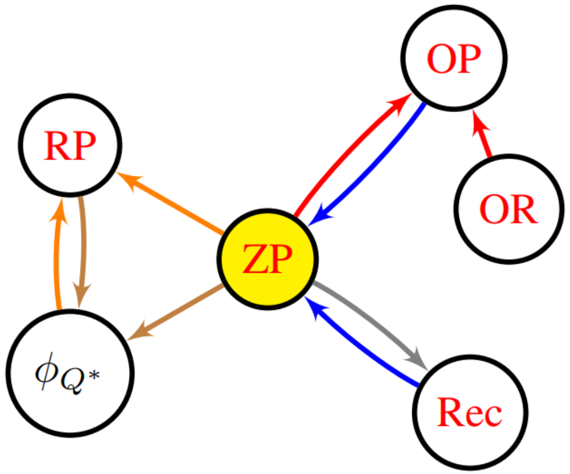
Implication Graph

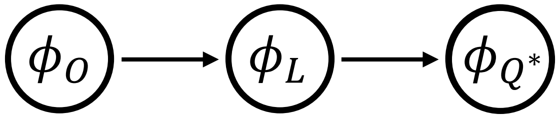
Authors prove that the conditions imply each other!

e.g., ZP + OR = OP (next latent prediction + obs reconstruction = next obs prediction)

In the same sense, abstractions also imply each other.

e.g., ϕ_O = OP + RP + Rec = ZP + OR + RP + Rec = ϕ_L + OR + Rec





*The source nodes of the edges with the same color together imply the target node.

Back to Introduction

Representation learning has become a hot topic in RL since around 2020.

ALM

LSFM

TD-MPC

Dreamer

OFENet

MuZero

EfficientZero

TCRL
(TD-MPC w/o planning)

PlaNet

VPN

PSR

TD7

DeepMDP

SPR

SAC-AE
(SAC + AutoEncoder)

CRAR

CURL

DRIML

DBC

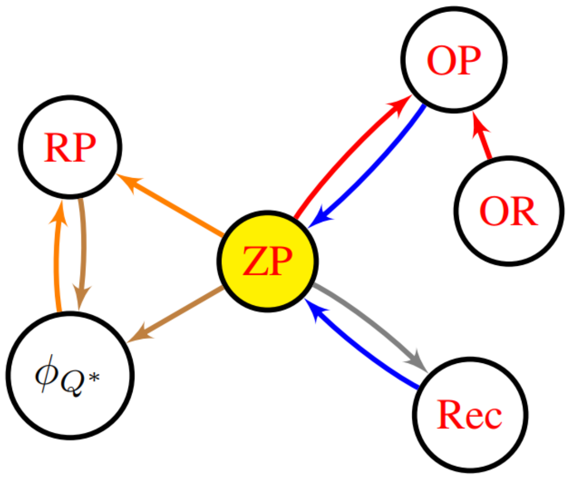
Back to Introduction

Representation learning has become a hot topic in RL since around 2020.

What if I told you that most of these just end up learning either ϕ_L or ϕ_O?

Work	PO?	Abstraction	Conditions	ZP loss	ZP target
Model-Free & Classic Model-Based RL	✗	ϕ_{Q^*}	ϕ_{Q^*}	N/A	N/A
(+VPN) MuZero (Schrittwieser et al., 2020)	✗	unknown	ϕ_{Q^*} + RP	N/A	N/A
MICo (Castro et al., 2021)	✗	unknown	ϕ_{Q^*} + metric	N/A	N/A
CRAR (François-Lavet et al., 2019)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP + reg.	ℓ_2	online
DeepMDP (Gelada et al., 2019)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP	W (ℓ_2)	online
SPR (Schwarzer et al., 2020)	✗	ϕ_L	ϕ_{Q^*} + ZP	cos	EMA
DBC (Zhang et al., 2020)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP + metric	FKL	detached
LSFM (Lehnert & Littman, 2020)	✗	ϕ_L	ϕ_{Q^*} + RP + EZP	SF	detached
Baseline in (Tomar et al., 2021)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP	ℓ_2	detached
EfficientZero (Ye et al., 2021)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP	cos	detached
TD-MPC (Hansen et al., 2022)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP	ℓ_2	EMA
ALM (Ghugare et al., 2022)	✗	ϕ_L	ϕ_{Q^*} + ZP	RKL	EMA
TCRL (Zhao et al., 2023)	✗	ϕ_L	RP + ZP	cos	EMA
OFENet (Ota et al., 2020)	✗	ϕ_O	ϕ_{Q^*} + OP	N/A	N/A
Recurrent Model-Free RL	✓	ϕ_{Q^*}	ϕ_{Q^*}	N/A	N/A
PBL (Guo et al., 2020)	✓	ϕ_L	ϕ_{Q^*} + ZP	ℓ_2	detached
AIS (Subramanian et al., 2022)	✓	ϕ_L, ϕ_O	RP + ZP or OP	ℓ_2, FKL	detached
(PlaNet) Belief-Based Methods	✓	ϕ_O	RP + ZP + OR	FKL	online
Causal States (Zhang et al., 2019)	✓	ϕ_O	RP + OP	N/A	N/A
Minimalist ϕ_L (this work)	✓	ϕ_L	ϕ_{Q^*} + ZP	ℓ_2, KL	stop-grad

