UNIVERSITÉ DE FRIBOURG
UNIVERSITÄT FREIBURG

eXascale Infolab

UNIVERSITY OF FRIBOURG

MASTER THESIS

Exploration of novel Methods in Reinforcement Learning using Black-Box-Optimization

Author:
Corina Masanti

Supervisor:
Dr. Giuseppe Cuccu

Co-Supervisor:
Prof. Dr. Philippe Cudré-Mauroux

January 01, 1970

eXascale Infolab
Department of Informatics

Boulevard de Pérolles 90 • 1700 Fribourg • Switzerland
phone +41 (26) 300 84 65 • diuf-secr@unifr.ch • www3.unifr.ch/inf

Abstract

Corina Masanti

Exploration of novel Methods in Reinforcement Learning using Black-Box-Optimization

Neural networks as generic function approximators can solve many challenging problems. However, they can only be applied successfully for a suited problem structure. In specific, neural networks require differentiability. But there are many areas where calculating an accurate gradient is non-trivial, including problems in Reinforcement Learning (RL). In contrast, Black-Box Optimization (BBO) techniques are less limiting. They presume no constraints on the problem structure, the model, or the solution. With this flexibility, we can study alternative models to neural networks that are unexplored in the context of RL. This thesis aims to achieve pleasing results with a function approximator other than neural networks. I analyze promising models optimized with BBO methods.

Problem -> Solution -> Results

Keywords: Black-Box Optimization, Reinforcement Learning

Contents

List of Figures

Chapter 1

Introduction

Your introduction chapter here.

1.1 Black-Box Optimization

1.1.1 Evolution Strategies

Evolution Strategies (ES) ...

1.1.2 Covariance Matrix Adaptation Evolution Strategy

Covariance Matrix Adaptation Evolution Strategy (CMA-ES) ...

1.2 Alternative Function Approximators

1.2.1 Polynomial

1.2.2 Fourier

1.2.3 Bézier

1.3 Previous Work

1.3.1 Benchmarks in Reinforcement Learning

When developing a novel algorithm, we have to discover how it performs compared to existing methods. For this evaluation, we need standard benchmark problems. OpenAI Gym [1] is a toolkit created for exactly this scenario. It contains a collection of benchmark problems with various levels of difficulty. However, not all benchmark problems are meaningful for the evaluation of an algorithm. If a problem is trivial to solve, the results do not reflect the quality of the model adequately.
In the paper *Analyzing Reinforcement Learning Benchmarks with Random Weight Guessing* (Oller, Glasmachers, and Cuccu (2020)), the authors analyze and visualize the complexity of standard RL benchmarks based on score distribution. They tested their approach on the five Classic Control benchmarks from the OpenAI Gym interface: `CartPole`, `Acrobot`, `Pendulum`, `MountainCar`, and `MountainCarContinuous`. Given an RL environment, the authors conducted a fixed series of experiments. For these experiments, they used three NN architectures ($N_{architectures} = 3$): a network without any hidden layers (0 HL), a network with a single hidden layer of 4 units (1 HL, 4 HU), and a network with two hidden layers of 4 units each (2 HL, 4 HU).

[1] `gym.openai.com`

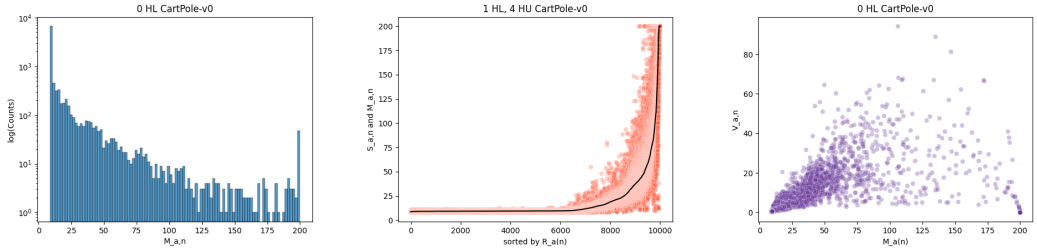
To avoid bias in the data, they did not include any learning. Instead, they chose the network weights i.i.d. from the standard normal distribution $\mathcal{N}(0,1)$ with Random Weight Guessing (RWG). With this, they initialized 10^4 samples ($N_{samples} = 10^4$) with different random weights. Each of these samples represents a controller that maps observations to actions in the environment. Therefore, neural networks denote the controllers. However, the goal of this thesis is to use an alternative function approximator representing the controller. The controllers were tested on an environment for 20 independent episodes ($N_{episodes} = 20$). For each episode, they saved the score in the score tensor S. Algorithm 1 illustrates the procedure with pseudocode.

