# TASK-INDUCED REPRESENTATION LEARNING

**Jun Yamada**[1]*, **Karl Pertsch**[2], **Anisha Gunjal**[2], **Joseph J. Lim**[2]
[1] University of Oxford, [2] University of Southern California

## ABSTRACT

A major bottleneck for applying deep reinforcement learning to real-world problems is its sample inefficiency, particularly when training policies from high-dimensional inputs such as images. A number of recent works use unsupervised representation learning approaches to improve sample efficiency. Yet, such unsupervised approaches are fundamentally unable to distinguish between *task-relevant* and *irrelevant* information. Thus, in visually complex scenes they learn representations that model lots of task-irrelevant details and hence lead to slower downstream task learning. Our insight: to determine which parts of the scene are important and should be modeled, we can exploit task information, such as rewards or demonstrations, from previous tasks. To this end, we formalize the problem of *task-induced* representation learning (TARP), which aims to leverage such task information in offline experience from prior tasks for learning compact representations that focus on modelling only task-relevant aspects. Through a series of experiments in visually complex environments we compare different approaches for leveraging task information within the TARP framework with prior unsupervised representation learning techniques and (1) find that task-induced representations allow for more sample efficient learning of unseen tasks and (2) formulate a set of best-practices for task-induced representation learning.

## 1 INTRODUCTION

A central capability of intelligent agents acting in complex environments is to filter their sensory inputs and retrieve the important bits of information. Humans are remarkably efficient at this: while hundreds of MB of raw pixel data enter our retina in every second, only roughly 40 bits/sec remain after passing the visual processing system (Zhaoping, 2006). This highly filtered and compressed representation of reality allows for more efficient learning and acting in the complex environments of our everyday lives. Similarly, our goal is to efficiently train *artificial* agents in complex environments by learning which parts of their sensory inputs should be modeled and which can be discarded.

Deep reinforcement learning (RL) replaces manually engineered perception pipelines with policies that are trained directly from raw visual inputs (Mnih et al., 2015; Levine et al., 2016), but requires a prohibitive amount of environment interactions for learning a good representation. Thus, *unsupervised* objectives for representation learning have been proposed (Lange and Riedmiller, 2010; Finn et al., 2016; Hafner et al., 2019; Laskin et al., 2020; Stooke et al., 2021). While substantially more efficient, these approaches are typically only evaluated in clean laboratory or simulated settings (Finn et al., 2016; Lee et al., 2020; Laskin et al., 2020) where most of the perceived information is important for the task at hand and therefore should be modeled.

In contrast, realistic environments feature lots of task-irrelevant detail, like clutter or objects moving in the background. Trying to capture all the details in the representation will make both representation learning and downstream task learning inefficient or even infeasible. Thus, we need mechanisms for filtering task-irrelevant details when learning representations in visually complex environments.

Fundamentally however, unsupervised representation learning techniques cannot address the filtering problem in such environments: they do not discriminate between task-relevant and irrelevant information, but instead are trained to model *all* perceived information. Without additional supervision, for them, modelling the movement of tree leaves on the road side might be as important as capturing an approaching car in front of the agent.

*Correspondence to: jyamada@robots.ox.ac.uk

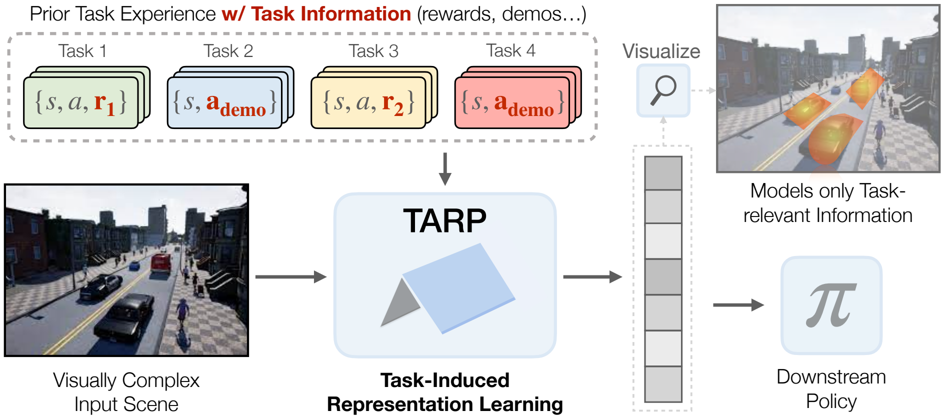

Prior Task Experience w/ Task Information (rewards, demos…)
Task 1
$\{s, a, \mathbf{r_1}\}$
Task 2
$\{s, \mathbf{a_{demo}}\}$
Task 3
$\{s, a, \mathbf{r_2}\}$
Task 4
$\{s, \mathbf{a_{demo}}\}$
Visualize
Models only Task-relevant Information
TARP
Visually Complex Input Scene
Task-Induced Representation Learning
$\pi$
Downstream Policy

Figure 1: Overview of the task-induced representation learning (TARP pipeline). By exploiting task information such as rewards or demonstrations from prior tasks, task-induced representation learning approaches learn representations of visually complex scenes that model only the task-relevant aspects and ignore irrelevant details. These representations can be transferred to efficiently learn policies on unseen downstream tasks.

Our core insight is that we can only decide which information to retain and which to filter by taking *task information* into account. In the example above, humans rely on a lifetime of experience solving roadside tasks to understand that the approaching car is more relevant than the tree leaves. To this end, we formalize the problem of **task-induced representation learning** (**TARP**, see Figure 1) which aims to leverage task-related information, such as rewards or demonstrations from prior tasks, for learning representations that focus on the task-relevant information in the observations. We analyze different approaches for leveraging task information within the TARP framework and show that the learned compact, task-driven representations accelerate the training of *unseen* tasks in domains of real-world visual complexity.

The contributions of our work are threefold: (1) we empirically show that common unsupervised representation learning techniques struggle at learning tasks in visually challenging environments since they cannot distinguish between relevant and irrelevant information, (2)we investigate the role of task information in representation learning by formalizing task-induced representation learning (TARP) that leverage task information from prior tasks and (3) we compare different approaches for leveraging such task information within the TARP framework with prior unsupervised representation learning techniques in three complex visual domains: distracting DMControl, ViZDoom, and CARLA. We find that task-induced representations allow for more sample efficient learning of unseen tasks and formulate a set of best-practices for task-induced representation learning.

## 2 Related Work

To enable agents to act from high-dimensional observations, classic approaches have relied on **engineered perception pipelines** that process the input into a manually defined "state" representation. This has allowed remarkable progress in challenging real-world domains like autonomous driving (Urmson et al., 2008) or humanoid robotics (Atkeson et al., 2015), but manually defining such perception interfaces for each task is costly and requires substantial human effort.