* SAC-AE (ϕ_O) = ϕ_{Q^*} + OR
* CURL (unknown) = ϕ_{Q^*} + weak OR via contrastive loss
* TD7 (unknown) = ZP



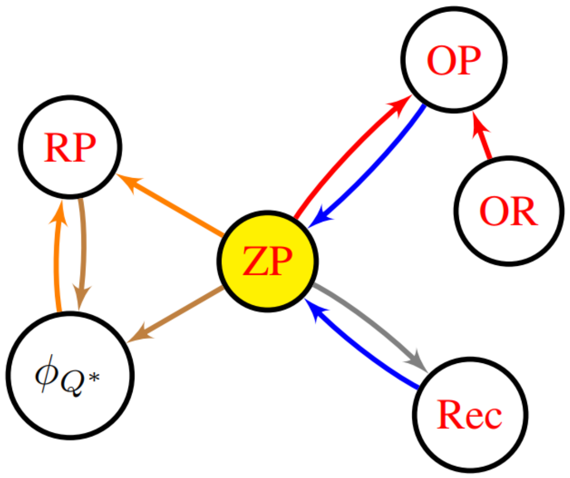
ϕ_L = RP + ZP

ϕ_O = RP + OP + Rec

Minimalist Representation Learning

If all these methods end up with the same abstraction, why not use the most minimal condition set?

Work	PO?	Abstraction	Conditions	ZP loss	ZP target
Model-Free & Classic Model-Based RL	✗	ϕ_{Q^*}	ϕ_{Q^*}	N/A	N/A
MuZero (Schrittwieser et al., 2020)	✗	unknown	ϕ_{Q^*} + RP	N/A	N/A
MICo (Castro et al., 2021)	✗	unknown	ϕ_{Q^*} + metric	N/A	N/A
CRAR (François-Lavet et al., 2019)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP + reg.	ℓ_2	online
DeepMDP (Gelada et al., 2019)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP	W (ℓ_2)	online
SPR (Schwarzer et al., 2020)	✗	ϕ_L	ϕ_{Q^*} + ZP	cos	EMA
DBC (Zhang et al., 2020)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP + metric	FKL	detached
LSFM (Lehnert & Littman, 2020)	✗	ϕ_L	ϕ_{Q^*} + RP + EZP	SF	detached
Baseline in (Tomar et al., 2021)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP	ℓ_2	detached
EfficientZero (Ye et al., 2021)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP	cos	detached
TD-MPC (Hansen et al., 2022)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP	ℓ_2	EMA
ALM (Ghugare et al., 2022)	✗	ϕ_L	ϕ_{Q^*} + ZP	RKL	EMA
TCRL (Zhao et al., 2023)	✗	ϕ_L	RP + ZP	cos	EMA
OFENet (Ota et al., 2020)	✗	ϕ_O	ϕ_{Q^*} + OP	N/A	N/A
Recurrent Model-Free RL	✓	ϕ_{Q^*}	ϕ_{Q^*}	N/A	N/A
PBL (Guo et al., 2020)	✓	ϕ_L	ϕ_{Q^*} + ZP	ℓ_2	detached
AIS (Subramanian et al., 2022)	✓	ϕ_L, ϕ_O	RP + ZP or OP	ℓ_2, FKL	detached
Belief-Based Methods	✓	ϕ_O	RP + ZP + OR	FKL	online
Causal States (Zhang et al., 2019)	✓	ϕ_O	RP + OP	N/A	N/A
Minimalist ϕ_L (this work)	✓	ϕ_L	ϕ_{Q^*} + ZP	ℓ_2, KL	stop-grad
Minimalist ϕ_O	,,	,,	ϕ_{Q^*} + OP	,,	,,



