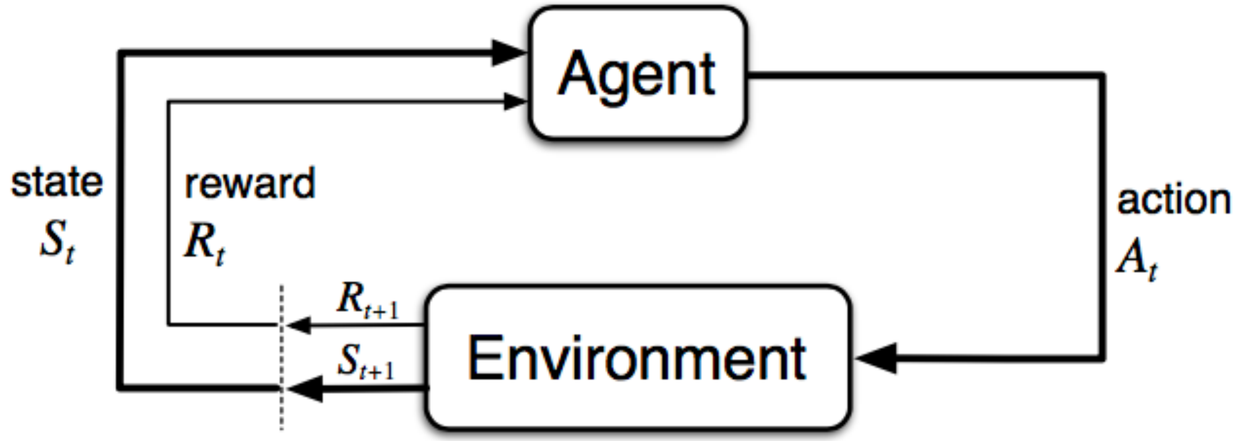
# Factored Adaptation for Non-stationary RL

Fan Feng, Biwei Huang, Kun Zhang, Sara Magliacane

# Reinforcement learning (RL)




## Environment (stationary):

**State-transition function:**

$$P_{ss'}^{a} = P\left(s' \mid s, a\right) = \mathbb{P}\left[S_{t+1} = s' \mid S_t = s, A_t = a\right] = \sum_{r \in \mathcal{R}} P\left(s', r \mid s, a\right)$$

**Reward function:**

$$R(s, a) = \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a\right] = \sum_{r \in \mathcal{R}} r \sum_{s' \in S} P\left(s', r \mid s, a\right)$$

## Agent:

**Policy function:**

$\pi(s) = a$ ***(Deterministic)*** $\pi(a \mid s) = \mathbb{P}_{\pi}[A = a \mid S = s]$ ***(Stochastic)***

***Objective: maximise the total reward***

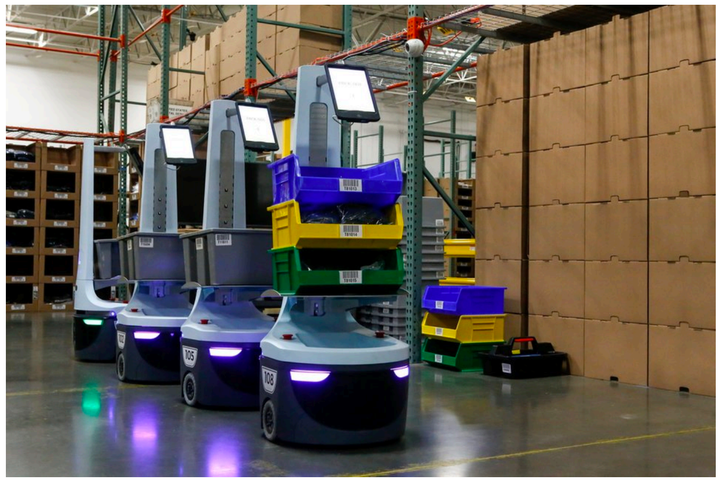
$$G_t = R_{t+1} + \gamma R_{t+2} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$



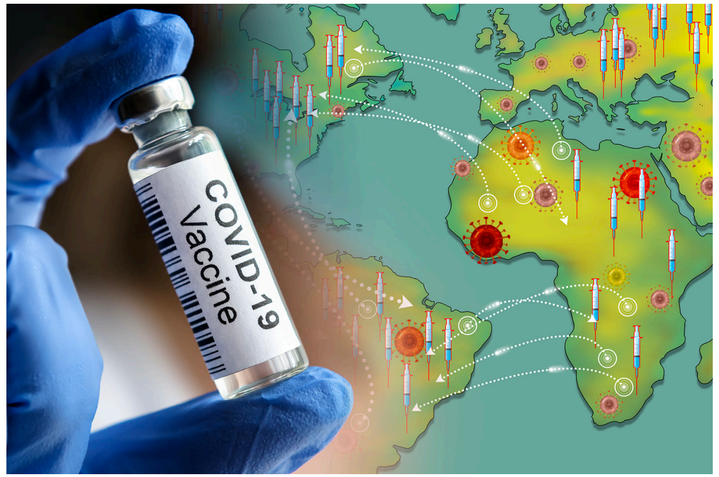
*(Image source: Sec. 3.1 Sutton & Barto (2017).)*

