A. Appendix

A.1. Algorithms

Algorithm 1
Online RL with decoupled ATC encoder (steps distinct from end-to-end RL in blue)

```
Require: $\theta_{ATC}, \phi_\pi$      ▷ ATC model parameters (encoder $f_\theta$ thru contrast $W$), policy parameters
$\mathcal{S} \leftarrow \{\}$      ▷ replay buffer of observations
$\bar{\theta}_{ATC} \leftarrow \theta_{ATC}$      ▷ initialize momentum encoder (conv and linear only)
repeat
    Sample environment and policy, through encoder:
    for 1 to m do      ▷ a minibatch
        $a \sim \pi(\cdot|f_\theta(s);\phi), s' \sim T(s,a), r \sim R(s,a,s')$
        $\mathcal{S} \leftarrow \mathcal{S} \cup \{s\}$      ▷ store observations (delete oldest if full)
        $s \leftarrow s'$
    end for
    Update policy by given RL formula:      ▷ on- or off-policy
    for 1 to n do      ▷ given number RL updates per minibatch
        $\phi_\pi \leftarrow \phi_\pi + RL(s,a,s',r;\phi_\pi)$      ▷ stop gradient into encoder
    end for
    Update encoder (and contrastive model) by ATC:
    for 1 to p do
        $s, s_+ \sim \mathcal{S}$      ▷ sample observations: anchors and positives
        $\theta_{ATC} \leftarrow \theta_{ATC} - \lambda_{ATC}\nabla_{\theta_{ATC}}\mathcal{L}^{ATC}(s,s_+)$      ▷ ATC gradient update
        $\bar{\theta}_{ATC} \leftarrow (1-\tau)\bar{\theta}_{ATC} + \tau\theta_{ATC}$      ▷ update momentum encoder (conv and linear only)
    end for
until converged
return Encoder $f_\theta$ and policy $\pi_\phi$
```

A.2. Additional Figures

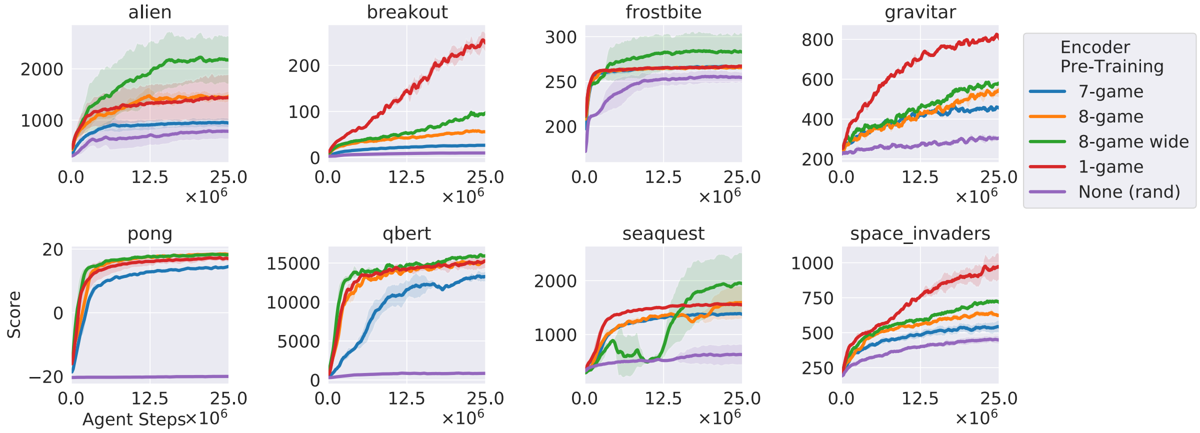
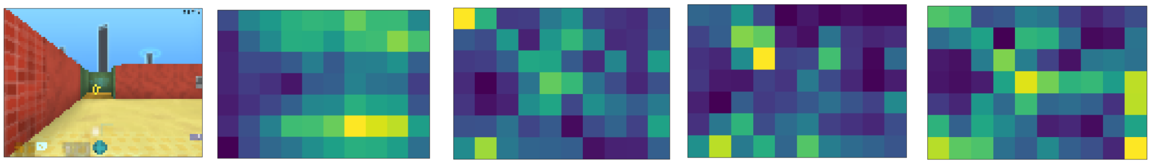
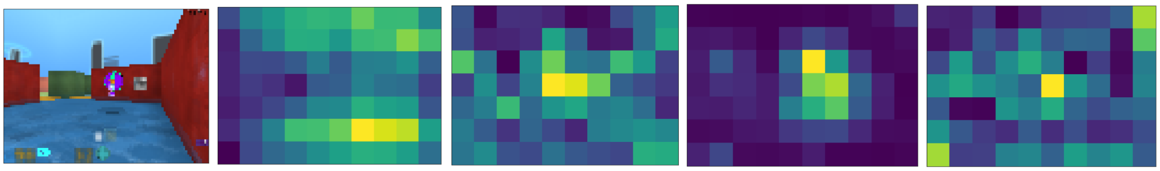


