Disentangling Dynamics and Content for Control and Planning

Ershad Banijamali[1] , **Ahmad Khajenezhad**[2], **Ali Ghodsi**[3] , **Mohammad Ghavamzadeh**[4]

[1]School of Computer Science, University of Waterloo
[2]Sharif University of Technology
[3]Department of Statistics and Actuarial Science, University of Waterloo
[4]DeepMind

sbanijam@uwaterloo.ca, khajenezhad@ce.sharif.edu
aghodsib@uwaterloo.ca, ghavamza@google.com

Abstract

In this paper, We study the problem of learning a controllable representation for high-dimensional observations of dynamical systems. Specifically, we consider a situation where there are multiple sets of observations of dynamical systems with identical underlying dynamics. Only one of these sets has information about the effect of actions on the observation and the rest are just some random observations of the system. Our goal is to utilize the information in that one set and find a representation for the other sets that can be used for planning and ling-term prediction.

1 Introduction

The world surrounding us is full of events that we only observe them through high-dimensional sensory data. However, in many cases, these events can be described by few features and simple relations. Discovering the simple low-dimensional feature space is an underlying task in many data processing algorithms. With the recent advances in the area of artificial neural networks, use of deep structures for learning the low-dimensional representations has been outstandingly increased in different applications. A good representation is defined based on the task in hand.

In the area of control, a good representation means a low-dimensional feature space, in which the relation between different states of the system can be modeled by simple functions. Finding such representation has been studied recently in different works [2]. Deep autoencoders have been used for obtaining an appropriate representation for control in [5, 8]. This problem has been also studied in action respecting embedding (ARE) framework [3]. Embed to control (E2C) [9], finds a low-dimensional locally-linear embedding of the observations that allows planning and long-term prediction by applying model predictive controllers, e.g. iterative linear quadratic regulator (iLQR). More recently, robust controllable embedding (RCE), [1], has been proposed, which can handle noise in the dynamics of the system.

In this paper, we address this problem in a more generalized setting. Suppose we have different sets of high-dimensional observations from the systems that have the same underlying dynamics. Therefore, in all of the observations there exist a common set of features that correspond to the dynamics of the system. Our goal is to extract this set of features using only one set of observations and use the learned dynamics to do planning and long-term prediction for the other sets. To do so, we design a model that disentangles the features that contribute in dynamics and those who just contribute in the content of the image. Building such model requires dynamics information (i.e.

knowing how the actions change our observation from the system) in one set and there is no need to have such information in other sets.

Learning disentangled features has various applications in image and video processing and text analysis and has been studied in different works [6]. More recently, authors in [7, 4] proposed a model in the framework of generative adversarial networks (GANs) that disentangles dynamics and content for video generation. However, to the best of our knowledge, our model is the first model that proposes disentangling dynamics and content for control, planning, and prediction.

2 Problem Statement

Suppose we have different sets of high-dimensional observations from the states of dynamical systems where the underlying dynamics of the systems is the same. For now, let us assume that we only have one dynamical system and there are just two observation sets from this system from different angles. We make this assumption just for the sake of simplicity in notations, but it can be easily relaxed. The two observation sets are denoted by X and Y that belong to the observation spaces $\mathcal{X}$ and $\mathcal{Y}$, respectively.

Let us denote by $\mathcal{S}$, the true state space of the system, in which $\mathbf{s}_t$ represents the state of the system at time step t. The dynamics of the system in this space is defined by $f_\mathcal{S}$:

$$\mathbf{s}_{t+1} = f_\mathcal{S}(\mathbf{s}_t, \mathbf{u}_t) + \mathbf{n}^\mathcal{S} \tag{1}$$

where $\mathbf{n}^\mathcal{S}$ is the noise in the state space. We do not have any information about the state space and want to estimate it based on our observations.

