State Representation Learning: An Overview

the firemakers

November 12, 2017

Abstract

Deep learning is ubiquitous, and robotics could not be a less deserved area worth penetrating. Soon robots will be able to benefit from deep learning frameworks to make complex choices and predictions in a robust and autonomous fashion. The interesting particularity of a robot is that its input perception is, as in humans, multimodal. The robot can use its camera, LIDAR, radar, microphone, etc., while it executes actions. As in many other traditional problems that deep learning is targeting, the classic challenge of the curse of dimensionality remains. A key issue is how to make an algorithm able to make predictions with several high dimension inputs and how to make it find hidden dependencies between them online. A solution is to reduce the dimensionality of each input by learning a mapping into a state representation space. The aim of state representation learning is to find modularities or few hidden parameters for each input. Once the hidden parameters are found, the task of finding dependencies between modularities is no more bothered by the issue of high dimensionality. This paper aims to cover the state of the art about state representation learning. It presents the different methods used to disentangle hidden parameters in datasets and to validate the learned state representation. This overview is particularly focused on learning representations in low dimensionality of known parameters such as, for instance, the state or position of a 3D object. This scope makes possible to better assess the representation learned.

1 Introduction

The use of deep learning in autonomous robots will be determinant in a near future, for example for autonomous vehicles. The data from hundreds of sensors of an autonomous robot has to be analysed and transformed into meaningful representations. Moreover, the need for validation and interpretability methods for those representations will be key for the acceptance of deep learning technologies in our lives. Reinforcement learning is one of the most used approaches to train neural networks for autonomous tasks [Mnih et al., 2015], [Lillicrap et al., 2015], [Schulman et al., 2015], [Wu et al., 2017], [Schulman et al., 2017]. The problem of these approaches is that they are very data consuming, in particular if the data is high dimensional. Furthermore, the more data is needed for reinforcement learning, the longer it takes to train. Learning to map data to compact representations is key to make reinforcement learning faster and more data efficient. Learning this mapping allows to produce state representations of data produced by sensors, in particular from cameras. A state representation is a low dimensional vector which characterizes the whole state of the data. Learning state representation can be seen as a dimensionality reduction method. Learning a state is implicitly done in the so-called end-to-end reinforcement learning. However, taking the task of representation learning away from reinforcement learning may accelerate reinforcement learning, make easier the training process, and make it more robust.

citations here

[Böhmer et al., 2015] presents the state of the art in terms of learning state representations for control. They show approaches based on deep auto-encoder networks followed by approaches on slow feature analysis (SFA), and it is particularly focused on learning state representations from images. Their vision of the notion of a good state representation is that it should:

- Be Markovian
- Be able to represent the true value of the current policy well enough for policy improvement
- Generalize the learned value-function to unseen states with similar futures

- Be low dimensional for efficient estimation

The rest of this article is structured as follows. First we present approaches which learn a representation for a time step t; secondly, we present works that use predictions in the future to learn a state representation, and finally, we present validation methods, frameworks and benchmarks used to assess the quality of the learned representations.

A broad branch of (generally unsupervised) approaches are mainly learned thanks to physics related or energy related (auto-encoders) functions, and they will be described in Section 2. Another possibility is to find the representation which makes possible to perform optimal actions or which maximizes rewards. The representations would then be learned thanks to interaction of the agent with the world and different reward functions. These representations can be learned through reinforcement learning algorithms, some of which will be summarized in section 3.

2 Unsupervised state representation learning

A common approach to teach agents useful tasks such as robotics control is via reinforcement learning. In this context, agents need to solve a task that requires one agent to consider a perspective view or perceptual state of the others. The concept of state can also be seen outside RL context, e.g., within the Theory of mind, which is the ability to assign distinct mental states (beliefs, intents, knowledge, etc.) to other members[1]. We focus on this paper on the former context mainly.

In the field of learning state representations and reinforcement learning for agents to interact and perform control in the environment, different strategies have been used. This section present strategies that learn a mapping from observation at time t o_t to a state representation s_t. The training strategies presented are unsupervised or weakly supervised. We support the believe that learning a state should not be supervised because the risk of overfitting is high [Bengio et al., 2012].

2.1 Learning states with Auto-encoders

In order to learn state representations, one common strategy is the use of auto-encoders. The architecture of an auto-encoder imposes a constraint of low dimensionality with a bottleneck between encoder and decoder. The assumption is that the best way to compactly synthesize an image is to learn a representation of the scene. Thus, the encoder E learns a mapping from the input image x to the state s_t, and the decoder D learns to reproduce the input given the state s_t to produce the output image $\hat{x}$. The loss function consists of the mean squared error between the input and the output, computed pixel-wise.

$$L = \| x - \hat{x} \|^2 \quad , \tag{1}$$

why this comma?

Using this loss function the encoder and decoder can be trained by gradient descent. The auto-encoder architecture can then be adapted to learn a particular representation. [Finn et al., 2015] use a spatial auto-encoder to learn spatial state representation of a controller. The representation learnt is used after to learn manipulation skills.

Unfortunately, even if theoretically the best solution is to learn a state representation, the training often leads to a "mean" image which optimizes quite well the loss for every input, focusing on mainly visual features and ignoring small but relevant ones [Lesort et al., 2017].

To make the training more robust to this kind of false optimization solution, denoising auto-encoders (DAE) [Vincent et al., 2008] can be used. This architecture adds noise to the input, and will make the "mean" image a worse solution than before; however, this architecture is tested in [van Hoof et al., 2016] to learn visual and tactile state representations. The authors compared state representations learned by a DAE and a variational auto-encoder (VAE) (see Section 2.2) by using the learned states in a reinforcement learning setting. They found that DAE state representations make possible to get less rewards than VAE state representation in most cases.

so can we show an positive example where DAE are worth using against VAE?

[1]Computational Neuroscience `http://neuro.cs.ut.ee/lab/`

2.2 Variational Learning

In the field of state representation learning, another solution explored is the use of variational inference [Kingma and Welling, 2013] [Jimenez Rezende et al., 2014] to learn representations of states. In this context, the variational inference is particularly used in a variational auto-encoder, as they learn to compress data into low dimensional representations.

A hidden parameters of the environment can be interpreted as a latent variable z of a distribution p that generates the input data x. We can find these latent variables by fitting $p(x,z)$ with a distribution $q(x,\hat{z})$. The variational approach aims to find this distribution q to extract a representation [Watter et al., 2015]. The problem is that $p(x)$ is intractable as well as $p(z|x)$, which makes the use of variational auto-encoders interesting.

why? if intractable, we need better motivation here, the previous sentence is also unclear (The problem is..)

This is because the variational auto-encoder learns an approximate distribution

hat?

$q(z|x)$ to fit the intractable true posterior $p(z|x)$. This distribution is called a probabilist encoder (or recognition model). It learns a mapping from an input to a representation $\hat{z}$. On the same way, $p(x|z)$ is called a probabilistic decoder: given a z, it produces a sample x of the distribution p. The marginal likelihood is composed by a sum over the marginal likelihoods of individual datapoints $log\, p(x^{(1)},...,x^{(N)}) = \sum_{i=1}^{N} \log p(x^{(i)})$, that can each be rewritten as:

$$\log p(x^{(i)}) = D_{KL}(q(z|x) \parallel p(x,z)) + \mathcal{L}(x^{(i)};\theta) \ , \tag{2}$$

Where θ indicates the parameters of p and q models. The first term is the KL divergence of the approximate from of the true posterior distribution. Since this KL-divergence is non-negative, the second term is called the variational lower bound on the marginal likelihood of datapoint i. The optimization of the lower bound can be achieved efficiently thanks to the re-parametrization trick of q(z|x) [Kingma and Welling, 2013], [Jimenez Rezende et al., 2014]. By re-arranging the lower bound we get:

$$\mathcal{L}(x^{(i)};\theta) = \mathbf{E}_{q(z|x)}[\log p(x,z) - \log q(z|x)] \leq \log p(x^{(i)}) \ , \tag{3}$$

$$= \mathbf{E}_{q(z|x)}[\log p(x|z) - D_{KL}(q(z|x) \parallel p(z))] \ , \tag{4}$$

The first term is called negative reconstruction error, while the second one is a regularizer error. The re-parametrization trick is an alternative method for generating samples from $q(z|x)$. Let be z a continuous random variable sampled from $q(z|x)$, we express z as a deterministic variable $z = g(\epsilon, x)$, where ϵ is auxiliary variable with independent marginal $p(\epsilon)$ and $g(.)$ is the function learned by the encoder.
The VAE implementation of this trick makes the encoder predict a vector of μ and σ. Then, ϵ is sampled as $\epsilon \sim \mathbf{N}(0, I)$. z is computed by:

$$z = \mu * \sigma + \mu \ , \tag{5}$$

which gives a fully differentiable way to produce a univariate Gaussian distribution $p(x|z) = \mathbf{N}(\mu, \sigma)$. In consequence, optimization of the lower bound is feasible by gradient descent.