# Real-world Systems are usually non-stationary

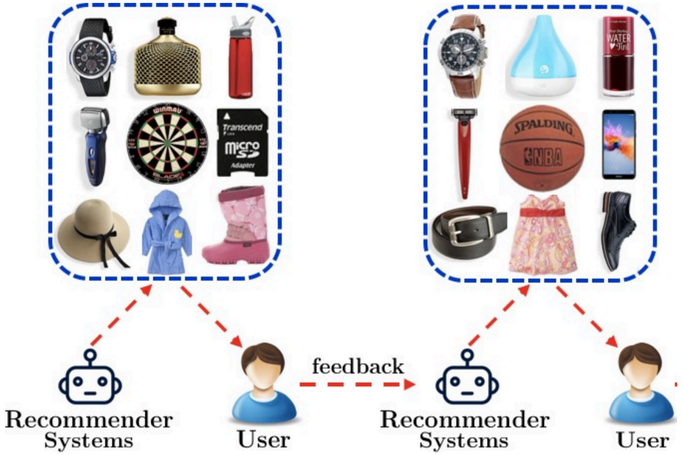
## Robot learning



## AI for Healthcare




## Recommendation systems

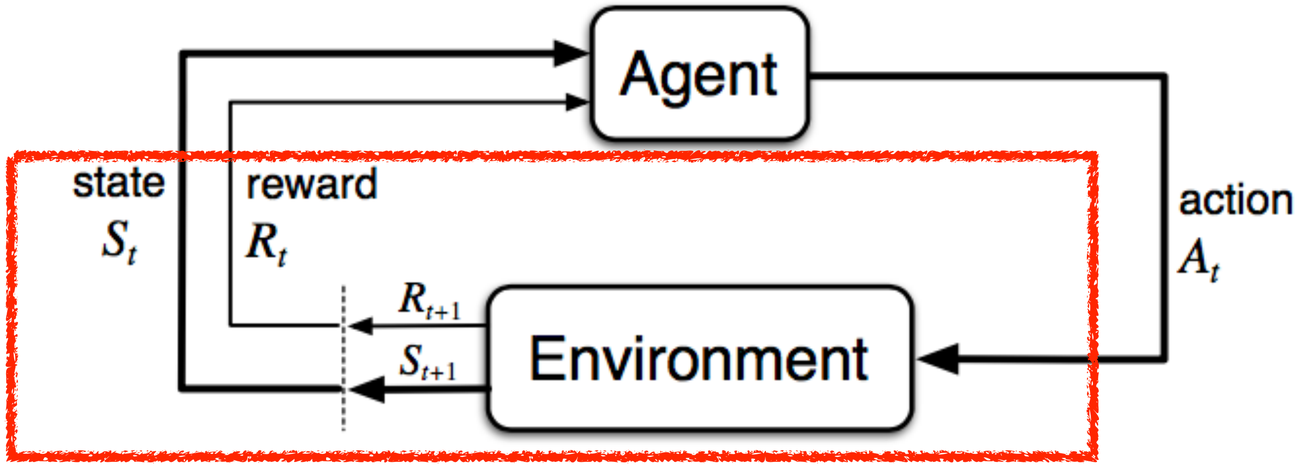


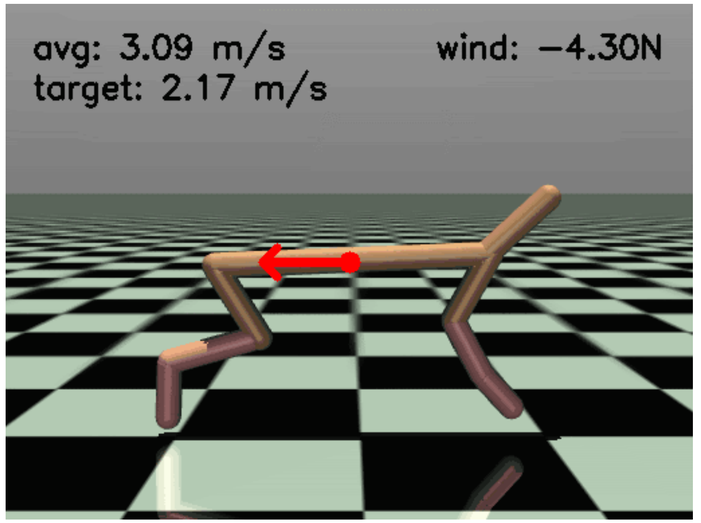

Nagabandi, A., Clavera, I., Liu, S., Fearing, R. S., Abbeel, P., Levine, S., & Finn, C. (2018). Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. ICLR 2018

Gottesman, O., Johansson, F., Komorowski, M., Faisal, A., Sontag, D., Doshi-Velez, F., & Celi, L. A. (2019). Guidelines for reinforcement learning in healthcare. *Nature medicine*

Zhao, X., Xia, L., Zhang, L., Ding, Z., Yin, D., & Tang, J. (2018). Deep reinforcement learning for page-wise recommendations. *ACM Conference on Recommender Systems 2018.*

# Non-stationary RL




**Time-varying environments**

**Stationary**

**State-transition function:**

$$P^a_{ss'} = P\left(s' \mid s, a\right) = \mathbb{P}\left[S_{t+1} = s' \mid S_t = s, A_t = a\right]$$

**Reward function:**

$$R\left(s, a\right) = \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a\right]$$

**Non-stationary**

**State-transition function:**

$$P^a_{ss'} = P\left(s' \mid s, a, \theta^s\right) = \mathbb{P}\left[S_{t+1} = s' \mid S_t = s, A_t = a, \theta^s_t\right]$$

**Reward function:**

$$R\left(s, a, \theta^r\right) = \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a, \theta^r_t\right]$$

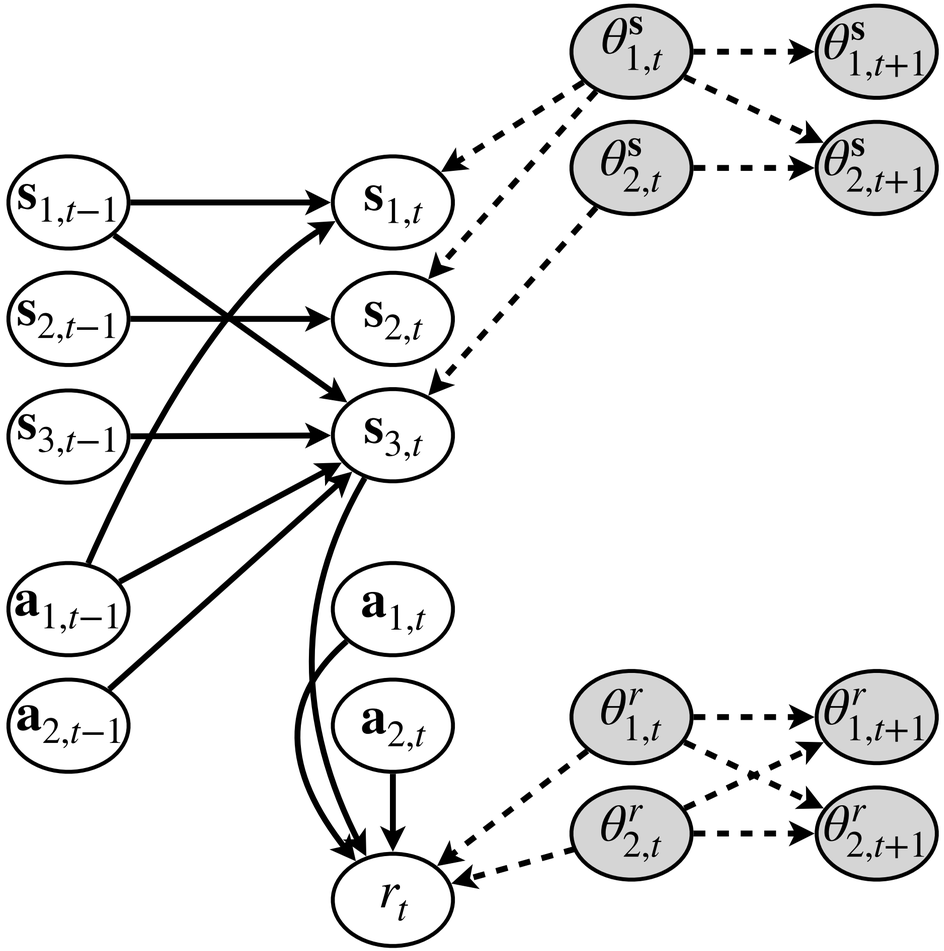
# How to capture the non-stationary and locate the exact changes explicitly?

# Factored Non-stationary MDP (FN-MDP)

- Bayesian Dynamic Network $\mathscr{G}$ over all dimensions of states, actions, rewards and changing factors;
- State transition: $\mathbb{P}_s\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{\theta}_t^s\right) = \prod_{i=1}^{d} \mathbb{P}_s\left(s_{i,t} \mid pa\left(s_{i,t}\right)\right)$
- Reward functions: $\mathscr{R}\left(\mathbf{s}_t, \mathbf{a}_t, \boldsymbol{\theta}_t^r\right) = \mathscr{R}\left(pa\left(r_t\right)\right)$
- Markov properties of change factors

$$\mathbb{P}_{\theta^s}\left(\boldsymbol{\theta}_t^s \mid \boldsymbol{\theta}_{t-1}^s\right) = \prod_{j=1}^{p} \mathbb{P}_{\theta^s}\left(\theta_{j,t}^s \mid pa\left(\theta_{ij,t}^s\right)\right)$$

$$\mathbb{P}_{\theta^r}\left(\boldsymbol{\theta}_t^r \mid \boldsymbol{\theta}_{t-1}^r\right) = \prod_{k=1}^{q} \mathbb{P}_{\theta^r}\left(\theta_{k,t}^r \mid pa\left(\theta_{k,t}^r\right)\right)$$



