Learning Task Informed Abstractions

Xiang Fu *1  Ge Yang *1 2  Pulkit Agrawal 1 2  Tommi Jaakkola 1

Abstract

Current model-based reinforcement learning methods struggle when operating from complex visual scenes due to their inability to prioritize task-relevant features. To mitigate this problem, we propose learning Task Informed Abstractions (TIA) that explicitly separates reward-correlated visual features from distractors. For learning TIA, we introduce the formalism of Task Informed MDP (TiMDP) that is realized by training two models that learn visual features via cooperative reconstruction, but one model is adversarially dissociated from the reward signal. Empirical evaluation shows that TIA leads to significant performance gains over state-of-the-art methods on many visual control tasks where natural and unconstrained visual distractions pose a formidable challenge. Project page: https://xiangfu.co/tia

1. Introduction

Consider results of a simple experiment reported in Figure 1. We train a state-of-the-art model-based reinforcement learning algorithm (Hafner et al., 2020) to solve two versions of the Cheetah Run task (Tassa et al., 2018): one with a simple and the other with a visually complex background (Zhang et al., 2021). For each version we train three model variants that contain 0.5× (small), 1× (medium) and 2× (large) parameters in their respective world models. The performance with the simple background is only marginally affected by model capacity indicating that even the smallest model has sufficient capacity to learn task-relevant features. With complex background, performance is much worse but increases monotonically with the model size. Because the amount of task-relevant information is unchanged between simple and complex background variants, these results demonstrate that excess model capacity is devoted to representing background information when learning from complex visual inputs. The background conveys no information about the task. It, therefore, interferes with the learning of task-relevant information by consuming model capacity. Here “relevant” refers to features needed to predict the optimal actions, whereas “irrelevant” refers to everything else that makes up the observation.

There are two main components of a model-based learner: (i) a forward dynamics model that predicts future events resulting from executing a sequence of actions from the current state and (ii) a reward predictor for evaluating possible future states. The policy performance depends critically on the prediction accuracy of the forward model, which is intimately tied to the feature space in which the future is predicted. Similar to the complex background version of Cheetah Run, visual observations obtained in the real world are full of irrelevant information. Therefore, if there is no bias for learning “task-relevant features”, the model will try to predict all the information. In such a scenario spurious features would unnecessarily increase the sample complexity and necessitate the use of much larger models than necessary. Sometimes it can also result in training failure.

Ways to learn good representations has been a major focus in model-based deep reinforcement learning. A popular choice is to reconstruct the input observations (Kingma & Welling,
Learning Task Informed Abstractions

2. Preliminaries

A Markov Decision Process is represented as the tuple \((S, O, A, T, r, \gamma, \rho_0)\) where \(O\) is a high-dimensional observation space, \(A\) is the space of actions, \(S\) is the state space, \(\rho_0\) is the initial state distribution, \(r: S \rightarrow \mathbb{R}\) is the scalar reward. The goal of RL is to learn a policy \(\pi^*(a|s)\) that maximizes cumulative reward \(J_{\pi} = \arg \max_{\pi} \mathbb{E} \sum_{t} \gamma^{t-1} r_t\) discounted by \(\gamma\).

Our primary contribution is in the method for learning forward dynamics and is agnostic to the specific choice of the model-based algorithm. We choose to build upon the state-of-the-art method Dreamer (Hafner et al., 2020). The main components of this model are:

\[
\begin{align*}
\text{Representation model:} & \quad p_{\theta}(s_t | o_t, s_{t-1}, a_{t-1}) \\
\text{Observation model:} & \quad q_{\theta}(o_t | s_t) \\
\text{Transition model:} & \quad q_{\theta}(s_t | s_{t-1}, a_{t-1}) \\
\text{Reward model:} & \quad q_{\theta}(r_t | s_t)
\end{align*}
\]

Model Learning Dreamer (Hafner et al., 2020) forecasts in a feature representation of images learned via supervi-
Learning Task Informed Abstractions

Policy Learning  Dreamer uses the learned forward dynamics model to train a policy using an actor-critic formulation described below:

Action model:  \( a_t \sim q_\phi(a_t | s_t) \)

Value model:  \( v_\psi(s_t) \approx \mathbb{E}_{q(s_{t+1} | s_t, a_t)} (\sum_{t=1}^{T} \gamma^{-t} r_t) \) (3)

The action model is trained to maximize cumulative rewards over a fixed horizon \( H \). Both the action and value models are learned using imagined rollouts from the learned dynamics. We refer the reader to (Hafner et al., 2020) for more details.