Instead, deep RL has allowed the training of **"end-to-end" policies** that directly map raw visual inputs to action commands, *without* the need for human-engineered perception modules (Mnih et al., 2015; Kalashnikov et al., 2018). Due to the need for large environment interactions, deep RL approaches are typically evaluated in clean lab settings with uncluttered input observations that allow for easier learning (Finn et al., 2016; Pathak et al., 2018; Zhang et al., 2019), with a few notable exceptions such as Gupta et al. (2018) and Kahn et al. (2021) that rely on large-scale data collection. In addition to sample inefficiency, representations learned through end-to-end task training are specific to the training task and cannot be reused for solving different downstream tasks. In contrast, our goal is to learn representations that enable sample efficient learning on *unseen* tasks.

To improve data efficiency in deep RL from high-dimensional observations, a number of recent works have taken inspiration from advances in computer vision and have explored using data augmentation (Yarats et al., 2021; Laskin et al., 2021) and **unsupervised representation learning** techniques during RL training. The latter can be categorized into (1) reconstruction (Lange and Riedmiller, 2010; Finn et al., 2016), (2) prediction (Hafner et al., 2019; Lee et al., 2020), and (3) contrastive learning (Laskin et al., 2020; Stooke et al., 2021; Zhan et al., 2020; Yang and Nachum, 2021) approaches. While these works have shown improved sample efficiency, fundamentally they are *unsupervised* and therefore cannot decide which information in the input is relevant or irrelevant to a certain task. We show that this leads to deteriorating performance in visually complex environments such as autonomous driving scenarios with lots of non-relevant information in the input observations. Task-induced representation learning addresses this problem by leveraging prior task experience to infer which input information is "interesting", allowing it to learn representations that enable sample efficient learning of new tasks in scenes of real-world complexity.

Closest to our work are recent approaches that explore reconstruction-free representation learning (Gelada et al., 2019; Zhang et al., 2021). Specifically, Zhang et al. (2021) use a bisimulation objective to learn a reward-induced representation which can ignore distractors. While Zhang et al. (2021) focus on single-task, online representation learning, we aim to leverage diverse offline datasets, collected across multiple tasks, which can enable efficient reuse of previously collected data from a variety of sources. Further, we compare multiple objectives for task-induced representation learning beyond bisimulation that can leverage diverse forms of task supervision.

## 3 Problem Formulation

Our goal is to efficiently learn a policy $\pi$ that solves a target task $T_{\text{target}}$ in an MDP defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, R, \rho, \gamma)$ of states, actions, transition probabilities, rewards, initial state distribution, and discount factor. The policy is trained to maximize the discounted return $\mathbb{E}_{a_t \sim \pi}\big[\sum_{t=0}^{T-1} \gamma^t R(s_t, a_t)\big]$.

We do not assume access to the underlying state $s$ of the MDP, but instead our policy receives a high-dimensional observation $x \in \mathcal{X}$ in every step, e.g., an image observation. To improve training efficiency, we aim to learn an encoder $\phi(x)$ which maps the input observation to a low-dimensional *state representation* that is input to the policy $\pi(\phi(x))$. To learn this representation, we assume access to a dataset of past interactions $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_N\}$ collected across $T_{1:N} \in \mathcal{T}$ tasks, with per-task datasets $\mathcal{D}_i = \{x_t, a_t, (r_t), ...\}$ of state-action trajectories and optional reward annotation. Crucially, the set of training tasks does not include the target task $T_{\text{target}} \notin T_{1:N}$. We require the training data to include task information for prior tasks, e.g., in the form of step-wise reward annotations $r_t$ or by consisting of task demonstrations.

We follow Stooke et al. (2021) and test all representation learning approaches in two phases: we first train the representation encoder $\phi(x)$ from the offline dataset $\mathcal{D}$, then freeze its parameters and optimize the policy $\pi(\phi(x))$ on the downstream task. This two-stage experimental protocol has two benefits: (1) by training representations solely from *offline* experience we are able to efficiently reuse data collected across prior tasks, e.g., from prior training runs (Fu et al., 2020; Çaglar Gülçehre et al., 2020) or human teleoperation (Mandlekar et al., 2018; Cabi et al., 2019), and (2) by freezing the encoder weights after pre-training we can evaluate the quality of the pre-trained representation in isolation. All tested approaches can be combined with finetuning of the learned representation on the downstream task to further improve performance, but we leave this investigation for future work.

## 4 Task-Induced Representation Learning

Our core idea is to use task information from prior tasks as a filter for which aspects of the environment are interesting, and which do not need to be modeled. This addresses a fundamental problem of the unsupervised representation learning approaches: by maximizing the mutual information between observations and representation they are trained to model every bit of information equally and thus struggle in visually complex environments with lots of irrelevant details.

Formally, the state of an environment $\mathcal{S}$ can be divided into task-relevant and task-irrelevant or nuisance components $\mathcal{S} = \{\mathcal{S}^i_{\text{task}}, \mathcal{S}^i_n\}$. The superscript $i$ indicates that this division is *task-dependent*, since some components of the state space are relevant for a particular task $T_i$ but irrelevant for others.

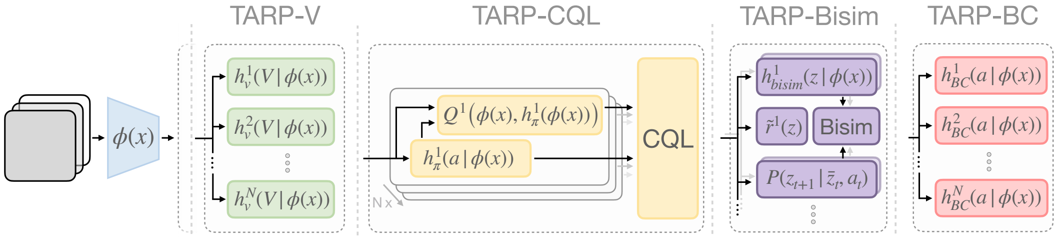


Figure 2: Instantiations of our task-induced representation learning framework. **Left to right**: Representation learning via multi-task value prediction (**TARP-V**), via multi-task offline RL (**TARP-CQL**), via bisimulation (**TARP-Bisim**) and via multi-task imitation learning (**right**, **TARP-BC**).

In visually complex environments, we can assume that $|\mathcal{S}_{\text{task}}| \ll |\mathcal{S}|$, i.e., that the task-relevant part of the state space is much smaller than all state information in the input observations (Zhaoping, 2006).

When choosing how much of this input information to model, end-to-end policy learning and unsupervised representation learning, form two ends of a spectrum. End-to-end policy learning only models $\mathcal{S}^i_{\text{task}}$ for its training task, which is efficient, but only allows transfer from task $i$ to task $j$ if $\mathcal{S}^j_{\text{task}} \subseteq \mathcal{S}^i_{\text{task}}$, i.e., if the task-relevant components of the target task are a subset of those of the training task. In contrast, unsupervised representation learning models the full $\mathcal{S} = \{\mathcal{S}_{\text{task}}, \mathcal{S}_n\}$, which allows for flexible transfer since $\mathcal{S}^j_{\text{task}} \subseteq \mathcal{S} \; \forall \; T_j \in \mathcal{T}$, but training can be inefficient since the learned representation contains many nuisance variables.