[Watter et al., 2015] presented a VAE model called "Embedded To Control" (E2C) that consists of a deep generative model that learns to generate image trajectories from a latent space in which the dynamics are constrained to be locally linear. The latent space is dependent on the task. They evaluate their model on four visual tasks: an agent in a plane with obstacles, a visual version of the classic inverted pendulum swing-up task, balancing a cart-pole system, and controlling of a three-link arm with larger images. While [Watter et al., 2015] learned control policies in a single shot, based on data under an exploration policy, [van Hoof et al., 2016] aims, using also VAE, to learn iteratively on-policy. The authors want to train state encoders in a way that respects the transition dynamics of the controlled system. The transition dynamics used to learn a representation make [van Hoof et al., 2016] also use a forward model (see Subsection 3.1).

Variational inference is also used in Deep Kalman Filters [Krishnan et al., 2015] to construct a continuous nonlinear-state space model. Deep Kalman filters (DKF) are an evolution to Kalman filter that use variational methods to replace all the linear transformations with non-linear transformations parameterized by neural nets. [Krishnan et al., 2015] apply DKF to model the "Healing MNIST" dataset where long-term structure, noise and actions are applied to sequences of digits

to achieve what?what is the healing MNIST, the moving numbers? add few adjectives clarifying a bit

.

This approximation of the posterior has also been used to extend Kalman filters into deep variational Bayes Filter (DVBF) in [Karl et al., 2016]. From raw non-Markovian sequence data, the authors create a nonlinear state space. They demonstrate that this approach can be used to estimate states of a pendulum and a bouncing ball.

2.3 Learning with priors

As in any auto-encoder, the error in VAEs is computed in order to reduce the distance between the input and the output. The learned representation is then not necessarily related to the state we want to find. If the state is not a main visual feature of the reconstruction, an auto-encoder will not learn to reproduce it. Another approach is then to not use the reconstruction of the image as criterion, but to use knowledge we have about the state representation we want to learn and use it to guide our search within the data. This a priori knowledge is called "priors". As described in [Bengio et al., 2012], and contrarily to Bayesian probability in the context of state representation learning, a prior is not an a priori distribution but a priori knowledge we have about the world. Most priors are concerned about the physics of the world [Scholz et al., 2014]. This knowledge help to learn a dense and efficient representation about the environment. They can be used to learn low dimensional representation in a robotics set up [Jonschkowski and Brock, 2015], or in simulation [Jonschkowski et al., 2017].

Different priors have been proposed to address the problem of sharing knowledge with neural networks. There are often related to the time dimension. In fact, for instance, one of our best advantage against neural networks is our comprehension of the interaction between entities through time. Neural networks have a lot of difficulties with long term dependencies. Explaining such information that can be extracted from temporal patterns could significantly help a neural network to understand and make decisions. It could also reduce the complexity of the inference because there is a lot of coherence through time.

"Here is" is informal

Below is a list of the priors, and the works that formulate or use them.

- **Simplicity**
  This prior assumes that it exists a low dimensional representation of a higher level input. This low dimensional representation is the one we want to learn. This prior imposes a low dimensional state representation and is implemented in the architecture of

  the again which one? you mean MOST?

  neural networks where a bottleneck forces the representation to be compact.

- **Slowness Principle**
  The slowness principle assumes that interesting features fluctuating slowly and continuously through time and that a radical change inside the environment has low probability [Wiskott and Sejnowski, 2002, Kompella et al., 2011]. It's also called time coherence or the prior of

$$L_{Slowness}(D, \hat{\phi}) = \mathbf{E}[\| \Delta \hat{s}_t \|^2] \ , \tag{6}$$

  This prior can have other naming depending on the unit of s_t, for example time coherence (time) or inertia (velocity).

- **Variation**

of what? shall it be called (Spatial?) Variability to be consistent with namings across bullet points?

The assumption of this prior is that positions of relevant objects vary, and learning state representations should then focus on moving objects.

$$L_{variation}(D, \hat{\phi}) = \mathbf{E}[e^{-\|\hat{s}_{t1} - \hat{s}_{t2}\|}] \ , \tag{7}$$

$e^{-distance}$ is used as a similarity measure that is 1 if the distance among states is 0 and that goes to 0 with increasing distance between the position states

state positions? not sure if this is a prior by itself or a general loss function component

, which is exactly what we want

unfinished sentence: we want when?, what for?, in which case? if nothing else: 'we are generally interested in depending on the rewards or actions taken"? Refine a bit.

- **Proportionality**
  The proportionality prior assumes that for a same action, the reactions of these actions will have proportional amplitude or effect. The representation will then vary in the same amount for two equal actions in different situation.

$$L_{Prop}(D, \hat{\phi}) = \mathbf{E}[(\| \Delta\hat{s}_{t_2} \| - \| \Delta\hat{s}_{t_1} \|)^2 | a_{t_1} = a_{t_2}] \ , \tag{8}$$

- **Repeatability**
  Two identical actions applied at similar states should provide similar state variations, not only in magnitude but also in direction.

$$L_{Rep}(D, \hat{\phi}) = \mathbf{E}[e^{-\|\hat{s}_{t_2} - \hat{s}_{t_1}\|^2} \| \Delta\hat{s}_{t_2} - \Delta\hat{s}_{t_1} \|^2 | \ a_{t_1} = a_{t_2}] \ , \tag{9}$$

- **Causality**
  The causality prior is concerned with rewards, and assumes that if we have two different rewards for two same actions, then the two states should be differentiated or *placed apart* in the representation space.

$$L_{Caus}(D, \hat{\phi}) = \mathbf{E}[e^{-\|\hat{s}_{t_2} - \hat{s}_{t_1}\|^2} \mid a_{t_1} = a_{t_2}, r_{t_1+1} \neq r_{t_2+1}] \ , \tag{10}$$

- **Controllability**
  Controllable objects are relevant for state representation learning. The elements that can be controlled by the

which robot in particular? here you want to mean 'by robots'

robot are likely relevant for their task. If a robot acts by applying forces, controllable things could be those whose accelerations correlate with the actions of the robot. Accordingly, we can define a loss function per action dimension i to optimize the covariance between an action dimension i and the accelerations in a state dimension i. Related with this prior is the notion of *empowerment*, defined as an information-theoretic capacity of an agent's actuation channel to influence its own evolution, as a universal agent-centric measure of control. This concept is motivated by classical utility functions that suffer from having to be designed and tweaked on a use-case basis [Klyubin et al., 2005]. Encoding priors based on such concept, as well as the assignment problem [Sutton, 1998] is worth exploring when the environment becomes multi-agent.

$$L_{controllability}(D, \hat{\phi}) = e^{-cov(a_{t,i}, s_{t+1,i})} \ , \tag{11}$$

- **Consciousness**
  This prior could be combined with other priors in order to help disentangling abstract factors from each other. It relates to the notion of awareness at a particular time instant, as a *powerful constraint on the representation in that such low-dimensional thought vectors can correspond to statements about reality which are true, highly probable, or very useful for taking decisions* [Bengio, 2017]. The idea behind aims at making predictions in an abstract richer (but more compact with less dimensions) space than in the pixel space, as a natural mapping from facts or rules.

  how to use the Consciousness prior? Nat:This is future work and promising direction. DAVID, shall we use the prediction (is it a prediction or loss function?) function in Bengio slides?

