Appendix

A. Experimental Details

A.1. Representation Network

We parametrize the representation function ϕ as a two-hidden layer fully connected neural network with 256 units per layer and output dimension 256. A Swish (Ramachandran et al., 2017) activation function is applied to the output of each hidden layer. We experimented with representation dimension sizes $16, 64, 256,$ and 512, and found 256 and 512 to generally work the best (see Figure 7 in Appendix B.3).

A.2. Transformer Network

The BERT-style transformer used in attentive contrastive learning (ACL) consists of one preprocessing layer of 256 units and ReLU activaiton, followed by a multi-headed attention layer (4 heads with 128 units each), followed by a fully connected feed forward layer with hidden dimension 256 and ReLU activation, finally followed by an output layer of 256 units (the same as ϕ's output). We experimented with different number of attention blocks and number of heads in each block, but did not observe significant difference in performance.

When masking input (sequences of state, actions, or rewards), we randomly choose to 'drop' each item with probability 0.3, 'switch' with probability 0.15, and 'keep' with probability 0.15. 'Drop' refers to replacing the item with a trainable variable of the same dimension. 'Switch' refers to replacing the item with a randomly sampled item from the same batch. 'Keep' refers to leaving the item unchanged. These probability rates where chosen arbitrarily and not tuned.

A.3. Action Prediction and Reconstruction

Whenever a loss includes action prediction or reconstruction, we follow Haarnoja et al. (2019), and (1) utilize an output distribution given by a tanh-squashed Gaussian and (2) apply an additive adaptive entropy regularizer to the action prediction loss.

A.4. Other Networks

With few exceptions, all other functions f, g mentioned in Section 4 are two-hidden layer fully connected neural networks with 256 units per layer and using a Swish (Ramachandran et al., 2017) activation.

The only exception is Momentum TCL, where f is the same structure but using a residual connection on the output.

A.5. Training

During pretraining, we use the Adam optimizer with learning rate 0.0001, except for the TCL variants, for which we found 0.0003 to work better. For Momentum TCL, we use a moving average with rate 0.05.

A.6. Convergence Failures

Representations learned under objectives including forward-raw model, VPN (with $k + 1 = 2$), and DeepMDP consistently diverge and output NaNs on offline and online RL, and are therefore removed from the results in Figure 2. The bisimulation objective on offline and online RL fails to converge in some runs but occasionally succeeds, therefore the means of succeeded runs are computed and shown in Figure 2.

B. Additional Experimental Results

B.1. Fully-Observable Online Environments

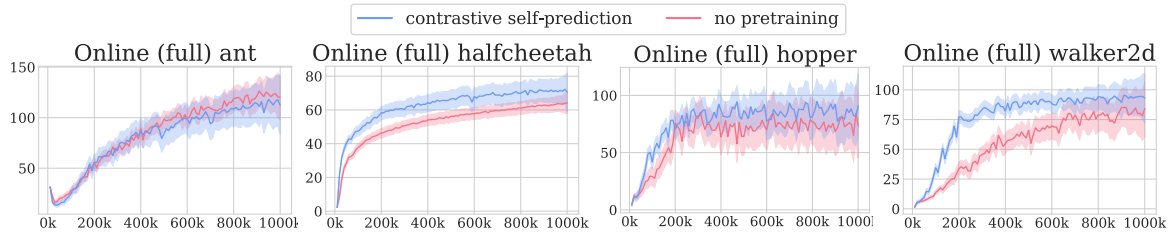

Figure 5. Average reward of best ACL ablation on fully-observable online RL compared to the baseline without pretraining.

