References

Anurag Ajay, Aviral Kumar, Pulkit Agrawal, Sergey Levine, and Ofir Nachum. Opal: Offline primitive discovery for accelerating offline reinforcement learning. *arXiv preprint arXiv:2010.13611*, 2020.

Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*, 2019.

Sanjeev Arora, Simon Du, Sham Kakade, Yuping Luo, and Nikunj Saunshi. Provable representation learning for imitation learning via bi-level optimization. In *International Conference on Machine Learning*, pages 367–376. PMLR, 2020.

Yusuf Aytar, Tobias Pfaff, David Budden, Thomas Paine, Ziyu Wang, and Nando De Freitas. Playing hard exploration games by watching youtube. *Advances in neural information processing systems*, 31, 2018.

Igor Babuschkin, Kate Baumli, Alison Bell, Surya Bhupatiraju, Jake Bruce, Peter Buchlovsky, David Budden, Trevor Cai, Aidan Clark, Ivo Danihelka, Antoine Dedieu, Claudio Fantacci, Jonathan Godwin, Chris Jones, Ross Hemsley, Tom Hennigan, Matteo Hessel, Shaobo Hou, Steven Kapturowski, Thomas Keck, Iurii Kemaev, Michael King, Markus Kunesch, Lena Martens, Hamza Merzic, Vladimir Mikulik, Tamara Norman, George Papamakarios, John Quan, Roman Ring, Francisco Ruiz, Alvaro Sanchez, Rosalia Schneider, Eren Sezener, Stephen Spencer, Srivatsan Srinivasan, Wojciech Stokowiec, Luyu Wang, Guangyao Zhou, and Fabio Viola. The DeepMind JAX Ecosystem, 2020. URL `http://github.com/deepmind`.

Bowen Baker, Ilge Akkaya, Peter Zhokov, Joost Huizinga, Jie Tang, Adrien Ecoffet, Brandon Houghton, Raul Sampedro, and Jeff Clune. Video pretraining (vpt): Learning to act by watching unlabeled online videos. *Advances in Neural Information Processing Systems*, 35:24639–24654, 2022.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/google/jax`.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

Xin Chen, Sam Toyer, Cody Wild, Scott Emmons, Ian Fischer, Kuang-Huei Lee, Neel Alex, Steven H Wang, Ping Luo, Stuart Russell, et al. An empirical investigation of representation learning for imitation. *arXiv preprint arXiv:2205.07886*, 2022.

Zichen Jeff Cui, Yibin Wang, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. From play to policy: Conditional behavior generation from uncurated robot data. *arXiv e-prints*, pages arXiv–2210, 2022.

Sarah Dean and Benjamin Recht. Certainty equivalent perception-based control. In *Learning for Dynamics and Control*, pages 399–411. PMLR, 2021.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Yiming Ding, Carlos Florensa, Pieter Abbeel, and Mariano Phielipp. Goal-conditioned imitation learning. *Advances in neural information processing systems*, 32, 2019.

Yan Duan, Marcin Andrychowicz, Bradly Stadie, OpenAI Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. One-shot imitation learning. *Advances in neural information processing systems*, 30, 2017.

Yonathan Efroni, Dipendra Misra, Akshay Krishnamurthy, Alekh Agarwal, and John Langford. Provable rl with exogenous distractors via multistep inverse dynamics. *arXiv preprint arXiv:2110.08847*, 2021.

Benjamin Eysenbach, Tianjun Zhang, Ruslan Salakhutdinov, and Sergey Levine. Contrastive learning as goal-conditioned reinforcement learning. *arXiv preprint arXiv:2206.07568*, 2022.

Chelsea Finn, Xin Yu Tan, Yan Duan, Trevor Darrell, Sergey Levine, and Pieter Abbeel. Deep spatial autoencoders for visuomotor learning. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 512–519. IEEE, 2016.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017a.

Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot visual imitation learning via meta-learning. In *Conference on robot learning*, pages 357–368. PMLR, 2017b.

Xiang Fu, Ge Yang, Pulkit Agrawal, and Tommi Jaakkola. Learning task informed abstractions. In *International Conference on Machine Learning*, pages 3480–3491. PMLR, 2021.

Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum, and Marc G Bellemare. Deepmdp: Learning continuous latent space models for representation learning. In *International Conference on Machine Learning*, pages 2170–2179. PMLR, 2019.

Dibya Ghosh, Abhishek Gupta, and Sergey Levine. Learning actionable representations with goal-conditioned policies. *arXiv preprint arXiv:1811.07819*, 2018.

Dibya Ghosh, Chethan Bhateja, and Sergey Levine. Reinforcement learning from passive data via latent intentions. *arXiv preprint arXiv:2304.04782*, 2023.

Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. *arXiv preprint arXiv:1910.11956*, 2019.

Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. Flax: A neural network library and ecosystem for JAX, 2023. URL `http://github.com/google/flax`.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.

Riashat Islam, Manan Tomar, Alex Lamb, Yonathan Efroni, Hongyu Zang, Aniket Didolkar, Dipendra Misra, Xin Li, Harm van Seijen, Remi Tachet des Combes, et al. Agent-controller representations: Principled offline rl with rich exogenous information. *arXiv preprint arXiv:2211.00164*, 2022.

Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022.

Ilya Kostrikov. JAXRL: Implementations of Reinforcement Learning algorithms in JAX, 10 2022. URL `https://github.com/ikostrikov/jaxrl2`. v2.

Ilya Kostrikov, Ofir Nachum, and Jonathan Tompson. Imitation learning via off-policy distribution matching. *arXiv preprint arXiv:1912.05032*, 2019.

Alex Lamb, Riashat Islam, Yonathan Efroni, Aniket Didolkar, Dipendra Misra, Dylan Foster, Lekan Molu, Rajan Chari, Akshay Krishnamurthy, and John Langford. Guaranteed discovery of controllable latent states with multi-step inverse models. *arXiv preprint arXiv:2207.08229*, 2022.

Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning*, pages 5639–5650. PMLR, 2020.

Kuang-Huei Lee, Ofir Nachum, Tingnan Zhang, Sergio Guadarrama, Jie Tan, and Wenhao Yu. Pi-ars: Accelerating evolution-learned visual-locomotion with predictive information representations. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1447–1454. IEEE, 2022.

Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *Conference on robot learning*, pages 1113–1132. PMLR, 2020.

Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. *arXiv preprint arXiv:2210.00030*, 2022.

Zakaria Mhammedi, Dylan J Foster, Max Simchowitz, Dipendra Misra, Wen Sun, Akshay Krishnamurthy, Alexander Rakhlin, and John Langford. Learning the linear quadratic regulator from nonlinear observations. *Advances in Neural Information Processing Systems*, 33:14532–14543, 2020.

Eric Mitchell, Rafael Rafailov, Xue Bin Peng, Sergey Levine, and Chelsea Finn. Offline meta-reinforcement learning with advantage weighting. In *International Conference on Machine Learning*, pages 7780–7791. PMLR, 2021.

Ofir Nachum and Mengjiao Yang. Provable representation learning for imitation with contrastive fourier features. *Advances in Neural Information Processing Systems*, 34:30100–30112, 2021.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Jyothish Pari, Nur Muhammad Shafiullah, Sridhar Pandian Arunachalam, and Lerrel Pinto. The surprising effectiveness of representation learning for visual imitation. *arXiv preprint arXiv:2112.01511*, 2021.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR, 2017.

Dean A Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 3(1):88–97, 1991.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, pages 5331–5340. PMLR, 2019.

Max Schwarzer, Nitarshan Rajkumar, Michael Noukhovitch, Ankesh Anand, Laurent Charlin, R Devon Hjelm, Philip Bachman, and Aaron C Courville. Pretraining representations for data-efficient reinforcement learning. *Advances in Neural Information Processing Systems*, 34:12686–12699, 2021.

Younggyo Seo, Kimin Lee, Stephen L James, and Pieter Abbeel. Reinforcement learning with action-free pre-training from videos. In *International Conference on Machine Learning*, pages 19561–19579. PMLR, 2022.

Younggyo Seo, Danijar Hafner, Hao Liu, Fangchen Liu, Stephen James, Kimin Lee, and Pieter Abbeel. Masked world models for visual control. In *Conference on Robot Learning*, pages 1332–1344. PMLR, 2023.

Shagun Sodhani, Amy Zhang, and Joelle Pineau. Multi-task reinforcement learning with context-based representations. In *International Conference on Machine Learning*, pages 9767–9779. PMLR, 2021.

Adam Stooke, Kimin Lee, Pieter Abbeel, and Michael Laskin. Decoupling representation learning from reinforcement learning. In *International Conference on Machine Learning*, pages 9870–9879. PMLR, 2021.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020.

David Venuto, Sherry Yang, Pieter Abbeel, Doina Precup, Igor Mordatch, and Ofir Nachum. Multi-environment pretraining enables transfer to action limited datasets. *arXiv preprint arXiv:2211.13337*, 2022.

William Whitney, Rajat Agarwal, Kyunghyun Cho, and Abhinav Gupta. Dynamics-aware embeddings. *arXiv preprint arXiv:1908.09357*, 2019.

Philipp Wu, Arjun Majumdar, Kevin Stone, Yixin Lin, Igor Mordatch, Pieter Abbeel, and Aravind Rajeswaran. Masked trajectory models for prediction, representation, and control. *arXiv preprint arXiv:2305.02968*, 2023.

Mengjiao Yang and Ofir Nachum. Representation matters: offline pretraining for sequential decision making. In *International Conference on Machine Learning*, pages 11784–11794. PMLR, 2021.

Mengjiao Yang, Sergey Levine, and Ofir Nachum. Trail: Near-optimal imitation learning with suboptimal data. *arXiv preprint arXiv:2110.14770*, 2021.

Sherry Yang, Ofir Nachum, Yilun Du, Jason Wei, Pieter Abbeel, and Dale Schuurmans. Foundation models for decision making: Problems, methods, and opportunities. *arXiv preprint arXiv:2303.04129*, 2023.

Denis Yarats, Amy Zhang, Ilya Kostrikov, Brandon Amos, Joelle Pineau, and Rob Fergus. Improving sample efficiency in model-free reinforcement learning from images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10674–10681, 2021.

Tianhe Yu, Chelsea Finn, Annie Xie, Sudeep Dasari, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot imitation from observing humans via domain-adaptive meta-learning. *arXiv preprint arXiv:1802.01557*, 2018.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pages 1094–1100. PMLR, 2020.

Kevin Zakka, Andy Zeng, Pete Florence, Jonathan Tompson, Jeannette Bohg, and Debidatta Dwibedi. Xirl: Cross-embodiment inverse reinforcement learning. In *Conference on Robot Learning*, pages 537–546. PMLR, 2022.

Hongyu Zang, Xin Li, Jie Yu, Chen Liu, Riashat Islam, Remi Tachet Des Combes, and Romain Laroche. Behavior prior representation learning for offline reinforcement learning. *arXiv preprint arXiv:2211.00863*, 2022.

Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. *arXiv preprint arXiv:2006.10742*, 2020.

Thomas T Zhang, Katie Kang, Bruce D Lee, Claire Tomlin, Sergey Levine, Stephen Tu, and Nikolai Matni. Multi-task imitation learning for linear dynamical systems. *arXiv preprint arXiv:2212.00186*, 2022.

A Extended related work

In this paper we focus specifically on pretraining methods that learn representations of high dimensional observations from multitask demonstration data with latent contexts for the purpose of imitation. There are many closely related problems that are studied in other papers that we did not have space to address fully in the main text that we more fully describe here. These are all very interesting and complementary lines of work, but are beyond the scope of this paper.

Perhaps the largest closely related line of work focuses on learning reward-directed representations in the context of reinforcement learning. This is a different setting than ours and methods from there are not applicable in our setting where we do not have access to rewards. Moreover, most of these methods do not consider multitask settings [Zhang et al., 2020, Gelada et al., 2019, Fu et al., 2021, Ghosh et al., 2018, Eysenbach et al., 2022, Sodhani et al., 2021].

Another line of work seeks to learn representations of actions or sequences of actions rather than observations. This is a directly complementary line of work to the ideas presented in this paper [Ajay et al., 2020, Yang et al., 2021, Lynch et al., 2020, Whitney et al., 2019].

