R.U "Angel Kunchev"

Studend science session 20.05.2021

Topic: Reward value effects on Q-learning agents

prepared: msc.eng. Martin Kaloev

Abstract

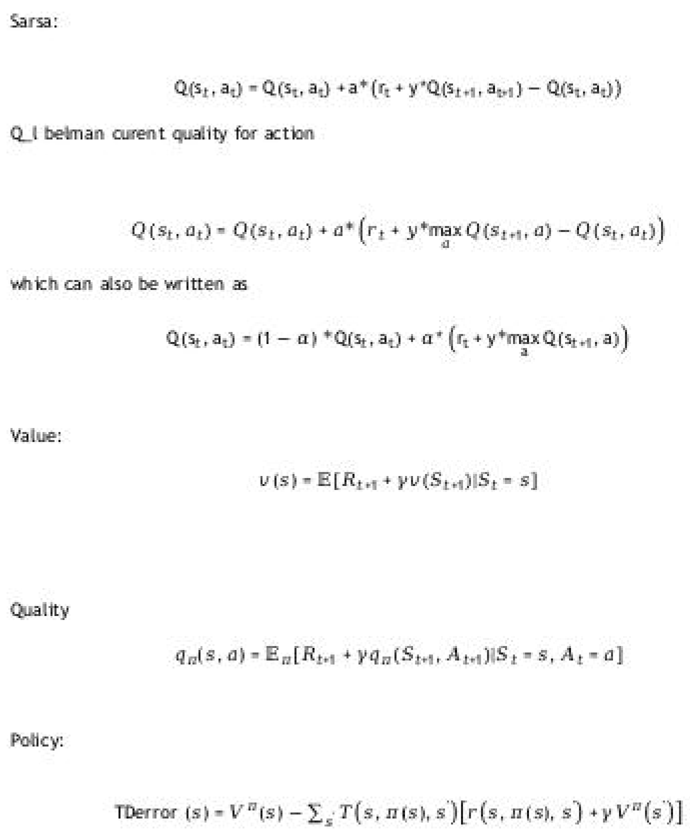
- One of the great advances in the field of artificial intelligence is the creation of general artificial intelligence (AGI). Or artificial intelligence allowing an agent to perform tasks of human complexity. This is due to the development of neural networks that implement in practice the concepts described by adaptive programming.
- Adaptive programming is a set of algorithms and methods describing the relationships between actions and states.
- Artificial intelligence of this type plays an important role in the modern world, in the areas like targeted advertising; digital assisted trade; robotics; transport and others.
- What all these agents have in common is the ability to learn from observed processes and interact with users.
- This article discusses the work of such an agent.The focus of the studie is the relationship between the change in the policy used by the agent and the change in the value of the rewards received from an action.
- The agent's task is to solve a two-dimensional spatial problem using a greedy epsilon policy. The algorithms used by the agent are an algorithm for determine the quality of actions, a sarsa algorithm and algorithms for evaluation quality of the policy.

Related work

- Articles on a similar topic can be found with the following titles:
- An Analysis of Q-Learning Algorithms with Strategies of Reward Function
- https://www.researchgate.net/publication/50247491_An_Analysis_of_Q-Learning_Algorithms_with_Strategies_of_Reward_Function
- Q-Learning Algorithms: A Comprehensive Classification and Applications
- https://www.researchgate.net/publication/335805245_Q-Learning_Algorithms_A_Comprehensive_Classification_and_Applications
- Playing Atari with Deep Reinforcement Learning
- https://arxiv.org/pdf/1312.5602v1.pdf
- Low-rank State-action Value-function Approximation
- https://arxiv.org/abs/2104.08805v1

Algorithms list

- The list of used algorithms:
- *-Sarsa:*
- *-Quality of action equation by R.E.Bellman*
- *-Value of action equation by R.E.Bellman*
- *-Policy based on action equation.*
- Those equations belongs to the family of discrete mathematics.
- Discrete mathematics is the study of mathematical structures that are countable or otherwise distinct and separable. Examples of structures that are discrete are combinations, graphs, and logical statements.
- Please note the slide named: **Model and equation**
- Manuals for application of algorithms:
- **Reinforcement Learning:**
- **An Introduction**
- by:
- *Richard S. Sutton and Andrew G. Barto*
- A Bradford Book
- The MIT Press
- Cambridge, Massachusetts
- London, England
- ***at: http://incompleteideas.net/book/first/ebook/node41.html***