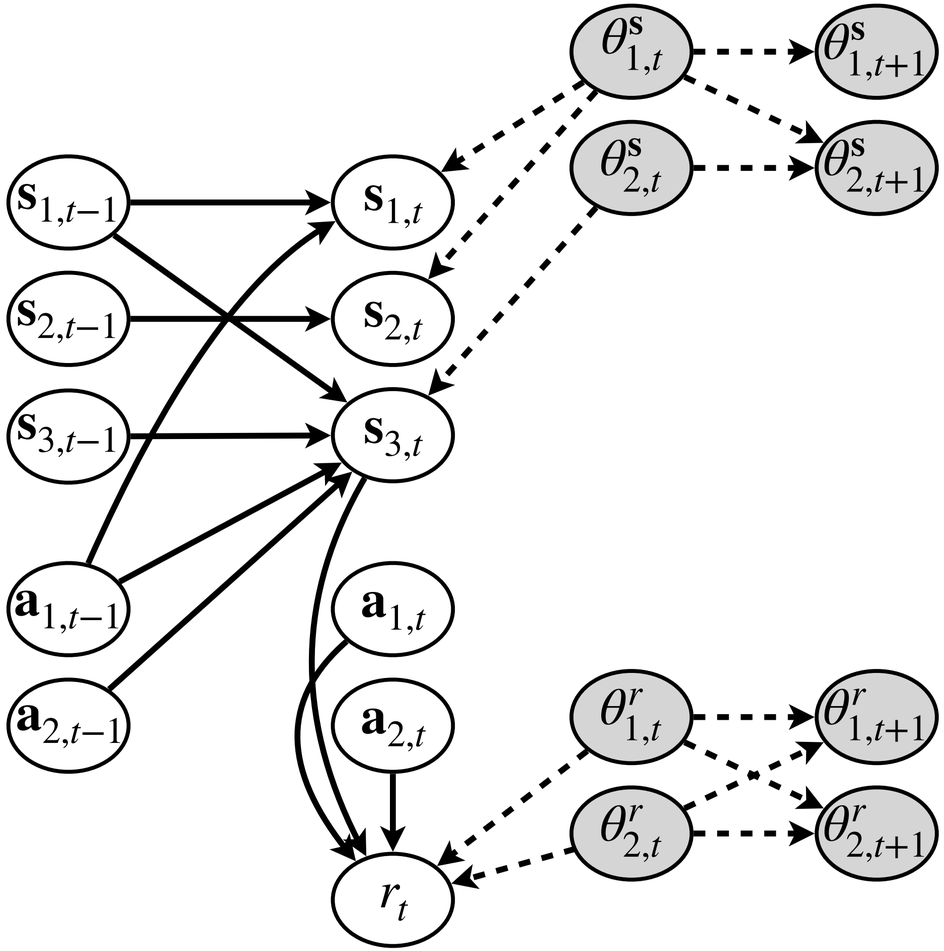
# Structural Equations to model FN-MDP

$$s_{i,t} = f_i\left(\boldsymbol{c}_i^{\boldsymbol{s}\to\boldsymbol{s}} \odot \boldsymbol{s}_{t-1}, \boldsymbol{c}_i^{\boldsymbol{a}\to\boldsymbol{s}} \odot \boldsymbol{a}_{t-1}, \boldsymbol{c}_i^{\boldsymbol{\theta}^{\boldsymbol{s}}\to\boldsymbol{s}} \odot \boldsymbol{\theta}_t^{\boldsymbol{s}}, \epsilon_{i,t}^{s}\right)$$

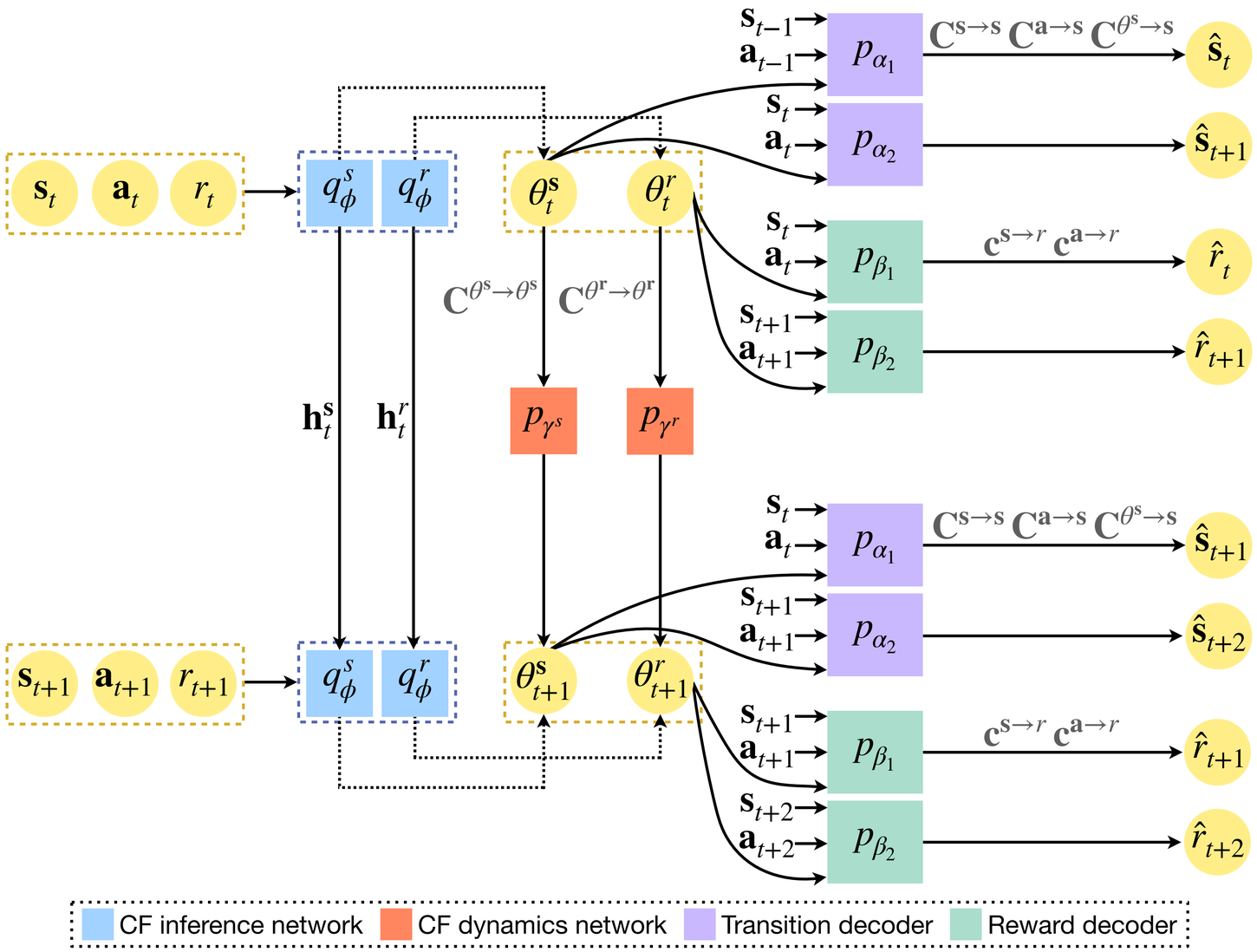
$$r_t = h\left(\boldsymbol{c}^{\boldsymbol{s}\to r} \odot \boldsymbol{s}_t, \boldsymbol{c}^{\boldsymbol{a}\to r} \odot \boldsymbol{a}_t, \boldsymbol{\theta}_t^{r}, \epsilon_t^{r}\right)$$

$$\theta_{j,t}^{s} = g^{\boldsymbol{s}}\left(\boldsymbol{c}_j^{\boldsymbol{\theta}^{s}\to\boldsymbol{\theta}^{\boldsymbol{s}}} \odot \boldsymbol{\theta}_{t-1}^{\boldsymbol{s}}, \epsilon_t^{\theta^{s}}\right)$$

$$\theta_{k,t}^{r} = g^{r}\left(\boldsymbol{c}_k^{\boldsymbol{\theta}^{r}\to\boldsymbol{\theta}^{r}} \odot \boldsymbol{\theta}_{t-1}^{r}, \epsilon_t^{\theta^{r}}\right)$$