Priors can be combined with other approaches such as the ones in [Finn et al., 2015], where the slowness principle is combined with the reconstruction of an auto-encoder. These priors can be used to learn representations in high dimensions in similar manners to how it is done in [Stewart and Ermon, 2016]; however, its main objective is beyond the scope of this paper.

2.4 Adversarial Learning

[Chen et al., 2016] propose to use the Generative Adversarial Network (GAN) framework to learn state representations. They present a model named InfoGAN that achieves the disentanglement of latent variables, especially in the context of this paper, on 3D pose of objects. GANs have been presented by [Goodfellow et al., 2014]. As described in [Chen et al., 2016], the goal is to learn a generator distribution $P_G(x)$ that matches the real distribution $P_{data}(x)$. Instead of trying to explicitly assign probability to every x in the data distribution, GANs learn a generator network G that generates samples from the generator distribution P_G by transforming a noise variable $z \sim P_{noise}(z)$ into a sample $G(z)$. This generator is trained by playing against an adversarial discriminator network D that aims to distinguish between samples from the true distribution P_{data} and the generator distribution P_G. They do so by maximizing the mutual information between a fixed small subset of the GAN's noise variables and the observations, which turns out to be relatively straight forward.

need to add other GAN model, probably Bigan for example- see GANs.md

Adversarial Feature Learning [Donahue et al., 2016] presents an extension of regular GANS to learn the inverse mapping: projecting data back into a latent space that allows the learned feature representation to be useful for auxiliary supervised discrimination tasks, and to be competitive with unsupervised and self-supervised feature learning.

3 Learning by Predicting

Theoretical and empirical proofs show that inference for a latent state should be performed using information from its future (i.e., looking at the true posterior), against recent work which mainly performs inference using only information from the past [Chung et al., 2015, Krishnan et al., 2017]. [Krishnan et al., 2017] introduce a new family of structured inference networks (parameterized by recurrent neural networks) that, by using Gaussian State Space Models, considers both inference and learning in a class of latent state variable that serves as a summary statistic for information from the past.

A broad branch of approaches focuses on reinforcement learning by using self-supervised approaches, i.e., using the future to either predict the next state to be reached after performing a given action (forward models), or predict the action needed to reach a desired future state (inverse models). However, predicting other environment signals such as visual or computational features are proxies shown to improve RL [Jaderberg et al., 2016]. This section summarizes this broad spectrum of research.

3.1 Predicting next state: Forward Models

In the reinforcement learning framework the basic elements are based on tuples (o_t, s_t, a_t, r_t) where from observation o_t, one can learn a corresponding abstract state s_t in an embedded space, in which performing action a_t returns reward r_t. In this section we present approaches that learn from an observation o_t and an action a_t to predict the next future state $\hat{s}_{t+1}$. This kind of approach learns what is called a *forward* model. In other words, given state and action taken in time t, predict next future state at $t+1$ that that action will lead us to. Formally:

$$\hat{s}_{t+1} = f(s_t, a_t; \theta_{fwd}) \tag{12}$$

Predicting context is critical for tackling unsupervised learning and being able to generate content. In unsupervised learning, self-supervised training approaches can be used to improve data efficiency without sacrificing return. They are approaches that in lack of labels, they make a pseudo-label out of the prediction error that comes from trial and error prediction of interactions with the environment. Self-supervised approaches can optimize tasks independent of reward, such as dynamics and inverse dynamics-related tasks. One example of this approach is predicting loss [Shelhamer et al., 2017], which can be used as a reward when extrinsic rewards are unavailable. In [Shelhamer et al., 2017], self-supervised auxiliary losses extend the limitations of traditional RL to learn from all experience, whether rewarded or not. Self-supervised pre-training and joint optimization using auxiliary losses in the absence of rewards improve the data efficiency and policy returns of end-to-end reinforcement learning. Self-supervision improvement is strongest early in training when the pre-training and policy distributions are close [Shelhamer et al., 2017][2].

[Oh et al., 2017] integrate model-free and model-based RL methods into a single neural network. In contrast to typical model-based RL methods, VPN learns a dynamics model whose abstract (discrete) states are trained to make option-conditional predictions of future values (discounted sum of reward, discount and the value of next state) rather than predictions of future observations. VPN has several advantages over both model-free and model-based baselines in a stochastic environment where careful planning is required but building an accurate observation-prediction model is difficult. Because they outperform Deep Q-Network (DQN) on several Atari games with short-lookahead planning, it can be a potential new way of learning state representations. A set of similar approaches integrating learning and planning in one architecture exist. One is the RNN-based *Predictron* [Silver et al., 2016], which works in uncontrolled settings but requires policy evaluation. Another one is Dyna-Q [Sutton, 1990], but for continuous control problems using model predictive control (MPC) [Lenz et al.,], or CNN-based Value Iteration Networks (VIN) [Tamar et al., 2016], which is constrained to lower dimensionality state spaces.

Another interesting related area concerns learning hierarchical intention states with *Multiple Spatio-Temporal Scales RNN*-based predictive coding [Choi and Tani, 2017, Ahmadi and Tani, 2017].

3.2 Predicting next action: Inverse Models

Without leaving the RL setting, and within a self-supervision paradigm of exploring, in order to learn about the environment, we can turn the forward model problem around and instead of learning a representation of states (given previous states and actions), use the states to predict actions.

In an inverse model, given a current state s_t and the next desired state s_{t+1}, we learn with supervised learning to predict the necessary action a_{t+1} to reach such future state [3]. This is a common problem in planning and navigation in AI, where the goal we want to reach is given and we need to find the actions that take us there. More formally, we learn $\hat{a}_t$:

$$\hat{a}_t = g(s_t, s_{t+1}; \theta_{inv}) \tag{13}$$

Equations 12 and 13 summarize forward and inverse models, where s_t and a_{t+1} are the world state and action applied time step t; s_{t+1}, a_{t+1} are the predicted state and actions, and θfwd and $Winv$ are parameters of the functions f and g that are used to construct the forward and inverse models.

Inverse models can also be learned via active learning [Baranes and Oudeyer, 2013] with intrinsically motivated goal exploration in robots. Connexions among the latter two families of models

[2]The policy gradient is augmented with auxiliary gradients from what is called *self-supervised* tasks. [Shelhamer et al., 2017] accuses the *gap between the initial optimization and recovery as a representation learning bottleneck in systems that are not end-to-end*

[3]Note that Inverse models are different from inverse RL, where rewards are generally unavailable and given states and actions, a reward function is learned. This area of research has also exploited for transferring learning from simulation to real world, through learning deep inverse dynamics models [Christiano et al., 2016]. However, we focus on the former, where predicting the action from states can be used to improved the state representation learning

exists: forward models can regularize inverse dynamics model. An example using both forward and inverse models is the Intrinsic Curiosity Module (ICM) [Pathak et al., 2017], which helps agents explore and discover the environment out of curiosity when extrinsic rewards are spare or not present at all, and integrates both an inverse and forward model where the 'surprise' element in the action prediction is used as signal for the forward model. This is an interesting way to bypass the hard problem of predicting pixels, unaffected by unpredictable aspects of the environment that do not affect the agent. In the inverse model, the neural network with parameters θ_{inv} is trained to optimize:

$$Loss_{inv} = min_{\theta_{inv}} L_{inv}(\hat{a_t}, a_t) \tag{14}$$

where, L_{inv} is the loss function that measures the discrepancy between the predicted and actual actions. The intrinsic reward signal is computed from the loss of the forward model:

$$Loss_{fwd} = Loss_{fwd}(\phi(s_t), \hat{\phi}(s_{t+1})) = \frac{1}{2} \parallel \hat{\phi}(s_{t+1}) - \phi(s_{t+1}) \parallel_2^2 \tag{15}$$

Their most interesting result is that using an observation space for computing curiosity poorly predicts the best action (in an inverse model) and is significantly worse than learning an embedding. When comparing with an architectures without inverse models (and add deconvolution layers to the forward model), the uncontrolable parts of the environment are ignored and in this way, ICM succeeds on harder exploration tasks that become progressively harder for a baseline such as A3C.

would be worth adding image in their website or avoid because of potential copyrights (since it its in their website may be ok? Also shall this description be added? from Shelhammer, literally, below:

3.3 Predicting next observation

A branch of approaches learns a mapping from observation o_t to a state s_{t+1} in a supervised manner. Even if the learning criterion does not depend on a state, the architecture of the model allows to extract a state representation as it happens by using auto-encoders.

