Task-Relevant Embeddings for Robust Perception in Reinforcement Learning

Eric Liang [1] Roy Fox [1] Joseph E. Gonzalez [1] Ion Stoica [1]

Abstract

Reinforcement learning is unreasonably sample inefficient in many real-world visual domains, which require relatively simple control behaviors but pose challenging perception problems. We show that even simple visual noise added to common reinforcement learning benchmark environments can significantly degrade learning efficiency and break common approaches such as the use of autoencoders. We propose new methods for learning task-relevant state representations, and show that they can discover image embeddings that are significantly more effective when robust perception is required.

1. Introduction

Agents that behave in realistic environments must deal with a wide range of environmental conditions, including varying illumination, weather, and transient occlusions. Dealing robustly with these conditions requires robust *perception*, which many Reinforcement Learning (RL) algorithms attempt to learn end-to-end as part of their training process. Due to the relative sample inefficiency of RL compared to supervised approaches, it has not been practical to use RL to train the types of deep convolutional networks that achieve state-of-the-art visual recognition results. While modern vision networks may have hundreds of layers dedicated to learning an underlying representation of the image, typical convolutional models used in RL, e.g., for Atari (Mnih et al., 2015), are more than an order of magnitude smaller.

As a result, alternatives to end-to-end training are sought that can tackle complex visual tasks. Researchers commonly employ supervised imitation learning, auxiliary tasks, data augmentation, heuristic preprocessors, or modular pipelines to make learning tractable. Without these steps, RL performance on visual tasks can be quite poor, with agents trained via state-of-the art algorithms often failing to drive down straight roads in photorealistic car simulators (Dosovitskiy et al., 2017), even after millions of timesteps of training, and when evaluated on the same scenario used for training.

Modular pipelines that leverage embedding models trained with supervised losses (e.g., to segment and label image regions) have been used to address the perception problem (Paden et al., 2016; López et al., 2017). However, in the absence of well-defined domain losses (e.g, scene segmentation) and labeled observation data, it is not always clear what supervised losses to use. Many domain-independent auxiliary or self-supervised losses such as inverse dynamics or frame prediction (Pathak et al., 2017b; Shelhamer et al., 2016; Ha & Schmidhuber, 2018; Jaderberg et al., 2017) help accelerate representation learning in RL, but we show that these approaches fail to learn *robust* state representations.

In this paper, we explore the problem of learning a complete state representation using task-relevant supervised losses. We propose two complementary methods using two domain-independent losses. First, we train a model to predict the outcomes of short policy *options*, learning a PSR-like embedding (Littman & Sutton, 2002) that transfers to the original action space. To capture longer-term task dependencies, this loss can be augmented with a forward-prediction loss in the embedded space. Second, we leverage the self-supervision built into deep image autoencoders by adversarially training a pair of autoencoders to separate task-relevant features from environmental noise. We show that these embeddings can be used directly with RL algorithms and improve performance over previously proposed embeddings.

Our modeling assumptions are as follows:

1. The environment is simple to solve given the latent state representation. This is relatively common in more complex environments, for example, the test scenarios in the Carla driving simulator (Dosovitskiy et al., 2017) can be solved with very coarse (discrete) vehicular inputs, yet popular RL agents perform very poorly.
2. Observations are perceptually complex, but most of the observation signal is irrelevant for the task at hand. Outside of laboratory environments, details such as illumination, shadows, and distractor objects (e.g., vegetation, clouds, skyscrapers) are commonly present yet uninformative for valuable behavior.

[1] University of California, Berkeley. Correspondence to: Eric Liang <ericliang@berkeley.edu>.

Accepted at the FAIM workshop "Prediction and Generative Modeling in Reinforcement Learning", Stockholm, Sweden, 2018.

3. We focus on the fully observable state setting and assume that a short sequence of observations (e.g., the last 4 frames) is sufficient to recover any hidden state. While in principle the techniques described in this paper could be applied to the partially observable setting, we leave that for future work.