B.2. Additional Contrastive Learning Results

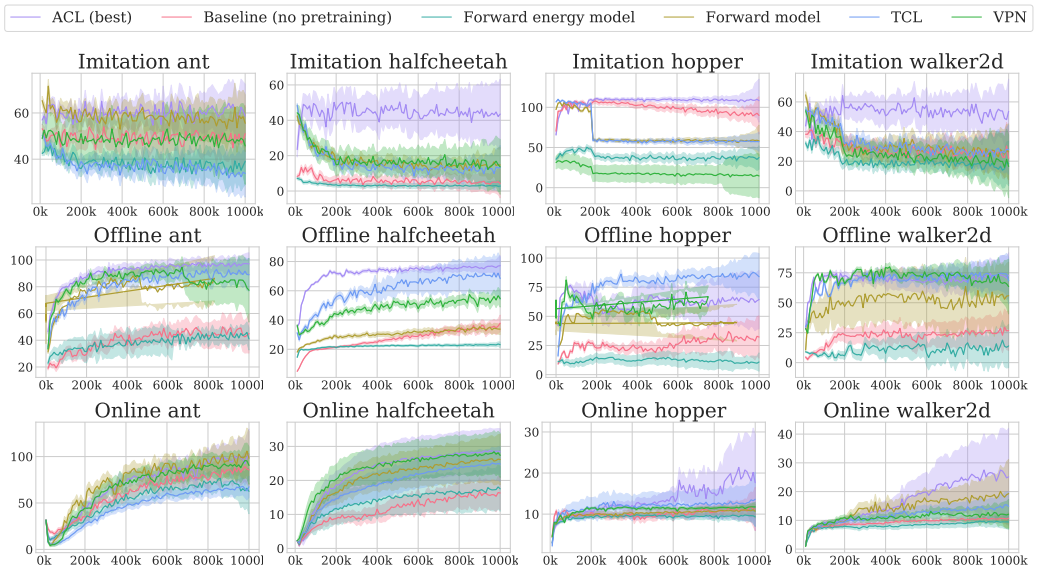

Figure 6. Additional training curves of contrastive learning objectives aggregated over different offline datasets in the same domain. Both in this figure and in Figure 1, we plot the best variant of ACL according to the ablation study, namely we set "input reward" to false in imitation learning, "reconstruct action" to true in offline RL, and "auxiliary loss" (in ant and halfcheetah) or "finetuning" (in hopper and walker2d) to true in online RL. The best variant of ACL generally performs the best compared to other contrastive learning objectives, although TCL's performance is competitive in offline RL.

B.3. Ablation over Representation Size

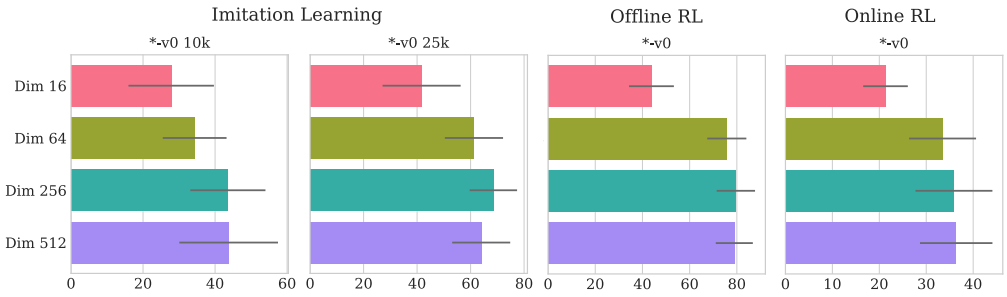

Figure 7. Average reward across domains and datasets with different representation dimensions. 256 and 512 work the best (this ablation is conducted with "reconstruct action" and "reconstruct reward" set to true).

B.4. Ablation over Pretraining Window Size

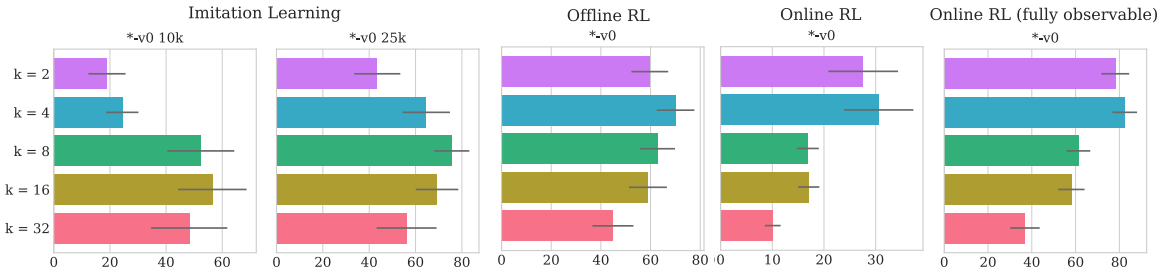

Figure 8. Average reward across domains and datasets with different pretraining window k in imitation learning, offline RL, and partially/fully observable online RL.