# Learn the FN-MDP

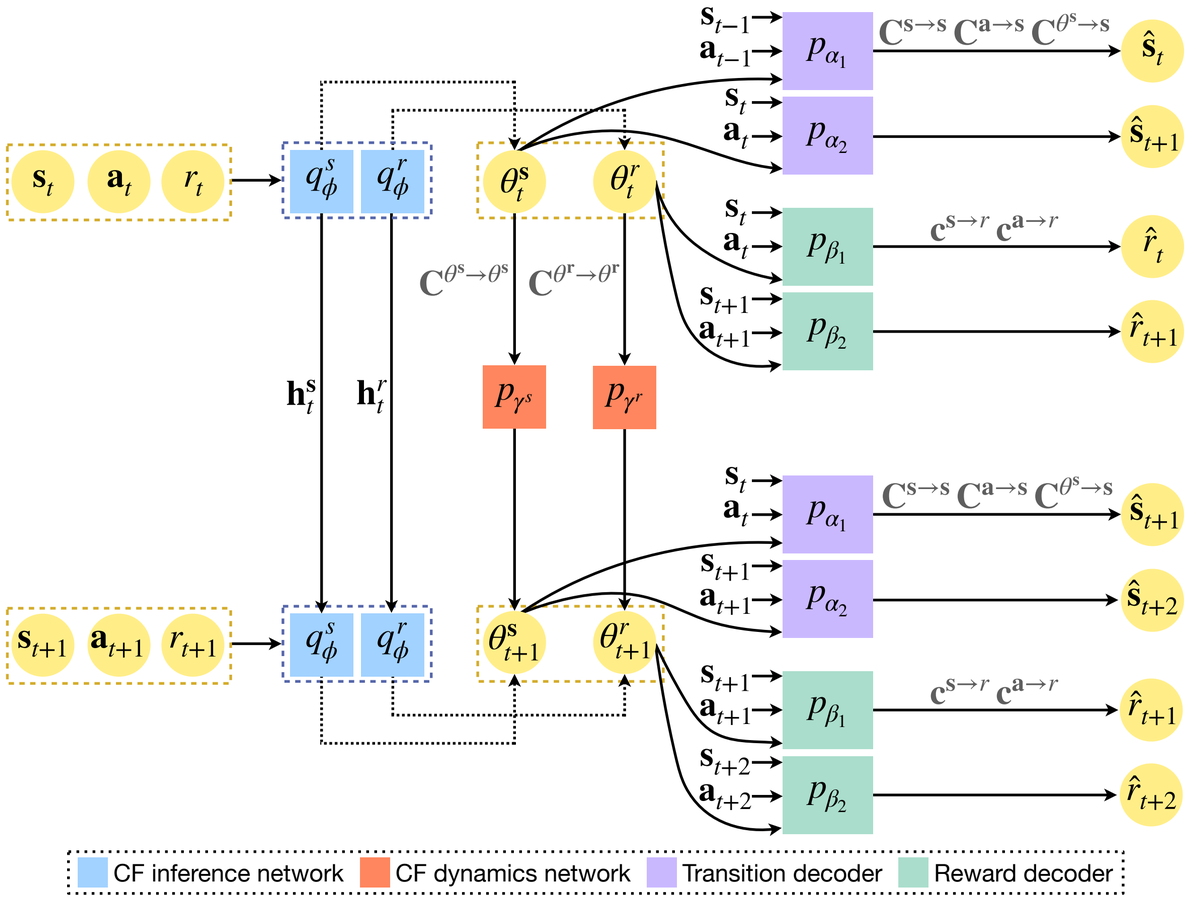


# Learn the FN-MDP

## CF inference network

- Infer the changes on dynamics:
  $q_{\phi^s}\left(\boldsymbol{\theta}_t^s \mid \boldsymbol{s}_t, \boldsymbol{a}_t\right) = q_{\phi^s}\left(\boldsymbol{\theta}_t^s \mid \boldsymbol{s}_t, \boldsymbol{a}_t, r_t, \boldsymbol{h}_{t-1}^s\right)$

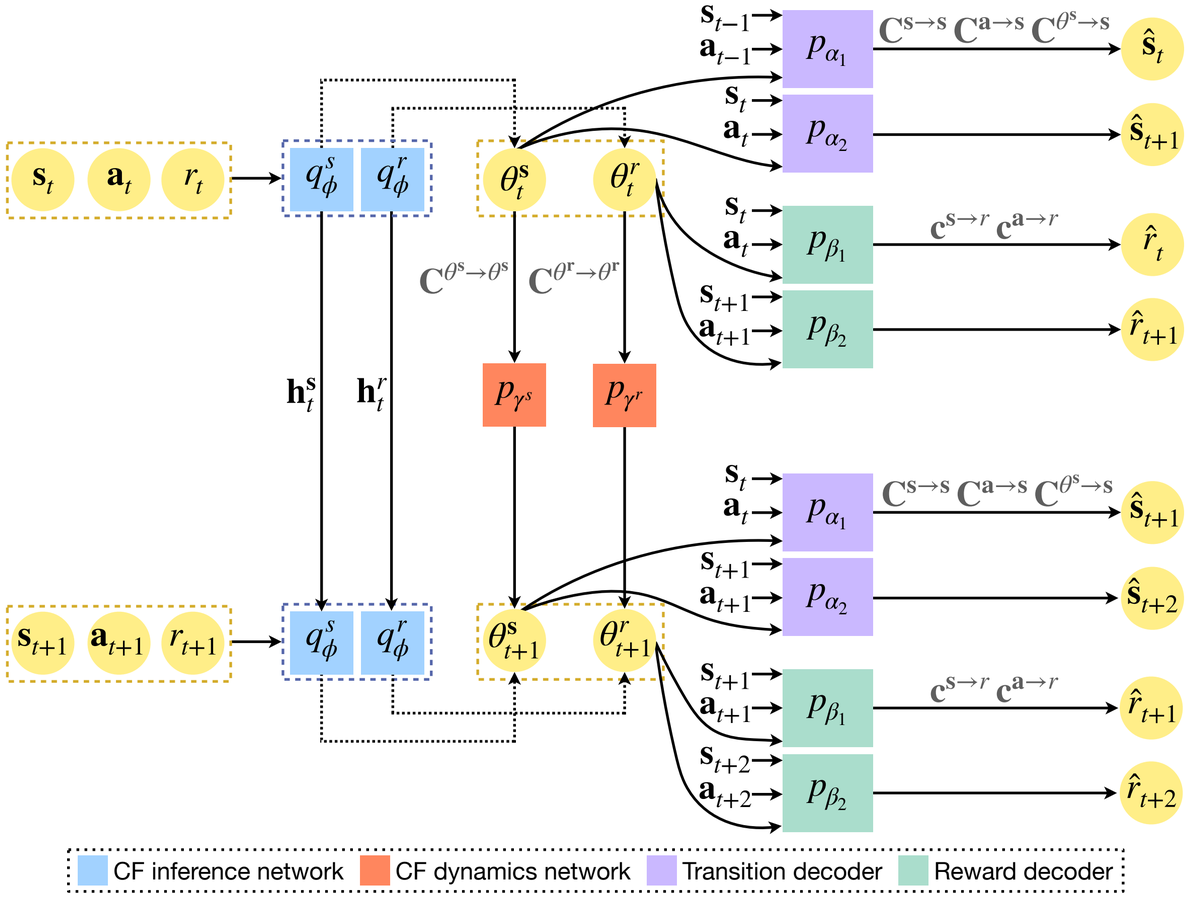
- Infer the changes on rewards:
  $q_{\phi^r}\left(\boldsymbol{\theta}_t^r \mid \boldsymbol{s}_t, \boldsymbol{a}_t, r_t\right) = q_{\phi^r}\left(\boldsymbol{\theta}_t^r \mid \boldsymbol{s}_t, \boldsymbol{a}_t, r_t, \boldsymbol{h}_{t-1}^r\right)$