Algorithm 1 Evaluation process taken from Oller, Glasmachers, and Cuccu (2020)

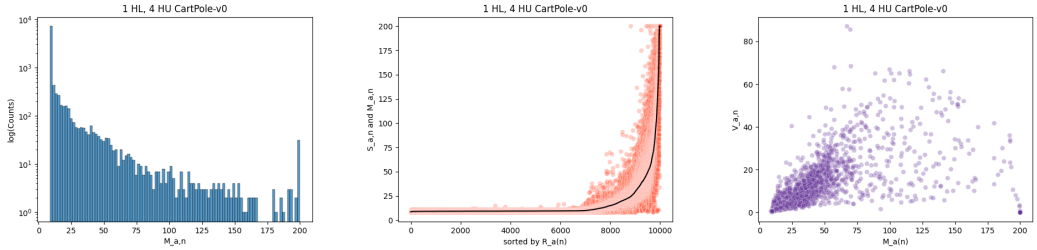
Initialize environment
Create array S of size $N_{architectures} \times N_{samples} \times N_{episodes}$
for $n = 1, 2, ..., N_{samples}$ **do**
    Sample NN weights randomly from $\mathcal{N}(0,1)$
    for $e = 1, 2, ..., N_{episodes}$ **do**
        Reset the environment
        Run episode with NN
        Store accured episode reward in $S_{a,n,e}$

After the authors obtained the scores, they calculated the mean performance over all episodes from a sample and its variance. The samples were ranked according to their mean score. They then visualized their results with three plots: a log-scale histogram of the mean scores, a scatter plot of the sample scores over their rank, a scatter plot of score variance over the mean score. I reproduced their results following the mentioned methodology. My findings for the environment `CartPole` are displayed in Figure 1.1.

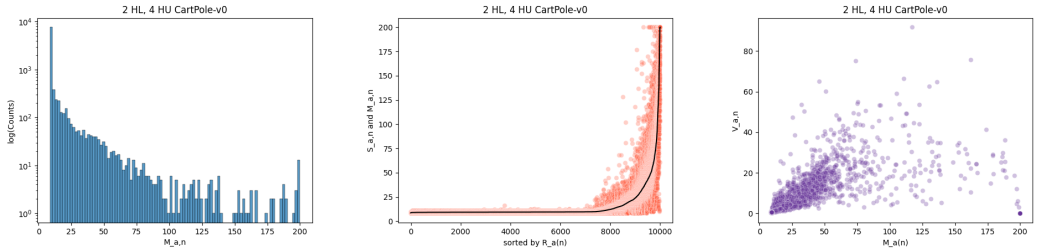
Describe method -> results -> conclusion
Describe plots, maybe add plots from other environments? now not really meaningful to show all architectures (not enough difference)



(A) Results of network architecture without hidden layers



(B) Results of network architecture with one hidden layer



(C) Results of network architecture with two hidden layers

FIGURE 1.1: **Results.** Results illustrated as histogram and scatter plots.

Chapter 2

Experiments

2.1 Experiments

2.2 Results

Chapter 3

Conclusion

3.1 Conclusion

In this work we...

3.2 Future Work

The continuation of this work includes...

Bibliography

Oller, Declan, Tobias Glasmachers, and Giuseppe Cuccu (Apr. 16, 2020). "Analyzing Reinforcement Learning Benchmarks with Random Weight Guessing". In: *arXiv:2004.07707 [cs, stat]*. arXiv: 2004.07707. URL: http://arxiv.org/abs/2004.07707 (visited on 12/07/2021).