1.1. Requirements for a Robust Embedding

An ideal state embedding robustly handles perception and separates clutter from essential state signal for the given control task. A key challenge is to preserve sufficient relevant information from the observation for the agent to complete its task. For good sample efficiency, the embedding should also be *policy independent*. Finally, the embedding should be compact and resistant to irrelevant observational features, including environmental noise that has dynamics of its own.

We examine limitations of three common self-supervised losses used for representation learning in RL:

Autoencoders: Deep autoencoders consist of an image *encoder* that produces a latent representation, followed by a *decoder* that seeks to reconstruct the image from the representation. The encoder and decoder are jointly trained by minimizing the L_2 loss between the original and reconstructed images. While autoencoders have proven effective in visually complex RL tasks (Ha & Schmidhuber, 2018), as we show in Section 3 they are easily distracted by structured observational noise.

Forward dynamics model: Forward dynamics models have been used effectively for model-based RL (Zhang et al., 2017; Pathak et al., 2017b; Ha & Schmidhuber, 2018). When operating in visual domains, they can be implemented as *one-step autoencoders*. In contrast to per-frame autoencoders, forward dynamics autoencoders are able to extract state information needed to predict the environment dynamics. Although robust to stationary noise (e.g., pixel noise), dynamics models are susceptible to structured noise (e.g., moving visual clutter) that follows the physics of the environment but has no impact on the control problem.

Inverse dynamics model: Inverse dynamics models seek to predict the action taken between two consecutive observations in a trajectory. The inverse dynamics loss has been used effectively in a variety of RL algorithms, e.g., for learning a physical model in a latent space (Pathak et al., 2017b), and for exploration (Pathak et al., 2017a). While an inverse dynamics model naturally captures only aspects of the observation that are controllable by the agent, it can fail to capture important aspects of the state for planning. For example, in the vehicular setting the difference in pose between two observations may be sufficient to completely capture the dynamics of the vehicle, but would omit important aspects of the environment such as upcoming obstacles.

2. Task-Relevant Embeddings

Truncated return prediction: We propose an alternative loss for representation learning based on predicting the truncated outcome of policy *options* (Sutton et al., 1999) taken by the agent. Predicting the outcome of options instead of the current policy makes the learned embedding policy-independent, yet relevant to the task. A distribution over truncation levels serves as data augmentation, extracting more supervisory signal from the training data.

A value function $V_\pi(s)$ predicts discounted future reward for a policy π from a state s. However, π changes as the policy is trained. We instead try to learn a more general function $\bar{V}(\pi, s, k)$, where π is a policy and k is the horizon after which rewards are truncated. Since it is infeasible to parameterize V by π, we approximate this by replacing π with a finite set of M policy options. As the policy space becomes more densely covered by these options, $\bar{V}(h, s, k)$ better approximates $\bar{V}(\pi, s, k)$ for any π.

From the returns (i.e., rewards) of these options, we learn a latent state representation $\phi(o_t)$ predictive of task-relevant outcomes. Given a sequence of observations $o_t ... o_{t+k-1}$, the option index h_t, rewards $r_t ... r_{t+k-1}$, a regressor f_ψ, and an optional preprocessor g for extracting extra ground truth statistics from the trace, the loss for ϕ_θ is as follows:

$$L_{pred}(f_\psi(\phi_\theta(o_t), h_t), r_t ... r_{t+k-1} || g(o_t ... o_{t+k-1})) \quad (1)$$

Such a representation $\phi(o_t)$ would act as a predictive state representation (PSR) (Littman & Sutton, 2002) if the returns were sufficient statistics for the task at hand. While the returns alone may not be sufficient to learn a PSR in many cases, other sources of signal can also provide supervision through g. For example, in vehicular simulators, ground truth pose and object proximity information is readily available, and indeed is often used for reward shaping.