Predictive models in general have shown success in robotics, health-care, and video understanding applications [Vondrick and Torralba, 2017]. Explicitly disentangling the model's memory from the prediction (by, e.g., transforming pixels in the past as in [Vondrick and Torralba, 2017] to generate short videos of plausible futures) helps the model learn desirable invariance.

In unsupervised learning contexts, since the acquisition of labeled data becomes increasingly expensive and impractical, it is possible to learn about physical interaction dynamics with action-conditioned video prediction models [Finn et al., 2016]. These use pixel motion to predict a distribution over it from previous frames and in this way produce video predictions.

3.4 To be INCLUDED OR NOT? Predicting the reward: Inverse Reinforcement Learning

In inverse reinforcement learning (IRL), no reward function is given; instead, the reward function is inferred given an observed behavior from an expert. The idea is to mimic the observed behavior that is often optimal or close to optimal [Ng et al.,]. One example, when rewards are not present or are scarceis *Compatible Reward IRL* [Metelli et al., 2017], where a model-free approach is presented. It does not require to specify a function space where to search for the expert's reward function; the algorithm generates basis functions that span the subspace that make the gradient vanish, and use this subspace to penalize the policies that deviate the most from the expert's policy. This approach can outperform behavioral cloning and true reward function in finite and continuous cases.

4 Current Research and Challenges

The challenge of high dimensionality spaces of actions [4] or continuity in action spaces are examples of difficult to tackle issues found in the current trends on representation learning. Some other challenges and current trends on reinforcement learning include:

[4]Approaches in large discrete action spaces have been sought, such as in [Dulac-Arnold et al., 2015]

we should keep only challenges related to the survey

- Generalizing from simulation [Peng et al., 2017] (the *reality gap*) and Simulation-to-real transfer learning [Tobin et al., 2017, Peng et al., 2017]. This issue concernts data sampling efficiency, since training in simulation is faster and less costly than on reality. A strategy that is helping speed up state representation learning is using more data sample efficient approaches such as demonstration learning [Hester et al., 2017].
- Including options [Machado et al., 2017a] and goal exploration processes [Forestier et al., 2017] as a way to improve state representation learning.
- Augmenting the dimensionality and continuity of the input observation space [Lesort et al., 2017].
- Adversarial feature learning [Donahue et al., 2016, Fu et al., 2017] for state representation learning.
- Unsupervised [Jaderberg et al., 2016] or self-supervised predictive [Shelhamer et al., 2017] auxiliary tasks as proxies supporting state representation learning. In the same arena, predicting the future as a proxy to learning to act [Dosovitskiy and Koltun, 2016, Oh et al., 2017].
- Integrating state representation learning on multi-task [Kulkarni et al., 2016, Andreas et al., 2017, Wilson et al., 2007, Teh et al., 2017], multi-agent [Foerster et al., 2016, Foerster et al., 2017], hierarchical [Wilson et al., 2007] and model-agnostic meta-learning [Finn et al., 2017] settings, e.g., involving human-robot value alignment as in cooperative inverse reinforcement learning (CIRL, a different notion of inverse RL [Hadfield-Menell et al., 2016]) and applications beyond robotics domains (see [Li, 2017, Kaelbling et al., 1996]).
- Full integrations of action, perception and learning (e.g., as done in [Le Goff et al., 2017] to produce a saliency map) shall be blended into a single pipeline process including both priors-based state representation learning and RL phases.
- Reward engineering. The need for handcrafting reward functions in RL is a current challenge impeding transfer learning. Automatic reward acquisition in high-dimensional settings with unknown dynamics is hard in practice, and approaches such as AIRL (Adversarial Inverse RL) are robust first steps [Fu et al., 2017] towards this goal.

In the area to tackle the "reality gap", i.e., the challenge of training in simulation for testing in reality, recent approaches are beating the state of the art, for instance [Peng et al., 2017].

Other latest approaches to bridge this gap use successful techniques such as domain randomization [Tobin et al., 2017] and adaptation [Bousmalis et al., 2017] but also, on top of that, dynamics randomization [Peng et al., 2017], which allows learning in real life with only data from (differently calibrated and sensor dynamics) simulators. In an object-pushing task, the authors demonstrate that training exclusively in simulation, allows to learn policies that are robust to calibration errors while maintaining performance levels.

Unfortunately, many of the successful approaches on transferring simulation to real life control and RL use methods that require motion capture devices for setting and tracking the goal when either the goal itself changes, or when the target object's relative position varies in between episodes. Often, these systems draw on motion capture systems such as *Optitrack* (for e.g. teaching Baxter robot a highly diverse repertoire of tasks including throwing a ball into a basket [Kim and Doncieux, 2017]) and *PhaseSpace* mocap system [Peng et al., 2017] (to determine that the goal should be considered satisfied if the puck being pushed is within a certain distance of the target). State representation learning could help to cross the reality gap for RL by using high level representation.

4.1 Promising Deep Reinforcement Learning Approaches for state representation learning

Research directions worth exploring include neuroscience-inspired AI [Hassabis et al., 2017] approaches as well as hybrid neural-symbolic [Garnelo et al., 2016] approaches, where learning from scratch, as is a common recent norm, is not the trend any more, but starting with pre-wired basic relationships, can allow better online-learning and notions such as *compositionality* in concept learning. One interesting approach in this area is the generative vision model which can be highlighted for his high data efficiency that is able to break text-based CAPTCHAs [George et al., 2017].

Likewise, adversarial approaches to feature learning must be explored for state representation learning [Donahue et al., 2016], as well as how to integrate options when multi-goal settings are present. Promising directions are in [Machado et al., 2017a], where they implicitly learn and discover options (acting at different time scales) through using Proto-value functions (PVFs, an approach for representation learning in MDPs) and *eigenpurposes*, intrinsic reward functions derived from the learned representations.

5 Validation Methods, Frameworks and Benchmarks

This section discusses application contexts, metrics and datasets used in the quality evaluation of states learned.

5.1 Interpretability and evaluation of learned representations

When it comes too give a quantitative value which estimates the quality of a representation, in the context of representation learning this can be harder than expected. The assessment should show if the representation we learned is conform to what we expect. While toolboxes to benchmark the robustness of machine learning models exists [Rauber et al., 2017], understanding intermediate layers during the development of a learning methods can be challenging. An approach in this direction is using linear classifier probes [Alain and Bengio, 2016], as well as using class-enhanced attentive responses [Kumar et al., 2017]. The latter enables the visualization of the activations for the most dominant classes. Other way of assessing dimensionality reduction algorithms is in terms of their loss of quality [Gracia et al., 2014], while adversarial methods can be tested with a suite of adversarial attacks [Rauber et al., 2017]. For instance, Foolbox[5] creates adversarial examples that fool neural networks with gradient-based and blackbox attacks.

The deep reinforcement learning (DRL) community has made available a large set of software benchmarks to test RL algorithms [Arulkumaran et al., 2017]. Despite the growing interest in making robotic applications work in real life settings, simulation environments are the de-facto start point and where most of the algorithms design, learning and evolution happens. Within the robotics domain, most used middleware is open-source: ROS, YARP, OROCOS, Player, etc. and some, also cross-platform [Ivaldi et al., 2014]. The top 7 most currently used simulators, in decreasing order, are Gazebo, ODE, Bullet, OpenRave, V-Rep, XDE, and Blender. Examples of most recently added RL Gym environments to PyBullet[6] Minitaur quadruped, MIT racecar, KUKA grasping, Ant, Hopper, Humanoid etc. These simulation environments are mainly designed for RL, but they could equally be used to test state representation algorithms.

