LEARNING REPRESENTATIONS FOR PIXEL-BASED CONTROL: WHAT MATTERS AND WHY?

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning representations for pixel-based control has garnered significant attention recently in reinforcement learning. A wide range of methods have been proposed to enable efficient learning, leading to sample complexities similar to those in the full state setting. However, moving beyond carefully curated pixel data sets (centered crop, appropriate lighting, clear background, etc.) remains challenging. In this paper, we adopt a more difficult setting, incorporating background distractors, as a first step towards addressing this challenge. We present a simple baseline approach that can learn meaningful representations with no metric-based learning, no data augmentations, no world-model learning, and no contrastive learning. We then analyze when and why previously proposed methods are likely to fail or reduce to the same performance as the baseline in this harder setting and why we should think carefully about extending such methods beyond the well curated environments. Our results show that finer categorization of benchmarks on the basis of characteristics like density of reward, planning horizon of the problem, presence of task-irrelevant components, etc., is crucial in evaluating algorithms. Based on these observations, we propose different metrics to consider when evaluating an algorithm on benchmark tasks. We hope such a data-centric view can motivate researchers to rethink representation learning when investigating how to best apply RL to real-world tasks.

1 INTRODUCTION

Learning useful representations for downstream tasks is a key component for success in rich observation environments [14, 36, 42, 49, 50]. Consequently, a significant amount of work proposes various representation learning objectives that can be tied to the original reinforcement learning (RL) problem. Such auxiliary objectives include the likes of contrastive learning losses [38, 34, 11], state similarity metrics like bisimulation or policy similarity [57, 56, 1], and pixel reconstruction losses [27, 19, 24, 23]. On a separate axis, data augmentations have been shown to provide huge performance boosts when learning to control from pixels [33, 31]. Each of these methods has been shown to work well for particular settings and hence displayed promise to be part of a general purpose representation learning toolkit. Unfortunately, these methods were proposed with different motivations and tested on different tasks, making the following question hard to answer:

What really matters when learning representations for downstream control tasks?

Learning directly from pixels offers much richer applicability than when learning from carefully constructed states. Consider the example of a self-driving car, where it is nearly impossible to construct a complete state description such that it describes the position and velocity of all objects of interest, such as road edges, highway markers, other vehicles, etc. In such real world applications, learning from pixels offers a much more feasible option. However, this requires algorithms that can discern between task-relevant and task-irrelevant components in the pixel input, i.e., learn good representations. Focusing on task-irrelevant components can lead to brittle or non-robust behavior when put in slightly different environments. For instance, billboard signs over buildings in the background has no dependence on the task in hand while a self-driving car tries to change lanes. However, if such task-irrelevant components are not discarded, they can lead to sudden failure when the car drives through a different environment, say a forest where there are no buildings or billboards. Avoiding brittle behavior is therefore key to efficient deployment of artificial agents in the real world.

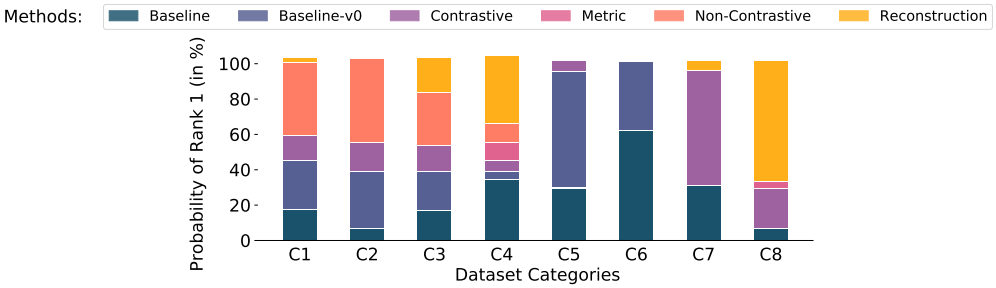

Figure 1: **Comparing pixel-based RL methods across finer categorizations** of evaluation benchmarks. Each category 'Cx' denotes different data-centric properties of the evaluation benchmark (e.g., C1 refers to discrete action, dense reward, without distractors, and with data randomly cropped [31, 33]). Exact descriptions of each category are provided in Table 3. Baseline-v0 refers to applying the standard deep RL agent (e.g., Rainbow DQN [48] and SAC [22]); Baseline refers to adding reward and transition prediction to baseline-v0, as described in Section 3; Contrastive includes algorithms such as CURL [34]; Metric denotes the state metric losses such as DBC [57]; Non-Contrastive includes algorithms such as SPR [41]; Reconstruction[2] includes DREAMER [24] and TIA [18]. For a given method, we always consider the best performing algorithm. Every method leads to varied performance across data categories, making a comparison which is an *average across all categories* highly uninformative. Details of the algorithm used for each method is given in Appendix 3 (Table 4).

There has been a lot of work recently that tries to learn efficiently from pixels. A dominant idea throughout prior work has been that of attaching an auxiliary loss to the standard RL objective, with the exact mechanics of the loss varying for each method [27, 57, 34]. A related line of work learns representations by constructing world models directly from pixels [40, 37, 21, 24]. We show that these work well when the world model is simple. However, as the world model gets even slightly more complicated, which is true of the real world and imitated in simulation with the use of video distractors [55, 29, 43], such approaches can fail. For other methods, it is not entirely clear what component/s in auxiliary objectives can lead to failure, thus making robust behavior hard to achieve. Another distinct idea is of using data augmentations [33, 31] over the original observation samples, which seem to be quite robust across different environments. However, as we will show, a lot of the success of data augmentations is an artifact of how the benchmark environments save data, which is not true of the real world [43], thus resulting in failure[1]. It is important to note that some of these methods are not designed for robustness but instead for enhanced performance on particular benchmarks. For instance, the ALE [7] benchmark involves simple, easy to model objects, and it becomes hard to discern if methods that perform well are actually good candidates for answering 'what really matters for robust learning in the real world.'