B.5. Ablation over Prediction Direction

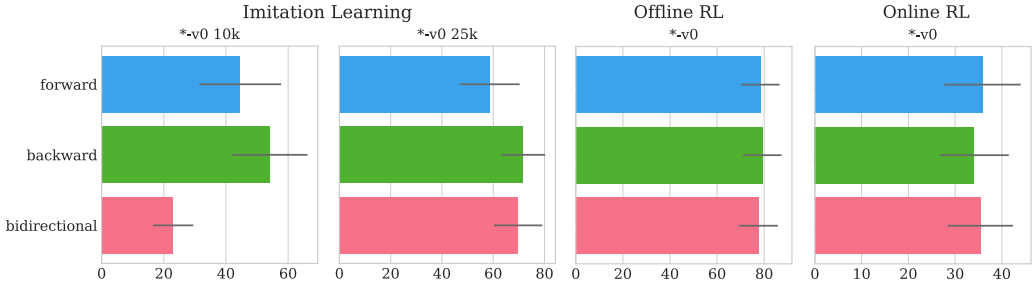

Figure 9. Average reward across domains and datasets with different prediction direction during embedding pretraining.

B.6. Ablation over Compounding Factors and on Sparse Reward

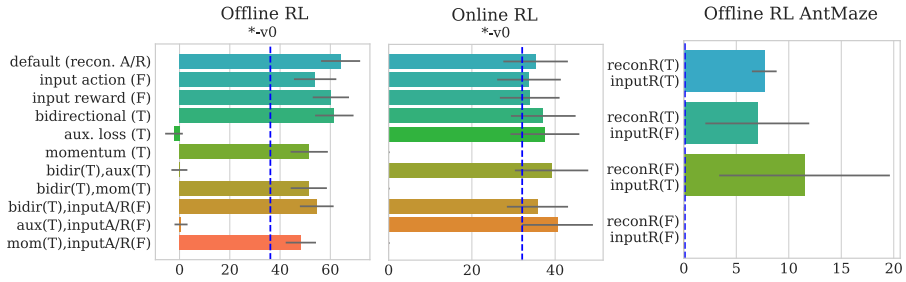

Figure 10. Left: Ablation (with compounding factors) with reconstructing action/reward as default. Right: reward ablation on antmaze-umaze with sparse reward.