3. Learning Task Informed Abstractions

Task Informed MDP  In many real-world problems, the state space of MDP cannot be directly accessed but needs to be inferred from high-dimensional sensory observations. Figure 2a shows the graphical model describing this common scenario. To explicitly segregate task-relevant and irrelevant factors, we propose to model the latent embedding space \( S \) with two components: a task-relevant component \( S^+ \) and a task-irrelevant component \( S^- \). We assume that the reward is fully determined by the task-relevant component \( r : S^+ \rightarrow \mathbb{R} \), and the task-irrelevant component contains no information about the reward:

\[ \text{MI}(r; s^-) = 0, \forall t \]

In the most general case, \( s^- \) and \( s^+ \) can depend on \( s^- \) and \( s^+ \). However, in many realistic scenarios the task-relevant and distractor features evolve independently (e.g. the cars on the road vs leaves flickering in the wind) and thus follow factored dynamics (Guestrin et al., 2003; Pitis et al., 2020). Such a situation greatly simplifies model learning. For this reason we further incorporate this factored structure into our formulation through the assumption:

\[ p(s_{t+1} | s_t, a_t) = p(s^+_t | s^+_t, a_t) p(s^-_t | s^-_t, a_t) \]

The resulting MDP, which we call Task Informed MDP (TiMDP) is illustrated in Figure 2b. Note that both \( S^+ \) and \( S^- \) generate the observation \( O \), and both forward models \( p(s^+_{t+1} | s^+_t, a_t) \) and \( p(s^-_{t+1} | s^-_t, a_t) \) admit the agent’s actions. For clarity, we summarize the assumptions we have made into Section 3.
Learning Task Informed Abstractions

![Diagram](image_url)

Figure 3: Components of Task Informed Abstraction Learning. (a) From the dataset of past experience, TIA uses the reward to factor the MDP into a task-relevant world model and a task-irrelevant one. (b) Only the forward dynamics in \( s_t^+ \) is used during policy learning. The Policy is trained using back-propagation through time. Note that the images are shown just for demonstration purposes and are not generated during policy learning.

### Task Model

We produce the mask \( M \) while the reward prediction head is trained to minimize the reward prediction loss, which we found to be simple and effective.

The reconstruction objective is obtained through a learned mixing with a mask \( M \) denoted element-wise product. The reconstruction objective is computed as \( \ln \mathcal{N}(a_t; \hat{a}_t, 1) \). An illustration of the mixing process is in Figure 3a.

### Distractor Model

**Reward Dissociation** for the distractor model is accomplished via the adversarial objective \( J_{\text{Radv}}^t \). This is a minimax setup where we interleave optimizing the distractor model’s reward prediction head (for multiple iterations/training step) with the training of the distractor model. While the reward prediction head is trained to minimize the reward prediction loss \(- \ln q(r_t | s_T^+) \), the distractor model maximizes this objective so as to exclude reward-correlated information from its learned features (Ganin & Lempitsky, 2015). The reward prediction loss is computed using \( \ln \mathcal{N}(r_t; \hat{r}_t, 1) \), where \( \mathcal{N}(\cdot; \mu, \sigma^2) \) denotes the Gaussian likelihood, and \( \hat{r}_t \) is the predicted reward.

**Distractor-model-only Reconstruction** One failure mode of the formulation being discussed is a degenerate solution where a distractor model that captures no information at all can still satisfy the two objectives described above. This would result in the task model reconstructing the entire observation by itself. To avoid such degeneracy, we add an additional image decoder that encourages the distractor model to reconstruct the entire input observation by itself, so as to capture as much information about the observation as possible. This reconstruction loss is denoted as \( J_{O_s} \).