Contributions. In this paper, we explore the major components responsible for the successful application of various representation learning algorithms. Based on recent work in RL theory for learning with rich observations [16, 4, 9], we hypothesize certain key components to be responsible for sample efficient learning. We test the role these play in previously proposed representation learning objectives and then propose an exceedingly simple *baseline* (see Figure 2) which takes away the extra "knobs" and instead combines two simple but key ideas, that of reward and transition prediction, to produce a potentially robust representation scheme. We conduct experiments across multiple settings, including the MuJoCo domains with natural distractors [55, 29, 43], DMC Suite [46], and Atari100K [28] from ALE [7]. Following this, we identify the failure modes of previously proposed objectives and highlight why they result in comparable or worse performance than the proposed baseline. Our observations suggest that relying on a particular method across multiple evaluation settings does not work, as the efficacy varies with the exact details of the task, even within the same benchmark (see Figure 1). We note that a finer categorization of available benchmarks based on metrics like density of reward, presence of task-irrelevant components, inherent horizon of tasks, etc., play a crucial role in determining the efficacy of a method. We list such categorizations as suggestions for more informative future evaluations. The findings of this paper advocate for a more data-centric view of evaluating RL algorithms [13], largely missing in current practice. We hope these insights can lead to better representation learning objectives for real-world applications.

[1]Moreover, it is hard to pick exactly which data augmentation will work for a particular environment or task.

[2]For ALE we use the performance of DREAMER after 1M steps, whereas for DMC we consider the performance after 500k steps.

2 RELATED WORK

Prior work on **auxiliary objectives** includes the Horde architecture [45], UVFA [39] and the UNREAL agent [27]. These involve making predictions about features or pseudo-rewards, however only the UNREAL agent used these predictions for learning representations. Even so, the benchmark environments considered there always included only task-relevant pixel information, thus not pertaining to the hard setting we consider in this work. Representations can be also be fit so as to obey certain state similarities. If these **state metrics** preserve the optimal policies and are easy to learn/given a priori, such a technique can be very useful. Recent works have shown that we can learn efficient representations either by learning the metrics like that in bisimulation [17, 57, 56, 8], by recursively sampling states [10] or by model invariance [47]. **Data augmentations** modify the input image into different distinct views, each corresponding to a certain type of modulation in the pixel data. These include cropping, color jitter, flip, rotate, random convolution, etc. Latest works [33, 53] have shown that augmenting the states samples in the replay buffer with such techniques alone can lead to impressive gains when learning directly from pixels. Recently, Stone et al. [43] illustrated the advantages and failure cases of augmentations. **Contrastive learning** involves optimizing for representations such that positive pairs (those coming from the same sample) are pulled closer while negative pairs (those coming from different samples) are pushed away [38, 11]. The most widely used method to generate positive/negative pairs is through various data augmentations [34, 41, 44, 52]. However, temporal structure can induce positive/negative pairs as well. In such a case, the positive pair comes from the current state and the actual next state while the negative pair comes from the current state and any other next state in the current batch [38]. Other ways of generating positive/negative pairs can be through learnt state metrics [1] or encoding instances [24]. Another popular idea for learning representations is learning world models [21] in the pixel space. This involves learning prediction models of the world in the pixel space using a **pixel reconstruction** loss [19, 23, 24]. Other methods that do not explicitly learn a world model involve learning representations using reconstruction based approaches like autoencoders [51].

Quite a few papers in the past have analysed different aspects of the RL problem. [15] and [3] have focused on analysing different policy optimization methods with varying hyperparameters. Our focus is specifically on representation learning methods that improve sample efficiency in pixel-based environments. [26] showed how RL methods in general can be susceptible to lucky seedings. Recently, [2] proposed statistical metrics for reliable evaluation. Despite having similar structure, our work is largely complimentary to these past investigations. [5] analysed reward and transition but only focused on the Atari 200M benchmark and pixel reconstruction methods. In comparison, our work is spread across multiple evaluation benchmarks, and our results show that reconstruction can be a fine technique only in a particular benchmark category.

3 METHOD

We model the RL problem using the framework of contextual decision processes (CDPs), a term introduced in Krishnamurthy et al. [32] to broadly refer to any sequential decision making task where an agent must act on the basis of rich observations (context) to optimize long-term reward. The true state of the environment is not available and the agent must construct it on its own, which is required for acting optimally on the downstream task. Furthermore, the emission function which dictates what contexts are observed for a given state is assumed to only inject noise that is uncorrelated to the task in hand, i.e. it only changes parts of the context that are irrelevant to the task [57, 43]. Consider again the example of people walking on the sides of a road while a self-driving car changes lanes. Invariance to parts of the context that have no dependence on the task, e.g. people in the background, is an important property for any representation learning algorithm since we cannot expect all situations to remain exactly the same when learning in the real world.