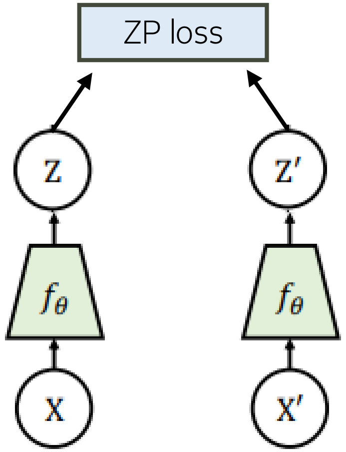
$$\phi_L = \text{RP} + \text{ZP}$$
$$\phi_O = \text{RP} + \text{OP} + \text{Rec}$$

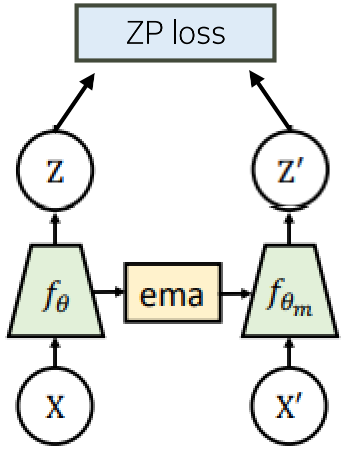
How Should We Learn ZP? a.k.a. theoretical justification for stop-grad

The self-predictive nature of latent prediction (ZP) poses a significant challenge.

- The infamous representation collapse: ZP can be satisfied by mapping every input into a single latent.
- Mitigated by (1) Other losses like RP, or (2) Contrastive loss, or (3) Detaching the target encoder's gradient

Work	PO?	Abstraction	Conditions	ZP loss	ZP target
Model-Free & Classic Model-Based RL	✗	ϕ_{Q^*}	ϕ_{Q^*}	N/A	N/A
MuZero (Schrittwieser et al., 2020)	✗	unknown	ϕ_{Q^*} + RP	N/A	N/A
MICo (Castro et al., 2021)	✗	unknown	ϕ_{Q^*} + metric	N/A	N/A
CRAR (François-Lavet et al., 2019)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP + reg.	ℓ_2	online
DeepMDP (Gelada et al., 2019)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP	W (ℓ_2)	online
SPR (Schwarzer et al., 2020)	✗	ϕ_L	ϕ_{Q^*} + ZP	cos	EMA
DBC (Zhang et al., 2020)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP + metric	FKL	detached
LSFM (Lehnert & Littman, 2020)	✗	ϕ_L	ϕ_{Q^*} + RP + EZP	SF	detached
Baseline in (Tomar et al., 2021)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP	ℓ_2	detached
EfficientZero (Ye et al., 2021)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP	cos	detached
TD-MPC (Hansen et al., 2022)	✗	ϕ_L	ϕ_{Q^*} + RP + ZP	ℓ_2	EMA
ALM (Ghugare et al., 2022)	✗	ϕ_L	ϕ_{Q^*} + ZP	RKL	EMA
TCRL (Zhao et al., 2023)	✗	ϕ_L	RP + ZP	cos	EMA
OFENet (Ota et al., 2020)	✗	ϕ_O	ϕ_{Q^*} + OP	N/A	N/A
Recurrent Model-Free RL	✓	ϕ_{Q^*}	ϕ_{Q^*}	N/A	N/A
PBL (Guo et al., 2020)	✓	ϕ_L	ϕ_{Q^*} + ZP	ℓ_2	detached
AIS (Subramanian et al., 2022)	✓	ϕ_L, ϕ_O	RP + ZP or OP	ℓ_2, FKL	detached
Belief-Based Methods	✓	ϕ_O	RP + ZP + OR	FKL	online
Causal States (Zhang et al., 2019)	✓	ϕ_O	RP + OP	N/A	N/A
Minimalist ϕ_L (this work)	✓	ϕ_L	ϕ_{Q^*} + ZP	ℓ_2, KL	stop-grad





How Should We Learn ZP? a.k.a. theoretical justification for stop-grad

tl;dr

Theoretically, latent self-predictive losses are problematic in stochastic environments.