In this work, we formalize the problem of task-induced representation learning (TARP), which aims to combine the best of both approaches. Similar to end-to-end policy learning, task-induced representation learning approaches use task information to learn compact, task-relevant representations. Yet, by combining the task information from a wide range of prior tasks they learn a representation that *combines* the task-relevant components of all tasks in the training dataset $\mathcal{D}$: $\mathcal{S}_{\text{TARP}} = \mathcal{S}^1_{\text{task}} \cup \cdots \cup \mathcal{S}^N_{\text{task}}$. Thus, such a representation allows for transfer to a wide range of *unseen* tasks for which $\mathcal{S}^{\text{target}}_{\text{task}} \subseteq \mathcal{S}_{\text{TARP}}$. Yet, the representation filters a large part of the state space which is not relevant for *any* of the training tasks, allowing sample efficient learning on the target task.

### 4.1 Approaches for Task-Induced Representation Learning

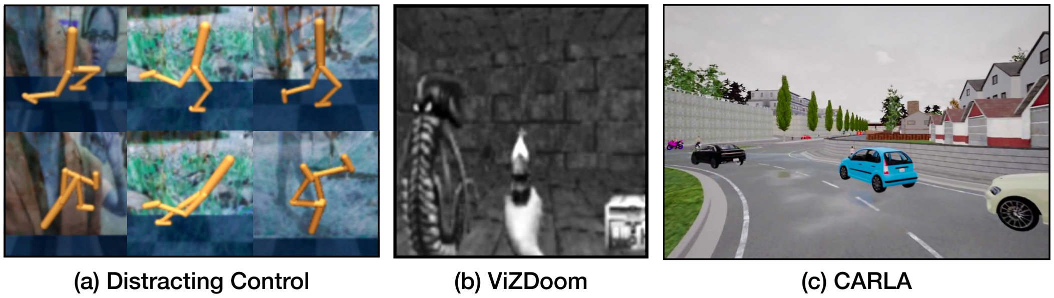
We compare multiple approaches for task-induced representation learning which can leverage or even mix different forms of task supervision. In this work we focus on two forms, reward annotations and task demonstrations; but, future work can extend the TARP framework to other forms of task supervision like language commands or human preferences. All instantiations of TARP train a shared encoder network $\phi(x)$ with separate task-supervision heads (see Figure 2).

**Value Prediction (TARP-V).** We can leverage reward annotations as task supervision by estimating the future discounted return of the data collection policy. Intuitively, a representation that allows estimation of the value of a state needs to include all task-relevant aspects of this state. We introduce separate value prediction heads $h^i_v$ for each task $i$ and train the representation $\phi(x)$ by minimizing the error in the predicted discounted return:

$$\mathcal{L}_{\text{TARP-V}} = \sum_{i=1}^{N} \mathbb{E}_{(x_t, r_t)\sim\mathcal{D}_i} \left( h^i_v(\phi(x_t)) - \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'} \right)^2. \tag{1}$$

**Offline RL (TARP-CQL).** Alternatively, we can learn a representation by training a policy to directly maximize the discounted reward of the training tasks. Since we aim to use offline training data only, we leverage recently proposed methods for offline RL (Levine et al., 2020; Kumar et al., 2020) and introduce separate policy heads $h^i_\pi$ and critics $Q^i$ for each task. Following Kumar et al. (2020), we train the policy to maximize the entropy-augmented future return:

$$\mathcal{L}_{\text{TARP-CQL}} = -\sum_{i=1}^{N} \mathbb{E}_{x\sim\mathcal{D}_i} \Big( Q^i(x, h^i_\pi(\phi(x))) + \alpha\mathcal{H}\big(h^i_\pi(\phi(x))\big) \Big). \tag{2}$$



(a) Distracting Control

(b) ViZDoom

(c) CARLA

Figure 3: Environments with high visual complexity and substantial task-irrelevant detail for testing the learned representations. (a) **Distracting DMControl** (Stone et al., 2021) with randomly overlayed videos, (b) **ViZDoom** (Wydmuch et al., 2018) with diverse textures and enemy appearances, and (c) **CARLA** (Dosovitskiy et al., 2017) with realistic outdoor driving scenes and weather simulation.

Here $\mathcal{H}(\cdot)$ denotes the entropy of the policy's output distribution and $\alpha$ is a learned weighting factor (Kumar et al., 2020; Haarnoja et al., 2018b).

**Bisimulation (TARP-Bisim).** We use a bisimulation objective for reward-induced representation learning (Larsen and Skou, 1991; Ferns et al., 2011). It groups states based on their "behavioral similarity", measured as their expected future returns under arbitrary action sequences. Specifically, we build on the approach of Zhang et al. (2021) that leverages deep neural networks to estimate the bisimulation distance, and modify it for multi-task learning by adding per-task embedding heads $z^i = h^i_{\text{bisim}}\big(\phi(x)\big)$. The representation learning objective is:

$$\mathcal{L}_{\text{TARP-Bisim}} = \sum_{i=1}^{N} \mathbb{E}_{\substack{(x_j, a_j, r_j) \\ (x_k, a_k, r_k)} \sim \mathcal{D}_i} \left( |z^i_j - z^i_k| - |r_j - r_k| - \gamma \mathcal{W}_2\big(P(\cdot|\bar{z}^i_{j,t}, a_{j,t}), P(\cdot|\bar{z}^i_{k,t}, a_{k,t})\big) \right)^2 \quad (3)$$

Here $P(\cdot|z,a)$ is a learned latent transition model and $\mathcal{W}_2$ refers to the 2-Wasserstein metric which we can compute in closed form for Gaussian transition models. Following Zhang et al. (2021) we use a target encoder updated with the moving average of the encoder's weights for producing $\bar{z}$ and we add an auxiliary reward prediction objective with per-task reward predictors $\tilde{r}^i_{\text{bisim}}(z)$.

**Imitation Learning (TARP-BC).** We can also train task-induced representations from data without reward annotation by directly imitating the data collection policy, thus learning to represent all elements of the state space that were important for the policy's decision making. We choose behavioral cloning (BC, Pomerleau (1989)) for imitation learning since it is easily applicable especially in the offline setting. We introduce $N$ separate imitation policy heads $h^i_{\text{BC}}$ and minimize the negative log-likelihood of the training data's actions:

$$\mathcal{L}_{\text{TARP-BC}} = -\sum_{i=1}^{N} \mathbb{E}_{(x,a)\sim\mathcal{D}_i} \left( \log h^i_{\text{BC}}(a|\phi(x)) \right) \quad (4)$$

# 5 Experiments

## 5.1 Experimental Setup

We compare the performance of different representation learning approaches in three visually complex environments (see Figure 3). We will briefly describe each environment; for more details on environment setup and data collection, see appendix, Section A.

**Distracting DMControl.** We use the environment of Stone et al. (2021), in which the visual complexity of the standard DMControl "Walker" task (Tassa et al., 2018) is increased by overlaying randomly sampled natural videos from the DAVIS 2017 dataset (Pont-Tuset et al., 2017). We train on data collected from standing, forward and backward walking policies and test on a downstream running task. Task-induced representations should focus on modeling the agent while ignoring irrelevant information from the background video.

