POLITECNICO DI MILANO
Master's Degree in Computer Science and Engineering
Dipartimento di Elettronica, Informazione e Bioingegneria

FAST REINFORCEMENT LEARNING USING DEEP STATE-ACTION FEATURES

AI & R Lab
Laboratorio di Intelligenza Artificiale
e Robotica del Politecnico di Milano

Supervisor: Prof. Marcello Restelli
Co-supervisor: Matteo Pirotta, Ph.D.

Master's Thesis by:
Daniele Grattarola (student ID 853101)

Academic Year 2016-2017

To. . .

Contents

Summary

Il sommario deve contenere 3 o 4 frasi tratte dall'introduzione di cui la prima inquadra l'area dove si svolge il lavoro (eventualmente la seconda inquadra la sottoarea più specifica del lavoro), la seconda o la terza frase dovrebbe iniziare con le parole "Lo scopo della tesi è ..." e infine la terza o quarta frase riassume brevemente l'attivitàï¿½ svolta, i risultati ottenuti ed eventuali valutazioni di questi.

NB: se il relatore effettivo è interno al Politecnico di Milano nel frontesizo si scrive Relatore, se vi è la collaborazione di un altro studioso lo si riporta come Correlatore come sopra. Nel caso il relatore effettivo sia esterno si scrive Relatore esterno e poi bisogna inserire anche il Relatore interno. Nel caso il relatore sia un ricercatore allora il suo Nome COGNOME dovràï¿½ essere preceduto da Ing. oppure Dott., a seconda dei casi.

Acknowledgments

I thank...

Chapter 1

Introduction

L'introduzione deve essere atomica, quindi non deve contenere nè sottosezioni nè paragrafi nè altro. Il titolo, il sommario e l'introduzione devono sembrare delle scatole cinesi, nel senso che lette in quest'ordine devono progressivamente svelare informazioni sul contenuto per incatenare l'attenzione del lettore e indurlo a leggere l'opera fino in fondo. L'introduzione deve essere tripartita, non graficamente ma logicamente:

1.1 Inquadramento generale

La prima parte contiene una frase che spiega l'area generale dove si svolge il lavoro; una che spiega la sottoarea più specifica dove si svolge il lavoro e la terza, che dovrebbe cominciare con le seguenti parole "lo scopo della tesi è ...", illustra l'obbiettivo del lavoro. Poi vi devono essere una o due frasi che contengano una breve spiegazione di cosa e come è stato fatto, delle attività sperimentali, dei risultati ottenuti con una valutazione e degli sviluppi futuri. La prima parte deve essere circa una facciata e mezza o due

1.2 Breve descrizione del lavoro

La seconda parte deve essere una esplosione della prima e deve quindi mostrare in maniera più esplicita l'area dove si svolge il lavoro, le fonti bibliografiche più importanti su cui si fonda il lavoro in maniera sintetica (una pagina) evidenziando i lavori in letteratura che presentano attinenza con il lavoro affrontato in modo da mostrare da dove e perché è sorta la tematica di studio. Poi si mostrano esplicitamente le realizzazioni, le direttive future di ricerca, quali sono i problemi aperti e quali quelli affrontati e si

ripete lo scopo della tesi. Questa parte deve essere piena (ma non grondante come la sezione due) di citazioni bibliografiche e deve essere lunga circa 4 facciate.

1.3 Struttura della tesi

La terza parte contiene la descrizione della struttura della tesi ed è organizzata nel modo seguente. "La tesi è strutturata nel modo seguente.

Nella sezione due si mostra ...

Nella sez. tre si illustra ...

Nella sez. quattro si descrive ...

Nelle conclusioni si riassumono gli scopi, le valutazioni di questi e le prospettive future ...

Nell'appendice A si riporta ... (Dopo ogni sezione o appendice ci vuole un punto)."

I titoli delle sezioni da 2 a M-1 sono indicativi, ma bisogna cercare di mantenere un significato equipollente nel caso si vogliano cambiare. Queste sezioni possono contenere eventuali sottosezioni.

Chapter 2

Background

Introduce the section

2.1 Deep Learning

Deep Learning is a branch of machine learning which exploits *Artificial Neural Networks* (ANN) with more than one hidden layer to learn an abstract representation of the input space [1].

Deep learning techniques can be applied to the three main classes of problems of machine learning (supervised, semi-supervised, and unsupervised), and have been used to achieve state-of-the-art results in a variety of learning tasks.