Figure 13. RL using multi-task encoders (all with weights frozen) for eight Atari games gives mixed performance, partially improved by increased network capacity (8-game-wide). Training on 7 games and testing on the held-out one yields diminished but non-zero performance, showing some limited feature transfer between games.



Stacked inputs Random Pixel Control RL-trained ATC (ours)

Figure 14. Attention map in LASERTAG. UL encoder with pixel control focuses on the score, while UL encoder with the proposed ATC focuses properly on the coin similar to RL-trained encoder.



Stacked inputs Random Pixel Control RL-trained ATC (ours)

Figure 15. Attention map in the LASERTAG which shows that UL encoders focus properly on the enemy similar to RL-trained encoder.

A.3. RL Settings

Table 1. DMControl, RAD-SAC Hyperparameters.

Hyperparameter	Value
Observation rendering	$(84, 84)$, RGB
Random Shift Pad	± 4
Replay buffer size	1×10^5
Initial steps	1×10^4
Stacked frames	3
Action repeat	2 (Finger, Walker) 8 (Cartpole) 4 (rest)
Optimizer	Adam
$(\beta_1, \beta_2) \rightarrow (f_\theta, \pi_\psi, Q_\phi)$	$(.9, .999)$
$(\beta_1, \beta_2) \rightarrow (\alpha)$	$(.5, .999)$
Learning rate $(f_\theta, \pi_\psi, Q_\phi)$	2×10^{-4} (Cheetah) 1×10^{-3} (rest)
Learning rate (α)	1×10^{-4}
Batch Size	512 (Cheetah, Pendulum) 256 (rest)
Q function EMA τ	0.01
Critic target update freq	2
Convolution filters	$[32, 32, 32, 32]$
Convolution strides	$[2, 2, 2, 1]$
Convolution filter size	3
Encoder EMA τ	0.05
Latent dimension	50
Hidden units (MLP)	$[1024, 1024]$
Discount γ	$.99$
Initial temperature	0.1

Table 2. Atari, PPO Hyperparameters.

Hyperparameter	Value
Observation rendering	$(84, 84)$, Grey
Stacked frames	4
Action repeat	4
Optimizer	Adam
Learning rate	2.5×10^{-4}
Parallel Environments	16
Sampling interval	128
Likelihood ratio clip, ϵ	0.1
PPO Epochs	4
PPO Minibatches	4
Convolution filters	$[32, 64, 64]$
Convolution filter sizes	$[8, 4, 3]$
Convolution strides	$[4, 2, 1]$
Hidden units (MLP)	$[512]$
Discount γ	.99
Generalized Advantage Estimation λ	0.95
Learning rate annealing	linear
Entropy bonus coefficient	0.01
Episodic lives	False
Repeat action probability	0.25
Reward clipping	± 1
Value loss coefficient	1.0

Table 3. DMLab, PPO Hyperparameters.

Hyperparameter	Value
Observation rendering	$(72, 96)$, RGB
Stacked frames	1
Action repeat	4
Optimizer	Adam
Learning rate	2.5×10^{-4}
Parallel Environments	16
Sampling interval	128
Likelihood ratio clip, ϵ	0.1
PPO Epochs	1
PPO Minibatches	2
Convolution filters	$[32, 64, 64, 64]$
Convolution filter sizes	$[8, 4, 3, 3]$
Convolution strides	$[4, 2, 1, 1]$
Hidden units (LSTM)	[256]
Skip connections	conv 3, 4; LSTM
Discount γ	.99
Generalized Advantage Estimation λ	0.97
Learning rate annealing	none
Entropy bonus coefficient	0.01 (Explore) 0.0003 (Lasertag)
Value loss coefficient	0.5

A.4. Online ATC Settings

Table 4. Common ATC Hyperparameters.

