Stochastic Latent Actor-Critic: Deep Reinforcement Learning with a Latent Variable Model (SLAC)

백승언

10 Apr, 2023

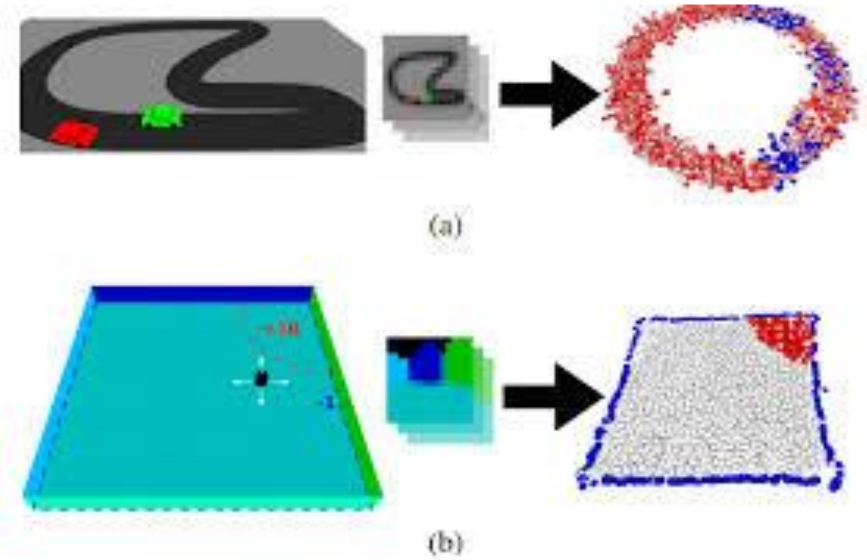
Contents

- **Introduction**
  - Challenge of Representation Learning in Reinforcement Learning
- **Stochastic Latent Actor-Critic(SLAC)**
  - SLAC
- **Experiment results**

Introduction

Challenge of Representation Learning in Reinforcement Learning

- **In visual control problems, unifying the observation representation and task-specific information into single end-to-end training is difficult**
  - Previous model-based methods are computationally expensive
    - Because they learn the model and policy separately(PlaNet, Dreamer, …)
  - Conventional model-free methods are confused
    - Because they learn the model and policy using reward solely(TD3, SAC, D4PG, …)
  - A number of prior works have explored the use of various approaches in RL to learn such representations
    - Learning auxiliary tasks
    - Data augmentation: DrQ
    - Latent dynamics: Flare, DeepMDP
    - Self-supervised learning: Plan2Explore, CURL



Representation learning in RL

Stochastic Latent Actor-Critic (SLAC)

https://arxiv.org/pdf/1907.00953.pdf

SLAC (I)

Overview of the SLAC

- Unlike the existing end-to-end RL to learn directly from image observation, SLAC use of explicit representation learning with RL for sample efficiency and potential capability to increase the complexity of tasks

- They simultaneously learned the observation representation and task-specific policy with joint objective modeling(substituted with lower bound)
  - $\log p(\mathbf{x}_{1:\tau+1}, \mathcal{O}_{\tau+1:T} | \mathbf{a}_{1:\tau})$

- They proposed a novel approach that integrates learning stochastic sequential models and RL into a single method, performing RL in the model's learned latent space
  - $\mathbf{z}_1^1 \sim p(\mathbf{z}_1^1), \mathbf{z}_1^2 \sim p_\psi(\mathbf{z}_1^2 | \mathbf{z}_1^1),\ \ \mathbf{z}_{t+1}^1 \sim p_\psi(\mathbf{z}_{t+1}^1 | \mathbf{z}_t^2, \mathbf{a}_t), \mathbf{z}_{t+1}^2 \sim p_\psi\left(\mathbf{z}_{t+1}^2 | \mathbf{z}_{t+1}^1, \mathbf{z}_t^2, \mathrm{a}_t\right),\ \ \mathbf{x}_t \sim p_\psi(\mathbf{x}_t | \mathbf{z}_t^1, \mathbf{z}_t^2)$
  - $Q(\mathbf{z}_t, \mathbf{a}_t) = r(\mathbf{z}_t, \mathbf{a}_t) + \mathbb{E}_{\mathbf{z}_{t+1}}[V(\mathbf{z}_{t+1})],\ \ V(\mathbf{z}_t) = \log \int \exp\left(Q(\mathbf{z}_t, \mathrm{a}_t)\right) d\mathbf{a}_t$