Theoretically, using stop gradient makes the problem less bad.

Authors will use stop-gradient from now on.

How Should We Learn ZP? a.k.a. theoretical justification for stop-grad

The self-predictive nature of latent prediction (ZP) poses a significant challenge.

The infamous representation collapse: ZP can be satisfied by mapping every input into a single latent.

Ideal ZP loss is unusable due to double sampling issue (i.e., cannot sample from the environment twice).

$$\mathcal{L}_{ZP,\mathbb{D}}(\phi, \theta; h, a) := \mathbb{E}_{z\sim\mathbb{P}_\phi(|h)}[\mathbb{D}(\mathbb{P}_\theta(z' \mid z, a) \mid\mid \mathbb{P}_\phi(z' \mid h, a))], \tag{1}$$

Instead, practical L2 or KL loss are used. However, these are theoretically okay only in deterministic MDPs.

$$J_\ell(\phi, \theta, \widetilde{\phi}; h, a) := \mathbb{E}_{o'\sim P(|h,a)}\left[\|g_\theta(f_\phi(h), a) - f_{\widetilde{\phi}}(h')\|_2^2\right], \tag{2}$$

$$J_{D_\mathtt{f}}(\phi, \theta, \widetilde{\phi}; h, a) := \mathbb{E}_{z\sim\mathbb{P}_\phi(|h), o'\sim P(|h,a)}\left[D_\mathtt{f}\left(\mathbb{P}_{\widetilde{\phi}}(z' \mid h') \mid\mid \mathbb{P}_\theta(z' \mid z, a)\right)\right], \tag{3}$$

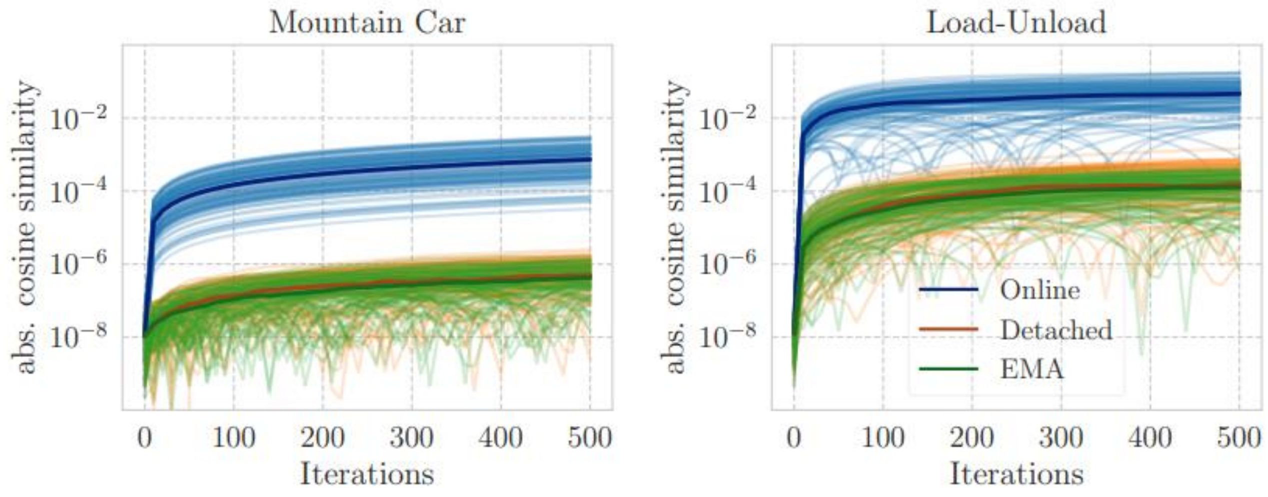
Proposition 1 (The practical ℓ_2 objective Eq. 2 is an **upper bound** of the ideal objective Eq. 1 $\mathcal{L}_{ZP,\ell}(\phi, \theta; h, a)$ that targets EZP condition. The equality holds in deterministic environments.)**.**

Proposition 2 (The practical $\mathtt{f}$-divergence objective Eq. 3 is an **upper bound** of the ideal objective Eq. 1 $\mathcal{L}_{ZP,D_\mathtt{f}}(\phi, \theta; h, a)$ that targets ZP condition. The equality holds in deterministic environments.)**.**

How Should We Learn ZP? a.k.a. theoretical justification for stop-grad

Luckily, stop gradient operation mitigates the problem.