**ViZDoom.** A simulator for the ego-shooter game Doom (Wydmuch et al., 2018). We use pre-trained models from Dosovitskiy and Koltun (2017) for data collection and vary their objectives to get diverse behaviors such as maximizing the number of collected medi-packs or minimizing the loss of health points. Learned representations should focus on important aspects such as the location of enemies or medi-packs, while ignoring irrelevant details such as the texture of walls or appearance features of medi-packs etc. We test on the full "battle" task from Dosovitskiy and Koltun (2017).

**Autonomous Driving.** We simulate realistic first-person driving scenarios using the CARLA simulator (Dosovitskiy et al., 2017). We collect training data from intersection crossings as well as right and left turns using pre-trained policies and test on a long-range point-to-point driving task that requires navigating an unseen part of the environment. For efficient learning, representations need to model relevant driving information such as position and velocity of other cars, while ignoring irrelevant details such as trees, shadows or the color and make of cars in the scene.

We compare the different instantiations of the task-induced representation learning framework to prior representation learning approaches:

- **Reconstruction**: Trains a beta-**VAE** (Higgins et al., 2017) or stochastic video prediction model ("**Pred-O**") on the training data and transfers the encoder.
- **Reconstruction + Task-Induced**: Combines video prediction with reward prediction for pre-training ("**Pred-R+O**"), similar to Hafner et al. (2019).
- **Contrastive Learning**: Uses the contrastive learning objective from Stooke et al. (2021) for pre-training ("**ATC**").

All approaches are trained on the same offline dataset $\mathcal{D}$ used for training of the TARP approaches. We also compare to a **policy transfer** baseline which pre-trains a policy on $\mathcal{D}$ using BC and then finetunes the full policy on the downstream task as an alternative to representation transfer[1]. In distracting DMControl we further report results for an **oracle** baseline whose representation is trained with direct supervision from the low-dimensional state representation[2].

After pre-training of the representation, we transfer the frozen encoder weights to the target task policy, which we train with soft actor-critic (SAC, Haarnoja et al. (2018a)) on continuous control tasks (distracting DMControl, CARLA), and with PPO (Schulman et al., 2017) on discrete action tasks (ViZDoom). For more implementation details and hyperparameters, see Appendix E.

### 5.2 Downstream Task Learning Efficiency

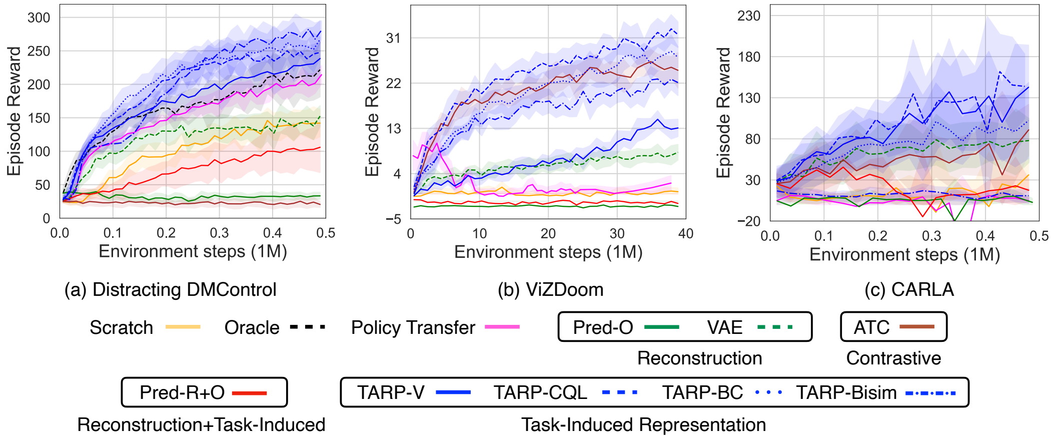
We report downstream task learning curves for task-induced and unsupervised representation learning methods as well as direct policy transfer in Figure 4. The low performance of the *policy transfer* baseline (purple) shows that in most tested environments the downstream task requires significantly different behaviors than those observed in the pre-training data, a scenario in which transferring representations can be beneficial over directly transferring behaviors[3]. All reconstruction-based approaches (green) struggle with the complexity of the tested scenes, especially the method that attempts to predict the scene dynamics (*Pred-O*). We find that adding reward prediction to the objective improves performance (red), especially the method that predicts the scene dynamics with the task-induced objective (*Pred-O+R*). Yet, we find that downstream learning is still slow, possibly because the reconstruction objective leads to task-irrelevant agent appearance information being modeled in the learned representation.

The non-reconstructive contrastive approach *ATC* (brown) achieves stronger results, particularly in the VizDoom environment which features less visual distractors, but its performance deteriorates substantially in environments with more task-irrelevant details, i.e. distracted DMControl and CARLA.

[1] We also tried pre-training policies on each of the individual task datasets $\mathcal{D}_1, \ldots, \mathcal{D}_N$ but found the finetuning performance of the policies trained on the full dataset $\mathcal{D}$ to be superior.

[2] The other environments do not provide a low-dimensional state representation.

[3] The policy transfer baseline achieves good performance in the distracting DMControl environment since it can reuse behaviors from the *walk-forward* training task for the target *run-forward* task.



(a) Distracting DMControl

(b) ViZDoom

(c) CARLA

Figure 4: Performance of transferred representations on unseen target tasks. The task-induced representations (**blue**) work better than the fully unsupervised representations or direct policy transfer and achieve comparable performance to the Oracle baseline. All results averaged across three seeds.

Overall, we find that task-induced representations enable more sample efficient learning. Among the TARP instantiations, we find that TARP-V and TARP-Bisim representations can lead to lower transfer performance, since they rely on the expressiveness of the reward function: if future rewards can be predicted without modeling all task-relevant features, the representations will not capture them. In contrast, TARP-CQL and TARP-BC learn representations via policy learning, which can enable more efficient transfer to downstream policy learning problems. On the distracting DMControl task we find that TARP representations even outperform representations trained with direct supervision through a handcrafted oracle state representation, since they can learn to represent concepts like joint body parts of the walker during pre-training, while the oracle needs to learn these during downstream RL.

### 5.3 Probing Task-Induced Representations

To better understand the improved learning efficiency of the TARP approaches over the unsupervised methods, we visualize what information is captured in the representation in Figure 5: we compare input saliency maps[4] for representations learned with task-induced and unsupervised objectives. We find that task-induced representations can focus only on the important aspects of the scene, such as the walker agent in distracting DMControl and other cars in CARLA. In contrast, the unsupervised approaches have high saliency values for scattered parts of the input and often represent task-irrelevant aspects such as changing background videos, buildings and trees, since they cannot differentiate task-relevant and irrelevant information.

To quantitatively analyze whether task-induced representations effectively represent task-relevant information and discard task-irrelevant features, we train probing networks on top of the learned representations in distracting DMControl. We test whether (1) task-relevant information is modeled by predicting oracle joint states and (2) whether task-irrelevant information is ignored by classifying the ID of the used background video. The more irrelevant background information is captured in the representation, the better the probing network will be at classifying the video.