Another body of literature focuses on learning representations that can be transferred across domain and embodiment gaps and even trained directly from videos without access to actions at all [Oord et al., 2018, Aytar et al., 2018, Seo et al., 2022, Ma et al., 2022, Zakka et al., 2022, Ghosh et al., 2023]. In this paper, we focus on the simpler task of pretraining a representation within one MDP with consistent dynamics and access to demonstration actions, but with varied tasks. This choice allows us to make more clear comparisons between algorithms and rigorous claims about when representations will be effective, but also limits the generality of the representations that are learned.

There are a variety of new methods that rely on transformer architectures to construct interesting new pretraining objectives [Yang and Nachum, 2021, Seo et al., 2023, Wu et al., 2023]. In this paper we focus on simple methods that can all use the same simple convolutional architecture operating on transition tuples to provide the most controlled comparison that we can. It is an interesting direction for future work to see how our insights in the Markovian case could be leveraged to inform sequence level models of partially observed problems.

Another line of work avoids pretraining representations directly and instead meta-learns a policy that can adapt to new tasks [Duan et al., 2017, Finn et al., 2017a,b, Yu et al., 2018, Rakelly et al., 2019, Mitchell et al., 2021]. This approach is beyond the scope of this paper which focuses on representation learning. Moreover, these meta-learning algorithms require the pretraining trajectories to be partitioned by task so that each task has multiple trajectories. Since we focus on pretraining data where we don't have access to the latent context, it is unclear how to create these meta-training datasets.

Finally, recent work has shown the promise of zero-shot generalization for multitask imitation, especially when the task identifying information is expressed in natural language to leverage advances in language models [Ding et al., 2019, Jang et al., 2022, Cui et al., 2022, Brohan et al., 2022]. This is an exciting line of work, but beyond the scope of this project where we focus on data where the context information is latent. It is an interesting direction for future work to assess precisely how much performance can be improved via extra context information to gauge whether it is worth the cost of labeling trajectories with context information.

It is an interesting direction for future work to try to better synthesize some of the findings from across this broad array of approaches to pretraining in slightly different settings.

B Extended experimental results

In this section we present the experimental results that were excluded from the main text due to space constraints. In particular, Section B.1 presents representation analysis by predicting one representation from another, Section B.2 presents the per-dataset results of various sweeps over dataset size and type, Section B.3 presents per-dataset results for representation analysis, and Section B.4 presents results of an ablation over multistep dynamics.

B.1 Cross-representation prediction

In the main text, we evaluated representation quality by measuring accuracy of small MLPs to predict either the actions on the finetuning data or the low dimensional states on the pretraining

data. Here we present a similar analysis, but now where we use small MLPs to predict the other representations themselves. This is interesting since it lets us assess which representations contain enough information and shared structure to predict the other representations. Hypothetically, a representation that is easily able to recover another representation may be preferable since it retains more information.

Results presented in Figure 9 show the average across datasets of the cross-representation prediction error on a validation set from the pretraining distribution (normalized by the mean prediction error on each dataset). There are several possible takeaways from this experiments. First, looking a the rows, which correspond to the error when each method is used as the source, we can see that inverse dynamics generally has the lowest average error for predicting the other representations. This suggests that inverse dynamics is doing a good job of recovering the information that is shared among all the representations. Second, looking at the columns, which correspond to error when each representation is used as the target, we see that BC is the most difficult to predict and inverse dynamics is second most difficult. This is a somewhat surprising result, but suggests that these representations actually contain information that may have been thrown away (or at made least difficult to access via small MLP) within the other representations. Finally, note that the contrastive learner is both the worst source and easiest target, which is consistent with the idea that those representations are losing important task-relevant information.

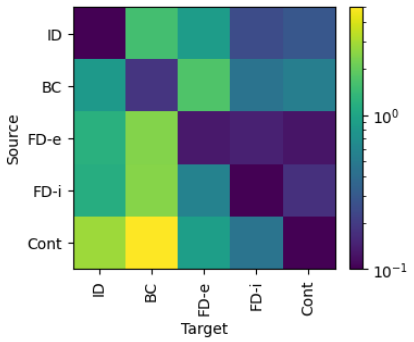

Figure 9: Cross-representation prediction error of a small MLP on a validation set from the pretraining distribution. Results are normalized per dataset by the mean error on that dataset and then averaged across datasets.

Full results on each dataset can be found in Appendix B.3 and full methodological details can be found in Appendix C.

B.2 Per dataset evaluation success results

In the main text and Section B.1 we have only presented aggregate results that average across datasets. These averages make it easier to summarize comparisons between methods, but they sacrifice the precision of how the results vary across datasets. In this section we present per dataset results for all of the relevant sweeps across dataset variations including pretraining size, finetuning size, and finetuning size when we ablate in distribution contexts or observability of the context in the observation.

Pretraining size. First, we present the full ablation over pretraining size, corresponding to the right panel of Figure 3. The full per dataset results are shown in Figure 10.

There are several findings in the dataset-specific results that are not visible in the aggregate reported in the main text:

- First, the kitchen environment is a clear outlier mainly due to the stochasticity in the data generating process and smaller dataset size compared to the others (see Appendix C.1 for more detailed description of the data). As a result of the noise added to the low dimensional states, training from States actually underperforms training from Pixels + Aug. We hypothesize that this is due to some implicit regularization that arises from training from the rendered noisy observations instead of the low dimensional noisy states. Importantly, inverse dynamics is much better able to handle the stochasticity than the alternative methods given the relatively small pretraining dataset and is the only method that is able to perform comparably to training from scratch.

- Point mass is the only environment where the externally pretrained representations (R3M and Imagenet) substantially outperform training from Pixels + Aug and they are substantially outperformed on kitchen and the metaworld datasets. We hypothesize that this shows how it is quite difficult to transfer features across domains and see consistent benefits on challenging tasks.