While this choice allows roboticists to pick the best middleware for their needs, it also makes comparing AI models very challenging, since dynamics models and simulators' performance comparison easily introduces noise, calibration issues and instabilities that are susceptible of making the algorithms sensible to them.

Despite the above being the best softwares upon user ratings [Ivaldi et al., 2014], the challenge of not having reproducible datasets for robotics both in simulation and real live makes these simulators practically unusable as platforms to support research benchmarks. This is why most research builds on RL toy tasks or the more broad set of Atari 2600 video games from the Arcade Learning Environment (ALE) [Bellemare et al., 2013, Machado et al., 2017b]. However, the further development of similar standard datasets and benchmarks for robotic control tasks is a crucial gap yet to filled.

Examples of challenging games that involve further planning and strategic thinking include Freeway and Montezuma's revenge. *Freeway* is a game in which a chicken is expected to cross the road while avoiding cars, well suited to observe options in which the agent clearly wants to reach a specific lane in the street, since a random walk over primitive actions does not explore the environment properly as shown in [Machado et al., 2017a]. In Montezuma's revenge, the objective

[5]https://pypi.python.org/pypi/foolbox
[6]http://pybullet.org

is to navigate a room to pickup a key to open a door, and thus, option discovery and valuation is key (see [Kulkarni et al., 2016] and [Machado et al., 2017a]).

5.2 Validation metrics

This section provides a listing on metrics and embedding quality evaluation techniques used across the literature. These are summarized by table 1.

Table 1: State representations quality evaluation criteria

Metric	**Evaluation target**	**Context/Observations**
Disentanglement metric score [Higgins et al., 2016]	Data-generating latent factor disentanglement	Transfer learning
Distortion [Indyk, 2001]	Preservation of local and global geometry coherence	Unsupervised Representation learning
NIEQA [Zhang et al., 2012]	Normalization Independent Embedding Quality Assessment	Manifold learning

Visual assessment of the representation's quality can be done using a Nearest-Neighbors approach as in [Sermanet et al., 2017] for example, or in [Pinto et al., 2016]. The idea is to look at the Nearest Neighbors in the learned representation space, and for each neighbor, retrieve their corresponding image and associated ground truth state-representative dimension we are interested in measuring.

Specially when learning state representations with priors, since we want to impose local coherence (especially the temporal prior), a good representation should have local coherence, and therefore, the associated ground truth states should be close. While the nearest neighbor coherence can be assessed visually, KNN-MSE is a quantitative metric from this information[Lesort et al., 2017]. Using the ground truth value for every image, it measures the distance between the value of the original image and the value on the nearest neighbor images retrieved in the learned state space[7].

For an image I, KNN-MSE is computed as follows:

$$\text{KNN-MSE}(I) = \frac{1}{k} \sum_{I' \in KNN(I,k)} ||\phi(I) - \phi(I')||^2 \tag{16}$$

where KNN(I, k) returns the k nearest neighbors of I in the learned state space and $\phi(I)$ gives the ground truth (s) associated to I.

Other metrics comparable to KNN-MSE are distortion [Indyk, 2001] and NIEQA [Zhang et al., 2011]; both share the same principle as two quantitative measures of the global quality of a representation: the representation space should, as much as possible, be an undistorted version of the original space.

NIEQA (Normalization Independent Embedding Quality Assessment) [Zhang et al., 2012] is a more complex evaluation that measures the local geometry quality and the global topology quality of a representation. NIEQA local part checks if the representation is locally equivalent to an Euclidean subspace that preserves the structure of local neighborhoods. NIEQA objectives are aligned with KNN-MSE [Lesort et al., 2017], as a measure to assess the quality of the representation, especially locally. The global NIEQA measure is also based on the idea of preserving original structure in the representation space, but instead of looking at the neighbors, it samples "representative" points in the whole state space. Then, it considers the preservation of the geodesic distance between those points in the state space.

Regarding the assessment of next observation's predictions, complementary feature learning strategies beyond MSE include multi-scale architectures, adversarial training methods, and image gradient difference loss functions as proposed in [Mathieu et al., 2015]. More concretely, the *Peak Signal to Noise Ratio, Structural Similarity Index Measure* and *image sharpness* show to be the proxies for next frame prediction assessment [Mathieu et al., 2015]. Architectures and losses may

[7]A low distance means that a neighbor in the ground truth is still a neighbor in the learned representation, and thus, local coherence is conserved

be used as building blocks for more sophisticated prediction models, involving memory and recurrence. Because unlike most optical flow algorithms, the model is fully differentiable, it can be fine-tuned for task transfer learning.

I havent added these to the table because they evaluate next observation, not the state representation formally, not sure if makes sense to add any

5.3 Datasets and benchmarks

Datasets used to validate state representation learning include varied, but mainly simulated environments. Unlike in image recognition challenges where MNIST digits or ImageNet dataset prevail, in state representation learning, a varied set of regular video games or visuomotor tasks in robotics can be found as a test suite for robotics control.

We list in this subsection a compilation of benchmarks used in the cited papers, from classic RL test beds to the latest most challenging Atari games. Evaluation benchmarks found evaluated in mentioned papers in this survey include Mario Bros, octopus game, labyrinths, mazes, driving cars, or controlled open source simulated RL environments such as the inverted and regular pendulum, car in mountain scenario, or the bouncing ball. Many of the latter are part of the Universe and OpenAI Gym[8][Brockman et al., 2016] or DeepMind Labs [Beattie et al., 2016]. Few of them involve a 3D input space; e.g. VizDoom (as used in the Intrinsic Curiosity Module of [Pathak et al., 2017]) or in the robotics domain, Baxter robot performing tasks such as pushing a button [Lesort et al., 2017] or poking objects [Agrawal et al., 2016].

Some of the most challenging games being tackled at the moment are those that require ahead thinking in order to learn states, actions and a more longer term planning. These include the road-crossing chicken game [Machado et al., 2017a] or Montezuma's Revenge Atari game, which happens to be the one with the worse performance when it comes to human-level control by AI agents [Mnih et al., 2015]. A comprehensive benchmark on DRL for continuous control is in [Duan et al., 2016], including locomotion and hierarchical tasks as main testbeds: Hopper, 2D Walker, Half-Cheetah, Simple/Full Humanoid, Cart-Pole Balancing, Inverted Pendulum, Mountain Car, Acrobot, Swimmer + Gathering/Maze task, and Ant + Gathering task. The benchmarked algorithms in continuous control include batch algorithms such as Truncated Natural Policy Gradient (TNPG), Reward-Weighted Regression (RWR), Trust Region Policy Optimization (TRPO), Cross Entropy Method (CEM), Relative Entropy Policy Search (REPS) and Covariance Matrix Adaptation Evolution Strategy (CMA-ES), as well as on-line algorithms such as Deep Deterministic Policy Gradient (DDPG). These are all mainly deep end-to-end approaches; however, learning an intermediate state representation could help boosting performance. Benchmarking tasks used in the most prominent state representation learning literature is summarized in Table 2.

Because choosing the state dimensionality or number of features a priori can be challenging, a promising direction avoiding this step is using return-based feature construction without relying on the value function to select the optimal action (at the cost of fixing the explained variance) [Parisi, 2017].

6 Discussion

This survey discussed different points of view and approaches to tackle the learning of state representations in unsupervised or semi-supervised manners. The following table (Table 3) summarizes the main particularities of each main work, including objective functions and mechanisms the current state of the art uses in order to achieve this goal. When the learning of the states is not done directly, we indicate if a proxy element is predicted in order to aid the learning of the state representations. We can see that the vast majority of approaches focus on end-to-end approaches for very specific tasks, although a varied set of works in the literature are nowadays blending different of these approaches at once to improve performance, the quality of the representations, their interpretability, or most importantly, their transferability to other tasks. Table 3 schematically summarizes the main works discussed with the hope of helping provide more contextual grounding and intuition to the different approaches, as well as intermediate proxies predicted, or used in the task of state representation learning.