- For L2 loss specifically, using stop gradient guarantees stationary point (online target doesn't).
- For linear models, stop gradient provably avoids representational collapse.



Minimalist ϕ_L Algorithm

ϕ_{Q^*} + ZP (with L2 loss and stop-gradient)

Any Model-free algorithm + Auxiliary latent self-prediction loss

Algorithm 1 Minimalist ϕ_L: learning self-predictive representations in RL

Require: Encoder $f_\phi : \mathcal{H}_t \to \mathcal{Z}$, Actor $\pi_\nu : \mathcal{Z} \to \mathcal{A}$, Critic $Q_\omega : \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$, Latent Transition Model $g_\theta : \mathcal{Z} \times \mathcal{A} \to \mathcal{Z}$. Learning Rate $\alpha > 0$ and Loss Coefficient $\lambda > 0$.

1: **procedure** UPDATE(h, a, o', r)

2: Compute any model-free RL loss $\mathcal{L}_{\text{RL}}$ (based on DDPG (Lillicrap et al., 2016) here) let $Q^{\text{tar}}(h', r) := r + \gamma Q_{\overline{\omega}}(f_{\overline{\phi}}(h'), \pi_{\overline{\nu}}(f_{\overline{\phi}}(h'))),$

$$\mathcal{L}_{\text{RL}}(\phi, \omega, \nu; h', r) = (Q_\omega(f_\phi(h), a) - Q^{\text{tar}}(h', r))^2 - Q_{\overline{\omega}}(f_{\overline{\phi}}(h), \pi_\nu(f_\phi(h))). \tag{4}$$

3: Compute the auxiliary ZP loss $\mathcal{L}_{\text{aux}}(\phi, \theta; h') = \|g_\theta(f_\phi(h), a) - f_{\overline{\phi}}(h')\|_2^2$.

4: Optimize all parameters using the sum of losses:

$$[\phi, \theta, \nu, \omega] \leftarrow [\phi, \theta, \nu, \omega] - \alpha \nabla (\mathcal{L}_{\text{RL}}(\phi, \omega, \nu; h', r) + \lambda \mathcal{L}_{\text{aux}}(\phi, \theta; h')). \tag{5}$$

Minimalist ϕ_L Algorithm

Enables comparing $\phi_{Q^*}, \phi_L, \phi_O$ without changing the RL algorithm.

Set $\lambda = 0 \rightarrow \phi_{Q^*}$
Change ZP to OP $\rightarrow \phi_O$

Algorithm 1 Minimalist ϕ_L: learning self-predictive representations in RL

Require: Encoder $f_\phi : \mathcal{H}_t \to \mathcal{Z}$, Actor $\pi_\nu : \mathcal{Z} \to \mathcal{A}$, Critic $Q_\omega : \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$, Latent Transition Model $g_\theta : \mathcal{Z} \times \mathcal{A} \to \mathcal{Z}$. Learning Rate $\alpha > 0$ and Loss Coefficient $\lambda > 0$.

1: **procedure** UPDATE(h, a, o', r)

2: Compute any model-free RL loss $\mathcal{L}_{\text{RL}}$ (based on DDPG (Lillicrap et al., 2016) here) let $Q^{\text{tar}}(h', r) := r + \gamma Q_{\overline{\omega}}(f_{\overline{\phi}}(h'), \pi_{\overline{\nu}}(f_{\overline{\phi}}(h')))$,

$$\mathcal{L}_{\text{RL}}(\phi, \omega, \nu; h', r) = (Q_\omega(f_\phi(h), a) - Q^{\text{tar}}(h', r))^2 - Q_{\overline{\omega}}(f_{\overline{\phi}}(h), \pi_\nu(f_\phi(h))). \quad (4)$$

3: Compute the auxiliary ZP loss $\mathcal{L}_{\text{aux}}(\phi, \theta; h') = \|g_\theta(f_\phi(h), a) - f_{\overline{\phi}}(h')\|_2^2$.