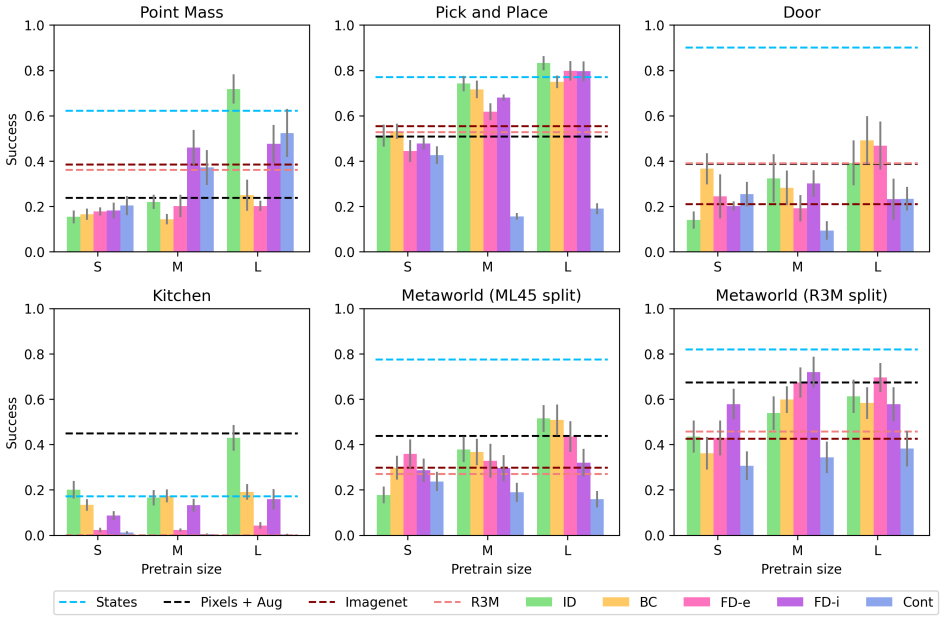

Figure 10: The per dataset results of sweeping over pretraining size, corresponding to the right panel of Figure 3. Error bars show standard error over seeds and contexts (as described in Table 1). Horizontal lines indicate mean performance of algorithms that do not depend on pretraining size.

- Note that performance of contrastive learning is substantially better relative to the alternatives on point mass. We hypothesize that this is due to the fact that random crop augmentations are actually a reasonable simulation of the dynamics in the pointmass environment specifically so that contrastive learning becomes more similar to implicit forward dynamics.

Finetuning size. Next, we present the full ablation over finetuning size, corresponding to the left panel of Figure 3. The full per dataset results are shown in Figure 11.

Again, as described above, Kitchen is a clear outlier due to stochasticity with inverse dynamics the best performer. Inverse dynamics is also the clear winner on point mass and a slight winner on pick and place. The other tasks are more ambiguous with many methods performing about the same, and none substantially better than training from scratch (across all pretraining sizes). Disaggregating the results here shows how even though inverse dynamics is clearly the best in aggregate, this is not necessarily true on every dataset. As we will see in Figure 12, we hypothesize that much of this weak performance can be attributed to the fact that the evaluation contexts in door and the two metaworld variants are truly out of distribution, making it difficult for any pretraining method to generalize.

Ablating in distribution contexts. Next, we present the full per dataset results when we ensure that all the evaluation contexts are included in the pretraining distribution, corresponding to Figure 4 in the main text. The full per dataset results are shown in Figure 12.

It is important to compare these results to those that include out of distribution evaluation contexts in Figure 11. First, note that the evaluation contexts on point mass and pick and place were already in distribution, so they are kept the same. However, on door and the two metaworld splits there is a *substantial* improvement, especially for inverse dynamics and BC. This shows how these methods can benefit from being applied on tasks that are contained in the pretraining distribution. Interestingly, even though the evaluation contexts are now in distribution, the forward dynamics representations do not see substantial improvements and are still outperformed by training from scratch on the more challenging datasets.

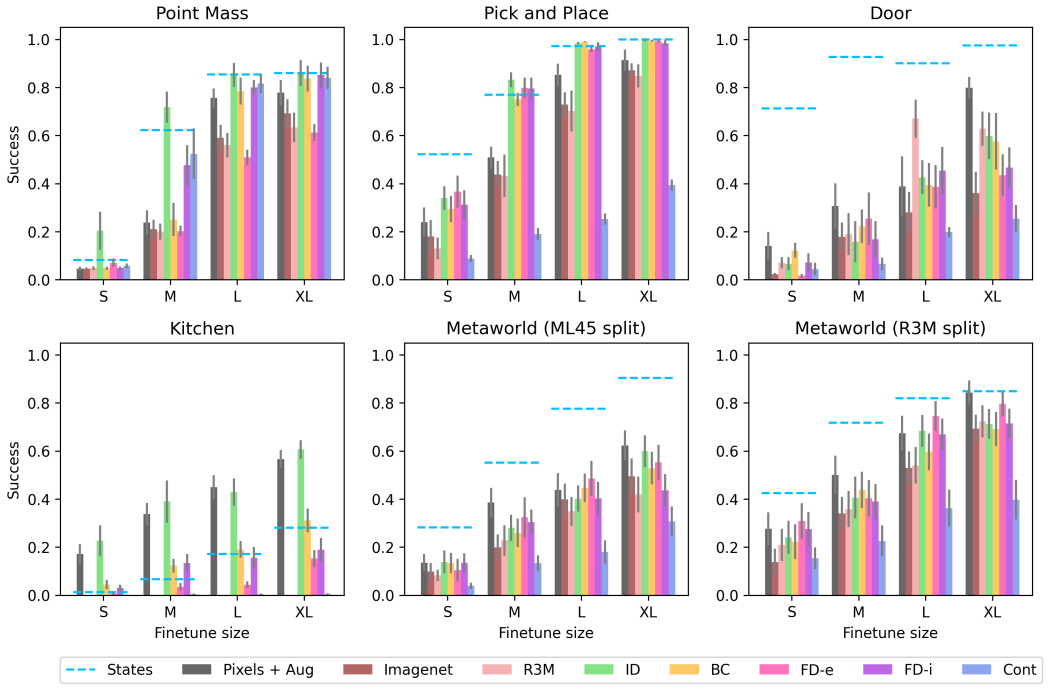

Figure 11: The per dataset results of sweeping over finetuning size, corresponding to the left panel of Figure 3. Error bars show standard error over seeds and contexts (as described in Table 1).

