Paper summary: Improving sample efficiency in model-free reinforcement learning from images

May 30, 2022

1 Idea in few sentances

2 Explanation of the central concept

3 Methodology

4 Initial rambly notes

4.1 Abstract

Fitting a high-capacity encoder to extract features (state information) from images with only the reward signal leads to poor performance. One option is to incorporate reconstruction loss into an off-policy algorithm, but that often leads to training instability. Investigtion into why shows variational autoencoders to be a problem.

4.2 Introduction

Some solutions to low sample efficiency are:

1. use an off-policy algorithm
2. add an auxiliary task with an unsupervised objective

The simplest auxiliary task is an autoencoder with a pixel reconstruction objective. Prior works uses a two-step training procedure, but this often leads to lower final performance.

> We confirm that a pixel reconstruction loss is vital for learning a good representation, specifically when trained jointly, but requires careful design choices to succeed.

There are 3 contributions:

1. methodical study of the issues involved with combining autoencoders with model-free RL in the off-policy setting, resulting in SAC+AE
2. demonstating SAC+AE does its thing robustly
3. open-source code of SAC+AE

4.3 Related work

The 2010 AE paper re-encodes after every AE update, which is unfeasible for large problems. In Finn et al. in Learning visual feature spaces for robotic manipulation with deep spatial autoencoders, authors pretrain and the linear policy is trained separately. This does not translate to end-to-end methods which are to be developed here. Other stuff was unstable and hindered policy learning performance. Model-based stuff is cool and is efficient, but it is super brittle and sensitive to hyperparameters due to multiple different auxiliary losses, ex. dynamics loss, reward loss, decoder loss, in addition to policy and/or value optimizations.

4.4 Background

4.4.1 SAC

Maximum entropy objective:

$$\pi^* = \underset{\pi}{\operatorname{argmax}} \sum_{t=1}^{T} \mathbb{E}_{(\boldsymbol{s}_t, \boldsymbol{a}_t) \sim \rho_\pi} \left[r_t + \alpha \mathcal{H}(\pi(\cdot|\boldsymbol{s}_t)) \right] \tag{1}$$

This is used to derive soft policy iteration. Soft Bellman residual:

$$J(Q) = \mathbb{E}_{(\boldsymbol{s}_t, \boldsymbol{a}_t, r_t, \boldsymbol{s}_{t+1}) \sim \mathcal{D}} \left[\left(Q(\boldsymbol{s}_t, \boldsymbol{a}_t) - r_t - \gamma \bar{V}(\boldsymbol{s}_{t+1}) \right)^2 \right] \tag{2}$$

where the target value function $\bar{V}$ is approximate via Monte-Carlo estimate of

$$\bar{V}(\boldsymbol{s}_t) = \mathbb{E}_{\boldsymbol{a}_t \sim \pi} \left[\bar{Q}(\boldsymbol{s}_t, \boldsymbol{a}_t) - \alpha \log \pi(\boldsymbol{a}_t | \boldsymbol{s}_t) \right] \tag{3}$$

The policy improvement step then attempts to project a parametric policy $\pi(\boldsymbol{a}_t|\boldsymbol{s}_t)$ by minimizing KL divergence between the policy and a Boltzmann distribution induced by the Q-function using the objective:

$$J(\pi) = \sim_{\approx} \sim \mathcal{D} \left[D_{KL}(\pi(\cdot|\boldsymbol{s}_t) || \mathcal{Q}(\boldsymbol{s}_t, \cdot)) \right] \tag{4}$$

where $\mathcal{Q}(\boldsymbol{s}_t, \cdot) \propto \exp\left\{ \frac{1}{\alpha} Q(\boldsymbol{s}_t, \cdot) \right\}$

4.4.2 Image-based observations and autoencoders

AE is represented as a convolutional encoder g_ϕ that maps an image observation $\boldsymbol{o}_t$ to a low-dimensional latent vector $\boldsymbol{z}_t$, and a deconvolutional decoder $f_\phi : \mathcal{Z} \to \mathcal{O}$. Both the encoder and decoder and trained simultaneously by maximizing the expected log-likelihood

$$J(AE) = \mathbb{E}_{\boldsymbol{o}_t \sim \mathcal{D}} \left[\log p_\theta(\boldsymbol{o}_t|\boldsymbol{z}_t)\right] \tag{5}$$

where $\boldsymbol{z}_t = g_\phi(\boldsymbol{o}_t)$.

In the β-VAE case, the objective is:

$$J(VAE) = \mathbb{E}_{\boldsymbol{o}_t \sim \mathcal{D}} \left[\mathbb{E}_{\boldsymbol{z}_t \sim q_\phi(\boldsymbol{z}_t|\boldsymbol{o}_t)} \left[\log p_\theta(\boldsymbol{o}_t|\boldsymbol{z}_t)\right]\right] - \beta D_{KL}(q_\phi(\boldsymbol{z}_t|\boldsymbol{o}_t)||p(\boldsymbol{z}_t)) \tag{6}$$

where the variational distribution is parametrized as $q_\phi(\boldsymbol{z}_t|\boldsymbol{o}_t) = \mathcal{N}(\boldsymbol{z}_t|\mu_\phi(\boldsymbol{o}_t), \sigma^2_\phi(\boldsymbol{o}_t))$. The latent vector $\boldsymbol{z}_t$ is the used by an RL algorithm instead of the unavailable true state $\boldsymbol{s}_t$.

4.5 Representation learning with image reconstruction

For a model-free RL algorithm, learning from pixels yield much worse results than learning from state. Prior works has shown that using auxiliary supervision to learn state representations helps. The focus of this work is to examine the use of image reconstruction loss as the auxiliary loss. Task-dependent auxiliary loss and world models are avoided in this work.

The authors tried to use a β-VAE, but only on current frames, instead of a sequence of frames. They tried alternating the training, and observed an positive correlation between performance and alternation frequency, but the final result didn't close the performance gap. They tried updating the β-VAE encoder with actor-critic gradients, but this led to severe instability in traning. They conclude that they resulted from the stochasic nature of β-VAEs and the non-stationary gradient from the actor.

4.5.1 Alternating representation learning with a β-VAE

First thing is to confirm that alternative training between the VAE and RL algorithm. Let N be the number of updates steps of one before the switch. The authors observe a positive correlation between N getting smaller and more efficient learning.

4.5.2 Joint representation learning with a β-VAE

Now the goal is to learn a latent representation that is well aligned with the underlying RL objective. This is achieved by updating the encoder network with actor and/or critic gradients along with reconstruction gradients. This is unstable and unusable with β-VAE.

4.5.3 Stabilizing joint representation learning

Smaller β values help due to their reduction in VAE stochasicity. Not using actor's gradients improves performance even further. This leads to the conclusion that stochasicity hurts and that a deterministic encoder will do better.

4.6 Method

4.7 Other stuff