We explore the utility of two main components, that of reward and transition prediction, in learning representations. A lot of prior work has incorporated these objectives either individually or in the presence of more nuanced architectures. Here, our aim is to start with the most basic components and establish their importance one by one. Particularly, we use a simple soft actor-critic setup (derived from SAC-AE [51]) as the base architecture, and attach the reward and transition prediction modules to it (See Figure 2). Note that the transition network is over the encoded state s_t and not over the observations [35]. Moreover, the transition model is fit between the encoded state and the reward

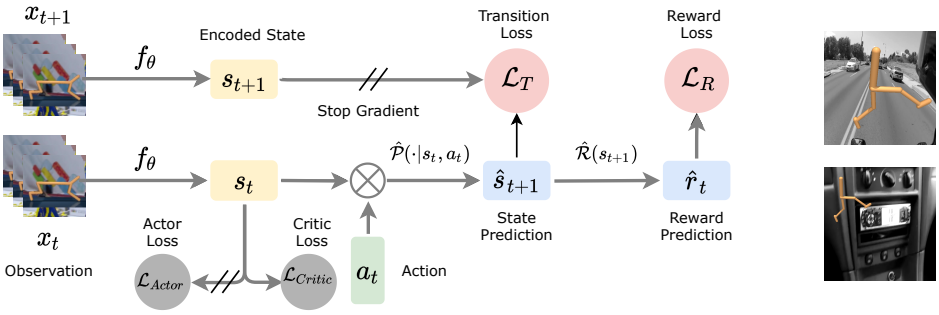

Figure 2: **(Left) Proposed Baseline for control over pixels**. We employ two losses besides the standard actor and critic losses, one being a reward prediction loss and the other a latent transition prediction loss. The encoded state s_t is the learnt representation. Gradients from both the transition/reward prediction and the critic are used to learn the representation, whereas the actor gradients are stopped. In the ALE setting, the actor and critic losses are replaced by a Rainbow DQN loss [48]. **(Right) Natural Distractor** in the background for standard DMC setting (top) and custom off-center settings (bottom).

model. Unless noted otherwise, we call this architecture as the *baseline* for all our experiments. Details about the network sizes and all hyperparameters is provided in Appendix 4 (Table 5).

4 Empirical Study

In this section, we analyze the baseline architecture across six DMC tasks: Cartpole Swingup, Cheetah Run, Finger Spin, Hopper Hop, Reacher Easy, and Walker Walk. Through our experiments, we show that the baseline adds the most minimal number of components to a base actor-critic setup while performing most robustly for the different tasks with distractor backgrounds. This highlights the importance of this baseline method, one that more sophisticated ideas can build over. We test the importance of having both the reward and transition modules individually, by removing each of them one by one.

4.1 Reward Prediction

Figure 3 (left) shows a comparison of 'with *vs* without reward prediction'. All other settings are kept unchanged and the only difference is the reward prediction. When the reward model is removed, there remains no grounding objective for the transition model. This results in a representation collapse as the transition model loss is minimized by the trivial representation which maps all observations to the same encoded state leading to degraded performance. This hints at the fact that without a valid grounding objective (in this case from predicting rewards), learning good representations can be very hard. Note that it is not the case that there is no reward information available to the agent, since learning the critic does provide enough signal to learn efficiently when there are no distractions present. However, in the presence of distractions the signal from the critic can be extremely noisy since it is based on the current value functions, which are not well developed in the initial stages of training. One potential fix for such a collapse is to not use the standard maximum likelihood based approaches for the transition model loss and instead use a contrastive version of the loss, which has been shown to learn general representations in the self-supervised learning setting. We test this later in the paper and conclude that although it does help prevent collapse, the performance is still heavily inferior to when we include the reward model. Complete performances for individual tasks are shown in Appendix 5.1.

Linear Reward Predictor. Furthermore, we also compare to the case when the reward decoder is a linear network instead of the standard 1 layer MLP. We see that performance decreases significantly in this case as shown in Figure 3 (middle) , but still does not collapse like in the absence of reward prediction. We hypothesize that the reward model is potentially removing useful information for predicting the optimal actions. Therefore, when it is attached directly to the encoded state, i.e., in the linear reward predictor case, it might force the representation to only preserve information required to predict the reward well, which might not be always be enough to predict the optimal actions well. For instance, consider a robot locomotion task. The reward in this case only depends on one variable,

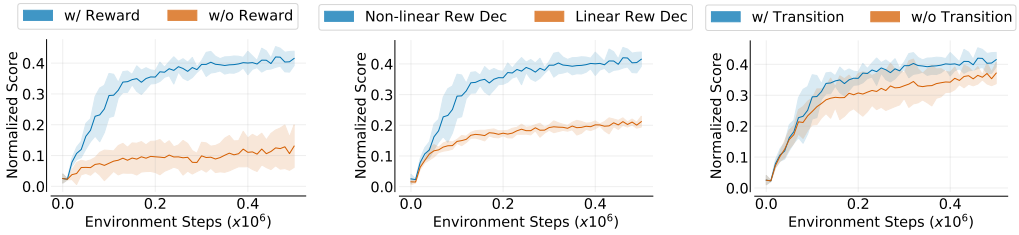

Figure 3: **Baseline Ablations**. Average normalized performance across six standard domains from DMC. Mean and std err. for 5 runs. **Left plot**: Baseline with *vs* without reward prediction **Middle plot**: Baseline with non-linear *vs* linear reward predictor/decoder. **Right plot**: Baseline with *vs* without transition prediction.

the center of mass, and thus the representation module would only need to preserve that in order to predict the reward well. However, to predict optimal actions, information about all the joint angular positions and velocities is required, which might be discarded if the reward model is directly attached to the encoded state. This idea is similar to why contrastive learning objectives in the self-supervised learning setting always enforce consistency between two positive/negative pairs *after* projecting the representation to another space. It has been shown that enforcing consistency in the representation space can remove excess information, which hampers final performance. We indeed see a similar trend in the RL case as well.

