Imperial College
London

IMPERIAL COLLEGE LONDON

DEPARTMENT OF COMPUTING

Project Interim Report

Author:
Dane G. Sherburn

CID:
00820119

Date: January 25, 2017

Contents

1 Introduction

A long term goal of artificial intelligence (AI) is the development of artificial general intelligence (AGI). A number of theoretical frameworks have been presented to formalize what it means to achieve AGI, notably by Hutter [1].

Reinforcement learning is fundamental in this framework. It's therefore quite exciting that deep reinforcement learning (DRL) systems have recently been able to master a wide range of tasks, including Atari games and Go. Though DRL systems have been remarkably successful, they still have a number of drawbacks [2]:

1. **Slow to learn**. Deep neural networks require large data sets and are therefore slow to learn.
2. **Fail to transfer past experience**. They often fail to perform well on tasks very similar to those they have mastered.
3. **Inability to reason abstractly.** They fail to exploit statistical regularities in the training data by using high-level processes like planning or causal reasoning.
4. **Hard to reason about.** It's often troublesome to understand why the DRL system chose the action it did.

Deep symbolic reinforcement learning (DSRL) is a recent advance which overcomes all of these drawbacks at once without the drawbacks from classical symbolic AI, namely the symbol grounding problem [2].

This novel architecture relies on the unsupervised extraction of compositional features, allowing for transfer learning and high-level cognitive processes. However, the unsupervised extraction of features from a wide range of scenes is still a challenge in AI research [4]. Fortunately methods are getting better, and the first unsupervised scalable model β-VAE was developed recently.

The aim of this project is to investigate compositional representations using β-VAE and evaluate their effectiveness in deep symbolic reinforcement systems. We plan to achieve this as follows:

1. Implement many flavours of autoencoders, including mixing and matchings of standard, convolutional and variational autoencoders, and investigate the relationship between the latent space and their final reconstructions. This will hopefully convey some intuition about the way scenes are typically represented in the latent space, their relevance in transfer learning and the significance of the problem we're trying to solve.
2. Next we focus our attention on convolutional variational autoencoders. We'll experiment with different values of β two ways: through heuristic visual inspection, and by comparing the disentanglement metric and reconstruction error. All experiments will be on a range of Atari 2600 games using the Atari Learning Environment.

3. Finally we can evaluate the value of a compositional structured latent space in a deep symbolic reinforcement system. We'll do this by training a number of reinforcement agents on the latent space and compare their performance to training on the on the original space. We can also train the one agent on multiple games to assess how well this compositional structure facilitates transfer learning.

2 Background

This chapter will contain the material necessary to understand later chapters and provide the context for our contributions.

We will introduce deep symbolic reinforcement learning which will motivate our interest in unsupervised disentangled representations. Here we'll explore state-of-the-art approaches to finding such representations, which will also require a brief introduction to autoencoders. Finally we'll mention less theoretical matters, including libraries used and the Arcade Learning Environment.

2.1 Autoencoders

2.1.1 Architecture

2.1.2 Convolutional autoencoders

2.1.3 Variational autoencoders

The autoencoders we've seen so far do the following: given input $\boldsymbol{x}$, compute its reconstruction $\widetilde{\boldsymbol{x}}$, after mapping it to a lower dimensional vector $\boldsymbol{z}$, the *latent space*. The aim of the variational autoencoder (VAE) is quite similar: given input samples from an unknown distribution $p(\boldsymbol{x})$, generate $\widetilde{\boldsymbol{x}}$ very similar to those in $p(\boldsymbol{x})$, after sampling a lower-dimensional vector $\boldsymbol{z}$. Since VAEs generate unseen, similar samples, they are *generative models* [5].

To justify the statements in the next paragraph, we have to mention that $p_\theta(\boldsymbol{z}|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{p(\boldsymbol{x})}$ is intractable due to the marginal likelihood $p(\boldsymbol{x})$ being exponentially computationally expensive, motivating the introduction of the probabilistic encoder $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ to approximate the posterior $p_\theta(\boldsymbol{z}|\boldsymbol{x})$. Since $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ appears in the loss function but is not differentiable (we can't differentiate random variables), we need to use the *reparameterisation trick*. The reparameterisation trick expresses $\boldsymbol{z} \sim q_\phi(\boldsymbol{z}|\boldsymbol{x})$ as a deterministic variable $\boldsymbol{z} = g_\phi(\boldsymbol{\epsilon}, \boldsymbol{x})$ where $\boldsymbol{\epsilon}$ is an auxilary variable with independent maginal and g_ϕ is a deterministic vector function parameterized by ϕ [8].

Learning the optimal $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ is formally expressed as the following objective:

$$\max_{\boldsymbol{\phi},\boldsymbol{\theta}} \mathbf{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - \mathcal{D}_{KL}[q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z})] \tag{1}$$

2.2 Deep symbolic reinforcement learning

2.2.1 Reinforcement learning

2.2.2 Symbolic artificial intelligence

2.2.3 Deep symbolic reinforcement learning

2.3 Representation learning

Suppose we were ordered to carry out the multiplication of two binary numbers, 1101_2 and 101101_2. Most of us would first convert these numbers to base-10 before performing the multiplication. We'd do this because the problem is significantly harder in base-2 than it is in base-10. It's therefore intuitive that a good representation is vital to the agent required to solve it [6].

