Department of Computer Science
Series of Publications A
Report A-2015-0

Deep Recurrent Neural Network for Robot Reinforcement Learning

Yuan Gao

University of Helsinki
Finland

Supervisor

Dorota Glowacka, University of Helsinki, Finland
Leo Kärkkäinen, Nokia Research Center, Finland
Honkala Mikko Nokia Research Center, Finland

Pre-examiners

Opponent

Custos

Contact information

Department of Computer Science
P.O. Box 68 (Gustaf Hällströmin katu 2b)
FI-00014 University of Helsinki
Finland

Email address: info@cs.helsinki.fi
URL: http://cs.helsinki.fi/
Telephone: +358 2941 911, telefax: +358 9 876 4314

ISSN 1238-8645
ISBN 000-00-0000-0 (paperback)
ISBN 000-00-0000-0 (PDF)
Computing Reviews (1998) Classification: A.0, C.0.0
Helsinki 2015
Unigrafia

Todo list

🟧 citation 1
🟧 citation 1

Deep Recurrent Neural Network for Robot Reinforcement Learning

Yuan Gao

Department of Computer Science
P.O. Box 68, FI-00014 University of Helsinki, Finland
gaoyuankidult@gmail.com
http://www.cs.helsinki.fi/u/yuangao/

PhD Thesis, Series of Publications A, Report A-2015-0
Helsinki, September 2015, 7 pages
ISSN 1238-8645
ISBN 000-00-0000-0 (paperback)
ISBN 000-00-0000-0 (PDF)

Abstract

Computing Reviews (1998) Categories and Subject Descriptors:

A.0	Example Category
C.0.0	Another Example

General Terms:

Additional Key Words and Phrases:

Acknowledgements

Contents

1 Introduction **1**

2 Reinforcement Learning **3**

2.1 Markov Decision Process 3

2.1.1 Partially Observable Markov Decision Process 3

2.1.2 Markov Decision Process with Continuous States . . 3

2.1.3 Value Functions 3

2.2 Reinforcement Learning Methods 3

2.2.1 Temporal Difference Learning 4

2.2.2 Q-Learning 4

2.2.3 Adaptive Heuristic Critic 4

2.2.4 Policy Gradient Methods 4

2.3 Classification of the Regarded RL Problems 4

2.3.1 High-Dimensionality 4

2.3.2 Real-World Samples 5

2.3.3 Under-Modelling and Model Uncertainty 5

2.3.4 Goal Specification 5

3 Deep Recurrent Neural Networks **7**

3.1 Deep Learning and its Recent Advances 7

3.2 Feedforward Neural Networks 7

3.3 Recurrent Neural Networks 7

3.3.1 Finite Unfolding in Time 8

3.3.2 Overshooting 8

3.3.3 Dynamical Consistency 8

3.4 Universal Approximation 8

3.4.1 Approximation by FFNN 8

3.4.2 Approximation by RNN 8

3.5 Training of RNN 8

3.5.1 Shared Weight Extended Backpropagation 9

3.5.2 Learning Long-Term Dependencies 9
3.5.3 Optimization Methods 9
3.6 Improved Model-Building with RNN 10
3.6.1 Handling Data Noise 10
3.6.2 Optimal Weights Initialization 10

4 Prior Arts of Combining Deep NN and RL 11
4.1 RL-LSTM based on Model/Critic 11
4.2 Deep Q Network 11
4.3 Recurrent Attention Model 11

5 Proposed Recurrent Neural Network Structure 13
5.1 Stacked LSTM Layers as Model-Critic Approach 13
5.2 Stacked Bidirectional LSTM Layers as Model-Critic Approach 13
5.3 Replacing LSTM with Mikolov's Context Layer 14

6 Experiment 15
6.1 System Implementation 15
6.2 Cart-pole Balancing 15
6.3 Stacking Wooden Blocks 15
6.4 Cart-pole Balancing Using Baxter Robot 16

7 Future Research 17
7.1 Combining Control with Vision Using Deep Neural Network 17
7.2 Transfer Learning for Learning Repetitive Behaviour Using Deep Neural Network 17

References 19

Chapter 1

Introduction

Controlling a complicated mechanical system to perform a certain task, for example making robot to dance, is a traditional problem studied in the field of control theory. Many successful applications like Google BigDog[10] and Google Self-driving car [**?**] have been made in accordance to the new theories found in this field.