4.2 Transition Prediction

Similarly, Figure 3 (right) shows a comparison of 'with *vs* without transition prediction'. The transition model loss enforces temporal consistencies among the encoded states. When this module is removed, we observe a slight dip in performance across most tasks, with the most prominent drop in cartpole as shown in Appendix 5.1 (Figure 15). This suggests that enforcing such temporal consistencies in the representation space is indeed an important component for robust learning, but not a sufficient component. To examine if this slight dip is an artifact of the exact architecture used, we explore other architectures below.

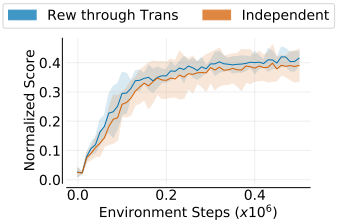

Figure 4: **Baseline reward through transition *vs* independent**. Average normalized performance across six standard domains from DMC.

Placement of Reward and Transition Modules. Having identified the importance of the reward and transition models, the remaining question is how to actually assemble these in the overall architecture. There are two workable possibilities for this, one being attaching both independently to the encoded state, i.e., $\mathcal{R}(s, a, s')$, and the other being attaching the transition model to the encoded state and then attaching the reward model to the output of the transition model, i.e., learning reward through transition $\mathcal{R}(s, a, \hat{s}')$. We test both of these and conclude that both perform equally well.

4.3 Connections to Value-Aware Learning

The baseline introduced above also resembles a prominent idea in theory, that of learning value aware models [16, 4]. Value-aware learning advocates for learning a model by fitting it to the value function of the task in hand, instead of fitting it to the true model of the world. The above baseline can be looked at as doing value aware learning in the following sense: the grounding to the representation is provided by the reward function, thus defining the components responsible for the task in hand and then the transition dynamics are learnt only for these components and not for all components in the observation space. There remains one crucial difference though. Value aware methods learn the dynamics based on the value function (multi-step) and not the reward function (1-step), since the value function captures the long term nature of the task in hand. To that end, we also test a more exact variant of the value-aware setup where we use the critic function as the target for optimizing the transition prediction, both with and without a reward prediction module (Table 1). Complete performances are provided in Appendix 5.5.

Table 1: **Truly value-aware objectives**. We report average final score after 500K steps across six standard domains from DMC.

	Baseline	Value-aware (w/ reward)	Value-aware (w/o reward)
Average Scores	0.42 ± 0.02	0.36 ± 0.03	0.23 ± 0.03

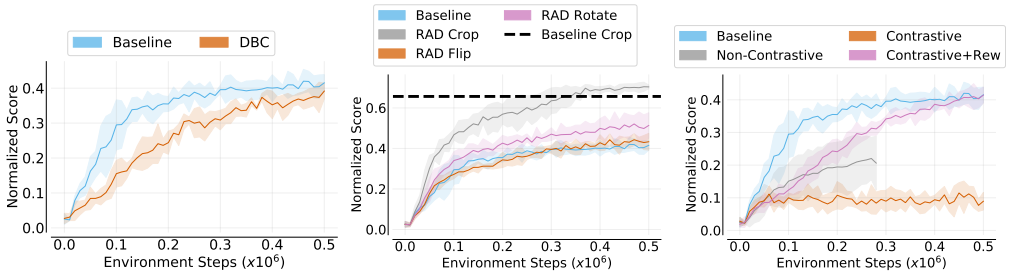

Figure 5: **Baseline Ablations**. Average normalized performance across six standard domains from DMC. Mean and std err. for 5 runs. **Left plot**: Baseline *vs* state metric losses (DBC [57]). The performance of baseline is compared with bisimulation metrics employed by DBC [3]. **Middle plot**: Data Augmentations. Cropping removes irrelevant segments while flip and rotate do not, performing similar to the baseline. Baseline with random crop performs equally good as RAD. **Right plot**: Contrastive and Non-Contrastive. The performance of baseline is compared with replacing transition loss with contrastive loss. Further, we consider simple contrastive (CURL [34]) and non-contrastive (variant of SPR [41] for DMC) as well.

5 Comparison

Until now, we have discussed why the two modules we identify as being vital for minimal and robust learning are actually necessary. Now we ask what other components could be added to this architecture which might improve performance, as has been done in prior methods. We then ask when do these added components actually improve performance, and when do they fail.

Metric Losses. Two recent works that are similar to the baseline above are DBC [57] and MiCO [10], both of which learn representations by obeying a distance metric. DBC learns the metric by estimating the reward and transition models while MiCO uses transition samples to directly compute the metric distance. We compare baseline's performance with DBC as shown in Figure 5 (left). Note that without metric, DBC is similar to the baseline barring architectural differences such as the use of probabilistic transition models in DBC compared to deterministic models in the baseline. Hence, we observe that the performance "without metric" exceeds that of "with metric".

Data Augmentations. A separate line of work has shown strong results when using data augmentations over the observation samples. These include the RAD [33] and DrQ [31] algorithms, both of which differ very minimally in their implementations. We run experiments for three different augmentations— 'crop', 'flip', and 'rotate'. The 'crop' augmentation always crops the image by some shifted margin from the center. Interestingly, the image of the robot is also always centered, thus allowing 'crop' to always only remove background or task-irrelevant information and never remove the robot or task-relevant information. This essentially amounts to not having background distractors and thus we see that this technique performs quite well as shown in Figure 5 (middle). However, augmentations that do not explicitly remove the distractors, such as rotate and flip, lead to similar performance as the baseline. This suggests that augmentations might not be helpful when distractor information cannot be removed, or when we do not know where the objects of interest lie in the image, something true of the real world. We test this by shifting the robot to the side, thus making the task-relevant components off-center and by zooming out i.e. increasing the amount of irrelevant information even after cropping. We see that performance of 'crop' drops drastically in this case, showcasing that most of the performance gains from augmentations can be attributed to how the data is collected and not to the algorithm itself. Additional ablations are provided in Appendix 5.2.