Suppose set X consists of triples $(\mathbf{x}_t, \mathbf{u}_t, \mathbf{x}_{t+1})$, i.e. observation of the system at time t, action that is applied to the system at time t, and the next observation after applying $\mathbf{u}_t$ to the system, respectively. Therefore, we know how the actions change our observations in X. We also assume that the observations in this set have Markov property. Set Y also has some observations of the system from a different point of view. However, there is no information about the actions and the effect of the actions on our observation in this set. We denote the observations in this set by $\mathbf{y}_t$. Note that $\mathbf{x}_t$ and $\mathbf{y}_t$ are two different observations of the state $\mathbf{s}_t$. Since X and Y, are observations from one system, the underlying dynamics is the same. Suppose that our goal is to do planning and long-term prediction in $\mathcal{Y}$. Our approach to achieve this goal is to extract the dynamics information from X and leverage this information to build a model for Y.

3 Model Description

There has been some efforts in finding a representation for high-dimensional observations of dynamical systems that is suitable for planning using neural networks. Recently, Robust Controllable Embedding (RCE) [1] has been proposed that shows good performance on this task. The RCE model is based on introducing a graphical model for the problem that describes the relation between pairs of observations and their embedded representations. Using deep variational learning, the lower bound of the conditional distribution of the observations is maximized.

We build our model up on RCE . However, instead of using only one latent variable, we assume that there are two independent variables in the latent space. One of these variables is related to the dynamics of the system and the other one is related to the content of the observation. Therefore we aim to disentangle the dynamics and content in the latent space. Such disentanglement allows us to model the dynamics of the observations, even though the content of them might be very different. Consider the graphical models in Fig. 1. Fig. 1a shows the model for X. In this figure, $\mathbf{z}_t$ and $\mathbf{w_x}$ are the two latent variables that we want to represent the dynamics and content information, respectively. Similar to RCE, we want to have locally-linear dynamics in the latent space, i.e.:

$$\hat{\mathbf{z}}_{t+1} = \mathbf{A}_t\mathbf{z}_t + \mathbf{B}_t\mathbf{u}_t + \mathbf{c}_t \tag{2}$$

where $\mathbf{A}_t$, $\mathbf{B}_t$, and $\mathbf{c}_t$ are matrices that are learned during training the model. Building this locally-linear model will allow us to use iLQR method for control. We use $\mathbf{z}$ and $\hat{\mathbf{z}}$ to distinguish between encoding of $\mathbf{x}$ and the variable after transition. Fig. 1b shows the model for Y. This set is encoded with two latent variables $\mathbf{v}_t$ and $\mathbf{w_y}$, representing dynamics and content, respectively. We would

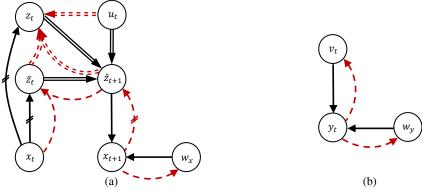(a)   (b)

Figure 1: Graphical models. The black arrows are generative links and the red dashed ones are recognition links. The parallel lines show the deterministic links. **(a)** Graphical model for set X. $\bar{\mathbf{z}}_t$ and $\mathbf{z}_t$ are two samples from $p(\mathbf{z}_t|\mathbf{x}_t)$. The neural networks that parameterize the links with hatch marks are hard tied, i.e. $p(\mathbf{z}_t|\mathbf{x}_t) = p(\bar{\mathbf{z}}_t|\mathbf{x}_t) = q(\hat{\mathbf{z}}_t|\mathbf{x}_t)$. **(b)** Graphical model for Y

like to have a locally-linear dynamics similar to Eq. 2 for $\mathbf{v}$. All of the conditional distribution on these graphical models are parameterized by neural networks.

The goal in this work can be interpreted as maximizing the likelihood of observations, while imposing a further constraint that if $\mathbf{x}_t$ and $\mathbf{y}_t$ are two high-dimensional observations of the same state of the dynamical system(s), then we want $q(\mathbf{z}_t|\mathbf{x}_t)$ and $q(\mathbf{v}_t|\mathbf{y}_t)$ be close to each other, e.g. have small KL divergence.