However more evidences show that in-cooperating with machine learning techniques in robotics can enable people to get rid of tedious engineering works of adjusting environmental parameters. Many researchers like Jan Peters, Sethu Vijayakumar, Stefan Schaal, Andrew Ng and Sebastian Thrun are the early explorers in this field. Based on the Partial Observable Markov Decision Process(POMDP) reinforcement learning, they contributed first several algorithms enabling robot to learn to perform a certain task overtime.

citation

citation

Recently, one sub-field of machine learning called deep learning gained a lot of attention as a method attempting to model high-level abstractions by using model architectures composed by multiple non-linear layers. (for example [6]). Several architectures of deep learning networks like deep belief network [4], deep Boltzman machine [11], convolutional neural network [6] and deep de-noising auto-encoder [12] have shown its advantages in specific areas. Especially, convolutional neural network, which was invented by Krizhevsky, outperformed all the traditional feature-based machine learning techniques in imagenet competition.

Based on the two trends we noticed, a natural path of research is to use deep learning methods for controlling movements of robot. Until the end of 2014, the main works of deep learning are more related to a category of robotics called perception, which deals with problems like Sensor Fusion [9], Nature Language Processing(NLP)[1] and Object Recognition[7][5]. Although considered briefly in Jürgen Schmidhuber's team[8], the other area of robotics, namely control, remains more-or-less unexplored in the realm of deep learning.

The researches done in Jürgen Schmidhuber's team provided several interesting structures that might be potentially useful for robot control. The name of one of these structures is called Long Short Term Memory (LSTM), which is one variation of Recurrent Neural Network(RNN). Several experiments like generating sequences[2], speech recognition[**?**] and neural turing machine [3] show that it has ability of extracting and storing temporal information from data. As a consequence, this specific structure of RNN, with modification, can be applied for control problems of robots.

There are two main focuses of this thesis. One main focus of this Thesis is to introduce general learning methods for robot control problem with an emphasis on deep learning method. With experiments, the algorithm is able to show it can provide better results than previous method. Another focus of this thesis is to hint that neural network solution is a nature way to combine vision and other sensory input. It is possible that neural networks can be unified model for robotics.

Chapter 2

Reinforcement Learning

- We introduce background of reinforcement learning.
- We introduce basic definition of reinforcement learning.

2.1 Markov Decision Process

- We formalize Markov Decision Process(MDP).

2.1.1 Partially Observable Markov Decision Process

- We extend our definition of MDP with stochastic elements to form Partial Observable Markov Decision Process(POMDP).

2.1.2 Markov Decision Process with Continuous States

- We consider discretization of the states.

2.1.3 Value Functions

- We consider two values functions of POMDP.
- One is state value function.
- Another one is state-action value function.

2.2 Reinforcement Learning Methods

- We introduce basic reinforcement learning algorithms(or concepts) that might be used in the following text.

- We also describe developmental background of these algorithms.

2.2.1 Temporal Difference Learning

- Temporal Difference(TD) learning is one of three basic methods in RL.

- We briefly introduce its relationship with Dynamic Programming and Monte Calo Method.

2.2.2 Q-Learning

- We introduce basic background of Q learning.

- Describe Q learning algorithm.

2.2.3 Adaptive Heuristic Critic

- We introduce basic background of Adaptive Heuristic Critic(AHC).

- Describe AHC architecture.

2.2.4 Policy Gradient Methods

- We introduce basic background of Policy Gradient(PG) methods.

- We discuss general approaches to policy gradient estimation.

- We discuss why PG is useful in area of reinforcement learning for robotics.

2.3 Properties of the Regarded RL Problems

- We tell readers what kind of RL problems we are considering.

- We quote four curses of applying reinforcement learning in robot from this paper. (paper)

2.3.1 High-Dimensionality

- High-Dimensionality is typical in the area of robotics. A lot of robots have many degrees of freedom.

2.3.2 Real-World Samples

- Safe exploration becomes a key issue of the learning process. Normally robot systems suffers from wear and tear.

2.3.3 Under-Modelling and Model Uncertainty

- Simulators may under-model the environment.
- A direct result is that the simulated robot can quickly diverge from the real-world system.

2.3.4 Goal Specification

- Define a reasonable reward function is difficult due to human also has not assumption of statistical property of the system.