Table 2: **RAD additional ablations**. We report average final score after 500K steps across Cheetah Run and Walker Walk domains from DMC. This illustrates that the performance of augmentations is susceptible to quality of data. Also, for the "Zoomed Out" setting, it is worth noting that both *crop* and *flip* settle to the same score.

	Standard	Off-center	Zoomed Out
RAD Crop	0.34 ± 0.14	0.30 ± 0.08	0.23 ± 0.10
RAD Flip	0.27 ± 0.08	0.29 ± 0.07	0.23 ± 0.07

Contrastive and Non-contrastive Losses. A lot of recent methods also deploy different contrastive losses to learn representations, which essentially refers to computing positive/negative pairs and

[3] DBC [57] performance data is taken from their publication.

pushing/pulling them apart respectively. In practise, this can be done for any kind of loss function, such as the encoding function f_θ [24], or using random augmentations [34], so on and so forth. We test a simple modification, that of using the contrastive variant of the transition prediction loss than the maximum likelihood version. We see that, in Figure 5 (right), the contrastive version leads to inferior results than the baseline, again suggesting that contrastive learning might not add a lot of performance improvement, as has been witnessed in the self-supervised literature with methods like SIMSIAM [12], BARLOW TWINS [54], and BYOL [20] getting similar or better performance than contrastive methods like SIMCLR [11]. Complete performances are provided in Appendix 5.3.

Note that in the ALE results (Figure 8), SPR [41] leads to the best results overall, which also deploys a specific similarity loss for transition prediction motivated by BYOL [20]. We follow the same setup and test a variant of the baseline which uses the cosine similarity loss from SPR and test its performance on DMC based tasks. We again show in Figure 5 (right) that there is very little or no improvement in performance as compared to the baseline performance. This again suggests that the same algorithmic idea can have a completely different performance just by changing the evaluation setting[5] (ALE to DMC).

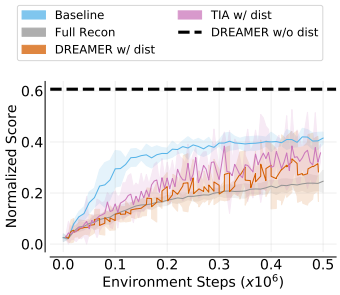

Figure 6: **Pixel reconstruction**. Average normalized performance across six DMC domains with distractors. Baseline achieves better performance than SOTA methods like DREAMER and TIA [4].

Learning World Models. We test DREAMER [24], a state of the art model-based method that learns world models though pixel reconstruction on two settings, with and without distractors. Although the performance in the "without distractors" case is good, we see that with distractors, DREAMER fails on some tasks, while performing inferior to the baseline in most tasks. This suggests that learning world models through reconstruction might only be a good idea when the world models are fairly simple to learn. If world models are hard to learn, as is the case with distractors, reconstruction based learning can lead to severe divergence that results in no learning at all. We also compare against the more recently introduced method from Fu et al. [18]. Their method, called TIA[18] incorporates several other modules in addition to DREAMER and learns a decoupling between the distractor background and task relevant components. We illustrate the performance of each of the above algorithms in Figure 6 along with a version where we add full reconstruction loss to the baseline. Complete performances are provided in Appendix 5.4.

Relevant Reconstruction and Sparse Rewards. Since we consider only dense reward based tasks, using the reward model to ground is sufficient to learn good representations. More sophisticated auxiliary tasks considered in past works include prediction of ensemble of value networks, prediction of past value functions, prediction of value functions of random cumulants, and observation reconstruction. However, in the sparse reward case, grounding on only the reward model or on past value functions can lead to representation collapse if the agent continues to receive zero reward for a long period of time. Therefore, in such cases where good exploration is necessary, tasks such as observation reconstruction can help prevent collapse. Although this has been shown to be an effective technique in the past, we argue that full reconstruction can harm the representations in the presence of distractors. Instead, we claim that reconstruction of *only* the task relevant components in the observation space results in learning good representations [18], especially when concerned with realistic settings like that of distractors. To that end, we show that in the sparse reward case, task-relevant reconstruction [6] is sufficient for robust performance. We show

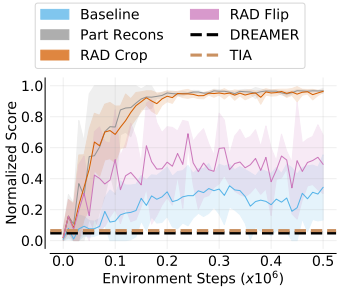

Figure 7: **Reconstruction and augmentations for sparse settings**. Normalized performance for ball-in-cup catch domain from DMC.

[4] TIA [18] performance data is taken from their publication.

[5] The SPR version without augmentations actually uses two separate ideas for improvement in performance, a cosine similarity transition prediction loss and a separate convolution encoder for the transition network, making it hard to attribute gains over the base DER [48] to just transition loss.

[6] *Part Recons.* in Figure 7 involves reconstructing the DMC robot over a solid black background.