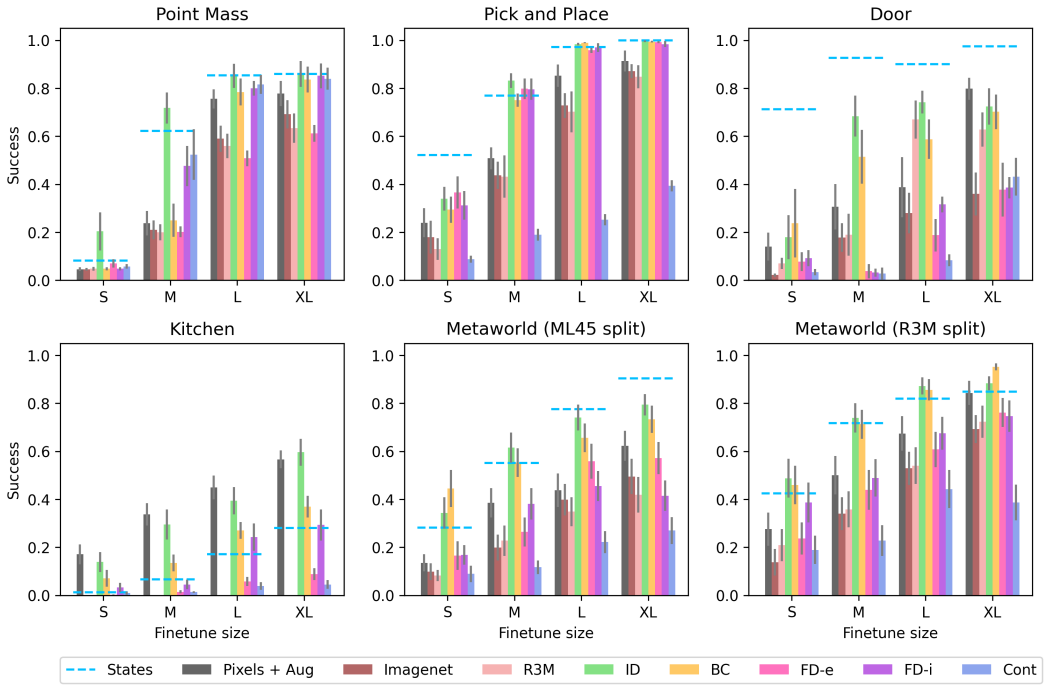

Figure 12: The per dataset results of sweeping over finetuning size when we include the evaluation tasks in the pretraining data, corresponding to Figure 4. Error bars show standard error over seeds and contexts (as described in Table 1).

693

Aggregating based on context observability. Finally, we present the full results for aggregations across whether the context is observable, corresponding to Figure 5 in the main text. Context is latent in point mass, pick and place, and kitchen, but inferrable in door and both metaworld splits. The results are shown in Figure 13. Note that these results are just grouped averages over the results presented in Figure 11.

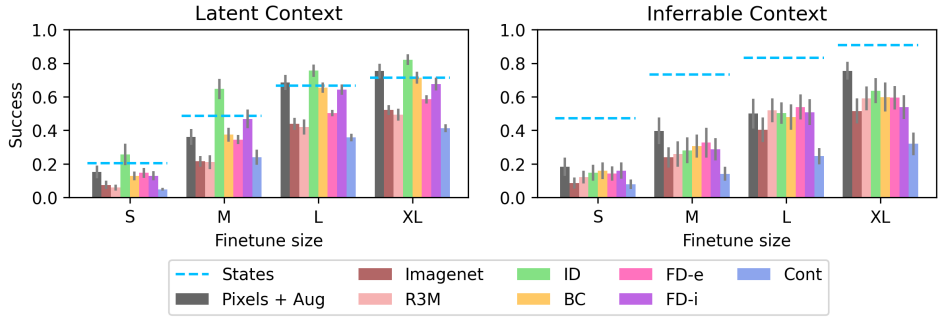

Figure 13: The full results of aggregating based on the observability of the context variable, corresponding to Figure 5. Error bars show standard error over seeds and contexts (as described in Table 1) then averaged across datasets.

Compared to Figure 5, we now include the results from all algorithms and also from the environments where the context is inferrable. As reported in the main text, there is a clear gap between inverse dynamics and BC when the context is latent, likley due to confounding. Here we see that this gap largely disappears in the datasets where the context is inferrable and generally the disparities between methods shrink.

B.3 Per dataset representation analysis

Now we present the per dataset results of the various methods of representation analysis based on predicting different target quantities of interest: the action, the low dimensional state, and the other representations themselves.

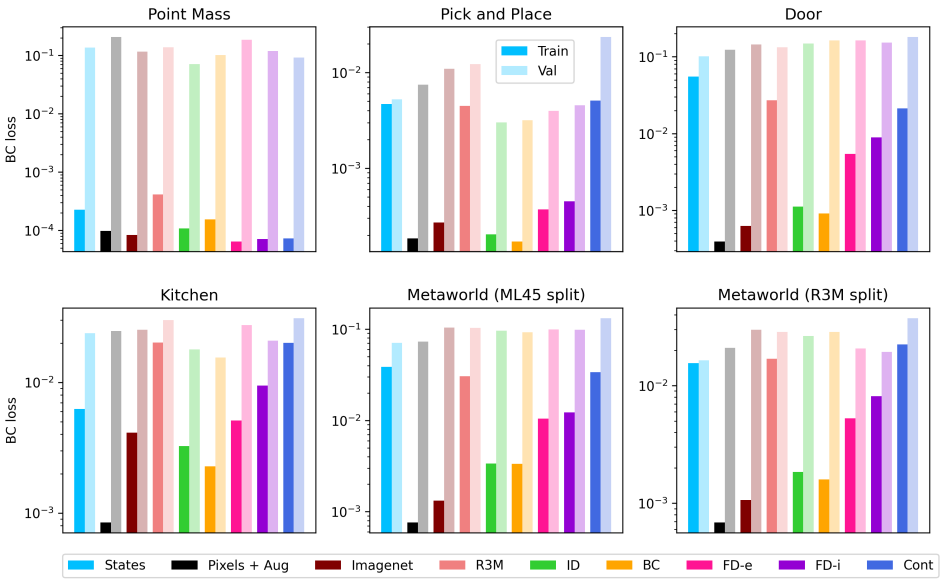

Figure 14: Full per dataset results of action prediction on the finetuning distribution.