Chapter 3

Deep Recurrent Neural Networks

- We introduce definition and background of Deep Learning(DL).

3.1 Deep Learning and its Recent Advances

- Several examples of DL are given including Convolutional Neural Network(CNN), Deep Boltzmann Machine(DBM) and Deep Bidirectional RNN.

- For each of them, several state-of-art applications are given. For example, Google LeNet(i.e. variation of CNN) is the best model for image classification task.

3.2 Feedforward Neural Networks

- We introduce basic structure of perceptron including bias neuron and non-linear transformation function.

- We then show how to fully connect several layers of perceptron to be a FeedForward Neural Network(FFNN).

3.3 Recurrent Neural Networks

- We introduce definition of recurrent connection and ”vanilla” RNN.

- We then discuss about properties of RNN and how they might be used for robot control.

3.3.1 Finite Unfolding in Time

- Finite unfolding in time is a typical way of understanding RNN and it is also preprocessing step before training
- We introduce the concept and important of finite unfolding in time.

3.3.2 Overshooting

- Overshooting happens when the training model does not have example any more.
- Here we discuss the concept, use and potential problem of overshooting.

3.3.3 Dynamical Consistency

- We introduce one solution(i.e. connect output of previous time step to input of next time step) to overshooting problem.

3.4 Universal Approximation

- We introduce Universal approximation theorem for both FFNN and RNN.

3.4.1 Approximation by FFNN

- We introduce Universal approximation theorem for FFNN(Honik's Universal Approximation Theorem)

3.4.2 Approximation by RNN

- We introduce Universal approximation theorem for RNN(Extension of Honik's Universal Approximation Theorem)

3.5 Training of RNN

- We introduce basic training method of RNN.
- We note that the method we use here is variant of Back-Propagation Through Time(BPTT) but there are other training methods like Real-time Recurrent Learning(RLRL) and Extended Kalman Filtering(EKF).

- We only discuss relevant ones.

3.5.1 Shared Weight Extended Backpropagation

- We first introduce Back Propagation for FFNN.
- We then extend that to BPTT for RNN.

3.5.2 Learning Long-Term Dependencies

- We introduce a special structure of RNN called Long Short Term Memory(LSTM).
- We discuss exploding and vanishing gradient problem of RNN.
- We then point out LSTM is able to solve this problem by adding control gates in the structure.
- We show evidences about how long can it keep information.

3.5.3 Optimization Methods

- We introduce optimization methods for training the network.
- It is not enough to just have BPTT as main algorithm. If one needs it to converge faster, one needs to consider other optimization method.
- We note that although here we only consider Nesterov's Momentum and Dropout, there are other methods like normal momentum and fisher-free optimization.
- We only consider relevant ones.

Nesterov's Momentum

- We introduce the concept and background of Nesterov's Momentum.
- We show why we use Nesterov's momentum for training the network.

Dropout

- Dropout is one typical technique used in CNN.
- However, this is used for regularization as a consequence, it can also be used for RNN for preventing overfit.

3.6 Improved Model-Building with RNN

- We show how we consider practical issue in this model.

3.6.1 Handling Data Noise

- We show filtering techniques that we used for handling data noise.

3.6.2 Optimal Weights Initialization

- We show that if we orthogonalize the weight matrix at the beginning of training, the algorithm is able converge faster.

Chapter 4

Prior Arts of Combining Deep NN and RL

- Some attempts have been made for combining deep NN or deep RNN(DRNN) with RL.
- We show some examples of these attempts.

4.1 RL-LSTM based on Model/Critic

- We show one model proposed by Bram Bakker.(paper)

4.2 Deep Q Network

- We show one architecture called Deep Q Network proposed by Google Deep Mind.(paper)

4.3 Recurrent Attention Model

- We show one architecture called Recurrent Attention Model proposed by Google Deep Mind.(paper)

Chapter 5

Proposed Recurrent Neural Network Structure

- We introduce our contribution to the field.

5.1 Stacked LSTM Layers as Model-Critic Approach

- We stack several layers of LSTM using the whole history of each epoch for training.

- In this model, first Deep LSTM model tries to predict action probability based on current state of system.

- Second LSTM model tried to take current state of system and action probability as input and output predicted reward.

5.2 Stacked Bidirectional LSTM Layers as Model-Critic Approach

- Bidirectional RNN has been a technique in recognition and prediction tasks. Recently, it has achieved state-of-art performance in speech recognition task.(paper)

- For solving more difficult reinforcement learning problem, the bidirectional RNN might be used for prediction.