The subject of what makes a good representation has generated much interest in recent times. According to [6], a good representation is simply one that makes subsequent tasks easier.

2.3.1 Disentangled representations

It's been suggested by [4] that a crucial factor in what makes a good representation is how *disentangled* it is. A disentangled representation is one where changes in latent factors correspond to changes in high-level factors in the output space [7].

To provide some intuition of what a disentangled representation is, we'll consider an example using the Frey Face data set.

The Frey Face data set consists of almost two-thousand 28×20 pictures of Brendan Frey, a Canadian machine learning researcher. The samples include Frey with varying facial expressions and orientations.

Shown in Figure (1) is a learned Frey Face manifold with two latent variables. One latent variable is sensitive to changes in the expression and one to changes in orientation, making the manifold a highly disentangled representation. In an entangled representation, changing one latent variable would change both the expression and the orientation.

Disentangled representations facilitate the reuse of previously learnt factors, called *knowledge transfer*; and the extreme case, *zero-shot inference* [6, 7].

2.3.2 Transfer learning

Transfer learning is the reuse of learnt factors from a previous task when learning a new one [6]. If the tasks are sufficiently similar, there are three ways the learning may be improved [9]:

Figure 1: A learned Frey Face manifold with two latent variables: one sensitive to the expression and one to the orientation. [8]

1. Initial performance
2. Faster to learn
3. Final performance

These improvements are shown on a performance-training graph in Figure 2.

1. Summarising quote by [10] below

”Without disentangled representations, it is difficult to interpret or re-use representations across tasks as no single component of the representation vector has a semantic meaning by itself” [10].

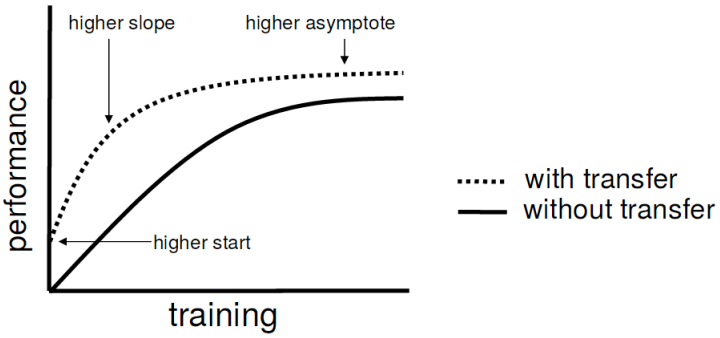

Figure 2: An illustration of the three ways in which transfer learning may improve the learning of a new task. [9]

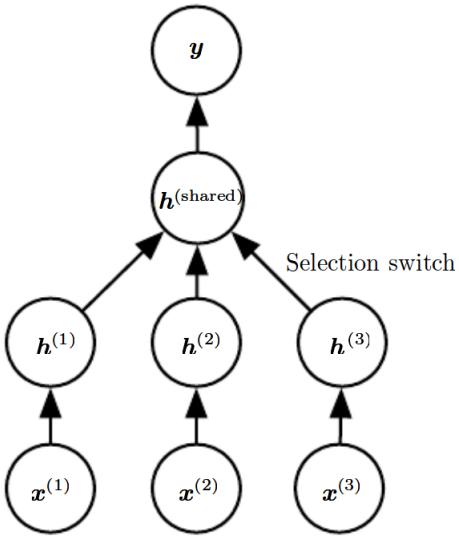

Figure 3: An illustration of how tasks may be translated into intermediate representations, then into a shared representation. The representation of different tasks in terms of common concepts is key to transfer learning. [6]

2.3.3 Zero-shot understanding

2.3.4 Data continuity

2.3.5 Statistical independence

2.3.6 Manifolds

2.4 Finding disentangled representations

2.4.1 Supervised algorithms

1. SIFT
2. Spin image
3. HoG
4. RIFT
5. Textons
6. GLOH

2.4.2 Semi-supervised algorithms

1. DC-IGN

2.4.3 Unsupervised algorithms

An overview of unsupervised algorithms learning disentangled representations is given in [4]. These include deep representations, such as deep Boltzmann machines and deep belief networks, in greedy layerwise unsupervised pre-training; graphical models; autoencoders; and manifold learning. The vast majority of approaches using these techniques (and an unmentioned approach, InfoGAN) assume the number of generative factors, priors and/or do not scale well. As far as we know, β-VAE is the only approach which has been shown to quantitatively outperform alternatives with respect to its ability to disentangle data, its stability in training and the relatively few assumptions about the data [7].