B.7. Ablation Results for Individual Domains and Datasets

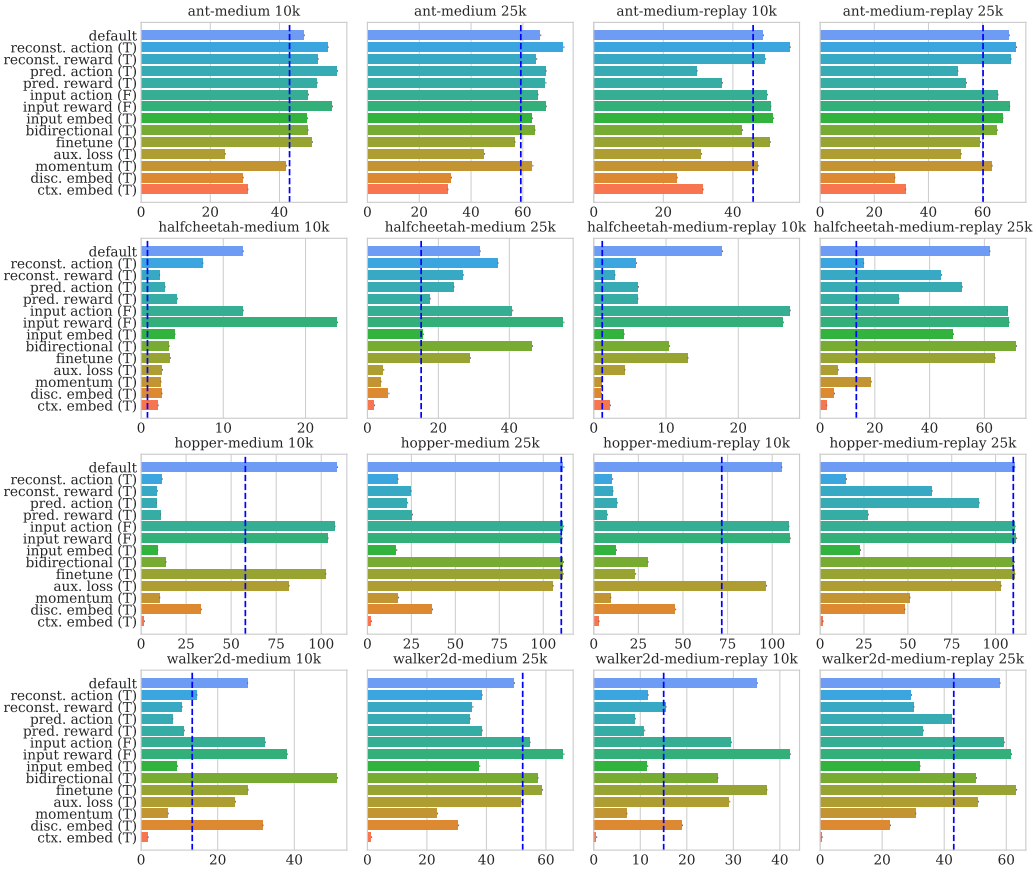

Figure 11. Imitation learning ablation on individual domains and datasets. The negative impact of inputting action and reward to pretraining is more evident in halfcheetah and walker2d. Reconstructing/predicting action/reward is especially harmful in halfcheetah, hopper, and walker2d. There always exists some variant of ACL that is better than without representation learning (blue lines) in all domain-dataset combinations.