# Learn the FN-MDP

## CF dynamics network

- Learn the Markov properties of change factors
  - $p_{\gamma^s}\left(\boldsymbol{\theta}^{\mathbf{s}}_{t+1} \mid \boldsymbol{\theta}^{\mathbf{s}}_{t}, \boldsymbol{C}^{\boldsymbol{\theta}^s \to \boldsymbol{\theta}^s}\right)$
  - $p_{\gamma^r}\left(\boldsymbol{\theta}^{r}_{t+1} \mid \boldsymbol{\theta}^{r}_{t}, \boldsymbol{C}^{\boldsymbol{\theta}^r + \boldsymbol{\theta}^r}\right)$



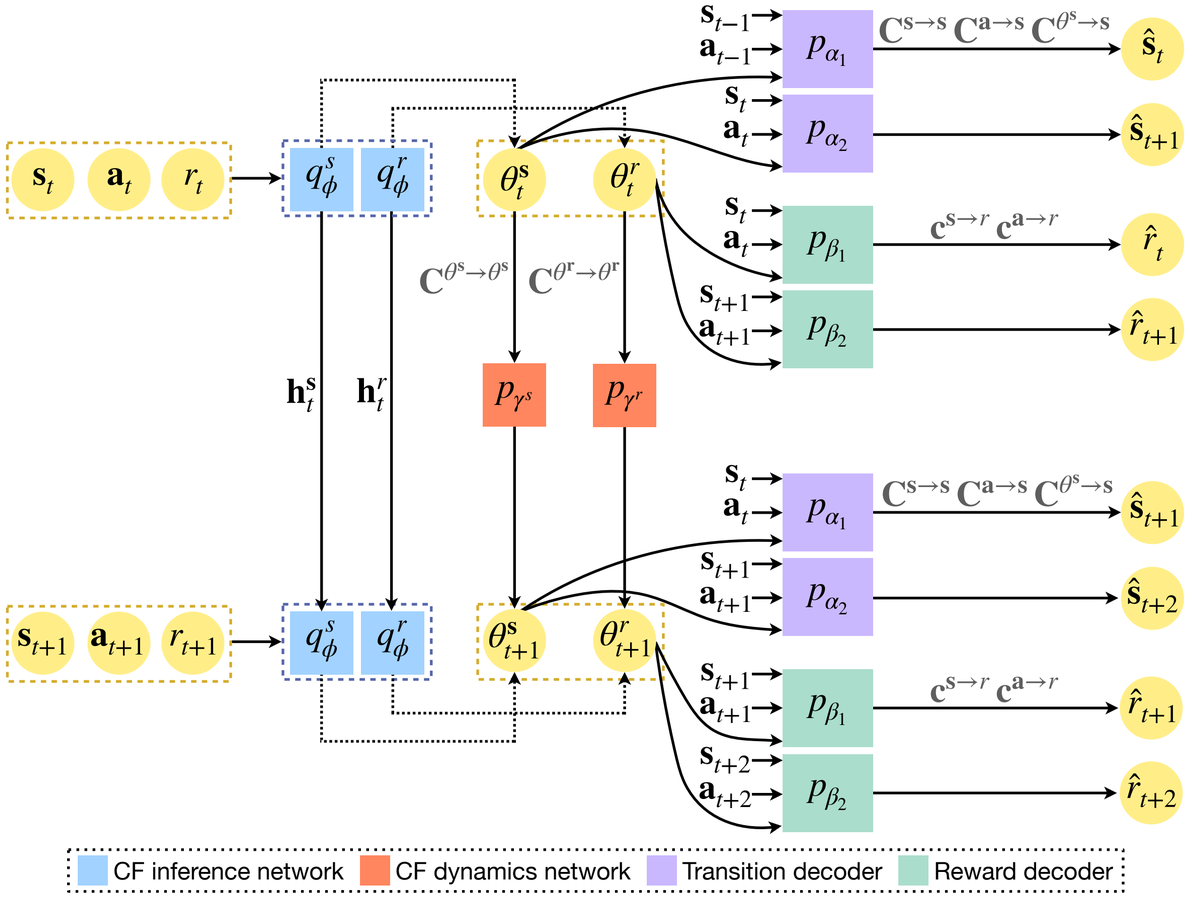
# Learn the FN-MDP

## Transition decoders

- Reconstruct the current states and predict the future states

$$\mathscr{L}_{\textbf{rec-dyn}} = \sum_{t=1}^{T-2} \mathbb{E}_{\theta_t^s \sim q_\phi} \log p_{\alpha_1}\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{\theta}_t^s; \boldsymbol{C}^{\cdot \to \boldsymbol{s}}\right)$$

$$\mathscr{L}_{\textbf{pred-dyn}} = \sum_{t=1}^{T-2} \mathbb{E}_{\theta_t^s \sim q_\phi} \log p_{\alpha_2}\left(\boldsymbol{s}_{t+1} \mid \boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{\theta}_t^s\right)$$