### Cooperative Reconstruction

By jointly reconstructing the image, the distractor model that’s biased towards capturing task-irrelevant information will enable the task model to focus on task-relevant features. We implement joint reconstruction through the objective \( J_{O_s}^t \). Starting with a sequence of observation and actions \( \{o_{t<\tau}, a_{t<\tau}\} \), we first pass this sequence through the two separate recurrent state space model (RSSM, Hafner et al. 2019) to produce the states \( s_t^+ \) and \( s_{\tau}^- \), which are then used to decode two images \( \hat{o}_t^+ \) and \( \hat{o}_{\tau}^- \) given the observation \( o_t \). The joint reconstruction is achieved through a learned mixing with a mask \( M_t \), which we found to be simple and effective.

We produce the mask \( M_t \) by letting the two image decoders additionally produce two tensors of size \( H \times W \times 3 \), where \( H, W \) are the height and width of the observed image. These two tensors are concatenated channel-wise to obtain a tensor of size \( H \times W \times 6 \), then is passed through a 1x1 convolution layer followed by sigmoid activation to obtain the \( H \times W \times 1 \) mask \( M_t \) with value between (0, 1). The final reconstruction is obtained through \( \hat{o}_t = \hat{o}_t^+ \odot M_t + \hat{o}_{\tau}^- \odot (1 - M_t) \), where \( \odot \) denotes element-wise product. The reconstruction objective is computed as \( \ln \mathcal{N}(a_t; \hat{a}_t, 1) \). An illustration of the mixing process is in Figure 3a.

**Policy Learning** is similar to Dreamer, except that we replace the world model with the task model. This way, the forward predictions are only made in the \( S^+ \) subspace.

Action model: \( a_t \sim q_\psi(a_t | s_t^+) \)

Value model: \( \psi(s_t^+) \approx \mathbb{E}_{q(s_t^+)}(\sum^t_{\tau=1} \gamma^{t-\tau} r_\tau) \)  

An illustration of the policy learning stage is in Figure 3b.

### 4. Experiments

Our empirical evaluation aims to answer if the proposed method outperforms existing methods when learning in environments with visually complex backgrounds containing task-irrelevant information. For this purpose, we make use of three environments that are described in Section 4.1. We compare our method against several baselines described in Section 4.2. Our code is available at https://github.com/kyonofx/tia.

#### 4.1. Environments

**ManyWorld** (Figure 4a) We first experimented on a custom environment suite called ManyWorld where the amount
Learning Task Informed Abstractions

Figure 4: Visualizing the information represented by Dreamer (Hafner et al., 2020) and the task and distractor models of our method on several environments. (a) In the ManyWorld environments, Dreamer mistakes the distractor (yellow) for the target object (blue). The task model of TIA isolates the target object (blue) and the goal (red). (b, c) Dreamer’s capacity is consumed at reconstructing the irrelevant video background, and it fails to capture the agent’s outline, which is the task-relevant information. In all domains, Dreamer reconstruction tries to capture every pixel of the raw observation but misses task-relevant information. TIA is able to capture task-relevant information with the task model and task-irrelevant information with the distractor model.

of distraction can be easily controlled. The task is to move a target block (in blue) to a goal location. The goal is visually indicated by a translucent red sphere. Other objects are distractors. The amount of distraction can be controlled by varying the number and/or the dynamics of these objects. We turned off physical collisions between objects, but they visually occlude each other if they are present at overlapping positions. The task-relevant objects and the distractor objects are visually similar, necessitating additional effort to resolve when learning a world model. The observation is RGB image of size $32 \times 32 \times 3$.

Kinematic Control with Natural Video Distraction (Figure 4b) We consider the DeepMind Control (DMC) suite with natural video background from the Kinetics dataset (Kay et al., 2017) used in prior work (Zhang et al., 2021), which was introduced specifically to test learning under natural visual distraction. These control tasks involve different types of challenges, such as long planning horizon (Hopper), contact and collision (Walker), and larger state/action space (Cheetah). The natural video backgrounds in this test suite contains many factors of variation which makes world-model learning challenging and negatively affects downstream policy optimization. The observation is RGB image of size $64 \times 64 \times 3$. The video backgrounds are from the class “driving car” and are grayscale as in Zhang et al. (2021).