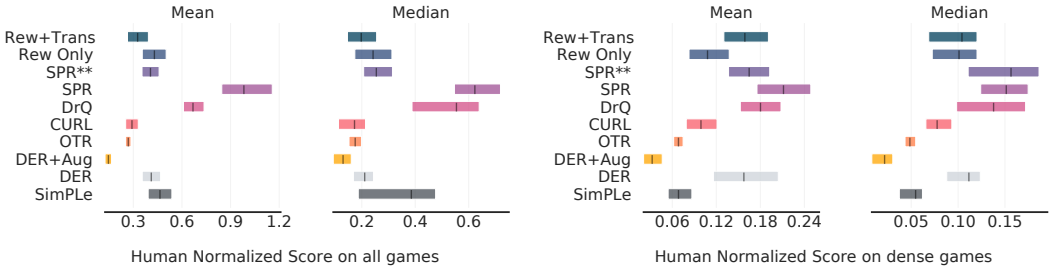

Figure 8: **Atari 100K**. Human normalized performance (mean/median) across 25 games from the Atari 100K benchmark. Mean and 95% confidence interval for 5 runs. **Left plot**: Comparison for all 25 games. **Right plot**: Comparison for only dense reward games (7 games from Table 6).

this in Figure 7 along with performance of baseline and augmentations. Of course, how one should come up with techniques that differentiate between task-relevant and task-irrelevant components in the observations, remains an open question [7]. Additional ablations are provided in Appendix 5.4.

Atari 100K. We study the effect of techniques discussed thus far for Atari 100K benchmark, which involves 26 Atari games and compares performance relative to human-achieved scores at 100K steps or 400K frames. We consider the categorization proposed by Bellemare et al. [6] based on the nature of reward (dense, human optimal, score exploit and sparse). We implemented two versions of the baseline algorithm, one with both the transition and reward prediction modules and the other with only reward prediction. Our average results over all games show that the baseline performs comparably to CURL [34], SimPLe [28], DER [48], and OTR [30] while being quite inferior to DrQ[8] [31, 2] and SPR [41]. In comparison to DER, since our implementation of the baseline is over the DER code, similar performance to DER might suggest that the reward and transition prediction do not help much in this benchmark. Note that the ALE does not involve the use of distractors and so learning directly from the RL head (DQN in this case) should be enough to encode information about the reward and the transition dynamics in the representation. This comes as a stark contrast to the without distractors case in DMC Suite, where transition and reward prediction still lead to better performance. Such differences might be attributed to the continuous *vs* discrete nature of DMC and ALE benchmarks. More surprisingly, we find that when plotting the same average performance results for only the dense reward environments in 100K, the gap in performance between DER and SPR/DrQ decreases drastically. Note that both SPR builds over DER but DrQ builds over OTR.

We further delve into understanding the superior performance of SPR and DrQ. In particular, SPR combines a non-contrastive transition prediction loss and data augmentations. To understand the effect of each of these individually, we run SPR without data augmentations, and call it SPR**[9]. We see that SPR** leads to performance similar to the baseline and the DER agent, suggesting that a non-contrastive loss does not lead to gains when run without data augmentations. Finally, we take the DER agent and add data augmentations to it (from DrQ). This is shown as DER + Aug in Figure 8. We see that this leads to collapse, with the worst performance across all algorithms. Note that DrQ builds over OTR and performs quite well whereas when the same augmentations are used with DER, which includes a distributional agent in it, we observe a collapse. This again indicates that augmentations can change data in a fragile manner, sometimes leading to enhanced performance with certain algorithms, while failing with other algorithms. Segregating evaluation of algorithms based on these differences is therefore of utmost importance. We show the individual performance on all 25 games in Appendix 5.3 (Table 6).

6 Discussion

The above description of results on Atari 100K point to a very interesting observation, that evaluation of different algorithms is very much correlated with a finer categorization of the evaluation benchmark,

[7] As also evident by TIA's [18] performance for DMC ball-in-cup catch experiments.

[8] We use the DrQ(ϵ) version from Agarwal et al. [2] for fair evaluation and denote it as DrQ.

[9] Note that this is different from the SPR without augmentations version reported in Schwarzer et al. [41] since that version uses dropout as well which is not a fair comparison.

Table 3: **Dataset Categorizations**. Dense+ refers to games under the 'Human Optimal' and 'Score Exploit' category from the categorization provided in [6] and Table 6. The performance of baseline, state metric, constrastive, non-contrastive and reconstruction in each of the categories are shown in Figure 1. The algorithm corresponding to each method is shown in Appendix 3 (Table 4). Also, since augmentations only modify the data, they should be included in the evaluation categorizations.

	C1	C2	C3	C4	C5	C6	C7	C8
Environment	ALE	ALE	ALE	DMC	DMC	DMC	DMC	DMC
Reward Structure	Dense	Dense	Dense+	Dense	Dense	Dense	Sparse	Dense
Distractor	No	No	No	Yes	Yes	Yes	Yes	No
Augmentations	None	Shift, Intensity	None	None	Crop	Flip	None	None

and not the whole benchmark itself. Specifically, focusing on finer categorizations such as density of reward, inherent horizon of the problem, presence of irrelevant and relevant task components, discreteness *vs* continuity of actions etc. is vital in recognizing if certain algorithms are indeed *better* than others. Figure 1 stands as a clear example of such discrepancies. These observations pave the way for a better evaluation protocol for algorithms, one where we rank algorithms for different categories, each governed by a specific data-centric property of the evaluation benchmark. Instead of saying that algorithm X is better than algorithm Y in benchmark Z, a better evaluation methodology is to say that algorithm X is better than algorithm Y in dense reward, short horizon problems (considered from benchmark Z), i.e. less emphasis on the benchmark itself and more on certain properties of a subset of the benchmark. Having started with the question of what matters when learning representations over pixels, our experiments and discussion clearly show that largely it is the data-centric properties of the evaluation problems that matter the most. Moreover, the baseline presented in the paper does indeed turn out to be robust to many changes in these properties, while a lot of other methods seem brittle.