Long-term dependencies: It is necessary for the embedding to encode features relevant to long-term outcomes, even with a finite truncation horizon. This can be addressed by adding a loss predicting the successor embedding $\phi_\theta(o_{t+k})$ after k steps of an option h (Figure 1). We can predict the successor state directly, similarly to using a forward dynamics model, without learning a full successor representation (Dayan, 1993). To stabilize training while ϕ_θ is changing rapidly, the successor is fitted against a periodically updated target embedding $\phi_{\theta_{target}}$, as in DQN (Mnih et al., 2015).

Task-aware autoencoder: We also explore leveraging a predictive loss to improve the effectiveness of visual autoencoders. As shown in Section 3, autoencoders work very well in the absence of visual noise, but poorly otherwise, suggesting a hybrid strategy may yield gains. This can be done by training a pair of autoencoders (Figure 2) that learns to model and, more importantly, separate task-

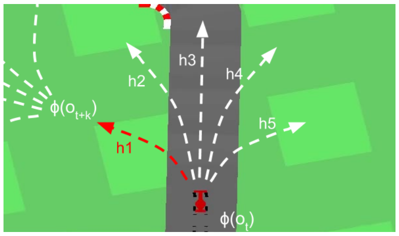


Figure 1. Options $h_1 ... h_5$ for CarRacing. In truncated return prediction, we learn an embedding $\phi(o_t)$ that can predict the returns of each option, which is a surrogate objective for being able to predict the value of any policy. Given h_t, it is also desirable for $\phi(o_t)$ to be predictive of the successor embedding $\phi(o_{t+k})$.

relevant and task-irrelevant image features. By explicitly modeling the visual clutter in the image (e.g., the falling ”snow” in the background of Figure 3), we are able to better separate and extract the task-relevant features. To force the specialization of each auto-encoder, we penalize the secondary auto-encoder for producing state representations that are predictive of future returns. We follow an approach inspired by GANs (Goodfellow et al., 2014) and predictability minimization (Schmidhuber, 1992).

We define the losses as follows, where (ϕ_θ, $\phi^{-1}_{\psi_1}$) and (ϕ_{θ_2}, $\phi^{-1}_{\psi_2}$) are the encoder-decoder pairs for the primary and secondary autoencoder respectively, as illustrated in Figure 2. The regressor f_{ψ_3} implements the truncated return prediction loss for ϕ_θ, and $f_{\psi_{adv}}$ is an adversarial regressor that encourages the secondary autoencoder to model only task-irrelevant features. During training, we minimize $\alpha L_{auto} + L_{pred} - L_{noise}$ while holding ψ_{adv} constant, and concurrently train ψ_{adv} using L_{noise} to improve the adversarial regressor. Here $\alpha = 0.01$ is a hyperparameter interpolating between the autoencoder and regressor losses. The encoder outputs are fused via a combining function c (e.g., pixel-wise max), and then compared against the input:

$$L_{auto}(c(\phi^{-1}_{\psi_1}(\phi_\theta(o_t)), \phi^{-1}_{\psi_2}(\phi_{\theta_2}(o_t))), o_{t+1}) \quad (2)$$

$$L_{pred}(f_{\psi_3}(\phi_\theta(o_t), h_t), r_i ... r_{t+k-1}) \quad (3)$$

$$L_{noise}(f_{\psi_{adv}}(\phi^{-1}_{\psi_2}(\phi_{\theta_2}(o_t, h_t))), r_i ... r_{t+k-1}) \quad (4)$$

3. Evaluation

We evaluate the proposed embeddings in comparison to autoencoder, inverse dynamics, and forward dynamics baselines, as well as training on the raw image. To simulate a complex visual environment where perception is the learning bottleneck, we add visual noise to image versions of Gym environments (Brockman et al., 2016) (Figure 3). To more directly measure the robustness of the embedding, we use the learned encoder ϕ as an image preprocessor (i.e., the RL agent never sees the original observation). We leave optimizations such as fine-tuning ϕ via RL or using it to supervise an auxiliary task to future work. We use TensorFlow (Abadi et al., 2016) to train the neural networks.