4: Optimize all parameters using the sum of losses:

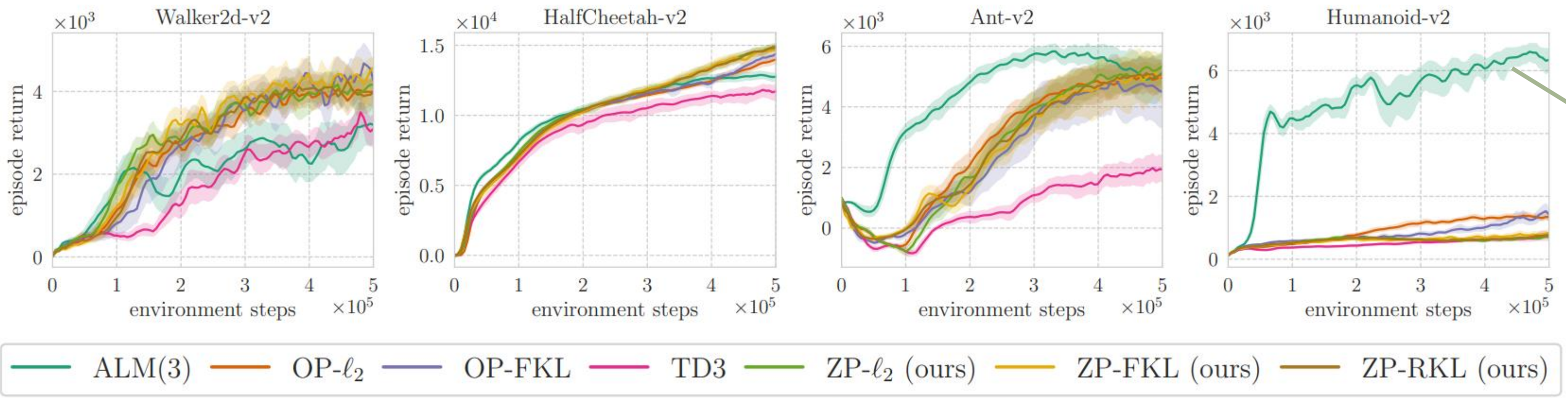
$$[\phi, \theta, \nu, \omega] \leftarrow [\phi, \theta, \nu, \omega] - \alpha \nabla(\mathcal{L}_{\text{RL}}(\phi, \omega, \nu; h', r) + \lambda \mathcal{L}_{\text{aux}}(\phi, \theta; h')). \quad (5)$$

Does The Minimalist Algorithm Work?

Environment: Mujoco

Base algorithm: ALM(3) → Mujoco SOTA

- Minimalist ϕ_L (ZP-L2, ZP-FKL, ZP-RKL) outperforms ALM(3) in all cases except Humanoid-v2.
- Both ϕ_L, ϕ_O outperformed L_{Q^*} (TD3).



"due to superior policy optimization"

Figure 3: **Decoupling representation learning from policy optimization using our algorithm based on ALM(3) (Ghugare et al., 2022).** Comparison between ϕ_{Q^*} (TD3), ϕ_L (our algorithm (ZP-ℓ_2, ZP-FKL, ZP-RKL) and ALM(3)), ϕ_O (OP-ℓ_2, OP-FKL), in the standard MuJoCo benchmark for 500k steps, averaged over 12 seeds. The observation dimension increases from left figure to right figure $(17, 17, 111, 376)$.

ϕ_L vs ϕ_O

Hypothesis: Observation prediction (ϕ_O) is fragile to distractors.

Environment: Distracting Mujoco (Gaussian noise)

Base algorithm: ALM(3) → Mujoco SOTA

- ϕ_L algorithms were more robust than ϕ_O (OP-L2, OP-FKL).