Suppose $q^\star = q(\mathbf{z}_t, \bar{\mathbf{z}}_t, \hat{\mathbf{z}}_{t+1}, \mathbf{w_x}|\mathbf{x}_t, \mathbf{x}_{t+1}, \mathbf{u}_t)$ and $q^\dagger = q(\mathbf{v}_t, \mathbf{w_y}|\mathbf{y}_t)$. Based on the graphical model we can consider these factorizations for $q^\star$ and $q^\dagger$:

$$q^\star = q_\phi(\mathbf{w_x}|\mathbf{x}_{t+1})q_\phi(\hat{\mathbf{z}}_{t+1}|\mathbf{x}_{t+1})q_\varphi(\bar{\mathbf{z}}_t|\hat{\mathbf{z}}_{t+1}, \mathbf{x}_t)\delta(\mathbf{z}_t|\hat{\mathbf{z}}_{t+1}, \bar{\mathbf{z}}_t, \mathbf{u}_t)$$

(3)

$$q^\dagger = q_\phi(\mathbf{w_y}|\mathbf{y}_t)q_\phi(\mathbf{v}|\mathbf{y}_t)$$

where ϕ and φ stand for encoder and transition network parameters, respectively. We also have the following factorization for the generative links in the graphical model:

$$p(\mathbf{x}_{t+1}, \mathbf{z}_t, \bar{\mathbf{z}}_t, \hat{\mathbf{z}}_{t+1}, \mathbf{w_x}|\mathbf{x}_t, \mathbf{u}_t) = p(\bar{\mathbf{z}}_t|\mathbf{x}_t)p(\mathbf{z}_t|\mathbf{x}_t)\delta(\hat{\mathbf{z}}_{t+1}|\bar{\mathbf{z}}_t, \mathbf{z}_t, \mathbf{u}_t)p(\mathbf{x}_{t+1}|\hat{\mathbf{z}}_{t+1}, \mathbf{w_x})p(\mathbf{w_x})$$

(4)

In this model, we want to maximize the likelihood of all the observations. Since we consider Markov property for set X, maximizing the likelihood of observations in X boils down to log-likelihood of the conditional distribution of the pair of observations. Therefore we will have:

$\log p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t) + \log p(\mathbf{y}_t)$

$$\geq \mathbb{E}_{q^\star}\big[\log p(\mathbf{x}_{t+1}, \mathbf{z}_t, \bar{\mathbf{z}}_t, \hat{\mathbf{z}}_{t+1}, \mathbf{w_x}|\mathbf{x}_t, \mathbf{u}_t) - \log q^\star\big] + \mathbb{E}_{q^\dagger}\big[\log p(\mathbf{y}_t, \mathbf{v}_t, \mathbf{w_y}) - \log q^\dagger\big]$$

$$= \mathbb{E}_{\substack{q_\phi(\hat{\mathbf{z}}_{t+1}|\mathbf{x}_{t+1})\\q_\phi(\mathbf{w_x}|\mathbf{x}_{t+1})}}\big[\log p(\mathbf{x}_{t+1}|\hat{\mathbf{z}}_{t+1}, \mathbf{w_x})\big] - \mathbb{E}_{q_\phi(\hat{\mathbf{z}}_{t+1}|\mathbf{x}_{t+1})}\big[\text{KL}\big(q_\varphi(\bar{\mathbf{z}}_t|\hat{\mathbf{z}}_{t+1}, \mathbf{x}_t) \parallel p(\bar{\mathbf{z}}_t|\mathbf{x}_t)\big)\big]$$

$$+ \text{H}\big(q_\phi(\hat{\mathbf{z}}_{t+1}|\mathbf{x}_{t+1})\big) + \mathbb{E}_{\substack{q_\phi(\hat{\mathbf{z}}_{t+1}|\mathbf{x}_{t+1})\\q_\varphi(\bar{\mathbf{z}}_t|\mathbf{x}_t, \hat{\mathbf{z}}_{t+1})}}\big[\log p(\mathbf{z}_t|\mathbf{x}_t)\big] - \text{KL}\big(q_\phi(\mathbf{w_x}|\mathbf{x}_t) \parallel p(\mathbf{w_x})\big)$$