We report the probing networks' state prediction error in Figure 6a and the background classification accuracy in Figure 6b to compare representations learned via TARP to unsupervised objectives (we choose VAE and ATC as comparisons since they achieved the best transfer performance among reconstruction and non-reconstruction baselines respectively). From this comparison we can see that probing networks trained on top of task-induced representations can more accurately predict the task-relevant state information while they are successfully filtering information about the background video and thus obtain lower background classification accuracy. Therefore, these quantitative results support the qualitative intuition from Figure 5.

[4] Saliency maps visualize the average gradient magnitude for each input pixel with respect to the output $\phi(x)$ and thus capture the contribution of each part of the input to the representation.

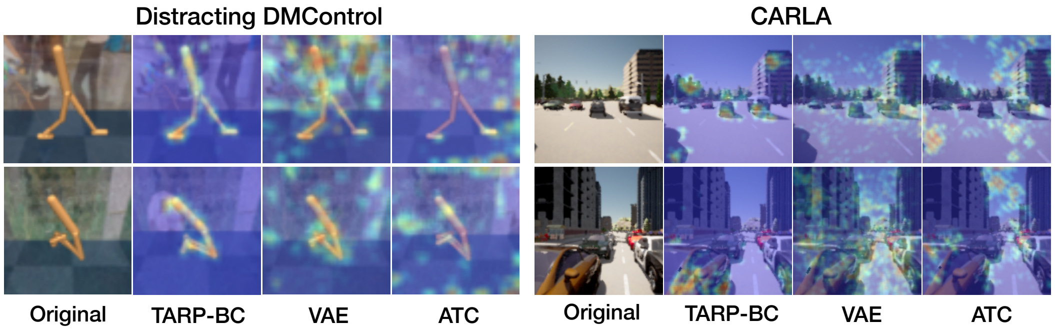
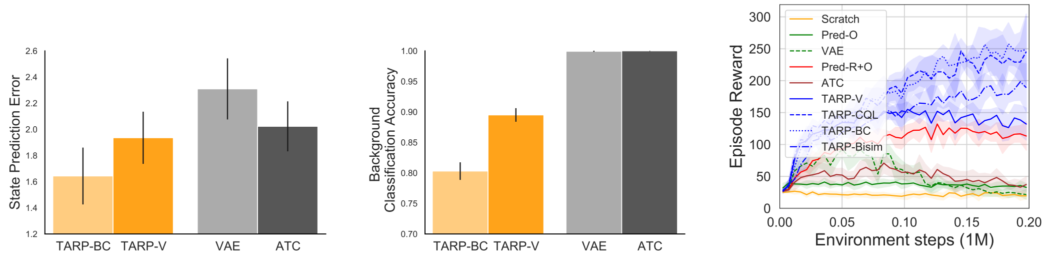


Figure 5: Visualization of the learned representations. **Left to right**: saliency maps for representations learned with task-induced representation learning (TARP-BC) and the highest-performing comparisons for reconstruction-based (VAE) and reconstruction-free (ATC) representation learning. **Left**: Distracting DMControl environment. **Right**: CARLA environment. Only task-induced representations can ignore distracting information and focus on the important aspects of the scene.



(a) State prediction

(b) Background classification

(c) Transfer for imitation learning

Figure 6: (a) Task-induced representations (TARP-BC/V) allow for more accurate prediction of the task-relevant joint states. Unsupervised approaches aim to model all information in the input – thus probing networks can still learn to predict state information, but struggle to achieve comparably low error rates. (b) Task-induced representations successfully filter task-irrelevant background information and thus cannot confidently classify the background video, while unsupervised approaches fail to filter the irrelevant information and thus achieve perfect classification scores. (c) Transfer performance for IL in distracting DMControl. Task-induced representations achieve superior sample efficiency.

### 5.4 Transfer Representation for Imitation Learning

We show that the representations learned via task-induced representation learning are general and can be used for multiple downstream learning approaches: in addition to the model-free RL results above we demonstrate their effectiveness for imitation learning (IL). We train policies with the representations learned on the same datasets used in Section5.2 by Soft-Q Imitation Learning (SQIL, Reddy et al. (2020)) on the distracting DMControl target running task. In Figure 6c we show that task-induced representations also improve downstream performance in visually complex environments for IL and allow for more efficient imitation, since they model only task-relevant information. Again, TARP-BC and TARP-CQL lead to the best learning efficiency. This shows that the benefit of TARP is not constrained to RL, but the *same* representations can be used to accelerate IL.

### 5.5 Data Analysis Experiments

Task-induced representation learning approaches are designed to leverage data from a variety of sources, such as prior training runs or human teleoperation. Thus, analyzing what characteristics of this training data lead to successful transfer is key to their practical use – our goal in this section is to derive a set of best practices when collecting datasets for task-induced representation learning.

**Many tasks vs. lots of data per task.** When collecting large datasets there is a trade-off between collecting a lot of data for few tasks vs. collecting less data each for a larger set of tasks. To analyze the effects of this trade-off on TARP, we train TARP-BC on data from a varying number of pre-training tasks in distracting DMControl. When training on data from fewer tasks, we collect more data on each of these tasks, to ensure that the size of the training datasets is constant across all experiments. The results in Figure 7 show: **training on fewer data from a larger number of tasks is beneficial over training on lots of data from few tasks**. Intuitively, since downstream tasks can leverage the union of task-relevant components of all training tasks, having a diverse set of tasks is more important for transfer than having lots of data for few tasks.

**More data vs. optimal data.** Another trade-off in data collection exists between the amount of data and its optimality: a method that can learn from sub-optimal trajectory data is able to leverage much larger and more diverse datasets. Since training on diverse data is important for successful transfer (see above) robustness to sub-optimal training data is an important feature of any representation learning approach. We test the robustness of task-induced representation learning approaches by training on sub-optimal trajectory data, collected from only partially trained and completely random policies, in distracting DMControl. The downstream RL performance comparison in Figure 8 shows that TARP approaches can learn strong representations from low-performance trajectory data and even when trained from random data does not decrease performance over a SAC baseline trained from scratch because, intuitively, task-induced representation learning does not pre-train a model on "how to act" but merely "what to pay attention to". We find that TARP-CQL's performance can even increase slightly with the suboptimal data, which we attribute to its increased state coverage. Thus we conclude that task-induced representation learning is robust to the optimality of the pre-training data and **collecting larger, diverse datasets is more important than collecting optimal data**.

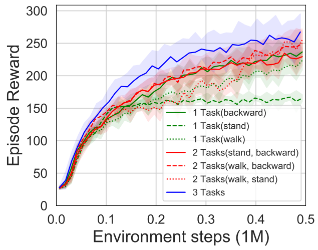


Figure 7: Transfer performance vs. number of pre-training tasks. Collecting training data from a larger number of tasks is more important for transfer performance than having lots of data for few tasks.