- Evaluation demonstrates that SLAC outperforms both model-free and model-based alternatives in terms of final performance and sample efficiency

SLAC (II)

- **Latent variable model**
  - To learn representations for RL, the authors used latent variable models trained with amortized variational inference.
  - To learn such a model, they utilized the evidence lower bound for the log-likelihood of entire generative process($p(x) = \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})dz$)
    - $\log p(\mathbf{x}) \geq \mathbb{E}_{\mathbf{z}\sim q}\left[\log p(\mathbf{x}|\mathbf{z})\right] - \mathrm{D_{KL}}(q(\mathbf{z}|\mathbf{x}) \,||\, p(\mathbf{z}))$

- **Sequential latent variable model**
  - They proposed a fully stochastic sequential latent variable model to consider a POMDP
    - They note that $\mathbf{x}_t$ does not provide all necessary information to infer $\mathbf{z}_t$, and prior observations must be taken into account during inference
    - $\log p(\mathbf{x}_{1:\tau+1}|\mathbf{a}_{1:\tau}) \geq \mathbb{E}_{\mathbf{z}_{1:\tau+1}\sim q}\left[\Sigma_{t=0}^{\tau} \log p(\mathbf{x}_{t+1}|\mathbf{z}_{t+1}) - \mathrm{D_{KL}}\left(q(\mathbf{z}_{t+1}|\mathbf{x}_{t+1}, \mathbf{z}_t, \mathbf{a}_t) \,||\, p(\mathbf{z}_{t+1}|\mathbf{z}_t, \mathbf{a}_t)\right)\right]$
  - They used the generative model given by $p_\psi(\mathbf{z_1}), p_\psi(\mathbf{z}_{\mathrm{t}+1}|\mathbf{z}_t, \mathbf{a}_t)$ and $p_\psi(\mathbf{x}_t|\mathbf{z}_t)$, and inference model given by $q_\psi(\mathbf{z}_1|\mathbf{x}_1)$ and $q_\psi(\mathbf{z}_{t+1}|\mathbf{x}_{t+1}, \mathbf{z}_t, \mathbf{a}_t)$.
    - These distributions are diagonal Gaussian, where the mean and variance are given by outputs of NN.
    - Objectives: $J_M(\psi) = \mathbb{E}_{\mathbf{z}_{1:\tau+1}\sim q_\psi}\left[\Sigma_{t=0}^{\tau} - \log p_\psi(\mathbf{x}_{t+1}|\mathbf{z}_{t+1}) + \mathrm{D_{KL}}\left(q_\psi(\mathbf{z}_{t+1}|\mathbf{x}_{t+1}, \mathbf{z}_t, \mathbf{a}_t) \,||\, p_\psi(\mathbf{z}_{t+1}|\mathbf{z}_t, \mathbf{a}_t)\right)\right]$

SLAC (III)