7 Conclusion

In this paper we explore what components in representation learning methods matter the most for robust performance. We focused on the DMC Suite with distractors as the main setting while also extending our observations to DMC Suite without distractors and the Atari 100k benchmark. Our results show that a much simpler baseline, one involving a reward and transition prediction modules can be attributed to much of the benefits in DMC Suite with distractors. We then analysed why and when existing methods fail to perform as good or better than the baseline, also touching on similar observations on the ALE simulator. Some of our most interesting findings are as follows:

- Pixel reconstruction is a sound technique in the absence of clutter in the pixels, but suffer massively when distractors are added.
- Contrastive losses in and of itself does not provide gains when there is a supervised loss available in place of it.
- Certain augmentations ('crop') do well when data is centered while dropping in performance when data is off-center or when cropping fails to remove considerable amounts of task-irrelevant information. Other augmentations ('flip' and 'rotate') show the opposite behavior.
- Non-contrastive losses do not provide much gains when used alone. With data augmentations, they lead to more significant gains.
- Augmentations are susceptible to collapse in the presence of distributional Q networks.

These results elicit the observation that claiming dominance over other methods for an entire benchmark may not be an informative evaluation methodology. Instead, focusing the discussion to a more data-centric view, one where specific properties of the environment are considered, forms the basis of a much more informative evaluation methodology. We hope this work can spur further discussion in categorizing evaluation domains on a finer scale and then associating progress in state of the art specifically for each category. We believe such an evaluation protocol would be much more useful in driving the RL community to successful real-world application.

REFERENCES

[1] Agarwal, R., Machado, M. C., Castro, P. S., and Bellemare, M. G. Contrastive behavioral similarity embeddings for generalization in reinforcement learning. In *International Conference on Learning Representations*, 2021.

[2] Agarwal, R., Schwarzer, M., Castro, P. S., Courville, A., and Bellemare, M. G. Deep reinforcement learning at the edge of the statistical precipice. *arXiv preprint arXiv:2108.13264*, 2021.

[3] Andrychowicz, M., Raichuk, A., Stańczyk, P., Orsini, M., Girgin, S., Marinier, R., Hussenot, L., Geist, M., Pietquin, O., Michalski, M., et al. What matters in on-policy reinforcement learning? a large-scale empirical study. *arXiv preprint arXiv:2006.05990*, 2020.

[4] Ayoub, A., Jia, Z., Szepesvari, C., Wang, M., and Yang, L. Model-based reinforcement learning with value-targeted regression. In *International Conference on Machine Learning*, pp. 463–474. PMLR, 2020.

[5] Babaeizadeh, M., Saffar, M. T., Hafner, D., Erhan, D., Kannan, H., Finn, C., and Levine, S. On trade-offs of image prediction in visual model-based reinforcement learning. 2020.

[6] Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29:1471–1479, 2016.

[7] Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47: 253–279, 2013.

[8] Biza, O., Platt, R., van de Meent, J.-W., and Wong, L. L. Learning discrete state abstractions with deep variational inference. *arXiv preprint arXiv:2003.04300*, 2020.

[9] Castro, P. S. Scalable methods for computing state similarity in deterministic markov decision processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 10069–10076, 2020.

[10] Castro, P. S., Kastner, T., Panangaden, P., and Rowland, M. Mico: Learning improved representations via sampling-based state similarity for markov decision processes. *CoRR*, abs/2106.08229, 2021. URL https://arxiv.org/abs/2106.08229.

[11] Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

[12] Chen, X. and He, K. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15750–15758, 2021.

[13] Co-Reyes, J. D., Sanjeev, S., Berseth, G., Gupta, A., and Levine, S. Ecological reinforcement learning. *arXiv preprint arXiv:2006.12478*, 2020.

[14] Du, S., Krishnamurthy, A., Jiang, N., Agarwal, A., Dudik, M., and Langford, J. Provably efficient rl with rich observations via latent state decoding. In *International Conference on Machine Learning*, pp. 1665–1674. PMLR, 2019.

[15] Engstrom, L., Ilyas, A., Santurkar, S., Tsipras, D., Janoos, F., Rudolph, L., and Madry, A. Implementation matters in deep rl: A case study on ppo and trpo. In *International conference on learning representations*, 2019.

[16] Farahmand, A.-m., Barreto, A., and Nikovski, D. Value-aware loss function for model-based reinforcement learning. In *Artificial Intelligence and Statistics*, pp. 1486–1494. PMLR, 2017.

[17] Ferns, N., Panangaden, P., and Precup, D. Bisimulation metrics for continuous markov decision processes. *SIAM J. Comput.*, 40(6):1662–1714, December 2011. ISSN 0097-5397. doi: 10.1137/10080484X. URL https://doi.org/10.1137/10080484X.

[18] Fu, X., Yang, G., Agrawal, P., and Jaakkola, T. Learning task informed abstractions. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 3480–3491. PMLR, 18–24 Jul 2021. URL http://proceedings.mlr.press/v139/fu21b.html.

[19] Gelada, C., Kumar, S., Buckman, J., Nachum, O., and Bellemare, M. G. DeepMDP: Learning continuous latent space models for representation learning. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2170–2179. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/gelada19a.html.

[20] Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., Doersch, C., Pires, B. A., Guo, Z. D., Azar, M. G., et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020.

[21] Ha, D. and Schmidhuber, J. World models. *arXiv preprint arXiv:1803.10122*, 2018.

[22] Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/haarnoja18b.html.

[23] Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, pp. 2555–2565. PMLR, 2019.

[24] Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1lOTC4tDS.

[25] Hafner, D., Lillicrap, T. P., Norouzi, M., and Ba, J. Mastering atari with discrete world models. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=0oabwyZbOu.

