1. Method Implementation Details

In this section we go over implementation details for our method as well as our comparisons.

1.1. Architecture Details

Block Pushing Domain: All comparisons leverage a nearly identical architecture, and are trained on an Nvidia 2080 RTX. In the block pushing domain input observations are $[64, 64, 6]$ in the case of our model (GAP), as well as the ablations, and $[64, 64, 3]$ in the case of Standard.

All use an encoder $f_{\text{enc}}$ with convolutional layers (channels, kernel size, stride): $[(32, 4, 2), (32, 3, 1), (64, 4, 2), (64, 3, 1), (128, 4, 2), (128, 3, 1), (256, 4, 2), (256, 3, 1)]$ followed by fully connected layers of size $[512, 2 \times L]$ where $L$ is the size of the latent space (mean and variance). All layers except the final are followed by ReLU activation.

The decoder $f_{\text{dec}}$ takes a sample from the latent space of size $L$, then is fed through fully connected layers $[128, 128, 128]$, followed by de-convolutional layers (channels, kernel size, stride): $[(128, 5, 2), (64, 5, 2), (32, 6, 2), (3, 6, 2)]$. All layers except the final are followed by ReLU activation, except the last layer which is a Sigmoid in the case of Standard, and GAP (-Residual), and Tanh in the case of GAP and GAP (-Goal Cond).

For all models the dynamics model $f_{\text{dyn}}$ are a fully connected network with layers $[128, 128, 128, L]$, followed by ReLU activation except the final layer.

The inverse model baseline utilizes the same $f_{\text{enc}}$ and $f_{\text{dy}}$ as above, but $f_{\text{dec}}$ is instead a fully connected network of size $[128, 128, \text{action size}]$ where action size is 4 (corresponding to delta x,y, z motion and gripper control). All layers except the final are followed by ReLU activation.

Lastly, the RIG (Nair et al., 2018) baseline uses a VAE with identical $f_{\text{enc}}$ and $f_{\text{dec}}$ to the standard approach above, except learns a policy in the latent space. The policy architecture used is the default SAC (Haarnoja et al., 2018) from RLkit, namely 2 layer MLPs of size 256.

1.2. Training Details

Block Pushing Domain: We collect a dataset of 2,000 trajectories, each 50 timesteps with a random policy. All models are trained on this dataset to convergence for roughly 300,000 iterations. All models are trained with learning rate of $1 \times 10^{-4}$, and batch size 32.

The RIG baseline is trained using the default SAC example parameters in RLkit, for an additional 3 million steps.

BAIR Robot Dataset: We train on the BAIR Robot Dataset (Ebert et al., 2017) as done in the original SVG paper, except with action conditioning.

RoboNet: We train on the subset of the RoboNet dataset which considers only the sawyer arm and the front facing camera view, and use a random 80/20 train test split.

References


SVG+GAP: In all SVG (Denton and Fergus, 2018) based experiments on real robot data, the architecture used is identical to the SVG architecture as described in official repo\(^1\) with the VGG encoder/decoder. All BAIR dataset experiments take as input sequences of 2 frames and predict 10 frames, while all RoboNet experiments take as input 2 frames and predict 20 frames. The latent dimension is 64, and the encoder output dimension is 128. All models are trained with batch size 32.

\(1\)https://github.com/edenton/svg