- **Overall objective**
  - They jointly model the observation and learn maximum entropy polices by maximizing the marginal likelihood $p(\mathbf{x}_{1:\tau+1}, \mathcal{O}_{\tau+1:T} | \mathbf{a}_{1:\tau})$
    - $\log p(\mathbf{x}_{1:\tau+1}, \mathcal{O}_{\tau+1:T} | \mathbf{a}_{1:\tau})$
  - Instead of optimizing above objective, which is intractable, they optimized a tractable lower bound
    - $\log p(\mathbf{x}_{1:\tau+1}, \mathcal{O}_{\tau+1:T} | \mathbf{a}_{1:\tau}) \geq \mathbb{E}_{(\mathbf{z}_{1:T}, \mathbf{a}_{\tau+1:T}) \sim q}[\log p(\mathbf{x}_{1:\tau+1}, \mathcal{O}_{\tau+1:T}, \mathbf{z}_{1:T}, \mathbf{a}_{\tau+1:T} | \mathbf{a}_{1:\tau}) - \log q(\mathbf{z}_{1:T}, \boldsymbol{a}_{\tau+1:T} | \mathbf{x}_{1:\tau+1}, \boldsymbol{a}_{1:\tau})]$
      $= \mathbb{E}_{(\mathbf{z}_{1:T}, \mathbf{a}_{\tau+1:T}) \sim q}\Big[\Sigma_{t=0}^{\tau}\big(\log p(\mathbf{x}_{t+1} | \mathbf{z}_{t+1}) - \mathrm{D}_{\mathrm{KL}}\big(q(\mathbf{z}_{\mathrm{t}+1} | \mathbf{x}_{\mathrm{t}+1}, \mathbf{z}_{\mathrm{t}}, \boldsymbol{a}_{\mathrm{t}}) \,||\, p(\mathbf{z}_{\mathrm{t}+1} | \mathbf{z}_{\mathrm{t}}, \mathbf{a}_{\mathrm{t}})\big)\big)$
      $+ \Sigma_{t=\tau+1}^{T}(r(\mathbf{z}_t, \mathbf{a}_t) + \log p(\mathbf{a}_t) - \log \pi(\mathbf{a}_t | \mathbf{x}_{1:t}, \mathbf{a}_{1:t-1}))\Big], p(\mathcal{O}_t = 1 | \mathbf{z}_t, \mathbf{a}_t) = \exp(r(\mathbf{z}_t, \mathbf{a}_t))$

- **Actor-Critic model**
  - The value and policy of RL are parameterized with θ, ϕ
    - To learn value and policy for RL, the authors used maximum entropy RL objectives
    - $J_Q(\theta) = \mathbb{E}_{\mathbf{z}_{1:\tau+1} \sim q_\psi}\left[\frac{1}{2}\left(Q_\theta(\mathbf{z}_{\boldsymbol{\tau}}, \mathbf{a}_{\boldsymbol{\tau}}) - \left(r_\tau + \gamma V_{\bar{\theta}}(\mathbf{z}_{\tau+1})\right)\right)^2\right], V_\theta(\mathbf{z}_{\tau+1}) = \mathbb{E}_{\mathbf{a}_{\tau+1} \sim \pi_\phi}[Q_\theta(\mathbf{z}_{\tau+1}, \mathbf{a}_{\tau+1}) - \alpha \log \pi_\phi(\mathbf{a}_{\tau+1} | \mathbf{x}_{\mathbf{1}:\boldsymbol{\tau}+\mathbf{1}}, \boldsymbol{a}_{1:\tau})]$
    - $J_\pi(\phi) = \mathbb{E}_{\mathbf{z}_{1:\tau+1} \sim q_\psi}\left[\mathbb{E}_{\mathbf{a}_{\tau+1} \sim \pi_\phi}[\alpha \log \pi_\phi(\mathbf{a}_{\tau+1} | \mathbf{x}_{1:\tau+1}, \mathrm{a}_{1:\tau}) - Q_\theta(\mathbf{z}_{\tau+1}, \mathbf{a}_{\tau+1})]\right]$

SLAC (IV)

- **Pseudo code**
  - Environment and initial parameter initialization corresponds with the main and target networks
  - In the control step, action is inferred from the policy(not conditioned on the latent state)
    - Next, the environment proceeds one step using action
    - Also, a transition is placed in the replay buffer $\mathcal{D}$
  - In the update step, transitions are sampled from the replay buffer $\mathcal{D}$
    - Next, latent variable z is sampled from the encoder
    - And then, Critic loss and ELBO objectives of parameters(ψ, θ, ϕ) are computed and backpropagated with SGD
    - Finally, the Target network update with temperature param ν