[8]OpenAI Gym environment classic control problems in RL: `https://gym.openai.com/envs`

Table 2: Benchmark and Tasks Used to evaluate State Representation Learning Algorithms

Task/Benchmark	**Models using it**	**Observations**
1)Pushing and (Baxter) poking objects and 2)Robotic pushing dataset	1)[Peng et al., 2017, Agrawal et al., 2016], 2) SV2P [Babaeizadeh et al., 2017], video pixel networks (VPN) [Kalchbrenner et al., 2016, Reed et al., 2017]	Motor babbling be used for unsupervised learning of object saliency maps [Le Goff et al., 2017]
Robotic grasping	Using domain adaptation and simulation [Bousmalis et al., 2017]	
Cart-pole	[Watter et al., 2015], PVE [Jonschkowski et al., 2017]	
Ball in cup	[Kim and Doncieux, 2017],PVE [Jonschkowski et al., 2017]	
Inverted pendulum	[Watter et al., 2015], PVE [Jonschkowski et al., 2017]	
Dynamic Pendulum	Deep Variational Bayes Filters Unsupervised learning of state space models from raw data [Karl et al., 2016]	
Driving car 2D	[Jonschkowski and Brock, 2015]	
Shopping Cart Task	Physics based model prior [Scholz et al., 2014]	task is to push a shopping-cart to the goal
Apartment Rearrangement Task	Physics based model prior[Scholz et al., 2014]	task is a multi-object rearrangement problem in a simulated apartment
Control of a three-link arm	[Watter et al., 2015]	
The 2-link arm problem	[Munk et al., 2016]	
The octopus problem	[Munk et al., 2016]	
A real-robot manipulation task based on tactile feedback	[van Hoof et al., 2016]	
A PR2 robot on tasks including pushing a free-standing toy block	[Finn et al., 2015]	picking up a bag of rice using a spatula, and hanging a loop of rope on a hook at various positions
Bouncing ball	[Karl et al., 2016]	
NORB dataset of generated simulated movies	[Goroshin et al., 2015]	
simulated high-dimensional, vision-based mobile robot planning task	[Boots et al., 2009]	
Super Mario Bros & VizDoom	[Pathak et al., 2017]	2 and 3D resp.
Baxter pushing button RL with Image benchmark for robotic priors	[Lesort et al., 2017]	3D
Montezuma's revenge	[Machado et al., 2017a]	
Freeway (chicken crossing road)	[Machado et al., 2017a]	
Tetherball game (robot has to hit a ball hanging from a pole without giving the opponent the chance to unwind it)	Used in reward-weighted PCA (rwPCA) goal-driven algorithm [Parisi, 2017]	
(finite) Taxi domain and (continuous) Linear Quadratic Gaussian Regulator and Car on the Hill environments	Reward Compatible IRL [Metelli et al., 2017]	
Atari games and (determ. and stochastic -reward position change-) Collect Domain	VPN [Oh et al., 2017] blending learning and planning using lookahead for discrete control problems	*Collect* Objective: to pick as much rewards as possible in minimal time.

Table 3: Approaches with proxy tasks used for state representation learning: *Proxy*, *Reward* and *Time* columns indicate if the approach uses rewards (*Rw*), uses time-based observations (*Time*), or a third element proxy (*Proxy*) in order to learn the state (versus learning the state directly such as e.g., VAEs).

Model and Ref.	**Supervised**	**Objective funct.**	**Uses Re-ward**	**Uses Proxy**	**Time**	**Observations**	**Requires prior knowledge**
VAE [Kingma and Welling, 2013]	No	Variational free energy	no	0	0	yes	
InfoGAN [Chen et al., 2016]	No	Mutual information	no	0	0	scalable	yes
DC-IGN [Kulkarni et al., 2015]	Semi		no	0	0	yes	
betaVAE [Higgins et al., 2016]	Semi	Variational free energy [a]	no	0	0	stable	No
PVE [Jonschkowski et al., 2017]	No		no	0	0		yes
Loss as reward [Shelhamer et al., 2017]	No	Auxiliary task loss	No	Aux. tasks	Yes	Augments policy gradient with aux. gradients from self-supervised tasks	No

[a][Jordan et al., 1999] if b=1

In similar ways as priors about the physical world can help learning state representations [Jonschkowski and Brock, 2015, Lesort et al., 2017], and learning policies guided by sketches and curriculum learning improves multi-task learning with shared policies for discrete and continuous control and sparse rewards [Andreas et al., 2017], integrating broader state representation learning strategies into RL policy learning is a latent topic of interesting research.

7 Acknowledgements

This research is funded by the DREAM project under the European Union's Horizon 2020 research and innovation program under grant agreement No 640891.

References

[Agrawal et al., 2016] Agrawal, P., Nair, A. V., Abbeel, P., Malik, J., and Levine, S. (2016). Learning to poke by poking: Experiential learning of intuitive physics. In *Advances in Neural Information Processing Systems*, pages 5074–5082.

[Ahmadi and Tani, 2017] Ahmadi, A. and Tani, J. (2017). How can a recurrent neurodynamic predictive coding model cope with fluctuation in temporal patterns? robotic experiments on imitative interaction. *Neural Networks*.

[Alain and Bengio, 2016] Alain, G. and Bengio, Y. (2016). Understanding intermediate layers using linear classifier probes. *ArXiv e-prints*.

[Andreas et al., 2017] Andreas, J., Klein, D., and Levine, S. (2017). Modular multitask reinforcement learning with policy sketches. *arXiv preprint arXiv:1611.01796*.

[Arulkumaran et al., 2017] Arulkumaran, K., Deisenroth, M. P., Brundage, M., and Bharath, A. A. (2017). A brief survey of deep reinforcement learning. *CoRR*, abs/1708.05866.

[Babaeizadeh et al., 2017] Babaeizadeh, M., Finn, C., Erhan, D., Campbell, R. H., and Levine, S. (2017). Stochastic Variational Video Prediction. *ArXiv e-prints*.

[Baranes and Oudeyer, 2013] Baranes, A. and Oudeyer, P.-Y. (2013). Active learning of inverse models with intrinsically motivated goal exploration in robots. *Robotics and Autonomous Systems*, 61(1):49–73.

[Beattie et al., 2016] Beattie, C., Leibo, J. Z., Teplyashin, D., Ward, T., Wainwright, M., Küttler, H., Lefrancq, A., Green, S., Valdés, V., Sadik, A., Schrittwieser, J., Anderson, K., York, S., Cant, M., Cain, A., Bolton, A., Gaffney, S., King, H., Hassabis, D., Legg, S., and Petersen, S. (2016). Deepmind lab. *CoRR*, abs/1612.03801.

[Bellemare et al., 2013] Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. (2013). The arcade learning environment: An evaluation platform for general agents.

[Bengio, 2017] Bengio, Y. (2017). The consciousness prior. *CoRR*, abs/1709.08568.

[Bengio et al., 2012] Bengio, Y., Courville, A. C., and Vincent, P. (2012). Unsupervised feature learning and deep learning: A review and new perspectives. *CoRR*, abs/1206.5538.

[Boots et al., 2009] Boots, B., Siddiqi, S. M., and Gordon, G. J. (2009). Closing the learning-planning loop with predictive state representations. *CoRR*, abs/0912.2385.

[Bousmalis et al., 2017] Bousmalis, K., Irpan, A., Wohlhart, P., Bai, Y., Kelcey, M., Kalakrishnan, M., Downs, L., Ibarz, J., Pastor, P., Konolige, K., et al. (2017). Using simulation and domain adaptation to improve efficiency of deep robotic grasping. *arXiv preprint arXiv:1709.07857*.

[Brockman et al., 2016] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. (2016). Openai gym. *CoRR*, abs/1606.01540.