$$+ \mathbb{E}_{q^\dagger}\big[\log p(\mathbf{y_t}|\mathbf{v}_t, \mathbf{w_y})\big] - \text{KL}\big(q_\phi(\mathbf{v}_t|\mathbf{y}_t) \parallel p(\mathbf{v}_t)\big) - \text{KL}\big(q_\phi(\mathbf{w_y}|\mathbf{y}_t) \parallel p(\mathbf{w_y})\big)$$

(5)

To maximize this lower bound we use the deep variational learning framework. We assume that the prior of the content variables, $\mathbf{w_x}$ and $\mathbf{w_y}$, are Gaussian. Also we assume $p(\bar{\mathbf{z}}_t|\mathbf{x}_t)$ is Gaussian. The constraint of minimizing the KL divergence between $q(\mathbf{z}_t|\mathbf{x}_t)$ and $q(\mathbf{v}_t|\mathbf{y}_t)$ can be imposed by considering $q(\mathbf{z}_t|\mathbf{x}_t)$ as the prior for $p(\mathbf{v}_t)$, i.e.:

$$p(\mathbf{v}_t) = \mathcal{N}\big(\mu_\phi(\mathbf{x}_t), \sigma_\phi(\mathbf{x}_t)\big)$$
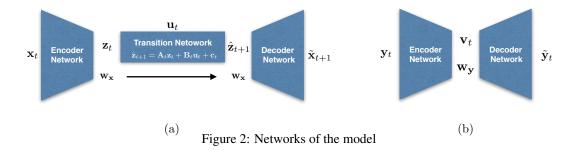
(6)

3

Figure 2: Networks of the model

Fig. 2 shows the high-level depiction of the networks in our model. In the case we use same networks for encoding and decoding the two observation sets (for example when the contents do not differ too much) , we can assume that $p(\mathbf{w_x})$ and $p(\mathbf{w_y})$ are two Gaussian distributions with different means.

4 Experiment Result

To evaluate the effectiveness of the proposed model, we consider the planar system domain. Consider an agent in a surrounded area, whose goal is to navigate from a corner to the opposite one, while avoiding the six obstacles in this area. The system is observed through a set of 40×40 pixel images taken from the top, which specify the agent's location in the area. Actions are two-dimensional and specify the direction of the agent's movement. Suppose that the difference between the two observation sets from this system is in the shape of the agent, as shown in Fig. 3. We use the same encoder and decoder for the two observation sets. We used 8000 samples (triples $(\mathbf{x}_t, \mathbf{u}_t, \mathbf{x}_{t+1})$) in the set X and only 2000 samples in set Y.

Fig. 3 shows the true map of the state-space of this system and the maps that are estimated using the model for the two observation sets. As we can see, the map that has been discovered using the information in X is very well preserved for the set Y. In this figure we can also see some predictions of the position of the agent for both sets given some actions versus the true position of the agent after applying those action. This shows that the model is successful in learning the dynamics for Y even though we did not have any information about the dynamics in this set.
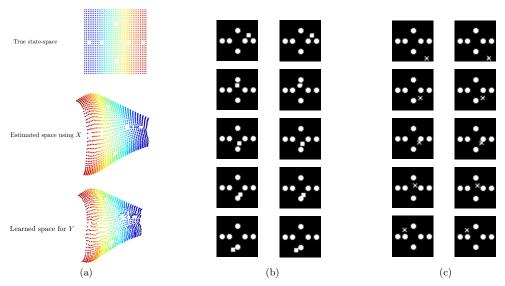

Figure 3: **(a)** Top: The true state space of the system. Middle: estimated locally-linear latent space from set X. Bottom: The hidden space learned for set Y. **(b)**: Left: An initial observation from X on top and its next observations after applying four random actions Right: Reconstruction of the initial state and prediction of the next observations. **(c)**: Left: An initial observation from Y on top and its next observations after applying four random actions Right: Reconstruction of the initial state and prediction of the next observations