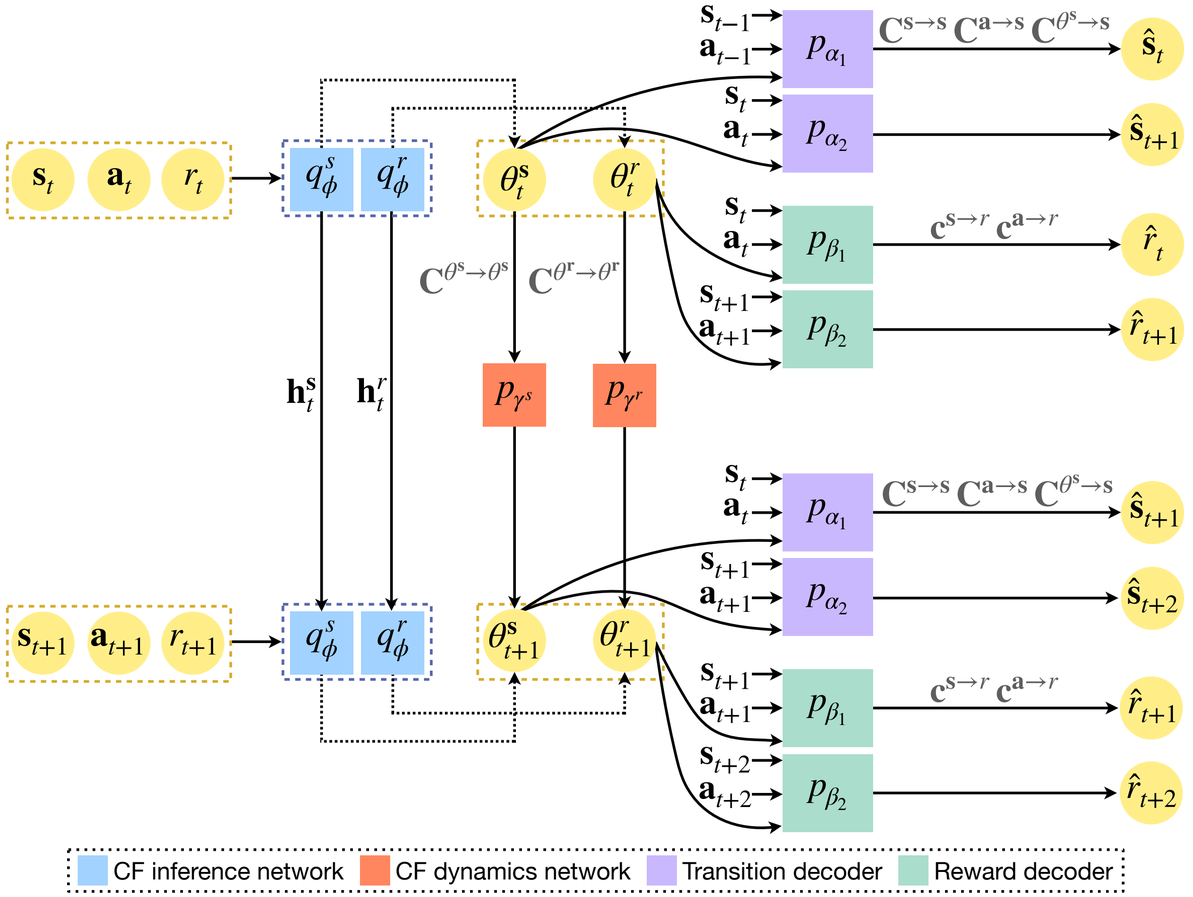
# Learn the FN-MDP

## Reward decoders

- Reconstruct the current rewards and predict the future rewards

$$\mathscr{L}_{\textbf{rec-rw}} = \sum_{t=1}^{T-2} \mathbb{E}_{\theta_t^r \sim q_\phi} \log p_{\beta_1} \left( r_t \mid \boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{\theta}_t^r; \boldsymbol{c}^{s \to r}, \boldsymbol{c}^{a \to r} \right)$$

$$\mathscr{L}_{\textbf{pred-rw}} = \sum_{t=1}^{T-2} \mathbb{E}_{\theta_t^r \sim q_\phi} \log p_{\beta_2} \left( r_{t+1} \mid \boldsymbol{s}_{t+1}, \boldsymbol{a}_{t+1}, \boldsymbol{\theta}_t^r \right)$$



# Policy Learning with FN-MDP

- With the learned FN-MDP, we can identify the minimal and sufficient sets of states and change factors for efficient policy learning

$$\tau_\psi\left(\boldsymbol{a}_t \mid \boldsymbol{s}_t^{\min}, q_\phi\left(\boldsymbol{\theta}_t^{\min} \mid \boldsymbol{\tau}_{0:t}\right)\right)$$

- To identify the minimal and sufficient sets:

$$s_{i,t} \in \boldsymbol{s}^{\min} \iff s_{i,t} \rightarrow \ldots \rightarrow r_{t+\tau} \text{ for } \tau \geq 1$$

$$\theta_i \in \boldsymbol{\theta}^{\min} \iff \theta_i \rightarrow \ldots \rightarrow r_{t+\tau} \text{ for } \tau \geq 1$$

# Evaluation
## Objectives

- Can the framework improve the learning performances under non-stationary RL?
- Can the framework learn a more compact latent representation against other baselines?
- Can the framework deal with a more general set of non-stationarity in RL?

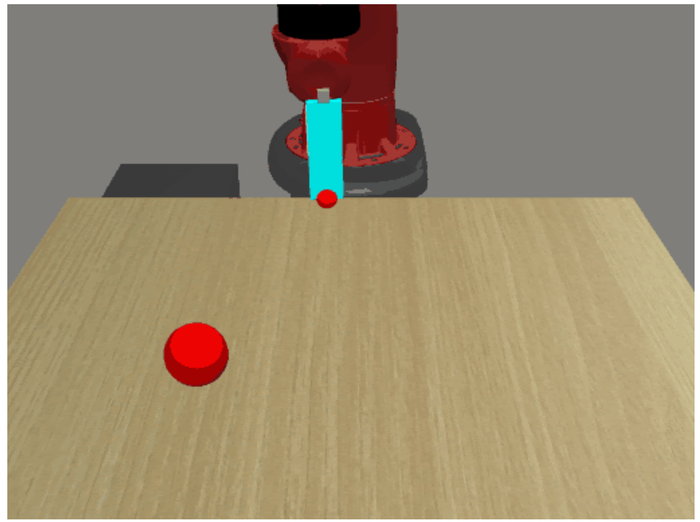
# Evaluation
## Baselines

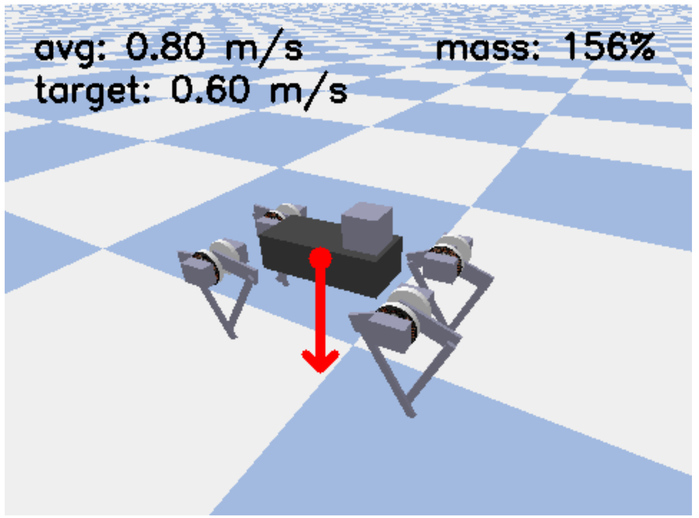
- Tracking non-stationarity via inferring the latent parameters via meta-learning on a set of tasks: TRIO (Poiani et al., IJCAI'21), VariBAD (Zintgraf et al., ICLR'21);

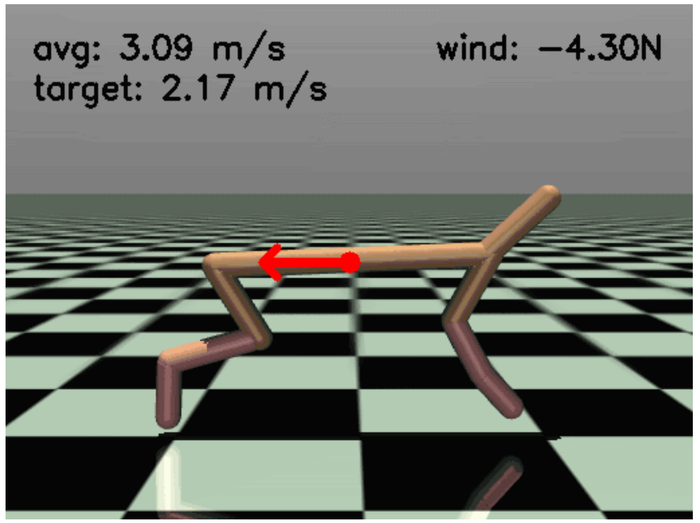
- Learn the latent parameters to capture the non-stationary components: LILAC (Xie et al., ICML'21), ZeUS (Sodhani et al., ArXiv'22)

# Evaluation

## Benchmarks






| | Sawyer | Mini-taur | Half-Cheetah |
|---|---|---|---|
| **Dyn changes** | | **Gravity** | **Wind speed** |
| **Reward changes** | **Target location** | **Target speed** | **Target speed** |