Algorithm 1 Stochastic Latent Actor-Critic (SLAC)

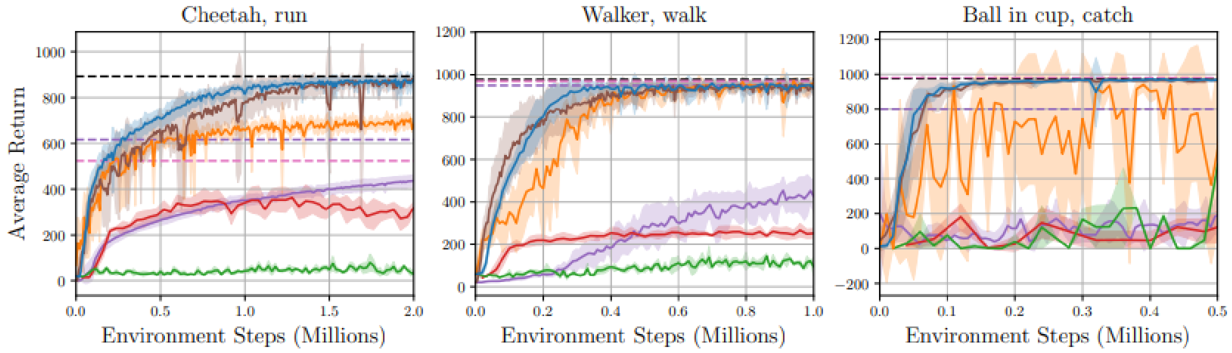
Require: Environment E and initial parameters $\psi, \phi, \theta_1, \theta_2$ for the model, actor, and critics.
  $\mathbf{x}_1 \sim E_{\text{reset}}()$
  $\mathcal{D} \leftarrow (\mathbf{x}_1)$
  for each iteration **do**
    for each environment step **do**
      $\mathbf{a}_t \sim \pi_\phi(\mathbf{a}_t | \mathbf{x}_{1:t}, \mathbf{a}_{1:t-1})$
      $r_t, \mathbf{x}_{t+1} \sim E_{\text{step}}(\mathbf{a}_t)$
      $\mathcal{D} \leftarrow \mathcal{D} \cup (\mathbf{a}_t, r_t, \mathbf{x}_{t+1})$
    for each gradient step **do**
      $\mathbf{x}_{1:\tau+1}, \mathbf{a}_{1:\tau}, r_\tau \sim \mathcal{D}$
      $\mathbf{z}_{1:\tau+1} \sim q_\psi(\mathbf{z}_{1:\tau+1} | \mathbf{x}_{1:\tau+1}, \mathbf{a}_{1:\tau})$
      $\psi \leftarrow \psi - \lambda_M \nabla_\psi J_M(\psi)$
      $\theta_i \leftarrow \theta_i - \lambda_Q \nabla_{\theta_i} J_Q(\theta_i)$ for $i \in \{1, 2\}$
      $\phi \leftarrow \phi - \lambda_\pi \nabla_\phi J_\pi(\phi)$
      $\bar{\theta}_i \leftarrow \nu \theta_i + (1 - \nu) \bar{\theta}_i$ for $i \in \{1, 2\}$