Arcade Learning Environments (ALE) or Atari, (Figure 6) is a standard benchmark for vision-based control. The visuals of these games naturally contain many distractor objects that are irrelevant to the game objective. Our limited compute resources only allowed us to experiment on six games. Each seed takes ten days on a V100 Volta GPU. We report results on games where state-of-the-art model-based algorithms perform significantly worse than model-free algorithms or human performance in the hope of closing this gap. The observation is a grayscale image of size $64 \times 64 \times 1$ as in Hafner et al. (2021).

4.2. Baseline Methods

We compare the proposed method against both model-based and model-free baselines. Some of the baseline methods were proposed specifically to tackle learning in the presence of visual distractions. In particular we compare against Dreamer (Hafner et al., 2020) which is a state of the art model-based algorithm on DMC. On ALE, we compare against an improved variant Dreamer(V2) (Hafner et al., 2021). We compare against a strong model-free method, Deep Bisimulation for Control (DBC) (Zhang et al., 2021), which uses the bisimulation metric developed specifically to be invariant to task-irrelevant features. Finally, we also include DeepMDP (Gelada et al., 2019) which augments the representation with a forward model, then uses model-free, distributional Q learning for the policy. The DeepMDP and DBC results are adapted from Zhang et al. (2021).

Representation learning through an inverse model Inverse model take the observations $o_t$ and $o_{t+1}$ as input and predict the intervening action $a_t$ (Agrawal et al., 2015; Jayaraman & Grauman, 2015). To investigate if features
Learning Task Informed Abstractions

Figure 5: Our method consistently outperforms Dreamer and other baseline methods in a variety of visual control tasks with distraction. The curves show mean and standard deviation, over five seeds for TIA, Dreamer, and Dreamer-Inverse. Results for DBC and DeepMDP are adopted from results reported in (Zhang et al., 2021) and used ten seeds. Our method is effective for both ManyWorld environments (a,b), which contains confusing distracting objects that look similar to the task-relevant components; and the DMC tasks with natural video backgrounds (c,d,e), where the distracting background contains rich information that would consume significant model capacity to capture.

Figure 6: (Left) Raw observation of Atari Robotank. (Right) The task model of TIA emphasizes task-relevant information such as the crosshair and the radar for tracking enemies, while ignoring task-irrelevant information such as textures in the raw observation.

learned by inverse model suffice for learning task-relevant features, we constructed the Dreamer-Inverse model. In this model, the learning objective becomes the following: \( J_{\text{Inverse}} = J_{\text{inv}} + \mathcal{J}_{D} + \beta \mathcal{J}_{D} \) where \( J_{\text{inv}} = \ln p(a_t \mid s_t, s_{t+1}) \) is the inverse model objective and \( J_{D} \) is the dynamics regularizer described in Section 2.

4.3. Can TIA disassociate task-irrelevant information?

We first evaluate our method on the ManyWorld domain, where relevant information comprises the agent (blue block) and the goal (red sphere). Figure 4a provides a qualitative comparison of the information represented by TIA and the baseline method of Dreamer. In many cases, Dreamer mistakes the distractor (yellow) for the agent (blue). On the other hand, the task model of TIA isolates the agent (blue) and the goal (red), and ignores the rest. On the other hand, the TIA’s distractor model successfully captures the distractors (yellow and green). The successful dissociation of task-irrelevant information is confirmed by quantitative results in Figure 5 (a) and (b) showing that TIA outperforms the baselines, and the performance gap increases with the number of distractors.

Next, we considered the DMC domains with natural video distractions. We report the learning curves of Cheetah Run, Walker Run and Hopper Stand are in Figure 5. The image reconstruction results Figure 4b and Figure 4c show that Dreamer performs poorly in capturing the full state of the agents and is distracted by the background. In contrast, the task model of our TIA method accurately recovers the relevant part of the raw observation, which happens to be the agent’s body in these examples. Quantitative performance reported in Figure 5 (c,d,e) clearly shows that TIA outperforms strong baseline methods described in Section 4.2. Overall, these results suggest that TIA is the new state-of-the-art in learning from cluttered observations. Results on additional DMC environments are included in the appendix.