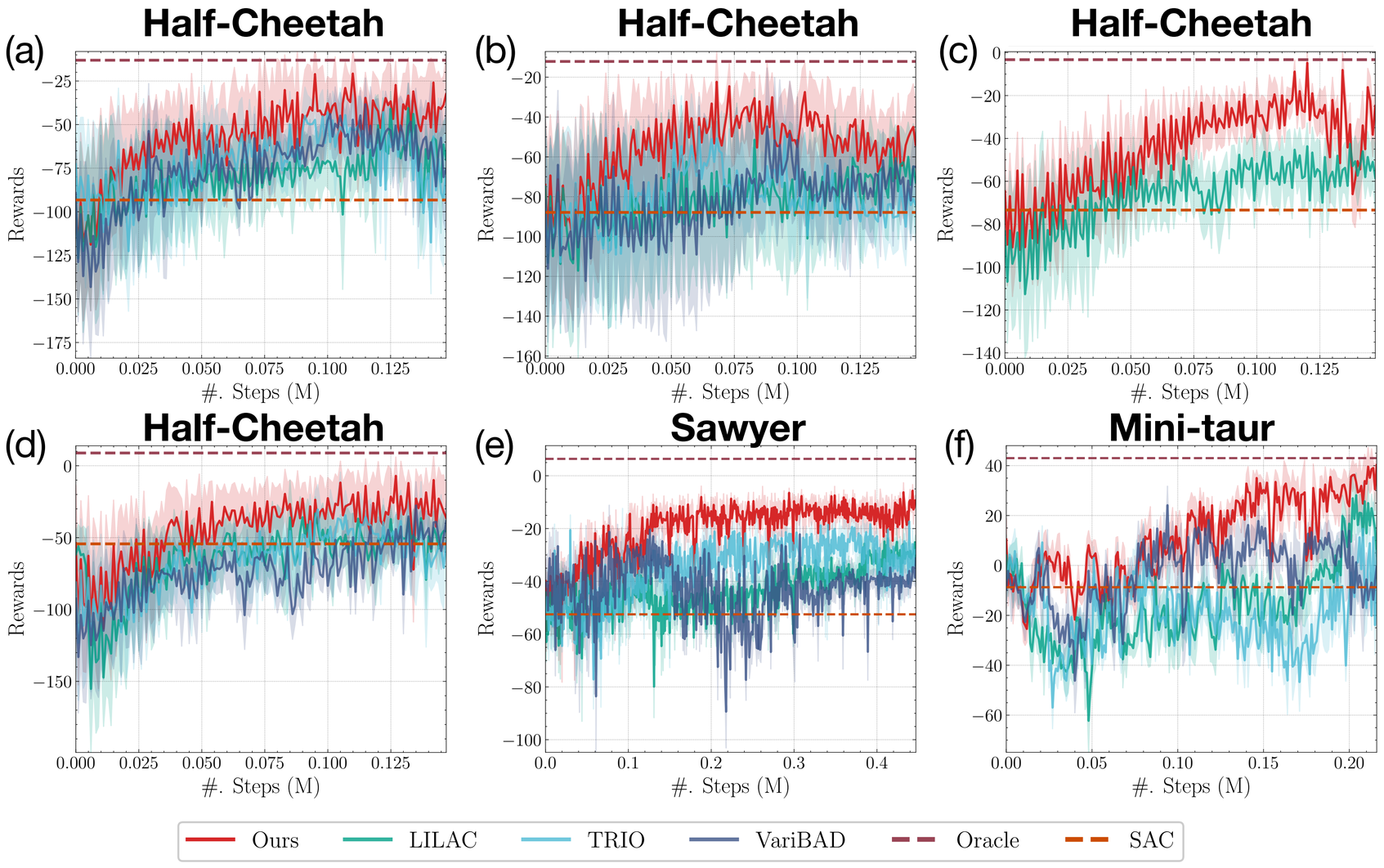
# Evaluation

## A more general non-stationary setting

| | LILAC | TRIO | VariBAD | ZeUS | Ours |
|---|---|---|---|---|---|
| Continuous changes | ✅ | | | | ✅ |
| Discrete changes | ✅ | ✅ | ✅ | ✅ | ✅ |
| Multiple changes | | | | ✅ | ✅ |
| Learning from pixels | | | | ✅ | ✅ |