[Böhmer et al., 2015] Böhmer, W., Springenberg, J. T., Boedecker, J., Riedmiller, M., and Obermayer, K. (2015). Autonomous learning of state representations for control: An emerging field aims to autonomously learn state representations for reinforcement learning agents from their real-world sensor observations. *KI - Künstliche Intelligenz*, pages 1–10.

[Chen et al., 2016] Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., and Abbeel, P. (2016). Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2172–2180.

[Choi and Tani, 2017] Choi, M. and Tani, J. (2017). Predictive coding for dynamic visual processing: Development of functional hierarchy in a multiple spatio-temporal scales RNN model. *CoRR*, abs/1708.00812.

[Christiano et al., 2016] Christiano, P., Shah, Z., Mordatch, I., Schneider, J., Blackwell, T., Tobin, J., Abbeel, P., and Zaremba, W. (2016). Transfer from simulation to real world through learning deep inverse dynamics model. *CoRR*, abs/1610.03518.

[Chung et al., 2015] Chung, J., Kastner, K., Dinh, L., Goel, K., Courville, A. C., and Bengio, Y. (2015). A recurrent latent variable model for sequential data. In *Advances in neural information processing systems*, pages 2980–2988.

[Donahue et al., 2016] Donahue, J., Krähenbühl, P., and Darrell, T. (2016). Adversarial feature learning. *CoRR*, abs/1605.09782.

[Dosovitskiy and Koltun, 2016] Dosovitskiy, A. and Koltun, V. (2016). Learning to act by predicting the future. *arXiv preprint arXiv:1611.01779.*

[Duan et al., 2016] Duan, Y., Chen, X., Houthooft, R., Schulman, J., and Abbeel, P. (2016). Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*, pages 1329–1338.

[Dulac-Arnold et al., 2015] Dulac-Arnold, G., Evans, R., Sunehag, P., and Coppin, B. (2015). Reinforcement learning in large discrete action spaces. *CoRR*, abs/1512.07679.

[Finn et al., 2017] Finn, C., Abbeel, P., and Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. *CoRR*, abs/1703.03400.

[Finn et al., 2016] Finn, C., Goodfellow, I., and Levine, S. (2016). Unsupervised learning for physical interaction through video prediction. In *Advances in Neural Information Processing Systems*, pages 64–72.

[Finn et al., 2015] Finn, C., Tan, X. Y., Duan, Y., Darrell, T., Levine, S., and Abbeel, P. (2015). Deep spatial autoencoders for visuomotor learning. *CoRR*, abs/1509.06113.

[Foerster et al., 2016] Foerster, J. N., Assael, Y. M., de Freitas, N., and Whiteson, S. (2016). Learning to communicate with deep multi-agent reinforcement learning. *CoRR*, abs/1605.06676.

[Foerster et al., 2017] Foerster, J. N., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. (2017). Counterfactual multi-agent policy gradients. *CoRR*, abs/1705.08926.

[Forestier et al., 2017] Forestier, S., Mollard, Y., and Oudeyer, P. (2017). Intrinsically motivated goal exploration processes with automatic curriculum learning. *CoRR*, abs/1708.02190.

[Fu et al., 2017] Fu, J., Luo, K., and Levine, S. (2017). Learning Robust Rewards with Adversarial Inverse Reinforcement Learning. *ArXiv e-prints.*

[Garnelo et al., 2016] Garnelo, M., Arulkumaran, K., and Shanahan, M. (2016). Towards deep symbolic reinforcement learning. *arXiv preprint arXiv:1609.05518.*

[George et al., 2017] George, D., Lehrach, W., Kansky, K., Lázaro-Gredilla, M., Laan, C., Marthi, B., Lou, X., Meng, Z., Liu, Y., Wang, H., Lavin, A., and Phoenix, D. S. (2017). A generative vision model that trains with high data efficiency and breaks text-based captchas. *Science.*

[Goodfellow et al., 2014] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. pages 2672–2680.

[Goroshin et al., 2015] Goroshin, R., Mathieu, M., and LeCun, Y. (2015). Learning to linearize under uncertainty. *CoRR*, abs/1506.03011.

[Gracia et al., 2014] Gracia, A., González, S., Robles, V., and Menasalvas, E. (2014). A methodology to compare dimensionality reduction algorithms in terms of loss of quality. *Information Sciences*, 270:1–27.

[Hadfield-Menell et al., 2016] Hadfield-Menell, D., Russell, S. J., Abbeel, P., and Dragan, A. (2016). Cooperative inverse reinforcement learning. In *Advances in neural information processing systems*, pages 3909–3917.

[Hassabis et al., 2017] Hassabis, D., Kumaran, D., Summerfield, C., and Botvinick, M. (2017). Neuroscience-inspired artificial intelligence. *Neuron*, 95(2):245–258.

[Hester et al., 2017] Hester, T., Vecerik, M., Pietquin, O., Lanctot, M., Schaul, T., Piot, B., Sendonaris, A., Dulac-Arnold, G., Osband, I., Agapiou, J., Leibo, J. Z., and Gruslys, A. (2017). Learning from demonstrations for real world reinforcement learning. *CoRR*, abs/1704.03732.

[Higgins et al., 2016] Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. (2016). beta-vae: Learning basic visual concepts with a constrained variational framework.

[Indyk, 2001] Indyk, P. (2001). Algorithmic applications of low-distortion geometric embeddings. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 10–33. IEEE.

[Ivaldi et al., 2014] Ivaldi, S., Padois, V., and Nori, F. (2014). Tools for dynamics simulation of robots: a survey based on user feedback. *arXiv preprint arXiv:1402.7050*.

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

[Jimenez Rezende et al., 2014] Jimenez Rezende, D., Mohamed, S., and Wierstra, D. (2014). Stochastic Backpropagation and Approximate Inference in Deep Generative Models. *ArXiv e-prints*.

[Jonschkowski and Brock, 2015] Jonschkowski, R. and Brock, O. (2015). Learning state representations with robotic priors. *Auton. Robots*, 39(3):407–428.

[Jonschkowski et al., 2017] Jonschkowski, R., Hafner, R., Scholz, J., and Riedmiller, M. A. (2017). Pves: Position-velocity encoders for unsupervised learning of structured state representations. *CoRR*, abs/1705.09805.

[Jordan et al., 1999] Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233.

[Kaelbling et al., 1996] Kaelbling, L. P., Littman, M. L., and Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285.

[Kalchbrenner et al., 2016] Kalchbrenner, N., Oord, A. v. d., Simonyan, K., Danihelka, I., Vinyals, O., Graves, A., and Kavukcuoglu, K. (2016). Video pixel networks. *arXiv preprint arXiv:1610.00527*.

[Karl et al., 2016] Karl, M., Soelch, M., Bayer, J., and van der Smagt, P. (2016). Deep Variational Bayes Filters: Unsupervised Learning of State Space Models from Raw Data. *ArXiv e-prints*.

[Kim and Doncieux, 2017] Kim, S. and Doncieux, S. (2017). Learning highly diverse robot throwing movements through quality diversity search. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '17, pages 1177–1178, New York, NY, USA. ACM.

[Kingma and Welling, 2013] Kingma, D. P. and Welling, M. (2013). Auto-Encoding Variational Bayes. *ArXiv e-prints*.

[Klyubin et al., 2005] Klyubin, A. S., Polani, D., and Nehaniv, C. L. (2005). Empowerment: A universal agent-centric measure of control. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 1, pages 128–135. IEEE.

[Kompella et al., 2011] Kompella, V. R., Luciw, M. D., and Schmidhuber, J. (2011). Incremental slow feature analysis: Adaptive and episodic learning from high-dimensional input streams. *CoRR*, abs/1112.2113.

[Krishnan et al., 2015] Krishnan, R. G., Shalit, U., and Sontag, D. (2015). Deep Kalman Filters. *ArXiv e-prints*.

[Krishnan et al., 2017] Krishnan, R. G., Shalit, U., and Sontag, D. (2017). Structured Inference Networks for Nonlinear State Space Models.