Sarsa:

$$Q(s_t, a_t) = Q(s_t, a_t) + a^*(r_t + y^*Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$$

Q_l belman curent quality for action

$$Q(s_t, a_t) = Q(s_t, a_t) + a^*\left(r_t + y^*\max_a Q(s_{t+1}, a) - Q(s_t, a_t)\right)$$

which can also be written as

$$Q(s_t, a_t) = (1 - \alpha)^*Q(s_t, a_t) + \alpha^*\left(r_t + y^*\max_a Q(s_{t+1}, a)\right)$$

Value:

$$v(s) = \mathbb{E}[R_{t+1} + \gamma v(S_{t+1}) | S_t = s]$$

Quality

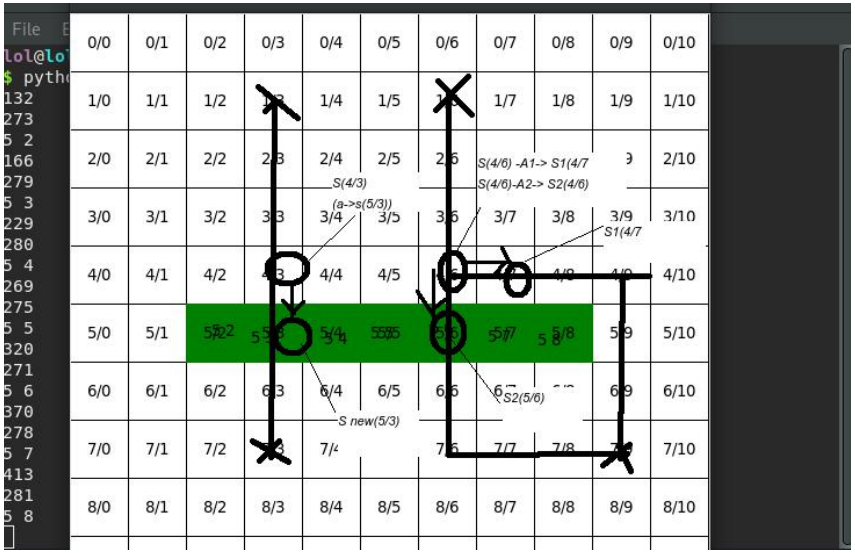
$$q_\pi(s, a) = \mathbb{E}_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) | S_t = s, A_t = a]$$

Policy:

$$\text{TDerror }(s) = V^\pi(s) - \sum_{s'} T\left(s, \pi(s), s'\right)\left[r\left(s, \pi(s), s'\right) + \gamma V^\pi\left(s'\right)\right]$$

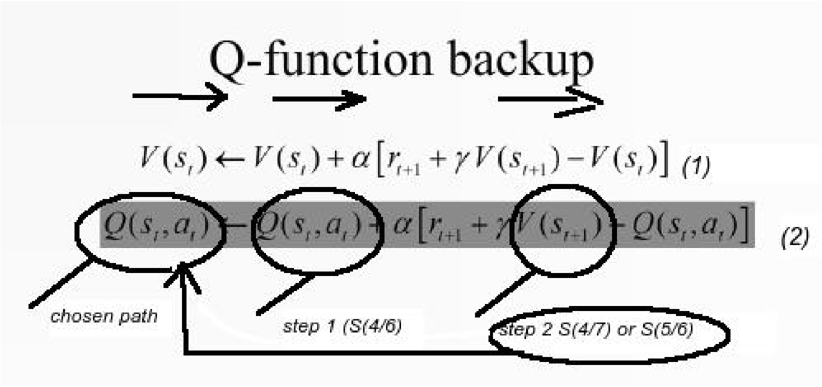
R.U "Angel Kunchev"

Model and Equation



```
lol@lo
$ pytho
132
273
5 2
166
279
5 3
229
280
5 4
269
275
5 5
320
271
5 6
370
278
5 7
413
281
5 8
```