Table 1: Planar System

Dataset	Reconstruction Loss	Prediction Loss	Planning Loss	Success Rate
with action (X)	3.6 ± 1.7	6.2 ± 2.8	21.4 ± 2.9	100%
without action (Y)	3.9 ± 2.2	6.3 ± 3.0	22.0 ± 2.4	100%

To evaluate the performance of the model in planning, we provide different sets of initial and final observations in $\mathcal{X}$ and $\mathcal{Y}$, and use the learned models to find the policy that leads the agent to reach the final observation within T steps. We present the performance of the model in table 1 in terms of: **1)** *Reconstruction Loss* is the loss in reconstructing current observation using the encoder and decoder. **2)** *Prediction Loss* is the loss in predicting next observations, given current observation and current action, using the encoder, decoder, and transition network. **3)** *Planning Loss* is computed based on the following quadratic loss:

$$J = \sum_{t=1}^{T} (\mathbf{s}_t - \mathbf{s}^f)^\top \mathbf{Q}(\mathbf{s}_t - \mathbf{s}^f) + \mathbf{u}_t^\top \mathbf{R}\mathbf{u}_t. \tag{7}$$

where $\mathbf{Q}$ and $\mathbf{R}$ are cost weighting matrices. $\mathbf{s}^f$ is the state corresponding to the final observation. We apply the sequence of actions returned by iLQR to the dynamical system and report the value of the loss in Eq. 7. **4)** *Success Rate* shows the number of times the agents reaches the goal within the planning horizon T, and remains near the goal in case it reaches it in less than T steps. For each of the sets, all the results are averaged over 20 runs.

5 Discussion

This model has potential applications in self-driving cars. Self-driving cars use many sensors to observe the surrounding environment that includes expensive sensors for dynamics estimation. They also use multiple cameras to monitor the area. Observations from the camera are rich in term of information about the content (objects in the area), however, extracting dynamics information using these observations is a hard task. On the other hand, the dynamics estimator sensors are poor in terms of the content information but provide information about action-state space with high accuracy. If we can find a way to transfer the learned dynamics from the sensor to the cameras, we can remove the sensor at the test time and reduce the cost of experiments.

References

[1] E. Banijamali, R. Shu, M. Ghavamzadeh, H. Bui, and A. Ghodsi. Robust locally-linear controllable embedding. In *arXiv preprint arXiv:1710.05373*, 2017.

[2] W. Böhmer, J. Springenberg, J. Boedecker, M. Riedmiller, and K. Obermayer. 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. *Künstliche Intelligenz*, 29(4):353–362, 2015.

[3] M. Bowling, A. Ghodsi, and D. Wilkinson. Action respecting embedding. In *Proceedings of the 22nd international conference on Machine learning*, pages 65–72, 2005.

[4] E. Denton and V. Birodkar. Unsupervised learning of disentangled representations from video. *arXiv preprint arXiv:1705.10915*, 2017.

[5] S. Lange and M. Riedmiller. Deep auto-encoder neural networks in reinforcement learning. In *Proceedings of the International Joint Conference on Neural Networks*, pages 1–8, 2010.

[6] M. F. Mathieu, J. J. Zhao, J. Zhao, A. Ramesh, P. Sprechmann, and Y. LeCun. Disentangling factors of variation in deep representation using adversarial training. In *Advances in Neural Information Processing Systems*, pages 5040–5048, 2016.

[7] S. Tulyakov, M.-Y. Liu, X. Yang, and J. Kautz. Mocogan: Decomposing motion and content for video generation. *arXiv preprint arXiv:1707.04993*, 2017.

[8] N. Wahlström, T. Schön, and M. Desienroth. From pixels to torques: Policy learning with deep dynamical models. In *arXiv preprint arXiv:1502.02251*, 2015.

[9] M. Watter, J. Springenberg, J. Boedecker, and M. Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in Neural Information Processing Systems*, pages 2746–2754, 2015.