4.4. Results on Atari

In environments tested so far the irrelevant factors were manually injected. Due to human intervention, it is possible these tasks were biased, which our method exploited. To investigate performance in more natural settings, we experimented on Atari games that innately contain significant visual distraction. E.g., images of Robotank game contain several visual signatures that change during the game: the number of enemy tanks destroyed, the rotating radar scanner, the green sprites, and so on. We evaluated performance on six Atari games that are known to be challenging for model-based methods (Hafner et al., 2020) without any hyperparameter tuning (i.e., just a single value chosen based on the intuition described in Section 4.5).

The results reported in Figure 7 demonstrate that we sub-
Learning Task Informed Abstractions

Figure 7: Atari performance at 100M steps. TIA significantly improves policy learning in Demon Attack, Robotank, and Yars Revenge, in which DreamerV2 fails to learn or has inferior sample complexity. We also add the performance of DreamerV2 at 200M steps, along with the performance of two model-free algorithms, DQN and Rainbow DQN, at 200M steps.

4.5. Hyperparameter Selection

The two important hyperparameters are $\lambda_{\text{Radv}}$ and $\lambda_{\text{Os}}$. The balance of these two terms enables the distractor model to capture the task-irrelevant part of states. One particular mode of failure is when the distractor model takes over the reconstruction. It strips the task model from task-relevant information, thus preventing the policy from learning meaningful behavior. Our reward dissociation scheme relies on informative reward signals to work. Yet, at the beginning of training, the reward collected by a random policy tend to be sparse and noisy, making $\lambda_{\text{Radv}}$ less effective at preventing distractor model from dominating. This scenario suggests using a large $\lambda_{\text{Radv}}$ at the beginning of training and slowly increasing the weight $\lambda_{\text{Os}}$ for the distractor reconstruction loss.

On the other extreme the distractor can become degenerate and capture no information of the input observations. In this case, TIA is equivalent to Dreamer. To avoid such degeneracy we would increase $\lambda_{\text{Os}}$ so that the distractor model is encouraged to capture more information.

To gain more insight into whether the reward dissociation process works as expected, we computed the how much reward specific information is captured by the distractor models during learning. For this we recorded the reward prediction error measured using the log-likelihood (as unit Gaussian) in Figure 8. Since the error is unit-less, for the sake of comparison, we estimated the upper bound of the prediction error using the running average of the reward in all historical data. This corresponds to an uninformed reward predictor (the “Mean Predictor”) that always guesses the average. The reward prediction loss of the distractor model trails behind this upper bound during learning (see Figure 8, in red), whereas the reward prediction loss of the main model remains small (in blue), slightly above the loss lower bound, which equals to $-\ln p(r) = -\ln \frac{1}{\sqrt{2\pi}} \approx 0.92$. This result indicates that the task model learns task-relevant information, while the distractor model’s learned feature is indeed reward-independent.

5. Other Related Works

State-of-the-art deep reinforcement learning algorithms often jointly optimize the discounted return together with an auxiliary representation learning objective such as image re-
construction (Watter et al., 2015; Wahlström et al., 2015) or contrastive learning (Sermanet et al. 2018; Oord et al. 2018; Yan et al. 2020; Lee et al. 2020; Mazoure et al. 2020). In model-based reinforcement learning methods, reward and value prediction (Oh et al., 2017; Racanière et al., 2017; Silver et al., 2017; Tamar et al., 2016; Feinberg et al., 2018; Schrittwieser et al., 2020) are also shown to improve performance, along with learning a world model. More recently, data augmentations are found to improve sample complexity (Laskin et al., 2020a; Yarats et al., 2021; Raileanu et al., 2020; Laskin et al., 2020b; Stooke et al., 2020) by taking advantage of domain knowledge of symmetry transformations in the space of data. A recent benchmark (Stone et al., 2021) however, shows that data augmentation helps but still performs poorly in the presence of complex visual distractions. We do find data augmentation to be an orthogonal and complementary approach to our proposal. Our approach focuses on finding ways to inform representation learning for RL which features are more useful for the task, and therefore should be learned first.