[Perhaps more research is needed here. Do we need to look harder for more unsupervised alternatives? Is more justification for using β-VAE needed? Currently we're relying solely on the survey of unsupervised algorithms by [7]]

2.5 Deeper investigation of β-VAE

2.5.1 Framework

Suppose we're given a set X of images $\boldsymbol{x} \in \mathbb{R}^N$ and ground truth data generative factors (that is, factors observed directly from the data as opposed to being inferred). We'll partition these factors into two sets, V and W, where vectors $\boldsymbol{v} \in \mathbb{R}^K$ are independent and $\boldsymbol{w} \in \mathbb{R}^H$ are dependent.

The aim of β-VAE is to factorise the latent space $\boldsymbol{z} \in \mathbb{R}^M$ into disentangled factors $\boldsymbol{v}$ and the rest. This enforces $M \geq K$, since we'd like the latent representation to be capable of learning all independent ground truth data factors of X.

β-VAE achieves this by adding an important multiplicative factor β to the KL-divergence term, making the objective:

$$\max_{\boldsymbol{\phi},\boldsymbol{\theta}} \mathbf{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - \beta \mathcal{D}_{KL}[q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z})] \tag{2}$$

The hyperparameter β balances reconstruction error with independence constraints [7].

2.5.2 Disentanglement metric

2.5.3 Quantitative performance

2.5.4 Qualitative performance

2.6 Libraries

2.6.1 Keras

Keras is a high-level neural network library written in Python [3] and was used to implement the frameworks in Chapter X. It was a suitable choice because it supports:

- Convolutional, deconvolutional and pooling layers
- Lambda functions, which is necessary for sampling in variational autoencoders
- Custom loss functions, which is necessary to implement the β-VAE framework
- Able to save weights, so the same network can be trained on multiple games
- Able to run on the GPU

2.6.2 Arcade Learning Environment

The Arcade Learning Environment (ALE) is a framework built on top of the Atari 2600 emulator Stella [11]. This library was a suitable choice because:

- Emulation details are abstracted away from the researcher
- The same agent can be used for all games, due to its modular design
- Frames can be saved during game play, which is used to collect training data
- A Python wrapper for the entire framework is provided, so our agents developed in Keras interact seamlessly

3 Project Plan

You should explain what needs to be done in order to complete the project and roughly what you expect the timetable to be. Dont forget to include the project write-up (the final report), as this is a major part of the exercise. Its important to identify key milestones and also fall-back positions, in case you run out of time. You should also identify what extensions could be added if time permits. The plan should be complete and should include those parts that you have already addressed (make it clear how far you have progressed at the time of writing). This material will not appear in the final report.

Table 1: Project plan in terms of remaining months.

	Milestones	**Fallbacks**
February	• Implement β-VAE • Reproduce results from [7] • Finish formal justification of VAE	• Implement β-VAE • Reproduce results from [7]
March	• Disentanglement metric script • Test script with simple data	• Disentanglement metric script • Test script with simple data
April	• Find suitable β for Pong • Find suitable β for Space Invaders • Graph latent vs output	• Find suitable β for Pong • Find suitable β for Space Invaders
May	• Complete report background • Complete report related work	
June	• Complete report results	

4 Evaluation plan

Project evaluation is very important, so it's important to think now about how you plan to measure success. For example, what functionality do you need to demonstrate? What experiments to you need to undertake and what outcome(s) would constitute success? What benchmarks should you use? How has your project extended the state of the art? How do you measure qualitative aspects, such as ease of use? These are the sort of questions that your project evaluation should address; this section should outline your plan.

References

[1] Marcus Hutter. *Universal Artificial Intelligence*, volume 1. 2005. pages 3

[2] Marta Garnelo, Kai Arulkumaran, and Murray Shanahan. Towards Deep Symbolic Reinforcement Learning. *arXiv Preprint*, pages 1–13, 2016. pages 3

[3] François Chollet. Keras: Deep Learning library for Theano and TensorFlow, 2015. pages 9

[4] Y Bengio, A Courville, and P Vincent. Representation Learning: A Review and New Perspectives. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):1798–1828, 2013. pages 3, 5, 8

[5] Carl Doersch. Tutorial on Variational Autoencoders. *arXiv*, pages 1–23, 2016. pages 4

[6] Aaron Courville Ian Goodfellow, Yoshua Bengio. Deep Learning Book. *Deep Learning*, 21(1):111–124, 2015. pages 5, 7

[7] Irina Higgins, Loic Matthey, Xavier Glorot, Arka Pal, Benigno Uria, Charles Blundell, Shakir Mohamed, and Alexander Lerchner. Early Visual Concept Learning with Unsupervised Deep Learning. *arXiv*, 2016. pages 5, 8, 10

[8] Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. *Iclr*, (Ml):1–14, 2014. pages 4, 6

[9] Lisa Torrey and Jude Shavlik. Transfer Learning. *Machine Learning*, pages 1–22, 2009. pages 5, 6

[10] William F Whitney, Michael Chang, Tejas Kulkarni, and Joshua B Tenenbaum. Understanding Visual Concepts with Continuation Learning. *arXiv*, pages 1–4, 2016. pages 6

[11] Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. In *IJCAI International Joint Conference on Artificial Intelligence*, volume 2015-Janua, pages 4148–4152, 2015. pages 9