0/0	0/1	0/2	0/3	0/4	0/5	0/6	0/7	0/8	0/9	0/10
1/0	1/1	1/2	1/3	1/4	1/5	1/6	1/7	1/8	1/9	1/10
2/0	2/1	2/2	2/3	2/4	2/5	2/6			9	2/10
3/0	3/1	3/2	3/3	3/4	3/5	3/6	3/7	3/8	3/9	3/10
4/0	4/1	4/2	4/3	4/4	4/5	4/6	4/7	4/8	4/9	4/10
5/0	5/1	5/2	5/3	5/4	5/5	5/6	5/7	5/8	5/9	5/10
6/0	6/1	6/2	6/3	6/4	6/5	6/6	6/7	6/8	6/9	6/10
7/0	7/1	7/2	7/3	7/4		7/6	7/7	7/8	7/9	7/10
8/0	8/1	8/2	8/3	8/4	8/5	8/6	8/7	8/8	8/9	8/10


Q-function backup

$$V(s_t) \leftarrow V(s_t) + \alpha\left[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)\right] \quad (1)$$

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\left[r_{t+1} + \gamma V(s_{t+1}) - Q(s_t,a_t)\right] \quad (2)$$

Model and Equation Explained

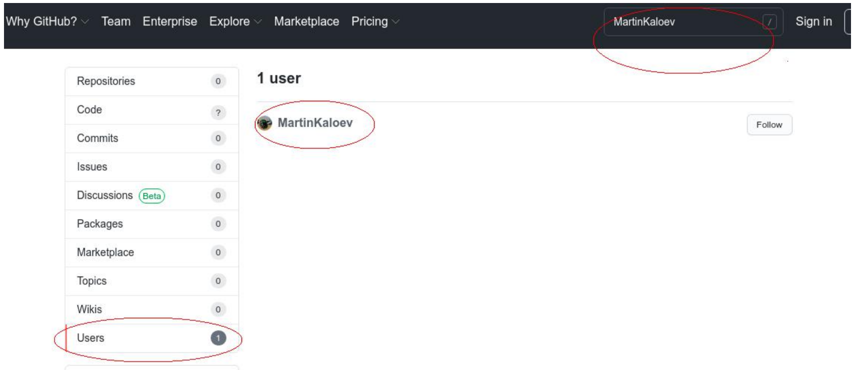
- In the slide: Model and equation you can see an equation that can be interpreted as:
- The quality of a certain action (Q) is determined by the change from initial state (S) to another state (St).
- Q has two attributes (S-state) and (A or V -value of action). Actions of higher quality have priority in choosing a policy or strategy.

- In this particular model, S (4/6) can choose to go into several new states. The state S (4/7) has a higher quality compared to the state S (5/6). From the agent's point of view, it's a better policy to go around a wall instead of crashing into it.

Test recreation

- To repeat the tests performed, please follow the steps below:
- 1) Use a virtual tool of your choice to navigate the World Wide Web to reach current github address
- 2) Download a copy of the software solution used in the tests. (Note **the github navigation** slides)
- 3) Use a virtual tool of your choice to navigate the World Wide Web to reach current jypiterlab address
- 4) Use the menus to access the online version of the software lab (Note **the jypiterlab navigation** slides)
- 5) Load a file named main _ ** _ main in the online version of jypiterlab and run the simulation. (Note the slides **jypiterlab: how to use it**)

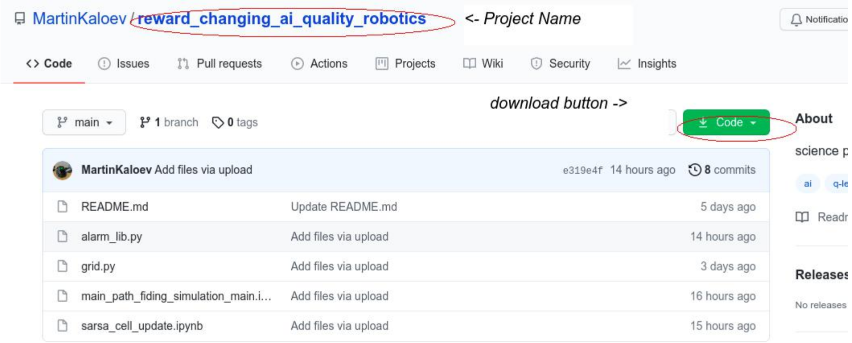
The github navigation



Repositories	0
Code	?
Commits	0
Issues	0
Discussions Beta	0
Packages	0
Marketplace	0
Topics	0
Wikis	0
Users	1