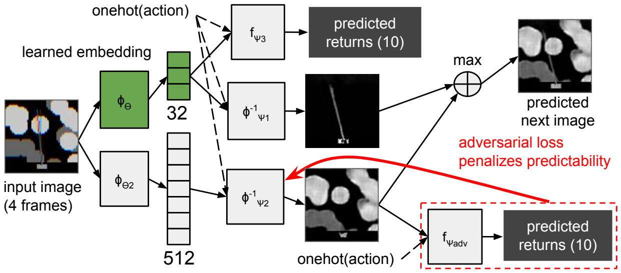
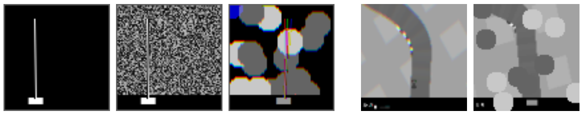


Figure 2. Combining truncated return prediction with a task-aware autoencoder that predicts the next frame. Green boxes highlight the embedding we want to learn. The primary autoencoder’s embedding is trained to predict truncated returns as well as produce a part of the final image. A secondary *high-bandwidth* autoencoder’s output is combined with that of the primary to reconstruct the original image. An adversarial loss from a separately trained regressor prevents the secondary autoencoder from including task-relevant features. We used convolutional layers from the Atari architecture (Mnih et al., 2015). Regressors are implemented by a single 64-unit ReLU hidden layer followed by a linear layer.



(a) Image cartpole

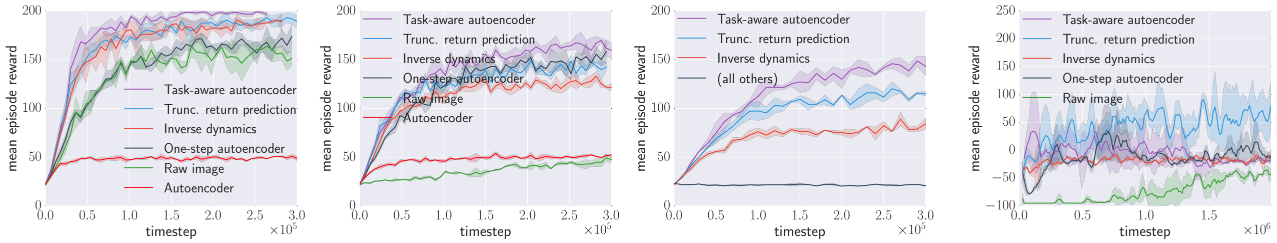
(b) CarRacing-v0

Figure 3. Sample observations for image cartpole and CarRacing-v0. We introduce white noise and structured noise to increase the difficulty of the perception component of the task. Structured noise has dynamics of its own (e.g., here behaving like falling ”snow”).

Pretraining procedure: For our experiments, we use a dataset of option traces gathered upfront for all preprocessor training. We gather this data with a ”bootstrap agent” that can adequately explore the environment (but not necessarily achieve the highest reward). The bootstrap agent is periodically interrupted to execute a random option h. In the online training scenario, bootstrapping can be replaced by alternating training of the agent and ϕ.

We perform minibatch SGD over the option traces with a batch size of 128, withholding 2% of the original dataset to serve as validation data. For the subsequent RL training, we pick the weights for ϕ that minimize validation loss.

For all experiments we use the Atari convolutional architecture (Mnih et al., 2015). When feature vectors are concatenated, they are postprocessed by a single 64-unit ReLU



(a) Image cartpole with (i) no noise, (ii) pixel white noise, (iii) and structured noise (”snow”). (b) Car racing, structured noise.

