Master thesis
Computing Science

Radboud University

Title Master Thesis

Author:
Niels van Velzen
s4269454

First supervisor/assessor:
Dr., N. H. Jansen
`n.jansen@cs.ru.nl`

Second supervisor:
MSc., D. M. Groß
`D.Gross@cs.ru.nl`

Second supervisor:
Ing., C. Schmidl
`christoph.schmidl.1@ru.nl`

December 22, 2021

Abstract

The abstract of your thesis is a brief description of the research hypothesis, scientific context, motivation, and results. The preferred size of an abstract is one paragraph or one page of text.

Contents

Chapter 1

Introduction

The introduction of your bachelor thesis introduces the research area, the research hypothesis, and the scientific contributions of your work. A good narrative structure is the one suggested by Simon Peyton Jones [**?**]:

- describe the problem / research question
- motivate why this problem must be solved
- demonstrate that a (new) solution is needed
- explain the intuition behind your solution
- motivate why / how your solution solves the problem (this is technical)
- explain how it compares with related work

Close the introduction with a paragraph in which the content of the next chapters is briefly mentioned (one sentence per chapter).

Chapter 2

Preliminaries

2.1 Reinforcement learning

RL, neural networks (incl non-linnear function approximation)

2.2 State-space dimensionality reduction

Definition, general info Methods:

2.2.1 Principal Component Analysis

algemene info pca

2.2.2 Autoencoder

Another way of projecting data onto a lower dimensional space, is using an *autoencoder*[2]. An autoencoder is a neural network consisting of two parts: an encoder network and a decoder network. The encoder projects the given input data onto a lower dimensional space, also called the *latent space*. The output of the encoder, called the *latent representation*, is used as input for the decoder. This decoder tries to reconstruct the original input from the latent representation as closely as possible. The autoencoder architecture is shown in figure2.1.

Formally, an autoencoder can be defined by two functions:

$$\psi : \mathbb{R}^n \to \mathbb{R}^m \tag{2.1}$$

$$\phi : \mathbb{R}^m \to \mathbb{R}^n \tag{2.2}$$

$$\phi, \psi = \underset{\phi,\psi}{\arg\min}\, \Delta(\psi \circ \phi(x)) \tag{2.3}$$

Equation (2.1) defines the encoder and equation (2.2) the decoder, both satisfying (2.3) where Δ is the reconstruction loss function for input x.

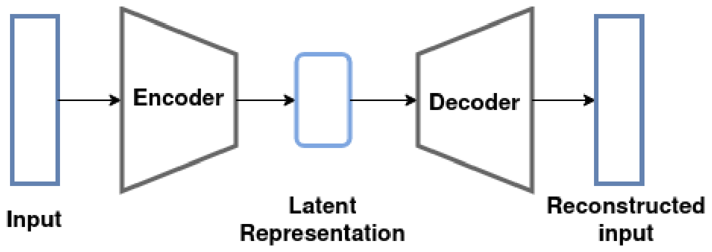

Figure 2.1: The architecture of an autoencoder.

A fundamental difference with using PCA, is the type of functions that can be approximated for lowering the dimensionality. Since an autoencoder uses neural networks, they can approximate nonlinear functions, as mentioned in section 2.1. This is in contrast with PCA, which can only approximate linear functions. Because of this, an autoencoder can learn more powerful generalisations which lead to lower information loss[2].

2.2.3 DeepMDP

Info over deepmdp

Chapter 3

Research

The aim of this paper is to examine the effect of reducing the dimensionality of the state-space in reinforcement learning (RL). In this section we will discuss the our research and its results. We will start by detailing our method in section 3.1; here we will explain the environment we used for our experiments, as well as the experiments that we ran. After this, we will show and discuss the results from these experiments in section 3.2. The discussion of the results will include an examination of how the different state-space reduction methods led to their results.

3.1 Method

In this section we will explain our method: how we researched the effect of state-space dimensionality reduction on an RL agent. Before going into the details of the different experiments that we ran in section 3.1.2, we will first look at the environment in which we ran the experiments in section 3.1.1.

3.1.1 Environment: Starcraft II

For our experiments we used the *StarCraft II* environment by *Blizzard*[1]. StarCraft II is a real-time strategy game, which has been used in RL research after the introduction of a learning environment created in collaboration with *DeepMind*, called *SC2LE* and a corresponding Python component called *PySC2*[3].

In particular we are using a PySC2 minigame called *MoveToBeacon*. This minigame simplifies the StarCraft II game. Here, the RL agent must select an army unit and move it to a given beacon. To simplify our RL agent, selecting the army unit is implemented as a script, thereby focusing our research on moving the army unit to the beacon. A screenshot of the game is given in figure 3.1.