[Kulkarni et al., 2016] Kulkarni, T. D., Narasimhan, K., Saeedi, A., and Tenenbaum, J. (2016). Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Advances in Neural Information Processing Systems*, pages 3675–3683.

[Kulkarni et al., 2015] Kulkarni, T. D., Whitney, W. F., Kohli, P., and Tenenbaum, J. (2015). Deep convolutional inverse graphics network. In *Advances in Neural Information Processing Systems*, pages 2539–2547.

[Kumar et al., 2017] Kumar, D., Wong, A., and Taylor, G. W. (2017). Explaining the Unexplained: A CLass-Enhanced Attentive Response (CLEAR) Approach to Understanding Deep Neural Networks. *arXiv preprint arXiv:1704.04133*.

[Le Goff et al., 2017] Le Goff, L. K., Mukhtar, G., Le Fur, P.-H., and Doncieux, S. (2017). Segmenting objects through an autonomous agnostic exploration conducted by a robot. In *Proceedings of the IEEE International Conference on Robotic Computing*, Taichung University. full length paper.

[Lenz et al.,] Lenz, I., Knepper, R. A., and Saxena, A. Deepmpc: Learning deep latent features for model predictive control.

[Lesort et al., 2017] Lesort, T., Seurin, M., Li, X., Rodríguez, N. D., and Filliat, D. (2017). Unsupervised state representation learning with robotic priors: a robustness benchmark. *CoRR*, abs/1709.05185.

[Li, 2017] Li, Y. (2017). Deep reinforcement learning: An overview. *CoRR*, abs/1701.07274.

[Lillicrap et al., 2015] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2015). Continuous control with deep reinforcement learning. *CoRR*, abs/1509.02971.

[Machado et al., 2017a] Machado, M. C., Bellemare, M. G., and Bowling, M. (2017a). A laplacian framework for option discovery in reinforcement learning. *arXiv preprint arXiv:1703.00956*.

[Machado et al., 2017b] Machado, M. C., Bellemare, M. G., Talvitie, E., Veness, J., Hausknecht, M. J., and Bowling, M. (2017b). Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *CoRR*, abs/1709.06009.

[Mathieu et al., 2015] Mathieu, M., Couprie, C., and LeCun, Y. (2015). Deep multi-scale video prediction beyond mean square error. *CoRR*, abs/1511.05440.

[Metelli et al., 2017] Metelli, A. M., Pirotta, M., and Restelli, M. (2017). Compatible reward inverse reinforcement learning. In *Advances in Neural Information Processing Systems*.

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

[Munk et al., 2016] Munk, J., Kober, J., and Babuska, R. (2016). *Learning state representation for deep actor-critic control*, pages 4667–4673. IEEE. Accepted Author Manuscript.

[Ng et al.,] Ng, A. Y., Russell, S. J., et al. Algorithms for inverse reinforcement learning.

[Oh et al., 2017] Oh, J., Singh, S., and Lee, H. (2017). Value Prediction Network. *ArXiv e-prints.*

[Parisi, 2017] Parisi, S., R.-S. P. J. (2017). Goal-driven dimensionality reduction for reinforcement learning. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS).*

[Pathak et al., 2017] Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. (2017). Curiosity-driven exploration by self-supervised prediction. In *ICML.*

[Peng et al., 2017] Peng, X. B., Andrychowicz, M., Zaremba, W., and Abbeel, P. (2017). Sim-to-real transfer of robotic control with dynamics randomization. *CoRR*, abs/1710.06537.

[Pinto et al., 2016] Pinto, L., Gandhi, D., Han, Y., Park, Y., and Gupta, A. (2016). The curious robot: Learning visual representations via physical interactions. *CoRR*, abs/1604.01360.

[Rauber et al., 2017] Rauber, J., Brendel, W., and Bethge, M. (2017). Foolbox v0.8.0: A python toolbox to benchmark the robustness of machine learning models. *CoRR*, abs/1707.04131.

[Reed et al., 2017] Reed, S. E., van den Oord, A., Kalchbrenner, N., Colmenarejo, S. G., Wang, Z., Belov, D., and de Freitas, N. (2017). Parallel multiscale autoregressive density estimation. *CoRR*, abs/1703.03664.

[Scholz et al., 2014] Scholz, J., Levihn, M., Isbell, C. L., and Wingate, D. (2014). A physics-based model prior for object-oriented mdps. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pages II–1089–II–1097. JMLR.org.

[Schulman et al., 2015] Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. (2015). Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 1889–1897.

[Schulman et al., 2017] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *CoRR*, abs/1707.06347.

[Sermanet et al., 2017] Sermanet, P., Lynch, C., Hsu, J., and Levine, S. (2017). Time-contrastive networks: Self-supervised learning from multi-view observation. *CoRR*, abs/1704.06888.

[Shelhamer et al., 2017] Shelhamer, E., Mahmoudieh, P., Argus, M., and Darrell, T. (2017). Loss is its own reward: Self-supervision for reinforcement learning. *arXiv preprint arXiv:1612.07307.*

[Silver et al., 2016] Silver, D., van Hasselt, H., Hessel, M., Schaul, T., Guez, A., Harley, T., Dulac-Arnold, G., Reichert, D., Rabinowitz, N., Barreto, A., et al. (2016). The predictron: End-to-end learning and planning. *arXiv preprint arXiv:1612.08810.*

[Stewart and Ermon, 2016] Stewart, R. and Ermon, S. (2016). Label-free supervision of neural networks with physics and domain knowledge. *CoRR*, abs/1609.05566.

[Sutton, 1990] Sutton, R. S. (1990). Integrated architectures for learning, planning, and reacting based on approximating dynamic programming.

[Sutton, 1998] Sutton, R. S. (1998). *Introduction to reinforcement learning*, volume 135.

[Tamar et al., 2016] Tamar, A., Wu, Y., Thomas, G., Levine, S., and Abbeel, P. (2016). Value iteration networks. In *Advances in Neural Information Processing Systems*, pages 2154–2162.

[Teh et al., 2017] Teh, Y. W., Bapst, V., Czarnecki, W. M., Quan, J., Kirkpatrick, J., Hadsell, R., Heess, N., and Pascanu, R. (2017). Distral: Robust multitask reinforcement learning. *CoRR*, abs/1707.04175.

[Tobin et al., 2017] Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. (2017). Domain randomization for transferring deep neural networks from simulation to the real world. *arXiv preprint arXiv:1703.06907.*

[van Hoof et al., 2016] van Hoof, H., Chen, N., Karl, M., van der Smagt, P., and Peters, J. (2016). Stable reinforcement learning with autoencoders for tactile and visual data. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3928–3934.

[Vincent et al., 2008] Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. (2008). Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 1096–1103, New York, NY, USA. ACM.

[Vondrick and Torralba, 2017] Vondrick, C. and Torralba, A. (2017). Generating the future with adversarial transformers. In *CVPR*.

[Watter et al., 2015] Watter, M., Springenberg, J., Boedecker, J., and Riedmiller, M. (2015). Embed to control: A locally linear latent dynamics model for control from raw images. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems 28*, pages 2746–2754. Curran Associates, Inc.

[Wilson et al., 2007] Wilson, A., Fern, A., Ray, S., and Tadepalli, P. (2007). Multi-task reinforcement learning: a hierarchical bayesian approach. In *Proceedings of the 24th international conference on Machine learning*, pages 1015–1022. ACM.

[Wiskott and Sejnowski, 2002] Wiskott, L. and Sejnowski, T. J. (2002). Slow feature analysis: Unsupervised learning of invariances. *Neural Comput.*, 14(4):715–770.

[Wu et al., 2017] Wu, Y., Mansimov, E., Liao, S., Grosse, R. B., and Ba, J. (2017). Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. *CoRR*, abs/1708.05144.

[Zhang et al., 2011] Zhang, P., Ren, Y., and Zhang, B. (2011). A new embedding quality assessment method for manifold learning. *CoRR*, abs/1108.1636.

[Zhang et al., 2012] Zhang, P., Ren, Y., and Zhang, B. (2012). A new embedding quality assessment method for manifold learning. *Neurocomputing*, 97:251–266.