5.3 Replacing LSTM with Mikolov's Context Layer

- Mikolov's context layer was Introduced by Tomas Mikolov in December 2014. It is shown that it can remember long time dependence just as LSTM do. However, Mikolov's context layer has less parameters.

- Replacing LSTM with Mikolov might be faster.

Chapter 6

Experiment

6.1 System Implementation

- Introduce what is experiment setting.
- Introduce the python-based GPU(Heterogeneous) deep learning library Theano.
- Introduce Qt-based physics simulator.
- Introduce ROS, Gazebo and moveit for Baxter simulation.
- We also describe software architecture and communication between training program and simulators.

6.2 Cart-pole Balancing

- We start from a simple cart-pole balancing task with both markovian and non-markovian states.
- This example is meant to show that recurrent neural network is able to approximate value function of markov decision process.
- Compare the results of my architectures with results mentioned in paper(paper).

6.3 Stacking Wooden Blocks

- In this experiment, robot needs to learn to stack three wooden blocks on top of each other.

- On one hand, this example is more difficult than previous one, as a consequence, it may give a better idea about what the algorithm can do.
- On the another hand, it may help me to do transfer learning research later.
- We also show the results of my architectures on this task.

6.4 Cart-pole Balancing Using Baxter Robot

- This test is meant for illustrating how my methods can help learning of real robot.

Chapter 7

Future Research

- Introduce what might be considered as future research directions.

7.1 Combining Control with Vision Using Deep Neural Network

- One may use unified neural network model for vision and control together.

- Deep neural network archived state-of-art or close to state-of-art result in video recognition tasks.(also in Image Recognition and Speech Recognition)

7.2 Transfer Learning for Learning Repetitive Behaviour Using Deep Neural Network

- In image classification task, people found out it takes much more less time to learn weights from a trained model, even if the model was trained from a very different dataset.

- For example, in convolutional neural network, no matter using what dataset, the weights of higher layer are similar.

- One may consider using it for learning repetitive behaviour.(For example, building a higher layer on trained RNN for training a task that might be similar to previously task.)

- The weights of neurons in higher layer may become similar(as they are all repetitive) by supervised or semi-supervised training. (inspired by works: (paper 1), (paper 2)

References

[1] K. Cho, B. van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*, arXiv, (2014).

[2] A. Graves, *Generating sequences with recurrent neural networks*, arXiv preprint arXiv:1308.0850, (2013), pp. 1–43.

[3] A. Graves, G. Wayne, and I. Danihelka, *Neural Turing Machines*, (2014), pp. 1–26.

[4] G. E. Hinton, S. Osindero, and Y. W. Teh, *A fast learning algorithm for deep belief nets.*, Neural computation, 18 (2006), pp. 1527–54.

[5] J. Hoffman, S. Guadarrama, E. Tzeng, R. Hu, J. Donahue, R. Girshick, T. Darrell, and K. Saenko, *LSDA: Large Scale Detection Through Adaptation*, (2014), pp. 1–9.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *ImageNet Classification with Deep Convolutional Neural Networks*, Advances In Neural Information Processing Systems, (2012), pp. 1–9.

[7] I. Lenz, H. Lee, and A. Saxena, *Deep Learning for Detecting Robotic Grasps*, CoRR, abs/1301.3 (2013).

[8] H. Mayer, F. Gomez, D. Wierstra, I. Nagy, A. Knoll, and J. Schmidhuber, *A system for robotic heart surgery that learns to tie knots using recurrent neural networks*, in IEEE International Conference on Intelligent Robots and Systems, 2006, pp. 543–548.

[9] P. O'Connor, D. Neil, S. C. Liu, T. Delbruck, and M. Pfeiffer, *Real-time classification and sensor fusion with a spiking deep belief network*, Frontiers in Neuroscience, (2013).

[10] M. Raibert, K. Blankespoor, G. Nelson, and R. Playter, *BigDog , the Rough-Terrain Quaduped Robot*, tech. rep., 2008.

[11] R. Salakhutdinov and G. Hinton, *Deep Boltzmann Machines*, Artificial Intelligence, 5 (2009), pp. 448–455.

[12] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, *Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion*, Journal of Machine Learning Research, 11 (2010), pp. 3371–3408.