Figure 4. As perception becomes more difficult, task-relevant embeddings outperform autoencoders, forward and inverse dynamics embeddings, and training on the raw image. Forward dynamics autoencoders learn to ignore white noise (ii), but completely fail to learn a useful embedding in the presence of structured noise (iii). Inverse dynamics embeddings fail to capture all relevant features (iii, b).

hidden layer. All embeddings consist of 32 float32 units. When training on the embedded image, two 256-unit ReLU hidden layers are used for the policy. We use a proximal variant of policy gradient (Schulman et al., 2017) for training cartpole, and A3C (Mnih et al., 2016) for car racing.

Image cartpole: We evaluate the proposed embeddings on an image variant of the toy CartPole-v0 control environment (Brockman et al., 2016). We define two options, h_1 and h_2, which repeat moving the cart left and right respectively, and gather 200k timesteps of option traces. To capture hidden state, the last 4 observations are framestacked prior to feeding to ϕ. We use truncation levels $k \in \{1, 2, 3, ..., 10\}$ for truncated return prediction. In Figure 4 we evaluate learning performance as increasingly complex visual noise is added to the environment. While training on the raw image works well in the absence of noise, even simple white noise degrades sample efficiency precipitously (not matching the embedding methods even after millions of timesteps), and structured noise causes learning to flatline entirely. Commonly used auxiliary losses such as inverse dynamics also struggle as they inherently cannot capture all relevant task features. In contrast, the task-relevant embeddings retain most of their performance even under heavy noise. In addition, the task-aware autoencoder produces the only embedding that leads to fully solving the task.

Car racing: We define $h_1 \ldots h_5$ as shown in Figure 1, use a framestack of 4, gather 1.5 million timesteps of option traces, and pretrain the embedding with truncation levels $k \in \{1, 4, 7, ..., 31\}$. Figure 4(b) shows that truncated return prediction outperforms other embeddings and also use of the raw image when there is noise. In the absence of noise, it improves to ~200 reward, and a standard autoencoder from < 0 to ~500. Note that none of the embeddings enable solving the task (900+ reward). A reward of 100-200 corresponds to successfully making a few turns before getting ”stuck” off-road. We hypothesize this is due to missing long-horizon features in the embedding, which could be addressed with successor embedding prediction (Section 2).

Table 1. Comparison of the effectiveness of embeddings at cross-modeling objectives. In each column we show the ratio of the achieved loss with a particular embedding to the best loss achieved by any embedding in the column. The task-aware autoencoder achieves close to the best loss for all objectives in image cartpole.

	AE Loss	IVD Loss	TRP Loss
Task-aware AE	1×	1×	1.05×
Trunc. Ret. P.	8.3×	1.02×	1×
Inv. Dyn.	8.3×	1.07×	1.5×
AE (1-step)	2.2×	1.47×	3.8×

Task-aware autoencoder: To better understand the performance of the task-aware autoencoder, in Table 1 we compare its effectiveness at objective cross-modeling. To measure this, we trained a multi-headed network with all losses, but gated the gradient to the embedding from all but one of the losses. The task-aware autoencoder achieves the best reconstruction loss due to its secondary autoencoder and nearly matches the best return prediction loss, indicating that some representational capacity was used for modeling the pole angle in lieu of pure return prediction. Interestingly, it also achieves the lowest inverse dynamics loss, despite not being optimized for that objective in particular.

Figure 2 shows the task-aware autoencoder can precisely predict the next cartpole frame from framestacked input. This is in contrast to standard autoencoders that only produce a blurry reconstruction of the pole or emit it entirely.

4. Future work

We plan to continue evaluating and improving the robustness of embeddings, tackling more complex photorealistic visual environments (e.g., Carla), and those with sparser rewards (e.g., Atari). Another natural question is how to best leverage automatic option extraction techniques for truncated return prediction. We are also investigating why the task-aware autoencoder does not converge to a good segmentation in the car racing environment. We expect this to be challenging due to the instability of GAN training.