Pseudo code of SLAC

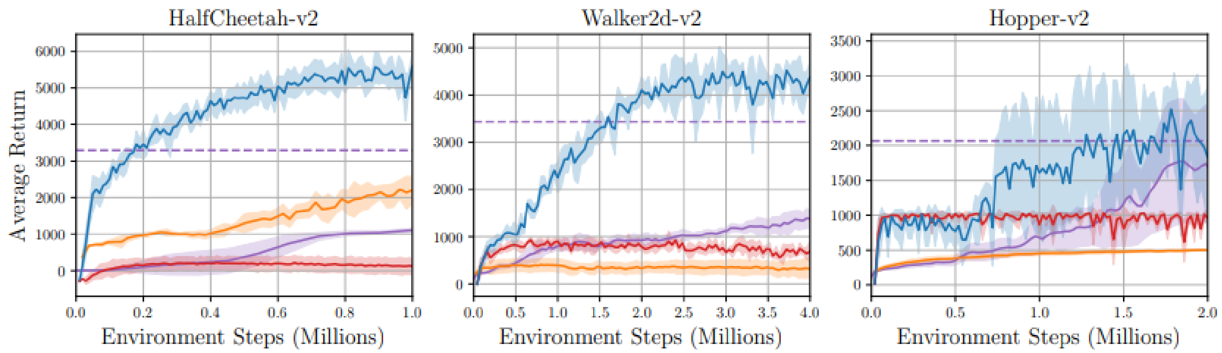
Experiment Results

Experiment results (I)

- **Comparison with previous methods**
  - Experiments show similar or better final performance compared to previous methods in DeepMind Control Suite(four tasks) and OpenAI Gym benchmark(four tasks)
    - SAC(state)
    - SAC
    - PlaNet
    - MPO(10^7 steps)
    - MPO
    - SLAC
    - D4PG(10^8 steps)
    - DVRL
    - DrQ
  - Experiments show that SLAC successfully learns complex continuous control benchmark tasks from raw image inputs.



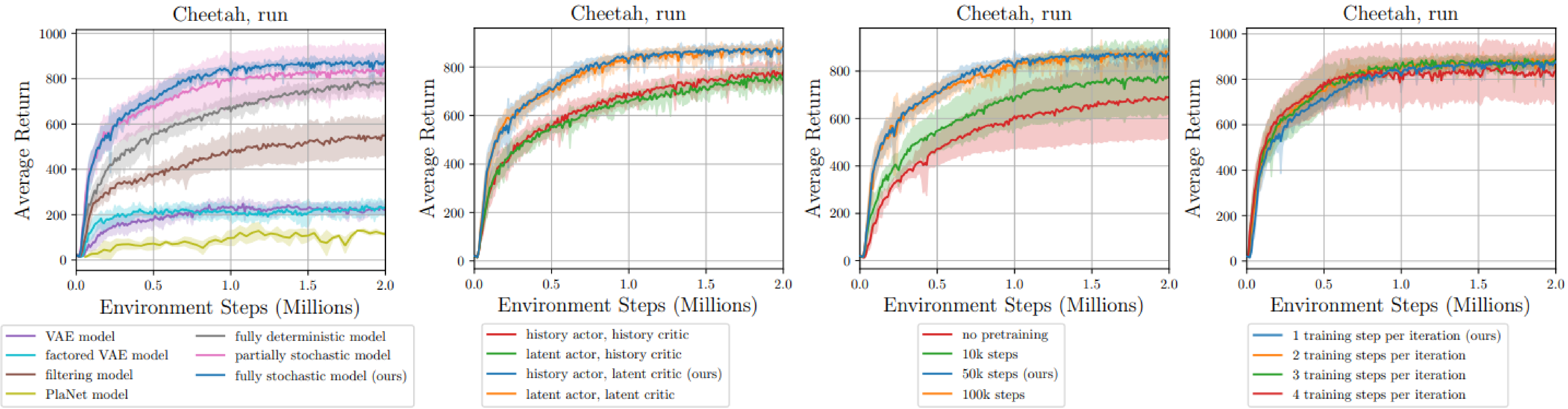
Average return in DeepMind Control Suit



Average return in Open AI Gym

Experiment results (II)

- **Ablation experiments**
  - Four sets of ablation experiments were performed to investigate how SLAC is affected by design choice
    - (a) Latent variable model
    - (b) The inputs, given to the actor and critic
    - (c) Number of model pretraining steps
    - (d) Number of training updates relative to the number of agent interactions



Comparisons of different design choices for (a), (b), (c), (d).

Thank you!

Q&A