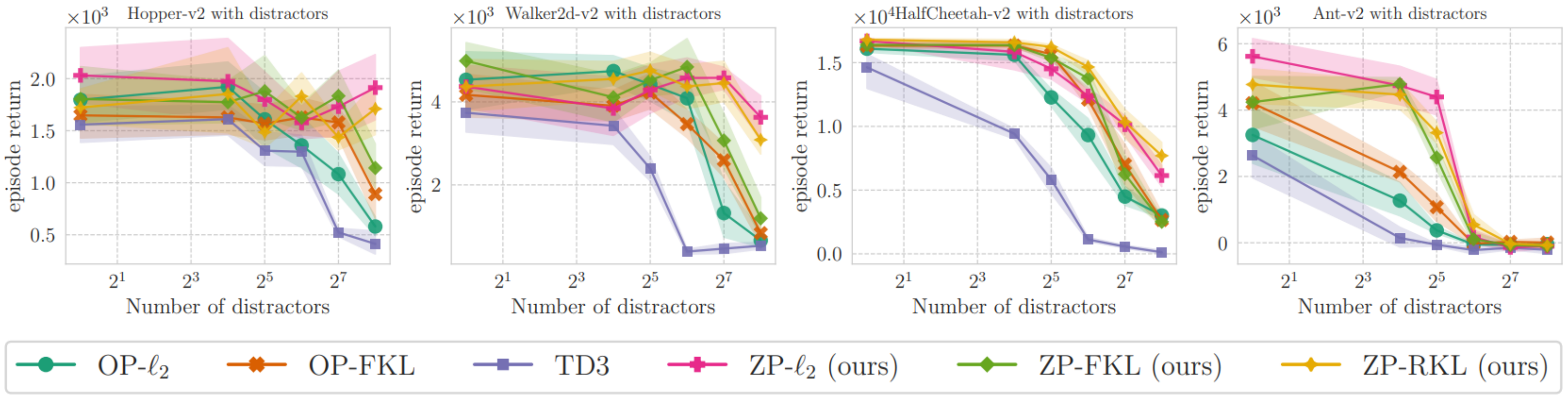


Figure 5: **Self-predictive representations are more robust.** Comparison between ϕ_{Q^*} (TD3), ϕ_L (ZP-ℓ_2, ZP-FKL, ZP-RKL) using our algorithm, ϕ_O (OP-ℓ_2, OP-FKL) in the **distracting** MuJoCo benchmark, varying the distractor dimension from 2^4 to 2^8, averaged over 12 seeds. The y-axis is final performance at 1.5M steps.

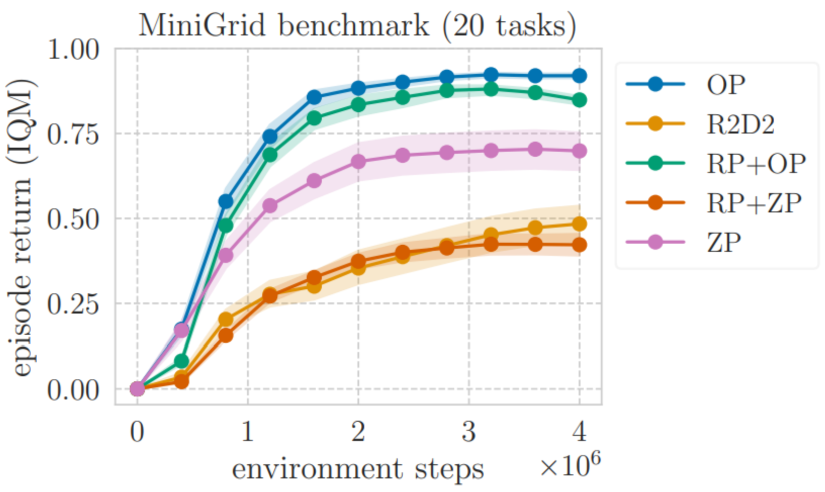
ϕ_L vs ϕ_O

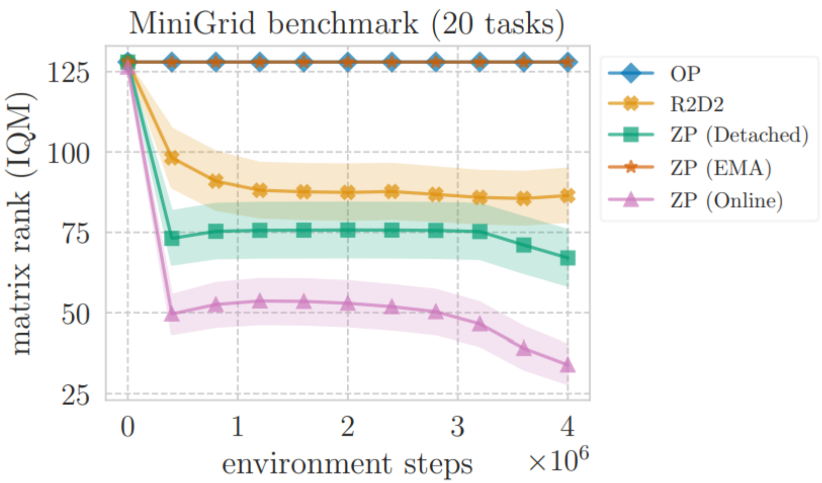
Hypothesis: Observation prediction (ϕ_O) is fragile to distractors.