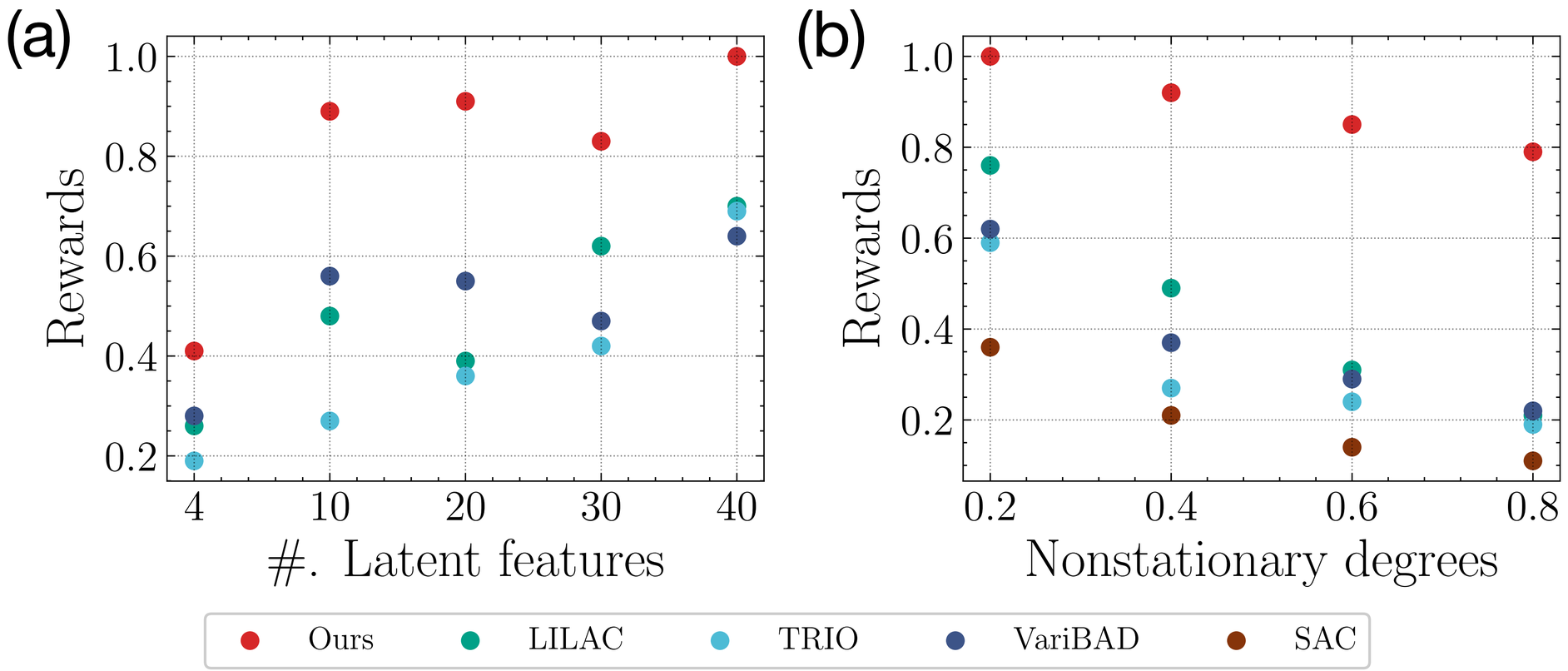
# Evaluation

## Learning curves



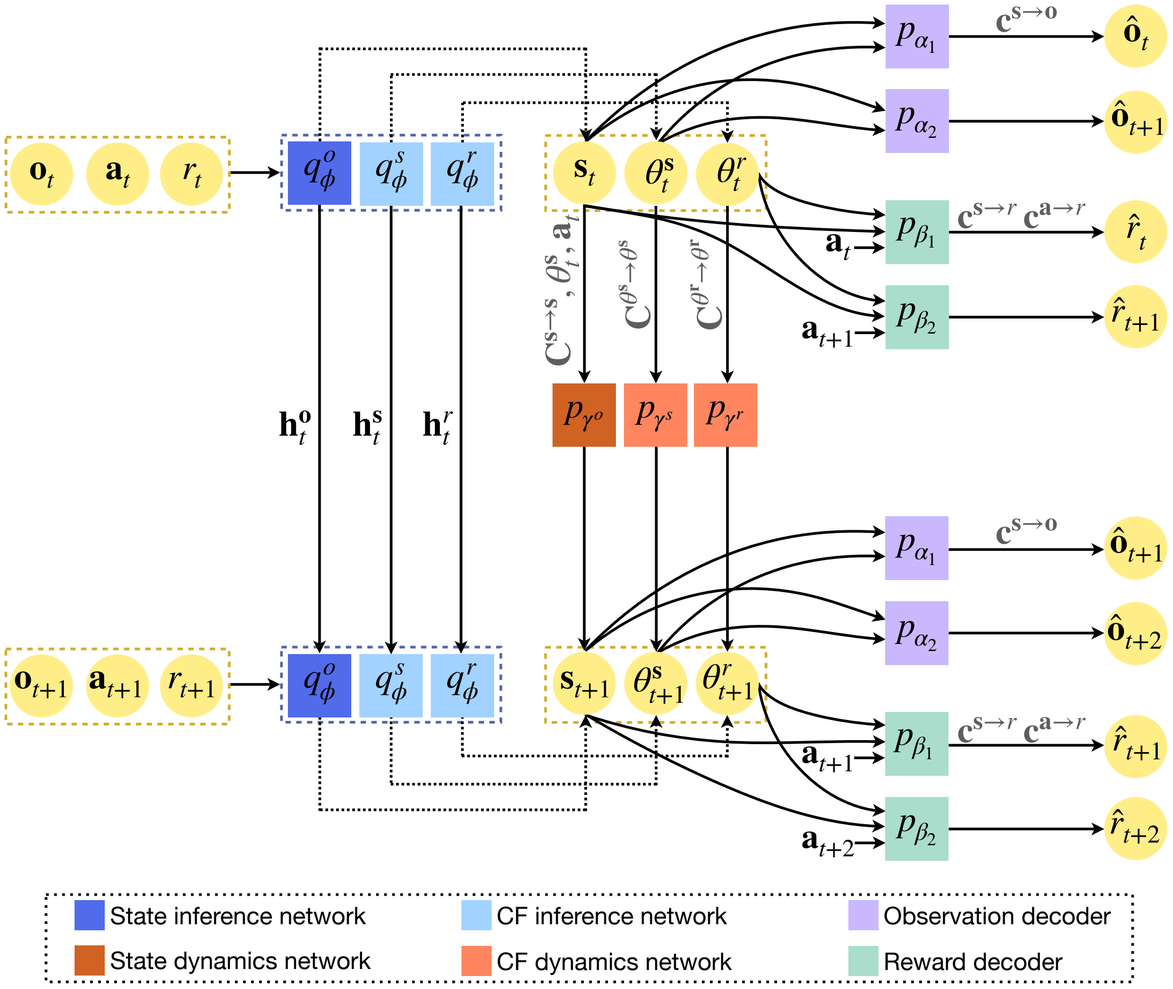
# Evaluation

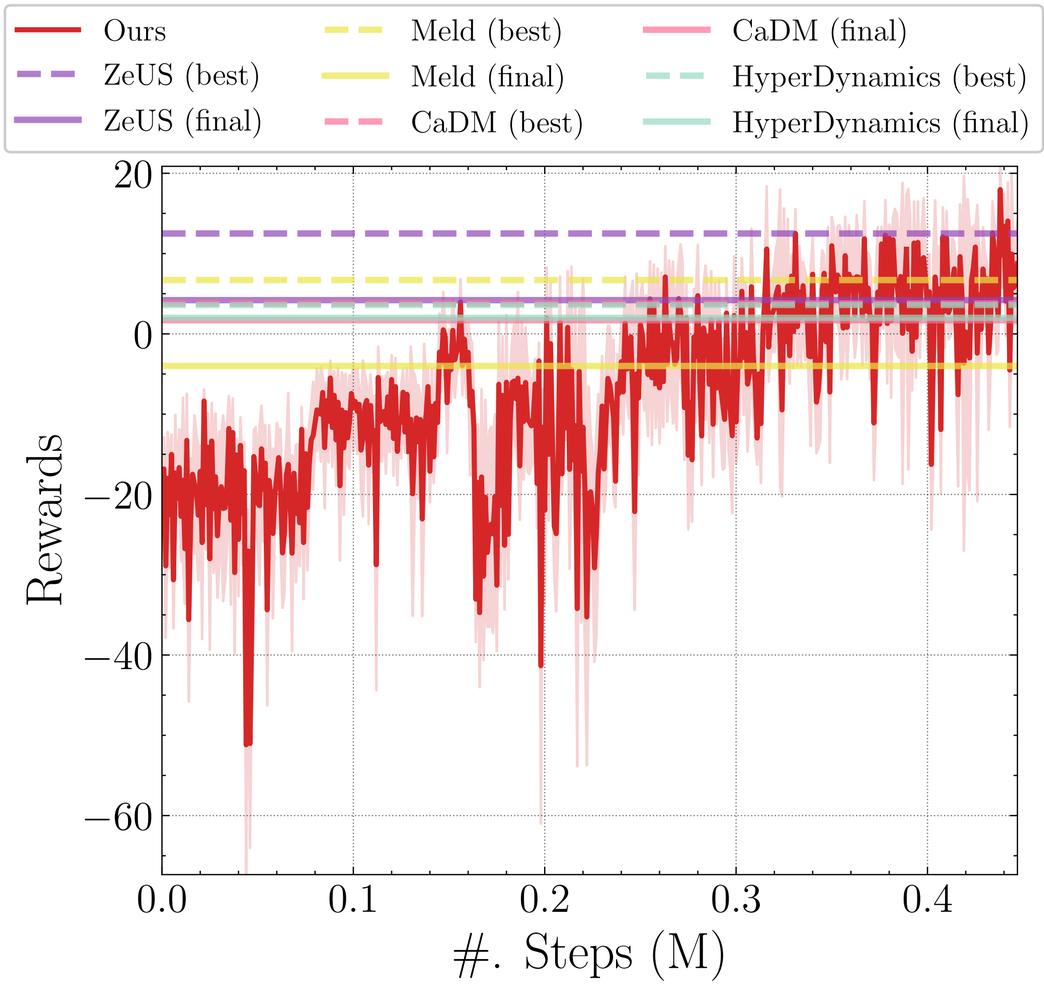
## Compactness and robustness



# Evaluation

## Learning from raw pixels



# Conclusions

- We propose a general non-stationary RL approach that learns a factored representation for adapting to changes in dynamics and reward functions.
- We formalise a unified framework that can handle different non-stationary settings, including discrete and continuous changes;
- Results outperform the state of the art on rewards, compactness of the latent space representation and robustness to varying degrees of non-stationarity.

# Thanks!
# Q&A