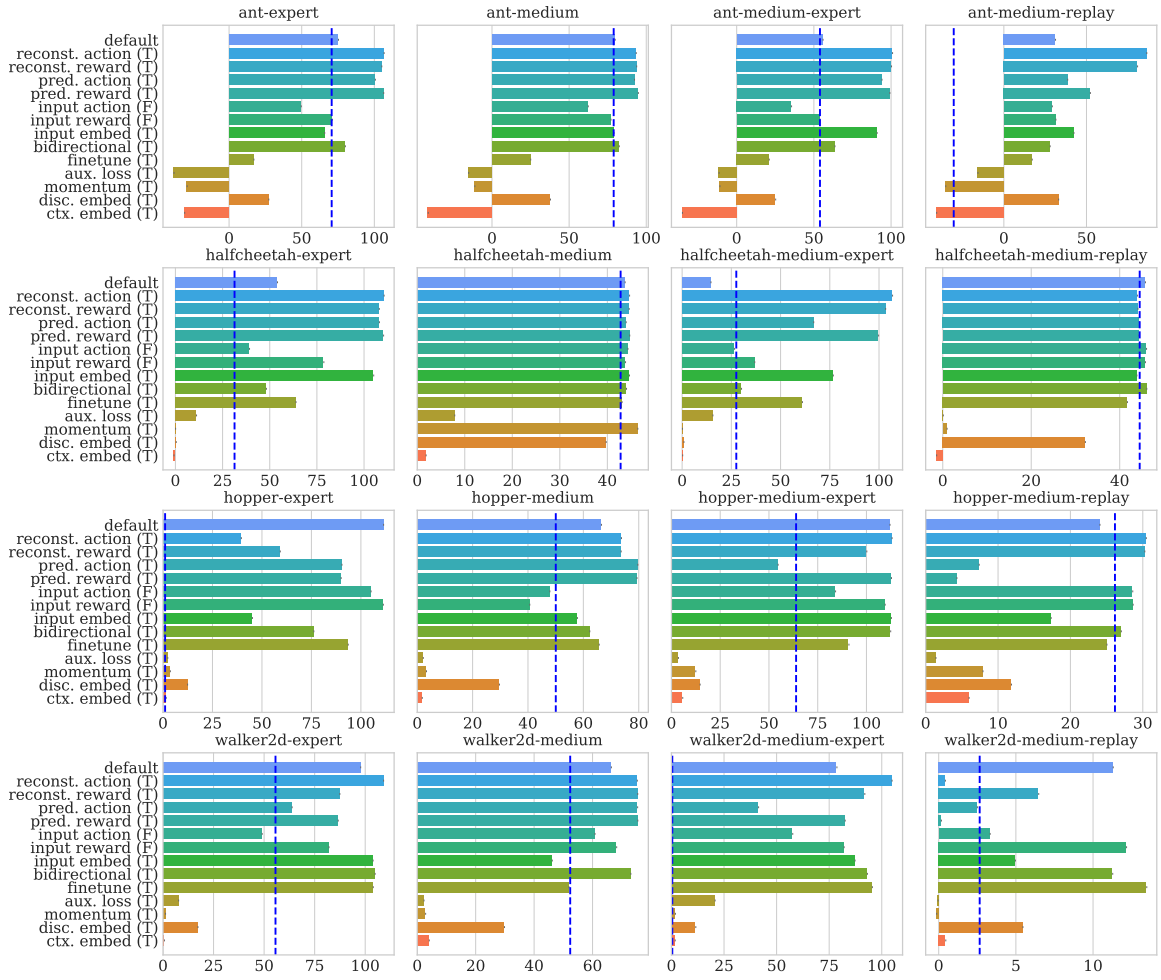

Figure 12. Offline RL ablation on individual domains and datasets. The benefit of representation learning is more evident when expert trajectories are present (e.g., expert and medium-expert) than when they are absent (medium and medum-replay). Reconstructing action and reward is more important in ant and halfcheetah than in hopper and walker2d.