An *observation* received by the agent in this minigame is given by a

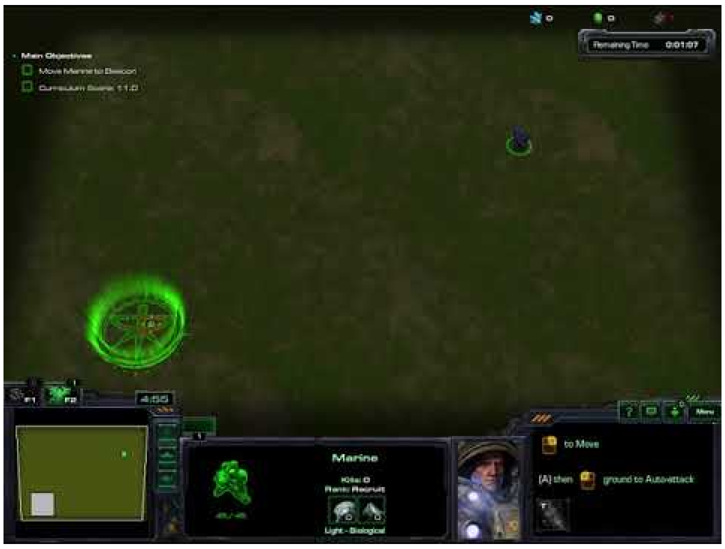

Figure 3.1: Screenshot of the minigame *MoveToBeacon* in *StarCraft II*.

32x32 grid, representing the entire state of the game, giving a total of 1024 *features*. Each cell in the grid represents a tile in the game. It can have one of three values: a 0 denoting an empty tile, a 1 denoting the army unit controlled by the agent, or a 5 denoting the beacon. The beacon comprises more than one tile, namely a total of 21 tiles; it comprises five adjecent rows, where the first comprises three adjecent columns, followed by three rows of five columns, followed by a row of three columns. Because of this, the beacon has 27 places where it could be, with the army unit having 1003 tiles left to be. This gives a total state-space of 32x32x3 with a cardinality of $27 \times 1003 = 731.187$. An example of such a state observation can be seen in figure 3.2.

Is this correct? Or are there actually perhaps only 3 features or some-thing?

Is this a correct usage of state-space?

An *action* taken by the agent is given by an (x, y) coordinate with $x, y \in \{0..31\}$. This denotes the (indices of the) cell in the grid that the army unit will move to.

Lastly, an *episode* takes 120 seconds. The goal is to move the army unit to the beacon as often as possible in this time limit, each time adding 1 point to the episode score. At the start of each episode, the beacon and army unit are placed randomly. Whenever the army unit reaches the beacon, only the beacon will be relocated randomly.

3.1.2 Experiments

Welke experiments/vergelijkingen gedaan + setup

- Base agent
- PCA agent

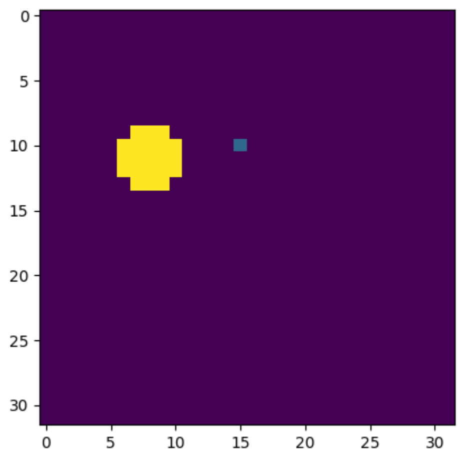

Figure 3.2: A state observation received by the RL agent, for the StarCraft II minigame MoveToBeacon. The yellow cells represent one beacon; the blue cell represents the army unit controlled by the player; all other cells are empty.

- Pretrained AE agent
- Online trained AE agent
- DeepMDP

3.2 Results

sectie opzet

3.2.1 Research results

Resultaten van de verschillende agents

3.2.2 Discussion

Resultaten van AE analyse

Chapter 4

Related Work

In this chapter you demonstrate that you are sufficiently aware of the state-of-art knowledge of the problem domain that you have investigated as well as demonstrating that you have found a *new* solution / approach / method.

Chapter 5

Conclusions

In this chapter you present all conclusions that can be drawn from the preceding chapters. It should not introduce new experiments, theories, investigations, etc.: these should have been written down earlier in the thesis. Therefore, conclusions can be brief and to the point.

Bibliography

[1] Blizzard. Blizzard/s2client-proto: Starcraft II Client - protocol definitions used to communicate with StarCraft II.

[2] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[3] Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John P. Agapiou, Julian Schrittwieser, John Quan, Stephen Gaffney, Stig Petersen, Karen Simonyan, Tom Schaul, Hado van Hasselt, David Silver, Timothy P. Lillicrap, Kevin Calderone, Paul Keet, Anthony Brunasso, David Lawrence, Anders Ekermo, Jacob Repp, and Rodney Tsing. StarCraft II: A new challenge for reinforcement learning. *CoRR*, abs/1708.04782, 2017.

Appendix A

Appendix

Appendices are *optional* chapters in which you cover additional material that is required to support your hypothesis, experiments, measurements, conclusions, etc. that would otherwise clutter the presentation of your research.