2.1.1 Artificial Neural Networks

Feed-forward Artificial Neural Networks (ANN) are function approximators inspired by the connected structure of neurons and synapses in the animal brain.

2.1.2 Convolutional Neural Networks

Convolutional Neural Networks (CNN) are a type of ANN inspired by the visual cortex in animal brains. CNNs exploit spatially-local correlations in the neurons of adjacent layers through the use of a *receptive field*, a set of weights which is used to transform a local subset of the input neurons of a layer.

2.1.3 Autoencoders

Autoencoders are a type of ANN which are used to learn a sparse and compressed representation of the input space, by sequentially compressing and reconstructing the inputs under some sparsity constraint (typically by minimizing the $L1$ norm of the activations in the innermost hidden layers).

2.2 Reinforcement Learning

2.2.1 Markov Decision Processes

Markov Decision Processes (MDP) are discrete-time, stochastic control processes, that can be used to describe the interaction of an *agent* with an *environment*.

Formally, MDPs are defined as 7-tuples $(S, S^t, A, P, R, \gamma, \mu)$, where:

- S is the set of observable states of the environment.
  When the set observable states coincides with the true set of states of the environment, the MDP is said to be *fully observable*. We will only deal with fully observable MDPs without considering the case of *partially observable* MDPs.

- $S^T \subseteq S$ is the set of *terminal states* of the environment, meaning those states in which the interaction between the agent and the environment ends. A sequence of states observed by an agent during an interaction with the environment and ending in a terminal state is usually called an *episode*.

- A is the set of actions that the agent can execute in the environment.

- $P : S \times A \times S \to [0, 1]$ is a *state transition function* which, given two states $s, s' \in S$ and an action $a \in A$, represents the probability of the agent going to state s' by executing a in s.

- $R : S \times A \to \mathbb{R}$ is a *reward function* which represents the reward that the agent collects by executing an action in a state.

- $\gamma \in (0, 1)$ is a *discount factor* with which the rewards collected by the agent are diminished at each step, and can be interpreted as the agent's interest for rewards further in time rather than immediate.

- $\mu : S \to [0, 1]$ is a probability distribution over S which models the probability of starting the exploration of the environment in a given state.

Episodes are usually represented as sequences of tuples

$$[(s_0, a_0, r_0, s_1), ..., (s_{n-1}, a_{n-1}, r_{n-1}, s_n)]$$

called *trajectories*, where (s_i, a_i, r_i, s_{i-1}) represent a transition of the agent to state s_{i+1} by taking action a_i in s_i and collecting a reward r_i, and $s_n \in S^T$. In Markov Decision Processes, the modeled environment must satisfy the *Markov property*, meaning that the reward and transition functions of the environment must only depend on the current state and action, rather than the past state-action trajectory of the agent.
In other words, an environment is said to satisfy the Markov property when its one-step dynamics allow to predict the next state and reward given only the current state and action.

Policy

The behavior of the agent in an MDP can be defined as a probability distribution $\pi : S \times A \to [0, 1]$ called a *policy*, which given $s \in S, a \in A$, represents the probability of selecting a as next action from s. An agent which uses this probability distribution to select its next action when in a given state is said to be *following* the policy.

Value Functions

Starting from the concept of policy, we can now introduce a function that evaluates how good it is for an agent following a policy π to be in a given state. This evaluation is expressed in terms of the expected return, i.e. the expected discounted sum of future rewards collected by an agent starting from a state while following π, and the function that computes it is called the *state-value function for policy* π.

Formally, the state-value function associated to a policy π is a function $V^\pi : S \to \mathbb{R}$ defined as:

$$V^\pi(s) = E_\pi[R_t | s_t = s] \tag{2.1}$$

$$= E_\pi[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s] \tag{2.2}$$

where $E_\pi[\cdot]$ is the expected value given that the agent follows policy π, and t is any time step of an episode of n steps $[s_1, ..., s_t, ..., s_n]$ where $s_t \in S, \forall t = 1, ..., n$.

Similarly, we can also introduce a function that evaluates the goodness of taking a specific action in a given state, namely the expected reward obtained by taking an action $a \in A$ in a state $s \in S$ and then following policy π. We call this function the *action-value function for policy* π denoted $Q^\pi : S \times A \to \mathbb{R}$, and defined as:

$$Q^\pi(s,a) = E_\pi[R_t | s_t = s, a_t = a] \tag{2.3}$$

$$= E_\pi[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a] \tag{2.4}$$

The majority of reinforcement learning algorithms is based on computing (or estimating) value functions to then control the behavior of the agent in order to maximize the expected reward collected during episodes.