Block MDPs (Du et al., 2019), bisimulation (Givan et al., 2003; Dean et al., 1997) and bisimulation metrics (Ferns et al., 2004; 2011; Ferns & Precup, 2014) exploit additional structure to learn state abstractions, leading to impressive empirical gains on natural scene domains (Zhang et al., 2021; Gelada et al., 2019; Agarwal et al., 2021). The work presented here falls under the broad umbrella of learning state and temporal abstractions and is the most closely related to utile distinction (McCallum, 1997) which is limited to finite-state machines. Like utile distinction, TIA is distinct from DBC and bisimulation embeddings in that TIA relies on the task specification for feature separation. It is not concerned about generalization across a set of MDPs related by dynamics.

The spiritual nearest neighbor to our work is the idea of using a primary-bias duo for model debiasing in supervised learning (Clark et al., 2019; Cadene et al., 2019; Wang et al., 2019; He et al., 2019; Bahg et al., 2020; Clark et al., 2020). These methods remove dataset bias by imposing independence constraints between the primary model and a model that's biased by design. Our approach focuses on RL and uses two models for removing task-irrelevant features for policy learning. Our implementation extends recent work in model-based RL from pixels (Watter et al., 2015; Finn & Levine, 2017; Banijamali et al., 2018; Ha & Schmidhuber, 2018; Kaiser et al., 2020; Hafner et al., 2020; 2021; Srinivas et al., 2018; Pertsch et al., 2020).

6. Conclusion and Discussion

This work demonstrates that the TiMDP formulation offers a way for a model-based agent to explain away task-irrelevant information through a distractor model, and successfully learn from cluttered visual inputs. Our approach of learning Task Informed Abstractions (TIA) outperforms previous state-of-the-art model-based RL methods on two standard benchmarks. TIA reduces to Dreamer when its distractor model degenerates and captures no information. TIA would outperform Dreamer when the hyperparameters are well-tuned and the disentangled state representation would help policy learning.

If we are able to distinguish task-relevant features from irrelevant ones, one might expect that such a scheme would also lead to better transfer to new scenarios. To investigate this, we ran experiments on DMC tasks, where the train and test backgrounds consisted of different videos. We found that when the train and test sets had different sets of natural video backgrounds, our method could easily generalize. However, when we trained on random noise background and test on natural video backgrounds without any fine-tuning, the performance suffered. This result is unsurprising, and the transfer problem can be alleviated by training on diverse data. Further, in our current setup, the learning of task-relevant information only happens during training. However, there is no mechanism to adapt what is relevant/irrelevant at test time. Investigating test-time adaptation methods can improve transfer performance to out-of-distribution situations.

A second issue worth mentioning is that while one set of hyperparameters works well across Atari games, for the DMC environments, the choice of hyperparameters, λRadv and λOs, is domain-dependent. We discussed good practices for choosing these hyperparameters in Section 4.5. We hypothesize that based the practice documented here, in the future, it might be possible to automatically tune the hyperparameters by considering the reconstruction and reward-prediction loss of the two models that constitute TIA.

It is possible that the factored dynamics assumption in TiMDP is too strong. Take ManyWorld as an example; this assumption is violated when distractor blocks interact with the target and are no longer task-independent. In practice, TIA implements a relaxed version of the factored dynamics constraint via cooperative reconstruction: Features that interact weakly (such as occlusion) can be filtered into the distractor model (see Figure 4a), whereas ones that interact strongly (such as via collision) are highly task-relevant, therefore will be included in the task model.

Our goal in this work was to build agents that operate from complex visual imagery. Another area of potential investigation is to characterize performance as a function of the sparsity of reward signals. We hypothesize that a low-noise reward signal is a key factor for the robustness of TIA, and the performance might drop in very sparse reward scenarios. Developing methods that can overcome this “potential” challenge is another avenue for future work.
Acknowledgement

We would like to thank Amy Zhang for sharing the performance data of DBC and DeepMDP and Danijar Hafner for sharing the performance data of DreamerV2, DQN, and Rainbow. This work is supported by the MIT-IBM grant on adversarial learning of multi-modal and structured data, the MIT-IBM Quest Program, the DARPA Machine Common Sense Program, and the National Science Foundation under Cooperative Agreement PHY-2019786 (The NSF AI Institute for Artificial Intelligence and Fundamental Interactions, http://iaifi.org/). The authors also acknowledge the MIT SuperCloud and the Lincoln Laboratory Supercomputing Center for providing HPC resources.

References


Learning Task Informed Abstractions


Learning Task Informed Abstractions