Predicting action. First we present the per dataset results for train and validation action prediction on the finetuning datasets using the default pretraining and finetuning size. These results correspond

to Figure 6 from the main text. Unlike in the main text, here we do not do any normalization of the losses, so the losses occur at different scales on each dataset depending on how difficult the prediction task is. Results are shown in Figure 14.

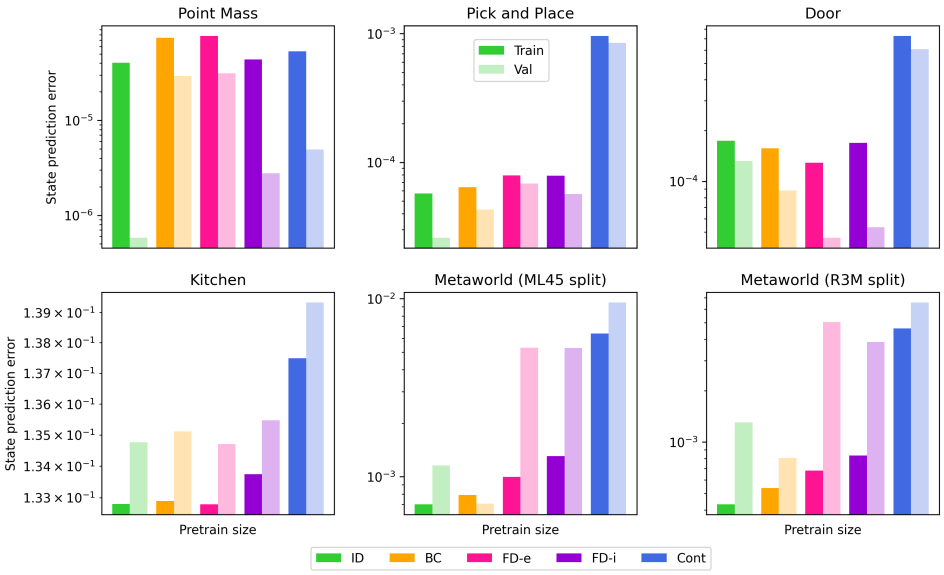

Figure 15: Full per dataset results for state prediction on the pretraining distribution.

Predicting state. Next, we present the per dataset results for predicting the low dimensional state on the pretraining distribution from the various learned representations. These results correspond to Figure 7 in the main text. Again, unlike in the main text, results are not normalized, so they occur at different scales across environments. Results are shown in Figure 15.

Note that as mentioned before, there is stochasticity added to the low dimensional states in the kitchen environment. This makes it difficult for any of the methods to substantially outperform the floor set by the noise level.

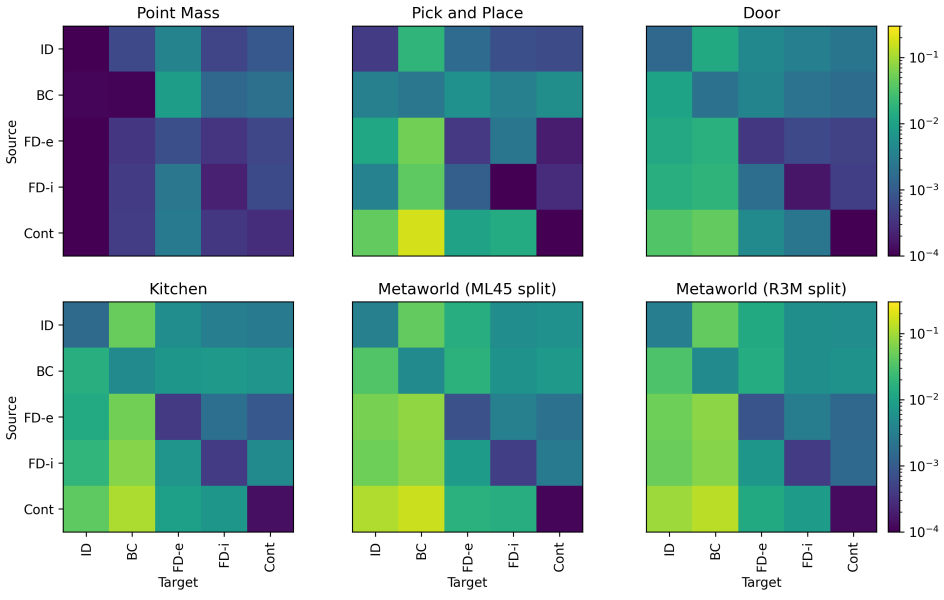

Figure 16: Per dataset results for cross-representation prediction on the pretraining distribution. Color shows the validation error of predicting target from source.

Predicting across representations. Finally, we present the per dataset results for predicting across the different learned representations on the pretraining distribution. These results correspond to Figure 9. Again, unlike in the averaged figure, this figure is not normalized, so the scales vary across datasets. We truncate the color scale at 1e-4 on the low end for easier visualization.

B.4 Ablation of multistep dynamics

As mentioned in the main text, some work argues for multistep dynamics models [Efroni et al., 2021, Lamb et al., 2022]. Note that this work focuses on settings with exogenous noise which are different from the simpler settings that we consider. To confirm that using multistep dynamics models does not help to learn better representations, we run an ablation of the number of steps included in the dynamics model on three environments: point mass, pick and place, and door and two algorithms: inverse dynamics and implicit forward dynamics. Results are shown in Figure 17. At a high level, we basically find little difference when ablating the number of steps, so we default to using one step models everywhere for simplicity.

Note: for inverse dynamics models, we learn a k step model by predicting a_t given o_t and o_{t+k}. For forward dynamics, we learn a k step model by predicting o_{t+k} given o_t and $a_{t:t+k}$.