**Multi-Source Task Supervision.** When collecting large datasets, it can be beneficial to pool data from multiple sources, e.g. data obtained from prior training runs or via human teleoperation. In Section 4 we introduced different instantiations of the TARP framework that are able to leverage different forms of task supervision. Here, we test whether we can train *a single representation* using multiple sources of task supervision *simultaneously*. In particular, we train "TARP-V+BC" models that assume reward annotations for only some of the tasks in the pre-training dataset and demonstration data for the remaining tasks (for more details on data collection, see Section B). We propagate gradients from all prediction heads into the same representation and compare downstream learning efficiency on distracting DMControl and CARLA in Figure 9. We find that the combined-source model trained from the heterogeneous dataset achieves comparable or superior performance to all single-source models, showing that practicioners should **collect diverse datasets, even if they have heterogeneous sources of task-supervision**.

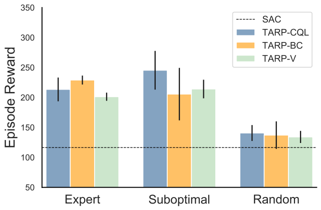


Figure 8: Robustness to pre-training data optimality. TARP approaches can learn good representations that allow for effective transfer even from sub-optimal data.

## 6 Conclusion

In this work, we formalize the problem of task-induced representation learning and analyze multiple approaches for leveraging offline experience of prior tasks for learning compact representations in visually complex scenes. Our empirical results show that task-induced representations retain task-relevant information and discard irrelevant features, leading to improved sample efficiency on downstream task compared to prior unsupervised representation learning approaches. Future work should investigate approaches for incorporating other sources of task information such as language commands to learn task-induced representation.

## 7 Reproducibility Statement

We perform all of our experiments on publicly available environments. We provide a detailed description of the procedures used for the offline dataset collection of prior tasks in Appendix A. Furthermore, in Appendix E we list all of the hyperparameters used for the pre-training phase (i.e. task-induced and unsupervised representation learning) as well as training RL policies. We include our codebase with example commands to reproduce our results in the supplementary materials.

## 8 Acknowledgement

We thank our colleagues from the CLVR lab for the valuable discussions that considerably advanced the research.

## References

Christopher G Atkeson, Benzun P Wisely Babu, Nandan Banerjee, Dmitry Berenson, Christoper P Bove, Xiongyi Cui, Mathew DeDonato, Ruixiang Du, Siyuan Feng, Perry Franklin, et al. No falls, no resets: Reliable humanoid behavior in the darpa robotics challenge. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 623–630. IEEE, 2015.

Serkan Cabi, Sergio Gomez Colmenarejo, Alexander Novikov, Ksenia Konyushkova, Scott Reed, Rae Jeong, Konrad Zolna, Yusuf Aytar, David Budden, Mel Vecerik, Oleg Sushkov, David Barker, Jonathan Scholz, Misha Denil, Nando de Freitas, and Ziyu Wang. Scaling data-driven robotics with reward sketching and batch reinforcement learning. *RSS*, 2019.

Alexey Dosovitskiy and Vladlen Koltun. Learning to act by predicting the future. *ICLR*, 2017.

Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

Norm Ferns, Prakash Panangaden, and Doina Precup. Bisimulation metrics for continuous markov decision processes. *SIAM Journal on Computing*, 40(6):1662–1714, 2011.

Chelsea Finn, Xin Yu Tan, Yan Duan, Trevor Darrell, Sergey Levine, and Pieter Abbeel. Deep spatial autoencoders for visuomotor learning. In *Proceedings of IEEE International Conference on Robotics and Automation*, 2016.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

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.

Abhinav Gupta, Adithyavairavan Murali, Dhiraj Gandhi, and Lerrel Pinto. Robot learning in homes: Improving generalization and reducing dataset bias. *NeurIPS*, 2018.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *ICML*, 2018a.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018b.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. *ICML*, 2019.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017.

Gregory Kahn, Pieter Abbeel, and Sergey Levine. Badgr: An autonomous self-supervised learning-based navigation system. *IEEE Robotics and Automation Letters*, 6(2):1312–1319, 2021.

Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pages 651–673, 2018.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *NeurIPS*, 2020.

Sascha Lange and Martin Riedmiller. Deep auto-encoder neural networks in reinforcement learning. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, 2010.

Kim G Larsen and Arne Skou. Bisimulation through probabilistic testing. *Information and computation*, 94(1):1–28, 1991.

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

Michael Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. In *Proceedings of Neural Information Processing Systems (NeurIPS)*, 2021.

Alex X Lee, Anusha Nagabandi, Pieter Abbeel, and Sergey Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *NeurIPS*, 2020.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Ajay Mandlekar, Yuke Zhu, Animesh Garg, Jonathan Booher, Max Spero, Albert Tung, Julian Gao, John Emmons, Anchit Gupta, Emre Orbay, Silvio Savarese, and Li Fei-Fei. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *CoRL*, 2018.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 02 2015.

Deepak Pathak, Parsa Mahmoudieh, Guanghao Luo, Pulkit Agrawal, Dian Chen, Yide Shentu, Evan Shelhamer, Jitendra Malik, Alexei A Efros, and Trevor Darrell. Zero-shot visual imitation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 2050–2053, 2018.

Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In *Proceedings of Neural Information Processing Systems (NeurIPS)*, pages 305–313, 1989.

Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017.

Siddharth Reddy, Anca D. Dragan, and Sergey Levine. {SQIL}: Imitation learning via reinforcement learning with sparse rewards. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1xKd24twB.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Austin Stone, Oscar Ramirez, Kurt Konolige, and Rico Jonschkowski. The distracting control suite – a challenging benchmark for reinforcement learning from pixels. *arXiv preprint arXiv:2101.02722*, 2021.

Adam Stooke, Kimin Lee, Pieter Abbeel, and Michael Laskin. Decoupling representation learning from reinforcement learning. *ICML*, 2021.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, MN Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, et al. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, 25(8):425–466, 2008.

Marek Wydmuch, Michał Kempka, and Wojciech Jaśkowski. Vizdoom competitions: Playing doom from pixels. *IEEE Transactions on Games*, 2018.

Mengjiao Yang and Ofir Nachum. Representation matters: Offline pretraining for sequential decision making. *arXiv preprint arXiv:2102.05815*, 2021.

Denis Yarats, Ilya Kostrikov, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *ICLR*, 2021.

Albert Zhan, Ruihan Zhao, Lerrel Pinto, Pieter Abbeel, and Michael Laskin. A framework for efficient robotic manipulation. *arXiv preprint arXiv:2012.07975*, 2020.

Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. *ICLR*, 2021.

Marvin Zhang, Sharad Vikram, Laura Smith, Pieter Abbeel, Matthew J Johnson, and Sergey Levine. Solar: deep structured representations for model-based reinforcement learning. *ICLR 2019*, 2019.

Li Zhaoping. Theoretical understanding of the early visual processes by data compression and data selection. *Network: Computation in Neural Systems*, 2006.