1 user

MartinKaloev



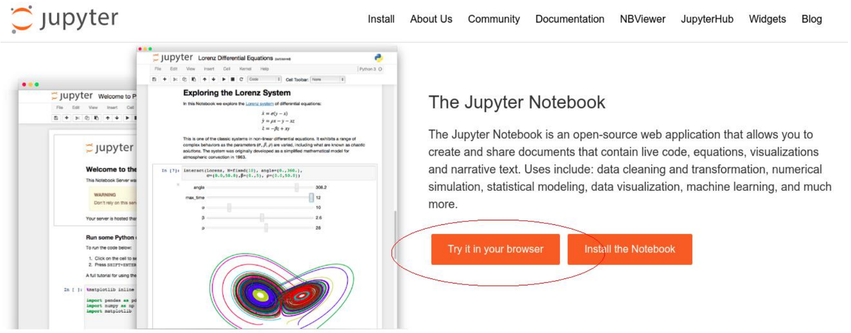
MartinKaloev / reward_changing_ai_quality_robotics <- Project Name

download button ->

MartinKaloev Add files via upload e319e4f 14 hours ago 8 commits

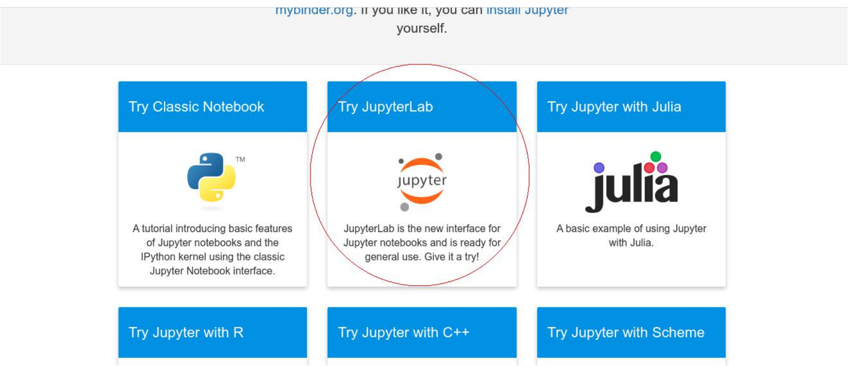
README.md	Update README.md	5 days ago
alarm_lib.py	Add files via upload	14 hours ago
grid.py	Add files via upload	3 days ago
main_path_fiding_simulation_main.i...	Add files via upload	16 hours ago
sarsa_cell_update.ipynb	Add files via upload	15 hours ago

The jypiterlab navigation



The Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.



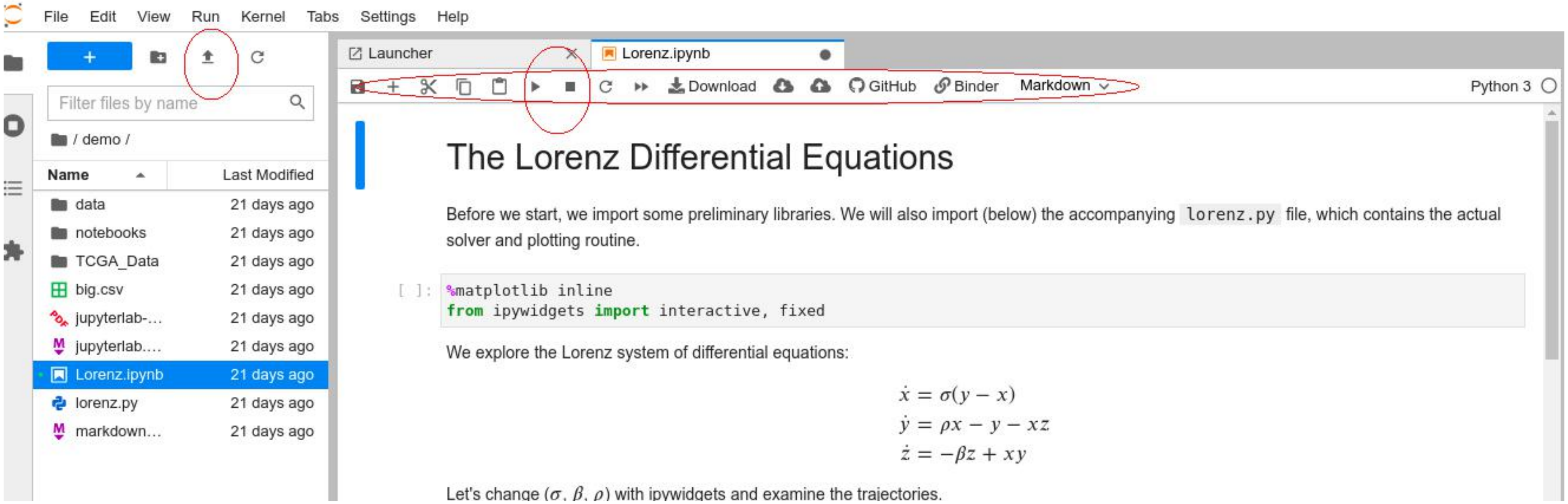
mybinder.org. If you like it, you can install Jupyter yourself.