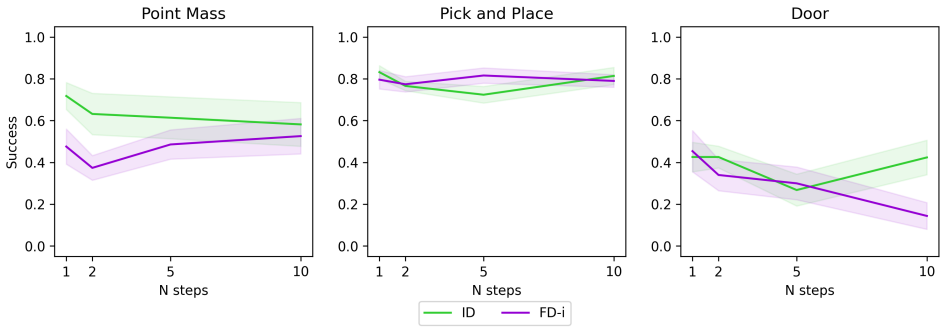

Figure 17: Sweep over the number of timesteps included in the dynamics models.

C Detailed experimental methodology

In this section we present a detailed account of out methodology. We also release our code that was used to perform the experiments for full transparency. We split up the description into Section C.1 which describes the environments and dataset generation, Section C.2 which describes the details of the pretraining pipeline, and Section C.3 which describes the details of the finetuning and evaluation pipeline.

C.1 Envionment and dataset details

Software dependencies. All of our environments are based on the MuJoCo simulator [Todorov et al., 2012]. The point mass environment is derived from the DM control suite [Tunyasuvunakool et al., 2020]. The kitchen environment and dataset was introduced in Gupta et al. [2019]. The rest of the environments are taken from Metaworld [Yu et al., 2020]. We describe each environment in detail and summarize the descriptions in Table 2.

Point mass. The point mass environment consists of an actuated point mass on a 2d plane. In our version, the context $c \in \mathbb{R}^2$ determines the goal location. Then, the demonstration policy π_c^* is a PD controller that moves the point from the current position x to the goal position c. Because the context variable is continuous, we sample an independent context for each trajectory in the pretraining dataset from the uniform distribution over possible goal states. The context is fully latent and not observable in the observation. The low dimensional state is the 2d position and the high dimensional images are 84x84x3.

Pick and place. The pick and place task is taken from the metaworld suite. In our version, the context $c \in \mathbb{R}^3$ determines the goal location for the block. The demonstration policy π_c^* is a scripted

policy from the metaworld repo. We remove the goal indicator from the image in this environment so that the context is fully latent and not observable from the observation. The low dimensional state is the 3d position of the gripper, 1d openness of the gripper, and 7d position and orientation of the block. The high dimensional observations are images of size 120x120x3.

Door. The door environment is also taken from the metaworld suite. In our version, the context $c \in [4]$ determines the index of the environment from door-close, door-open, door-unlock, and door-lock. For our default experiments we use door-close, door-open, and door-unlock as the pretraining contexts and door-lock as the eval context. For the ablation where we ensure that the eval context is in the pretraining distribution, we include door-lock in the pretraining data. The demonstration policy π_c^* is a scripted policy from the metaworld repo. Given the context, the initial position of the robot, initial position of the door, and goal position (which is visible in the observation image) are all randomized. Note, the context is inferrable since the initial position of the door and lock allow the learner to infer the context. The low dimensional state is the 3d position of the gripper, 1d openness of the gripper, 7d position and orientation of two objects in the scene, and 3d goal position. The high dimensional observations are images of size 120x120x3.

Kitchen. The kitchen environment and dataset are taken from Gupta et al. [2019]. Each trajectory contains a sequence of four tasks in a simulated kitchen collected by a human demonstrator. In our version, the context $c \in [24]$ is determined by the sequence of four tasks contained within the demonstration trajectory (of which there are 24 possibilities). We evaluate on three contexts: microwave-kettle-light switch-slide cabinet, bottom burner-top burner-slide cabinet-hinge cabinet, and kettle-bottom burner-top burner-light switch. In our default setup, we pretrain on the other 21 contexts, and in the ablation of in distribution evaluation we pretrain on all 24 contexts. The context is fully latent and not observable from the initial state. The low dimensional state is a 9d description of the arm position and a 21d description of the position of objects in the kitchen. The high dimensional observations are images of size 120x120x3.

Note: the kitchen environment is the only one that we consider that has added noise. The raw data from Gupta et al. [2019] contains gaussian noise added to the low dimensional states and actions, so this noise cannot be removed without re-generating the data. We render the images from the noisy states, so there is also noise present in the image observations. We also evaluate in an environment with the same noise added, so there is no gap between training and eval.

Metaworld (ML45 and R3M). Finally, we consider two variants of the full metaworld suite. Here the context $c \in [50]$ determines which metaworld task is used. We consider two different train-eval splits for our default environments. The ML45 split takes the eval tasks from the original metaworld ML45 task which are bin-picking, box-close, hand-insert, door-lock, and door-unlock. The R3M split takes the eval tasks that were chosen in the R3M paper [Nair et al., 2022]: assembly, bin-picking, button-press, drawer-open, and hammer. Given the context, the initial and goal positions are randomized. The goal position is visible in the observation. The low dimensional state is the 3d position of the gripper, 1d openness of the gripper, 7d position and orientation of (potentially) two objects in the scene, and 3d goal position. The high dimensional observations are images of size 120x120x3.

Table 2: A summary of the description of datasets above. Inferrable refers to whether the context is observable. OOD refers to whether the evaluation context is out of distribution.

Dataset	Policy	Context	Inferrable	OOD	Noise	State dim
Point mass	PD controller	$\mathbb{R}^2$	No	No	No	2
Pick and place	Script	$\mathbb{R}^3$	No	No	No	11
Door	Script	[4]	Yes	Yes	No	21
Kitchen	Human	[24]	No	Yes	Yes	30
Metaworld-ML45	Script	[50]	Yes	Yes	No	21
Metaworld-R3M	Script	[50]	Yes	Yes	No	21

C.2 Pretraining details

Software dependencies. We implement all of our training in JAX [Bradbury et al., 2018]. We use flax for neural networks [Heek et al., 2023] and optax for optimization [Babuschkin et al., 2020]. Our code is loosely based on Kostrikov [2022].