References

Abadi, Martin, Barham, Paul, Chen, Jianmin, Chen, Zhifeng, Davis, Andy, Dean, Jeffrey, Devin, Matthieu, Ghemawat, Sanjay, Irving, Geoffrey, Isard, Michael, Kudlur, Manjunath, Levenberg, Josh, Monga, Rajat, Moore, Sherry, Murray, Derek G., Steiner, Benoit, Tucker, Paul, Vasudevan, Vijay, Warden, Pete, Wicke, Martin, Yu, Yuan, and Zheng, Xiaoqiang. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pp. 265–283, 2016. URL https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf.

Brockman, Greg, Cheung, Vicki, Pettersson, Ludwig, Schneider, Jonas, Schulman, John, Tang, Jie, and Zaremba, Wojciech. OpenAI Gym. *CoRR*, abs/1606.01540, 2016. URL http://arxiv.org/abs/1606.01540.

Dayan, Peter. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5(4):613–624, 1993.

Dosovitskiy, Alexey, Ros, Germán, Codevilla, Felipe, López, Antonio, and Koltun, Vladlen. CARLA: an open urban driving simulator. *CoRR*, abs/1711.03938, 2017. URL http://arxiv.org/abs/1711.03938.

Goodfellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, and Bengio, Yoshua. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Ha, David and Schmidhuber, Jürgen. World models. *CoRR*, abs/1803.10122, 2018. URL http://arxiv.org/abs/1803.10122.

Jaderberg, Max, Mnih, Volodymyr, Czarnecki, Wojciech Marian, Schaul, Tom, Leibo, Joel Z., Silver, David, and Kavukcuoglu, Koray. Reinforcement learning with unsupervised auxiliary tasks. 2017.

Littman, Michael L and Sutton, Richard S. Predictive representations of state. In *Advances in neural information processing systems*, pp. 1555–1561, 2002.

López, Antonio M, Imiya, Atsushi, Pajdla, Tomas, and Álvarez, Jose M. *Computer Vision in Vehicle Technology: Land, Sea, and Air*. John Wiley & Sons, 2017.

Mnih, Volodymyr, Kavukcuoglu, Koray, Silver, David, Rusu, Andrei A, Veness, Joel, Bellemare, Marc G, Graves, Alex, Riedmiller, Martin, Fidjeland, Andreas K, Ostrovski, Georg, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

Mnih, Volodymyr, Badia, Adria Puigdomenech, Mirza, Mehdi, Graves, Alex, Lillicrap, Timothy, Harley, Tim, Silver, David, and Kavukcuoglu, Koray. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pp. 1928–1937, 2016.

Paden, Brian, Cáp, Michal, Yong, Sze Zheng, Yershov, Dmitry S., and Frazzoli, Emilio. A survey of motion planning and control techniques for self-driving urban vehicles. *CoRR*, abs/1604.07446, 2016. URL http://arxiv.org/abs/1604.07446.

Pathak, Deepak, Agrawal, Pulkit, Efros, Alexei A., and Darrell, Trevor. Curiosity-driven exploration by self-supervised prediction. In *ICML*, 2017a.

Pathak, Deepak, Agrawal, Pulkit, Efros, Alexei A, and Darrell, Trevor. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning (ICML)*, volume 2017, 2017b.

Schmidhuber, Jürgen. Learning factorial codes by predictability minimization. *Neural Computation*, 4(6):863–879, 1992.

Schulman, John, Wolski, Filip, Dhariwal, Prafulla, Radford, Alec, and Klimov, Oleg. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

Shelhamer, Evan, Mahmoudieh, Parsa, Argus, Max, and Darrell, Trevor. Loss is its own reward: Self-supervision for reinforcement learning. *CoRR*, abs/1612.07307, 2016. URL http://arxiv.org/abs/1612.07307.

Sutton, Richard S, Precup, Doina, and Singh, Satinder. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

Zhang, Richard, Isola, Phillip, and Efros, Alexei A. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *CVPR*, volume 1, pp. 5, 2017.