[26] Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., and Meger, D. Deep reinforcement learning that matters. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[27] Jaderberg, M., Mnih, V., Czarnecki, W. M., Schaul, T., Leibo, J. Z., Silver, D., and Kavukcuoglu, K. Reinforcement learning with unsupervised auxiliary tasks. *arXiv preprint arXiv:1611.05397*, 2016.

[28] Kaiser, L., Babaeizadeh, M., Milos, P., Osinski, B., Campbell, R. H., Czechowski, K., Erhan, D., Finn, C., Kozakowski, P., Levine, S., et al. Model-based reinforcement learning for atari. *arXiv preprint arXiv:1903.00374*, 2019.

[29] Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[30] Kielak, K. Importance of using appropriate baselines for evaluation of data-efficiency in deep reinforcement learning for atari. *arXiv preprint arXiv:2003.10181*, 2020.

[31] Kostrikov, I., Yarats, D., and Fergus, R. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *CoRR*, abs/2004.13649, 2020. URL https://arxiv.org/abs/2004.13649.

[32] Krishnamurthy, A., Agarwal, A., and Langford, J. Pac reinforcement learning with rich observations. *arXiv preprint arXiv:1602.02722*, 2016.

[33] Laskin, M., Lee, K., Stooke, A., Pinto, L., Abbeel, P., and Srinivas, A. Reinforcement learning with augmented data. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 19884–19895. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/e615c82aba461681ade82da2da38004a-Paper.pdf.

[34] Laskin, M., Srinivas, A., and Abbeel, P. Curl: Contrastive unsupervised representations for reinforcement learning. *Proceedings of the 37th International Conference on Machine Learning, Vienna, Austria, PMLR 119*, 2020. arXiv:2004.04136.

[35] Lee, A. X., Nagabandi, A., Abbeel, P., and Levine, S. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 741–752. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/08058bf500242562c0d031ff830ad094-Paper.pdf.

[36] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[37] Oh, J., Guo, X., Lee, H., Lewis, R., and Singh, S. Action-conditional video prediction using deep networks in atari games. *arXiv preprint arXiv:1507.08750*, 2015.

[38] Oord, A. v. d., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[39] Schaul, T., Horgan, D., Gregor, K., and Silver, D. Universal value function approximators. In *International conference on machine learning*, pp. 1312–1320. PMLR, 2015.

[40] Schmidhuber, J. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE Transactions on Autonomous Mental Development*, 2(3):230–247, 2010.

[41] Schwarzer, M., Anand, A., Goel, R., Hjelm, R. D., Courville, A., and Bachman, P. Data-efficient reinforcement learning with self-predictive representations. *arXiv preprint arXiv:2007.05929*, 2020.

[42] Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

[43] Stone, A., Ramirez, O., Konolige, K., and Jonschkowski, R. The distracting control suite–a challenging benchmark for reinforcement learning from pixels. *arXiv preprint arXiv:2101.02722*, 2021.

[44] Stooke, A., Lee, K., Abbeel, P., and Laskin, M. Decoupling representation learning from reinforcement learning. In *International Conference on Machine Learning*, pp. 9870–9879. PMLR, 2021.

[45] Sutton, R. S., Modayil, J., Delp, M., Degris, T., Pilarski, P. M., White, A., and Precup, D. Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pp. 761–768, 2011.

[46] Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., Casas, D. d. L., Budden, D., Abdolmaleki, A., Merel, J., Lefrancq, A., et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

[47] Tomar, M., Zhang, A., Calandra, R., Taylor, M. E., and Pineau, J. Model-invariant state abstractions for model-based reinforcement learning. *arXiv preprint arXiv:2102.09850*, 2021.

[48] van Hasselt, H. P., Hessel, M., and Aslanides, J. When to use parametric models in reinforcement learning? In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/1b742ae215adf18b75449c6e272fd92d-Paper.pdf.

[49] Wahlström, N., Schön, T. B., and Deisenroth, M. P. From pixels to torques: Policy learning with deep dynamical models. *arXiv preprint arXiv:1502.02251*, 2015.

[50] Watter, M., Springenberg, J. T., Boedecker, J., and Riedmiller, M. Embed to control: A locally linear latent dynamics model for control from raw images. *arXiv preprint arXiv:1506.07365*, 2015.

[51] Yarats, D., Zhang, A., Kostrikov, I., Amos, B., Pineau, J., and Fergus, R. Improving sample efficiency in model-free reinforcement learning from images. *arXiv preprint arXiv:1910.01741*, 2019.

[52] Yarats, D., Fergus, R., Lazaric, A., and Pinto, L. Reinforcement learning with prototypical representations. *arXiv preprint arXiv:2102.11271*, 2021.

[53] Yarats, D., Kostrikov, I., and Fergus, R. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=GY6-6sTvGaf.

[54] Zbontar, J., Jing, L., Misra, I., LeCun, Y., and Deny, S. Barlow twins: Self-supervised learning via redundancy reduction. *arXiv preprint arXiv:2103.03230*, 2021.

[55] Zhang, A., Wu, Y., and Pineau, J. Natural environment benchmarks for reinforcement learning. *arXiv preprint arXiv:1811.06032*, 2018.

[56] Zhang, A., Lyle, C., Sodhani, S., Filos, A., Kwiatkowska, M., Pineau, J., Gal, Y., and Precup, D. Invariant causal prediction for block MDPs. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 11214–11224. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/zhang20t.html.

[57] Zhang, A., McAllister, R., Calandra, R., Gal, Y., and Levine, S. Learning Invariant Representations for Reinforcement Learning without Reconstruction. *arXiv*, jun 2020. ISSN 23318422. URL http://arxiv.org/abs/2006.10742.