Architecture. All of our pretraining algorithms share exactly the same encoder architecture to ensure that we have a fair comparison. Since our tasks are relatively simple visually, and so as to allow for large scale experiments without too much compute, we use a relatively small convnet encoder. Specifically, we follow the architecture from Yarats et al. [2021] which consists of a 4 layer convnet with 3x3 filters, number of channels of (32, 64, 128, 256), and strides of (2,2,1,1). We add a modification to include a spatial softmax activation [Finn et al., 2016], which we found to be important for the manipulation tasks we consider. This is followed by a linear layer to project into the embedding dimension of 64 and finally a layernorm and tanh activation to normalize the embedding. We use the gelu activation function throughout.

Now we will birefly describe the architecture used for each pretraining algorithm, following their descriptions in Section 4.2:

- Inverse dynamics: the inverse dynamics head is an MLP that takes in $\phi(o), \phi(o')$ and produces an estimated action. This MLP has two hidden layers of width 256 and dropout of 0.1 during training.
- BC: the BC policy head is an MLP with two hidden layers of width 256 and dropout of 0.1 during training.
- Implicit forward dynamics: the implicit forward dynamics model uses an action encoder $\phi_a(a)$ which outputs a 64 dimensional normalized action embedding which is concatenated to $\phi(o)$ to form $\phi(o, a)$. Then there are two projection heads f_1, f_2 that take in $\phi(o, a)$ and $\phi(o')$ respectively and produce 64 dimensional embeddings that are normalized to have unit norm. All these networks (ϕ_a, f_1, and f_2) are MLPs with two hidden layers of width 256 and the relevant input and output dimensions.
- Explicit forward dynamics: the explicit forward dynamics model uses the same architecture to encode a with ϕ_a. Then, instead of projection heads, we require a convolutional decoder to produce an image. Following Yarats et al. [2021] we use an architecture that inverts the encoder, having a dense projection layer followed by channels of (256, 128, 64, 32) and strides of (1,1,2,2).
- Contrastive: the contrastive network is the same as the implicit forward dynamics network except that there is no action input and o' is replaced by an augmentation of o.

Training hyperparameters. For pretraining, we split the datasets into two categories: easy (point mass, pick and place, and door) and hard (kitchen, metaworld-ml45, and meatworld-r3m). On the easy tasks we train for 100k gradient steps and on the hard tasks we train for 200k gradient steps. Batch size is 256 for all methods except explicit forward dynamics where (due to the added compute required for the decoder) we use batch size of 128 to even out computational requirements across methods. All methods are trained with the adamw optimizer with learning rate 1e-3, a cosine learning rate decay schedule, and default weight decay of 1e-4.

Data augmentation. Following [Chen et al., 2022] and others, we note that cropping augmentations are the most important for training policies in simulated visual domains. As such, all of our pretraining algorithms (and the Pixels + Aug baseline) use random cropping augmentations, and we found this to be an important implementation detail. The one exception is explicit forward dynamics where we found it difficult to reconstruct images with augmentations, so we omit them for that algorithm.

Compute resources. Pretraining was all done on an internal cluster using RTX8000 GPUs. We estimate that the final training run needed to produce the results in the paper took approximately 600 GPU hours.

C.3 Finetuning and evaluation details

Training hyperparameters. The policy is always an MLP with two hidden layers of width 256. We use gelu activation and apply dropout with probability 0.1 during finetuning. We finetune on every

dataset for 10k gradient steps with batch size 256. All policies are trained with the adamw optimizer with learning rate 1e-3, a cosine learning rate decay schedule, and default weight decay of 1e-4.

As explained in Table 1 there are several seeds and evaluation contexts for each environment. For example, for the default results in Figure 1 we end up having a total of 80 different finetuning datasets per representation when sweeping across dataset, context, and seed so that Figure 1 is reporting aggregate results across 720 finetuning and evaluation runs.

Evaluation hyperparameters. Each evaluation is run for 100 episodes in the environment to estimate the success of the policy (except for the kitchen environment where we run 50 episodes due to slow rendering of that environment).

Compute resources. Finetuning and evaluation was all done on an internal cluster on CPU (since the finetuned policy network is small and environments run on CPU). We estimate that all the finetuning and evaluation in the final runs used to produce results for the paper took approximately 2000 CPU hours.

D Extended analysis discussion

Here we provide a more detailed discussion of related theoretical work.

One recent line of work focuses on learning representations for exploration [Efroni et al., 2021, Lamb et al., 2022] and offline RL [Islam et al., 2022] in the presence of *exogenous noise*. The exogenous noise setting means that the high dimensional observations contain information that is not effected by the actions; e.g., background dynamics that appear in image observations but do not affect the task. This line of work argues that inverse dynamics modeling is the best approach to ignore exogenous noise. Our results are complementary to this line of work in showing that even in settings *without* exogenous noise, inverse dynamics is still often preferable to alternatives for representation learning. Moreover, we consider a *multitask* imitation setting with latent contexts while they consider single task and reward-directed problems.

Another line of work proves that learning a forward dynamics model is a well-motivated approach for multitask imitation [Nachum and Yang, 2021]. While that work does not directly compare to inverse dynamics pretraining, we find that inverse dynamics pretraining outperforms forward dynamics modeling in our settings. Moreover, while this paper shows that if our representation learns a good forward dynamics model that it works well for imitation, it does not discuss how efficiently such a representation can be learned. So, while both methods are well-motivated, we find inverse dynamics modeling to be more efficient than learning the forward dynamics.

Finally, another line of work studies multitask representation learning for imitation by directly performing behavior cloning [Arora et al., 2019, Zhang et al., 2022]. These methods provide positive results for the approach, but require algorithms that have access to the latent context information which must be discrete so as to learn a separate policy for every pretraining context, thus avoiding confounding. This method requires extra information and is difficult to scale to very large numbers of contexts. In contrast, we find that inverse dynamics modeling is able to perform well without this extra information or added complexity of learning multiple models and naturally avoids confounding by the latent context information.