Çaglar Gülçehre, Ziyu Wang, Alexander Novikov, Thomas Paine, Sergio Gómez Colmenarejo, Konrad Zolna, Rishabh Agarwal, Josh Merel, Daniel J. Mankowitz, Cosmin Paduraru, Gabriel Dulac-Arnold, Jerry Li, Mohammad Norouzi, Matthew Hoffman, Nicolas Heess, and Nando de Freitas. Rl unplugged: A suite of benchmarks for offline reinforcement learning. In *NeurIPS*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/51200d29d1fc15f5a71c1dab4bb54f7c-Abstract.html.

# A Environments

## A.1 Distracting DMControl

We increase the visual complexity of DMControl "walker" task (Tassa et al., 2018) by overlaying randomly sampled videos from the DAVIS 2017 dataset (Pont-Tuset et al., 2017), similar to Stone et al. (2021). In our experiments we use expert policies to collect offline datasets for the pre-training tasks of standing, forward walking, and backward walking respectively. To collect these datasets, we pre-train polices with SAC(Haarnoja et al., 2018a) in the state space and collect rollouts of the visual observation. We test our representation on the downstream task of "running".

**Rewards:** We use reward functions provided by DMControl "walker" task (Tassa et al., 2018). For the backward walking task, we invert the sign of walk speed in "forward" task defined in DMControl so that the agent gets higher reward when the agent moves backward instead of forward.

## A.2 ViZDoom

Our experiments use "D3 battle"environment provided by Wydmuch et al. (2018) where the agent's objective is to defend against enemies while collecting medi-packs and ammunition. For collection of the prior task dataset, we use the pre-trained models provided by Dosovitskiy and Koltun (2017) and vary the reward weighting parameters (described below) to produce a diverse set of behaviours.

**Rewards:** A reward function is defined by a linear combination of three measurements (ammunition, health, and frags) with their corresponding coefficients.

$$R_{\text{ViZDoom}} = c_{ammo} \cdot \frac{x_t^{amm} - x_{t-1}^{amm}}{7.5} + c_{health} \cdot \frac{x_t^{health} - x_{t-1}^{health}}{30.0} + c_{frags} \cdot \frac{x_t^{frags} - x_{t-1}^{frags}}{1.0}. \tag{5}$$

where $x_t^{amm}$ is a measurement of ammunition, $x_t^{health}$ is health of the agent, and $x_t^{frags}$ is a number of frags at timestep $t$. The set of coefficients for ammunition, health and frags are represented as $(c_{ammo}, c_{health}, c_{frags})$. We use the coefficients of $(0, 0, 1)$, $(0, 1, 0)$, and $(1, 1, -1)$ for the prior tasks, and $(0.5, 0.5, 1.0)$ for the target task.

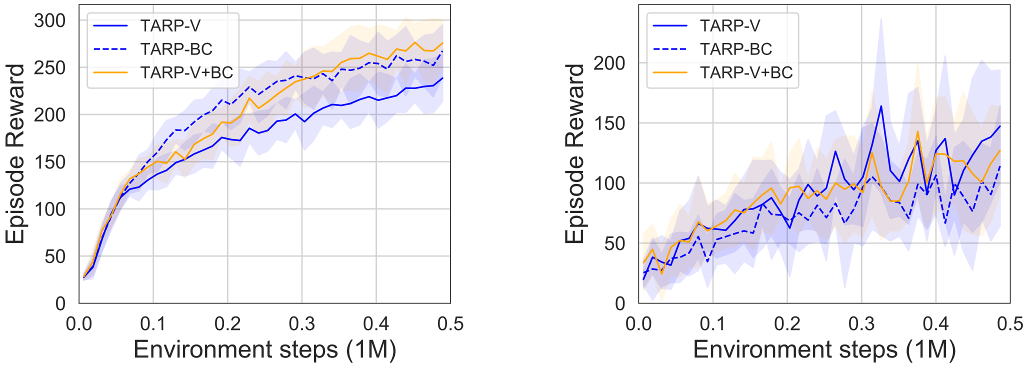
## A.3 CARLA

In our experiments, we use the map of "Town05" from the CARLA environment (Dosovitskiy et al., 2017). At the beginning of each episode, we randomly spawn 300 vehicles and 200 pedestrians. The initial location of the agent is randomly sampled from a task set containing multiple start and goal locations. We collect the datasets for the tasks of intersection crossing, taking a right turn, and taking a left turn using pre-trained policies. We pre-train the policies with SAC (Haarnoja et al., 2018a) using segmentation masks of the environment as the input, and then use the learnt policies to collect rollouts of visual observations for the datasets.

**Rewards:** We use the same reward function for all of the tasks. The reward function consists of terms for speed, centering on a road, angle of the agent, and collision.

$$\begin{aligned} R_{\text{speed}} &= \frac{v}{v_{min}} \cdot \mathbb{1}_{v \leq v_{min}} + (1.0 - \frac{v - v_{target}}{v_{max} - v_{target}}) \cdot \mathbb{1}_{v \geq v_{max}} + 1.0 \cdot \mathbb{1}_{v_{min} \leq v \leq v_{max}}. \\ R_{\text{centering}} &= \max(1.0 - \frac{d_{center}}{d_{max}}, 0). \\ R_{\text{angle}} &= \max(1.0 - |\frac{r}{(r_{max} \cdot \frac{\pi}{180})}|, 0). \\ R_{CARLA} &= R_{\text{speed}} + R_{\text{centering}} + R_{\text{angle}} - 10^{-4} \cdot collision\_intensity. \end{aligned} \tag{6}$$

where $v$ is velocity of the agent, $d_{center}$ is distance between the center of the road and the agent, $r$ is angle of the agent. We use constant values of $v_{min} = 15.0$, $v_{max} = 30.0$, $v_{target} = 25.0$, $d_{max} = 3$, and $r_{max} = 20$.



(a) Distracting DMControl

(b) CARLA

Figure 9: Transfer performance for single source and multiple source task-induced representation. Our task-induced representation with multiple supervision performs slightly better or as good as the single source representation. Thus, task-induced representation learning framework is compatible with heterogeneous datasets.