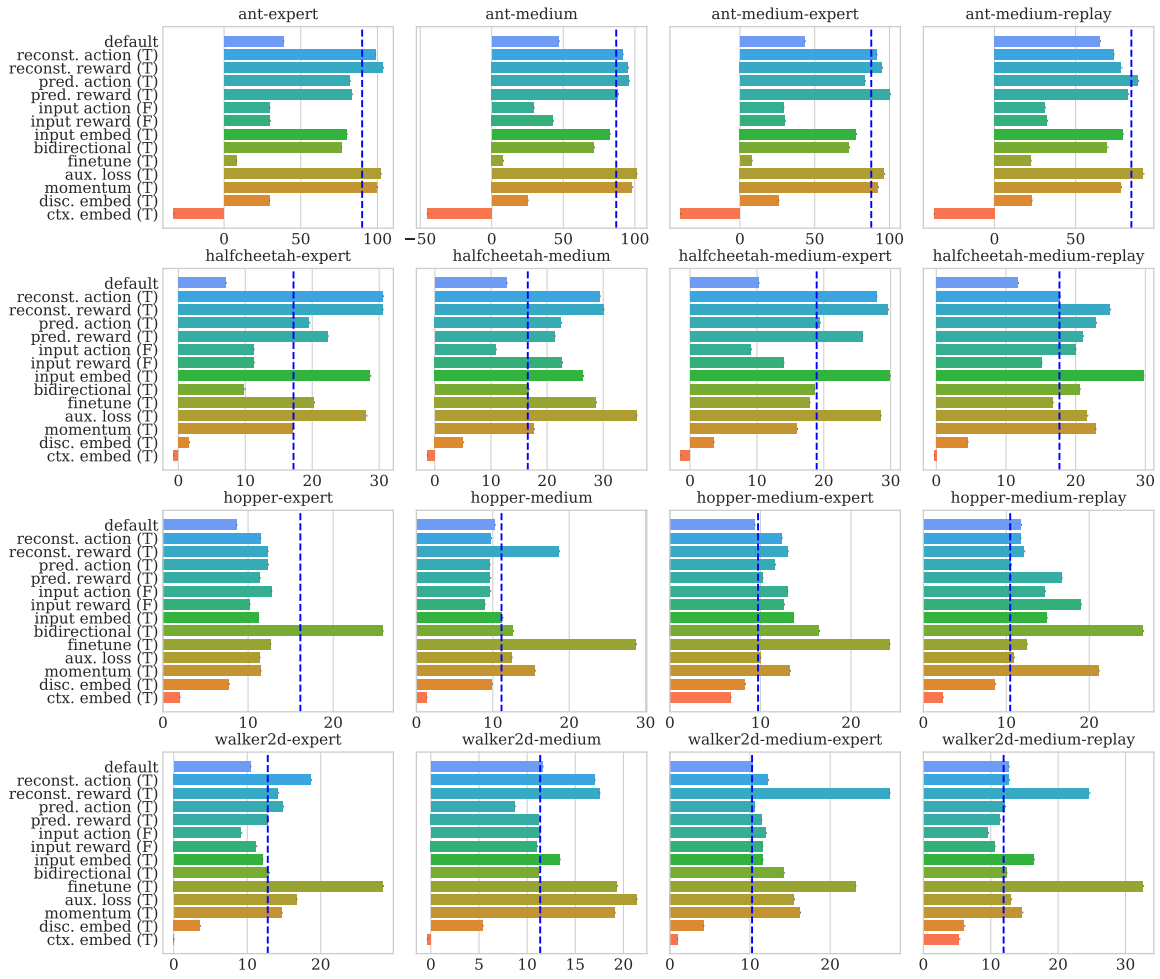

Figure 13. Online RL ablation on individual domains and datasets. Auxiliary loss generally improves performance in all domains and datasets. Finetuning improves halfcheetah, hopper, and walker2d but significantly impairs ant.

C. Additional Anecdotal Conclusions

1. **More ablations.** Although we present our ablations as only changing one factor at a time, we also experimented with changing multiple factors at a time. We did not find any of these additional ablations to change the overall conclusions.
2. **Reconstruct action.** One ablation that did work surprisingly well was to only reconstruct the action (with no other loss). This appeared to perform poorly on imitation learning, but well on other settings.
3. **More transformers.** We experimented with a different application of transformers than ACL. Namely, we attempted to treat each dimension of the state as a token in a sequence (as opposed to using the whole state observation as the token). We found this to provide promising results, although it did not convincingly improve upon the configuration of ACL. Still, it may merit further investigation by future work.
4. **Transformer architecture.** We experimented with a different number of attention blocks or number of heads in each block, but did not observe significant differences in performance.
5. **Normalized or regularized representations.** We experimented with applying an explicit normalization layer on the output of ϕ and found no benefits. We also experimented with a stochastic representation along with a KL-divergence regularizer to the standard normal distribution, and again found no benefits.

D. Additional Interpretations of Results

While we wanted to avoid making claims without exhaustive proof regarding why certain design choices are better than others, we believe many of our findings are interpretable, and here is a selection of our hypotheses:

1. Reward prediction helps in offline and online RL because rewards are critical to the downstream task.
2. Bisimulation-style approaches perform poorly because they are too eager to reduce the latent space (e.g., in the absence of reward, ϕ would be constant).
3. The poor performance of auxiliary training in offline RL may reflect the fact that offline RL is generally more liable to divergence in training, which would dominate gradients from any auxiliary objective and render the representation learning objective useless.
4. The poor performance of context embeddings in all setups may be explained as a consequence of an overly-rich representation – i.e., using context embedding means that in the downstream task the same state may appear multiple times as different representations (since it is in a different context), and this can complicate learning in near-Markovian environments, unlike in NLP.

E. Choice of Non-Image Based Inputs

Our choice to focus on representation learning of state observations (as opposed to image-based inputs) is deliberate, since it allows us to focus on representation learning independent of any prior knowledge on the type of observation. Many prior works focus on images and end up utilizing random cropping/translations, etc. (even using a CNN could be interpreted as using prior knowledge of the invariances of images). We believe our setups make the representation learning aspect *harder* precisely because one cannot use this prior knowledge and that achieving an improvement over the 'no pretraining' baseline is more difficult. In fact, at the beginning of our research we were intensely curious about whether we would still get gains from representation learning on such non-image domains, and were pleasantly surprised to find that significant gains are possible.