We also note a fundamental property of value functions, which satisfy particular recursive relationships like the following *Bellman equation for* V^π:

$$\begin{aligned}
V^\pi(s) &= E_\pi[R_t | s_t = s] \\
&= E_\pi[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s] \\
&= E_\pi[r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} | s_t = s] \qquad (2.5) \\
&= \sum_{a \in A} \pi(s,a) \sum_{s' \in S} P(s,a,s')[R(s,a) + \\
&\quad + \gamma E_\pi[\sum_{k=0}^{\infty} \gamma^k r_{t+k+2} | s_{t+1} = s]] \qquad (2.6) \\
&= \sum_{a \in A} \pi(s,a) \sum_{s' \in S} P(s,a,s')[R(s,a) + \gamma V^\pi(s')] \qquad (2.7)
\end{aligned}$$

Intuitively, this relation decomposes the state-value function as the sum of the immediate reward collected from a state s to a successor state s', and the value of s' itself; by considering the transition model of the MDP and the policy being followed, we see that Bellman equation simply averages the expected return over all the possible (s, a, r, s') transitions, by taking into account the probability that these transitions occur.

2.2.2 Optimal Value Functions

In general terms, *to solve* a reinforcement learning task is to identify a policy that yields a sufficiently high expected return. In the case of MDPs with finite state and actions sets [1], it is possible to define the concept of *optimal policy* as the policy which maximizes the expected return collected by the agent in an episode.
We start by noticing that state-value functions define a partial ordering over policies as follows:

$$\pi \geq \pi' \iff V^{\pi}(s) \geq V^{\pi'}(s), \forall s \in S$$

From this, the *optimal policy* π^* of an MDP is a policy which is better or equal than all other policies in the policy space.
The state-value function associated to π^* is called the *optimal state-value function*, denoted V^* and defined as:

$$V^*(s) = \max_{\pi} V^{\pi}(s), \forall s \in S$$

As we did when introducing the value functions, given an optimal policy for the MDP it is also possible to define the *optimal action-value function* denoted Q^* (the second equivalence in this definition highlights the relation between Q^* and V^*):

$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a) \tag{2.8}$$
$$= E[r_{t+1} + \gamma V^*(s_{t+1}|s_t = s, a_t = a] \tag{2.9}$$

2.2.3 Value-based optimization

SARSA

Q-learning

Fitted Q-Iteration

2.3 Deep Reinforcement Learning

2.3.1 Deep Q-Learning

2.3.2 Asynchronous Advantage Actor Critic

[1] This specification is only required for formality, but is not investigated further in this work. Refer to SUTTON, BARTO for more details on the subjet of non-finite MDPs.

Chapter 3

State Of The Art

Chapter 4

Effects of Feature Selection on the Performance of Agents

In questa sezione si deve descrivere l'obiettivo della ricerca, le problematiche affrontate ed eventuali definizioni preliminari nel caso la tesi sia di carattere teorico.

4.1 Formalism

4.2 Tree-based Recursive Feature Selection

4.3 Impact of RFS on Performance of Agents

4.4 RFS on Deep Features

Chapter 5

Technical Details and Implementation

Si mostra il progetto dell'architettura del sistema con i vari moduli.

5.1 Atari Games

5.2 Deep Neural Network for Feature Extraction

5.3 Recursive Feature Selection

5.4 Code Implementation

Chapter 6

Experimental Results

Si mostra il progetto dal punto di vista sperimentale, le cose materialmente realizzate. In questa sezione si mostrano le attività sperimentali svolte, si illustra il funzionamento del sistema (a grandi linee) e si spiegano i risultati ottenuti con la loro valutazione critica. Bisogna introdurre dati sulla complessità degli algoritmi e valutare l'efficienza del sistema.

Chapter 7

Conclusions and Future Developments

Si mostrano le prospettive future di ricerca nell'area dove si è svolto il lavoro. Talvolta questa sezione può essere l'ultima sottosezione della precedente. Nelle conclusioni si deve richiamare l'area, lo scopo della tesi, cosa è stato fatto,come si valuta quello che si è fatto e si enfatizzano le prospettive future per mostrare come andare avanti nell'area di studio.

7.1 Future Developments

Bibliography

[1] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.