Environment: MiniGrid (POMDP + Sparse reward)

Base algorithm: R2D2 (Distributed RNN)

- R2D2 + OP (ϕ_O) was more effective than R2D2 + ZP (ϕ_L)





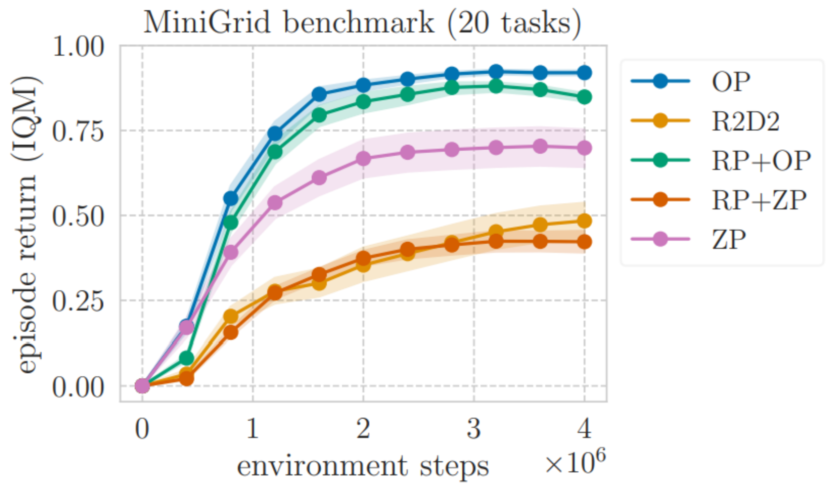
End-to-end vs Phased

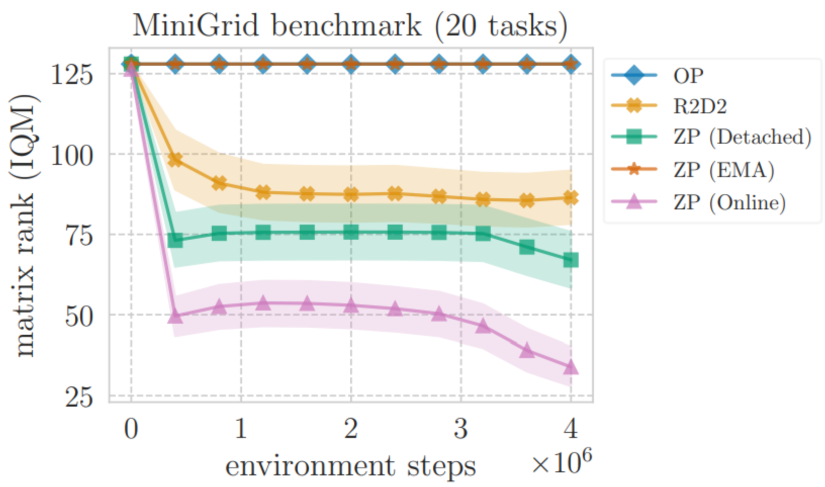
Hypothesis: End-to-end learning (e.g., TD-MPC) and phased learning (e.g., Dreamer) won't matter

Environment: MiniGrid (POMDP + Sparse reward)

Base algorithm: R2D2 (Distributed RNN)

End-to-end learning (OP, ZP) was way more effective than phased ones (RP+OP, RP+ZP)





Conclusion

Representation learning in RL boils down to latent self-prediction vs observation reconstruction.

End-to-end learning is just as effective as phased learning (TD-MPC vs Dreamer).

Choose either ϕ_L, ϕ_O over ϕ_{Q^*}.

ϕ_L vs ϕ_O depends on the task.

- Noisy/distracting tasks → Try ϕ_L
- Sparse-reward tasks → Try ϕ_O