## B Data Collection for Multi-Source Task Supervision

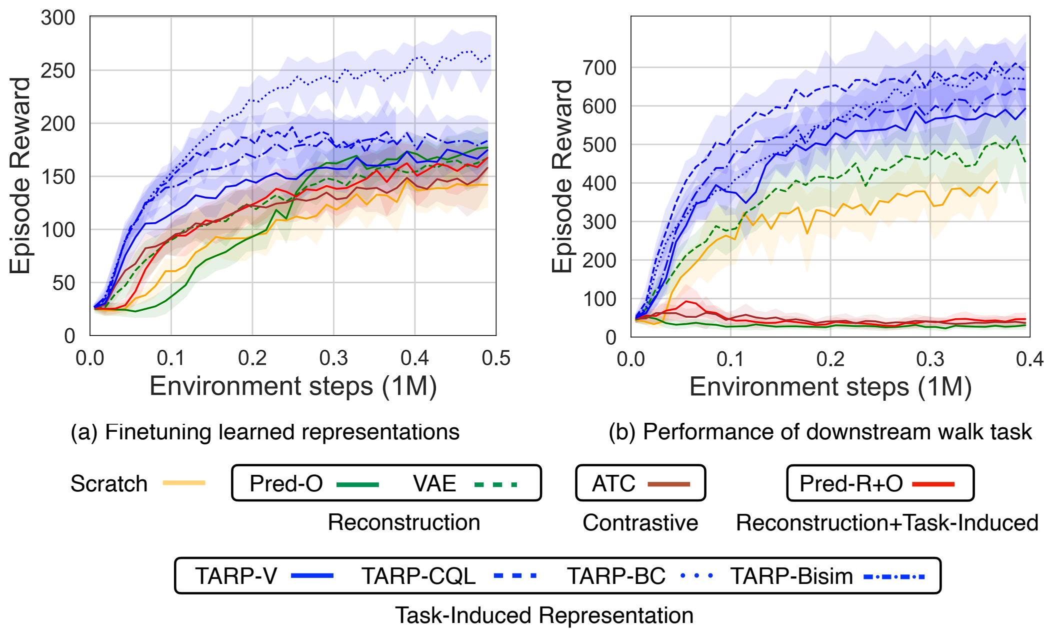
We use supervision from heterogeneous data sources to learn task-induced representations. For the distracting DMControl environment, the dataset is composed of demonstrations for the “forward walking“ and “backward walking“ tasks, and a dataset with reward annotations for the “stand“ task. For the CARLA environment, the task-induced representations are trained on a dataset composed of demonstrations for the “intersection crossing” task, and the datasets annotated with rewards for the “right turn” and “left turn” tasks. For both environments, the datasets are collected with the procedure described in Appendix A.

## C Finetuning learned representations

In our experimental evaluation in Section 5 we held the parameters of the pre-trained encoder fixed to cleanly evaluate the quality of the pre-trained representation. However, in practice pre-trained representations are often finetuned on the target task using target task rewards. Prior work on representation learning found mixed results when finetuning the pre-trained representations (Yang and Nachum, 2021). Thus, in this section we experimentally compare the different representation learning approaches on the Distracting DMControl environment *while finetuning the learned representations on the target task*. As illustrated in Figure 10(a), we find that task-induced representations show improved sample efficiency and better performance even in the fine-tuning setting, analogous to the results in Section 5.2.

## D Transfer Performance for different prior and target task sets

We experimentally validate that we can interchange pre-training vs target task sets. Figure 10(b) shows the downstream performance on the "walk" task with representations pre-trained on "run", "backward walking", and "standing" tasks. This quantitative result underlines that task-induced representation lead to more efficient downstream training than unsupervised methods – the same conclusions as in Section 4 apply in this scenario.



(a) Finetuning learned representations

(b) Performance of downstream walk task

Figure 10: (a) Performance of transferred representations with finetuning on unseen target tasks on distracting DMControl. The task-induced representations (**blue**) show better sample efficiency and performance compared to the representations learned with unsupervised objectives. (b) Performance of a downstream walk task on distracting DMControl with representations learned on offline datasets of running, backward walking, and standing tasks.

# E HYPERPARAMETERS

## E.1 HYPERPARAMETERS FOR RL

Table 1: Common SAC hyperparameter

| Parameter | Value |
|---|---|
| Stacked frames | 3 |
| Optimizer | Adam |
| Learning rate | 3e-4 |
| Discount factor ($\gamma$) | 0.99 |
| Latent dimension | 256 |
| Convolution filters | $[8, 16, 32, 64]$ |
| Convolution strides | $[2, 2, 2, 2]$ |
| Convolution filter size | 3 |
| Hidden Units (MLP) | [1024] |
| Nonlinearity | ReLU |
| Target smoothing coefficient ($\tau$) | 0.005 |
| Target entropy | $-\dim(\mathcal{A})$ |

Table 2: Distracting DMControl SAC hyperparameter

| Parameter | Value |
|---|---|
| Observation Rendering | (64, 64), RGB |
| Initial steps | $5 \times 10^3$ |
| Action repeat | 2 |
| Replay buffer size | $10^5$ |
| Minibatch size | 256 |
| Target update interval | 1 |
| Actor update interval | 1 |
| Initial temperature | 1 |

Table 3: CARLA SAC hyperparameter

| Parameter | Value |
|---|---|
| Observation Rendering | (128, 128), RGB |
| Initial steps | $3 \times 10^3$ |
| Action repeat | 1 |
| Replay buffer size | $10^5$ |
| Minibatch size | 128 |
| Target update interval | 2 |
| Actor update interval | 2 |
| Initial temperature | 0.1 |

Table 4: ViZDoom PPO hyperparameter

| Parameter | Value |
|---|---|
| Observation rendering | (64, 64) Grey |
| Stacked frames | 4 |
| Action repeat | 1 |
| Optimizer | Adam |
| Learning rate | 3e-4 |
| PPO epoch | 10 |
| Buffer size | 2048 |
| Convolution filters | $[8, 16, 32, 64]$ |
| Convolution filter sizes | $[2, 2, 2, 2]$ |
| Hidden units (MLP) | $[256]$ |
| Generalized advantage estimation $\lambda$ | 0.95 |
| Entropy bonus coefficient | 4e-3 |
| Discount factor ($\gamma$) | 0.99 |
| Minibatch size | 256 |
| Nonlinearity | ReLU |

Table 5: Hyperparameters for CQL

| Environment | Trade-off factor $\alpha$ for Q-values | Number of action samples |
|---|---|---|
| Distracting DMControl | 3. | 1 |
| ViZDoom | 1. | 1 |
| CARLA | 3. | 1 |

### E.2 Hyperparameters for pre-training

In TARP-BC, we use recurrent neural networks to predict a sequence of actions over prediction horizon $T$ to learn task-induced representations only in CARLA (see Table 7).

Table 6: Common model parameters

| Parameter | Value |
|---|---|
| Batch size | 128 |
| Hidden units (MLP) | [256] |
| Learning rate | 1e-4 |

Table 7: Hyperparameters for TARP-BC

| Environment | LSTM hidden units | Prediction horizon |
|---|---|---|
| Distracting DMControl | — | - |
| ViZDoom | — | - |
| CARLA | 512 | 8 |

Table 8: Hyperparameters for TARP-V

| Environment | Discount rate |
|---|---|
| All environments | 0.4 |

Table 9: Hyperparameters for TARP-Bisim

| Environment | Reward predictive loss weight | Bisimulation loss weight $T$ |
|---|---|---|
| Distracting DMControl | 1. | 0.1 |
| ViZDoom | 1. | 10. |
| CARLA | 1. | 0.01 |

Table 10: Hyperparameters for VAE

| Environment | $\beta$ constraint |
|---|---|
| Distracting DMControl | 100. |
| ViZDoom | 100. |
| CARLA | 10. |

Table 11: Hyperparameters for Pred-S and Pred-R+S

| Environment | $\beta$ constraint | Prediction horizon $T$ |
|---|---|---|
| Distracting DMControl | 50. | 6 |
| ViZDoom | 10. | 6 |
| CARLA | 5. | 8 |

Table 12: Hyperparameters for ATC

| Environment | Random shift probability | Temporal shift |
|---|---|---|
| All environments | 1. | 3 |