Hyperparameter	Value
Random shift pad	± 4
Learning rate	1×10^{-3}
Learning rate annealing	cosine
Target update interval	1
Target update τ	0.01
Predictor hidden sizes, h_ψ	[512]
Replay buffer size	1×10^5

Table 5. DMControl ATC Hyperparameters.

Hyperparameter	Value
Random shift probability	1
Batch size	as RL (individual observations)
Temporal shift, k	1
Min agent steps to UL	1×10^4
Min agent steps to RL	1×10^4
UL update schedule	as RL (2x Cheetah)
Latent size	128

Table 6. Atari ATC Hyperparameters.

Hyperparameter	Value
Random shift probability	0.1
Batch size	512 (32 trajectories of 16 time steps)
Temporal shift, k	3
Min agent steps to UL	5×10^4
Min agent steps to RL	1×10^5
UL update schedule	Annealed quadratically from 6 per sampler iteration (1×10^4 once at 1×10^5 steps for weight initialization)
Latent size	256

Table 7. DMLab ATC Hyperparameters.

Hyperparameter	Value
Random shift probability	1
Batch size	512 (individual observations)
Temporal shift, k	3
Min agent steps to UL	5×10^4
Min agent steps to RL	1×10^5
UL update schedule	2 per sampler iteration
Latent size	256

A.5. Offline Pre-Training Details

We conducted coarse hyperparameter sweeps to tune each competing UL algorithm. In all cases, the best setting is the one shown in our comparisons.

When our VAEs include a time difference between input and reconstruction observations, we include one hidden layer with action additionally input between the encoder and decoder. We tried both 1.0 and 0.1 KL-divergence weight in the VAE loss, and found 0.1 to perform better in both DMControl and Atari.

DMControl For the VAE, we experimented with 0 and 1 time step difference between input and reconstruction target observations and training for either 1×10^4 or 5×10^4 updates. The best settings were 1-step temporal, and 5×10^4 updates, with batch size 128. ATC used 1-step temporal, 5×10^4 updates (although this can be significantly decreased), and batch size 256 (including CHEETAH). The pretraining data set consisted of the first 5×10^4 transitions from a RAD-SAC agent learning each task, including 5×10^3 random actions. Within this span, CARTPOLE and BALL_IN_CUP learned completely, but WALKER and CHEETAH reached average returns of 514 and 630, respectively (collected without the compressive convolution).

DMLab For Pixel Control, we used the settings from (Hessel et al., 2019) (see the appendix therein), except we used only empirical returns, computed offline (without bootstrapping). For CPC, we tried training batch shapes, $batch \times time$ in (64, 8), (32, 16), (16, 32), and found the setting with rollouts of length 16 to be best. We contrasted all elements of the batch against each other, rather than only forward constrasts. In all cases we also used 16 steps to warmup the LSTM. For all algorithms we tried learning rates 3×10^{-4} and 1×10^{-3} and both 5×10^4 and 1.5×10^5 updates. For ATC and CPC, the lower learning rate and higher number of updates helped in LASERTAG especially. The pretraining data was 125×10^3 samples from partially trained RL agents receiving average returns of 127 and 6 in EXPLORE_GOAL_LOCATIONS_SMALL and LASERTAG_THREE_OPPONENTS_SMALL, respectively.

Atari For the VAE, we experimented with 0, 1, and 3 time step difference between input and reconstruction target, and found 3 to work best. For ST-DIM we experimented with 1, 3, and 4 time steps differences, and batch sizes from 64 to 256, learning rates 1×10^{-3} and 5×10^{-4}. Likewise, 3-step delay worked best. For the inverse model, we tried 1- and 3-step predictions, with 1-step working better overall, and found random shift augmentation to help. For pixel control, we used the settings in (Jaderberg et al., 2017), again with full empirical returns. We ran each algorithm for up to 1×10^5 updates, although final ATC results used 5×10^4 updates. We ran each RL agent with and without observation normalization on the latent image and observed no difference in performance. Pretraining data was 125×10^3 samples sourced from the replay buffer of DQN agents trained for 15×10^6 steps with epsilon-greedy $\epsilon = 0.1$. Evaluation scores were:

Table 8. Atari Pre-Training Data Source Agents.

GAME	EVALUATION SCORE
ALIEN	1,800
BREAKOUT	279
FROSTBITE	1,400
GRAVITAR	390
PONG	18
QBERT	8,800
SEAQUEST	11,000
SPACE INVADERS	1,200