A tutorial introducing basic features of Jupyter notebooks and the IPython kernel using the classic Jupyter Notebook interface.

JupyterLab is the new interface for Jupyter notebooks and is ready for general use. Give it a try!

A basic example of using Jupyter with Julia.

jypiterlab: How to use it



Name	Last Modified
data	21 days ago
notebooks	21 days ago
TCGA_Data	21 days ago
big.csv	21 days ago
jupyterlab-…	21 days ago
jupyterlab.…	21 days ago
Lorenz.ipynb	21 days ago
lorenz.py	21 days ago
markdown…	21 days ago

The Lorenz Differential Equations

Before we start, we import some preliminary libraries. We will also import (below) the accompanying `lorenz.py` file, which contains the actual solver and plotting routine.

```
[ ]: %matplotlib inline
     from ipywidgets import interactive, fixed
```

We explore the Lorenz system of differential equations:

$$\begin{aligned}\dot{x} &= \sigma(y-x)\\ \dot{y} &= \rho x - y - xz\\ \dot{z} &= -\beta z + xy\end{aligned}$$

Let's change (σ, β, ρ) with ipywidgets and examine the trajectories.

Test Explained

- The virtual tool main _ ** _ main creates a simulation of an agent solving a randomly generated maze in an optimal way. The user can change the value (1) of the actions for each step and the penalties on the assessment of action in violation of the rules of the maze.

- This allows different agent behavior to be observed at different values set by the user.
- In addition, the functionality in cell 7 can be replaced by a Bellman algorithm with a Sarsa algorithm (2).

- 1*see Slide Model and equation explained again
- 2*see sarsa_cell_update sniped

Results

- Tests are performed exhausting all possible combinations. After the tests are completed, a statistical analysis is performed as to which factors play the biggest role in the violation of the rules of the labyrinth by the agent. The results of the tests show that the difference between the value of the detour compared to the value of the penalty in the event of a breach plays the biggest role. The detour path means the path connecting two points that are closest to the breakthrough point.

Discussions and Optimizations

- A possibility to optimize the virtual agent is the addition of a library signaling a possible breakthrough. An example of such a library is shown in alarm_lib.py provided in the project repository.

$$fw = \begin{cases} true, [\sum_{n=1}^{Vp} \ n * Rp] > [\sum_{n=1}^{Vw} \ n * Rw] \\ false, [\sum_{n=1}^{Vp} \ n * Rp] < [\sum_{n=1}^{Vw} \ n * Rw] \end{cases}$$

Where are we?

- Let's take a few minutes to discuss the role of Q-learning as the basis of many of the technologies used in the modern world.

Conclusion

- The article analyzes the change in the behavior of an agent when changing the values of actions.
- Answers to questions about why rewards often have a negative value instead of a positive one. The moments in which the agent violates the rules of the given task are considered and analyzed. A solution for the specific problems of a problem derived in an equation and a software library is proposed.

Thank you for your attention