Can Reinforcement Learning be applied to Surgery?

Agostino Aiezzo
Department of Computer Science and Engineering
University of Bologna
Bologna, BO 40136
agostino.aiezzo@studio.unibo.it

Abstract

The idea behind this project is to replicate and enhance an original work [1] done by a group of gynecologists with minimal knowledge of deep learning. What they tried to do, in fact, is to apply reinforcement learning to surgery and, in particular, to hysterectomy that is the removal of the uterus. To do that, the first step 2 was to reproduce the environment proposed in [1] which is characterized by the object to be removed at the centre (in this case the uterus) and then a series of ligaments and arteries that must be avoided by the intelligent agent. Secondly 3, the algorithm and the model were defined. This latter, in particular, was made more efficient compared to the original one since it has far less parameters. Finally 4, the model was trained using the same number of episodes reported in [1], achieving the same results. Some conclusions 5 were inferred at the end as well as some considerations for future developments.

1 Introduction

The purpose of this project is to replicate and try to improve an original work [1] done by a group of gynecologists who tried to apply reinforcement learning to surgery. In particular, given the profession, they focused on a particular surgical procedure that is hysterectomy: the removal of the uterus. Despite the basic knowledge of deep learning techniques, they achieved interesting results by using a still improvable neural network and a DQN algorithm without particular changes or modifications. The main reasons for selecting this topic is related not only to the margins for improvement left by the original creators, but also because the application of reinforcement learning to a field that is so unusual is interesting and worth to be further explored.

2 The environment

The first step to reproduce the task proposed in [1] was to recreate the environment in which the agent moves. In [1], it was created starting by a publicly available game code for the game “Snake”. In order to save time and make it also more efficient and simpler, in this replica, the environment has been defined as a 10x10 matrix where each cell has a specific value that identify the correspondent element in the surgical area. In Figure 1, there’s an example of the final result.

At the center of the image, it's possible to observe the uterus that is the object the agent must remove. To do that, it must cut the target points: preferred cutting points that consider the densities of the arteries and the distance from the object. All around, there are ligaments and arteries that must be avoided by the agent. To conclude, there's the peritoneum that is the area the agent can navigate in. All the environment is closed by a frame.
Of course, this is a simplification and gamification of a real surgical procedure. Nonetheless, for the purpose of the project is more than sufficient.

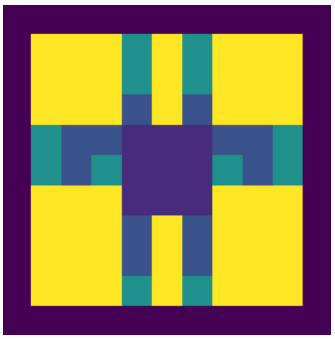

Figure 1: The surgical environment

3 The Agent

The second step has been to define the agent that is basically a knife whose objective is to cut all the target points avoiding the ligaments and the arteries in the area as well as hitting the frame. The original model proposed in [1] is represented in Figure 2:

```
Model: "sequential_layer"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_layer1 (Conv2D)       (None, 10, 10, 16)        80
_________________________________________________________________
conv2d_layer2 (Conv2D)       (None, 8, 8, 32)          4640
_________________________________________________________________
flatten_layer (Flatten)      (None, 2048)              0
_________________________________________________________________
dense_layer1 (Dense)         (None, 256)               524544
_________________________________________________________________
dense_layer2 (Dense)         (None, 4)                 1028
=================================================================
Total params: 530,292
Trainable params: 530,292
Non-trainable params: 0
_________________________________________________________________
```

Figure 2: The original model

It is possible to observe that the number of parameters of the aforementioned neural network is very large for a task that is all in all very simple. This is primarily due to the presence of convolution layers which may be a bit too much. Consequently, a much simpler neural network, mainly characterized by dense layers, was used as a further simplification of the original design. A summary of the new model can be seen in Figure 3. As you can see, the number of parameters is way lower. Precisely, the second model contains (more or less) $1/17$ of the parameters of the first one. For the composition of this smaller neural network, inspiration came from a model used for maze resolution [2], given the similarity with this application.

```
Model: "sequential_layer"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 121)               14762
_________________________________________________________________
p_re_lu (PReLU)              (None, 121)               121
_________________________________________________________________
dense_1 (Dense)              (None, 121)               14762
_________________________________________________________________
p_re_lu_1 (PReLU)            (None, 121)               121
_________________________________________________________________
dense_2 (Dense)              (None, 4)                 488
=================================================================
Total params: 30,254
Trainable params: 30,254
Non-trainable params: 0
_________________________________________________________________
```

Figure 3: The simpler model

For what concerns the algorithm used, it is a basic version of the DQN algorithm without any particular changes or modifications. More than anything else, the most challenging part here was to fine-tune all the hyper-parameters like, for example, the reduction factor for the ε, the batch size for the replay buffer as well as the number of epochs for the retraining, etc. Indeed, time played an important role and outputs were not always immediately available.

4 Results

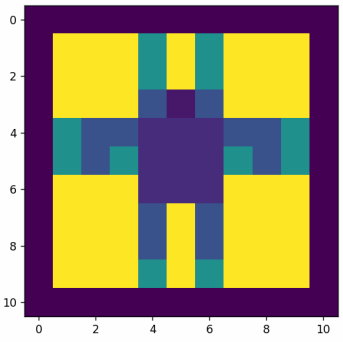

Figure 4: The agent in action

In Figure 4, a test of an agent, trained after 1500 episodes, can be observed.
The animated GIF shows that the path selected by the knife is the optimal one. This is possible thanks to the combination of rewards and punishments selected after some fine-tuning:

- if the agent cuts the uterus (the episode stops) -> -5
- if the agent hits the frame (the episode stops) -> -5
- if the agent cuts a target point -> +5
- if the agent cuts a ligaments or an artery -> -3
- if the agent walk in the peritoneum -> -1
- if the agent visits an already visited cell -> -3

In particular, the last two conditions allow the agent to find the shortest path. Despite the different reconstruction of the environment with respect to the original one and the simplification made to the model, the results obtained are good and comparable with the ones reported in [1].

5 Conclusions

With just few changes, slightly better results have been achieved. From Figure 5, in fact, can be easily deduced that even with less than 1500 episodes (like 1000 or something similar), the optimal

performance can be already reached. Of course, further improvements can be deepened like, for example, using an even simpler model or a more sophisticated DQN algorithm. Also, more difficult and more complex environments should be tested.
Anyway, what can be inferred at the end of this experiment is, for sure, that reinforcement learning applied to surgery is possible, always taking into account the significant simplifications. Certainly, it is just at the beginning and many more investigations and insights need to be conducted.

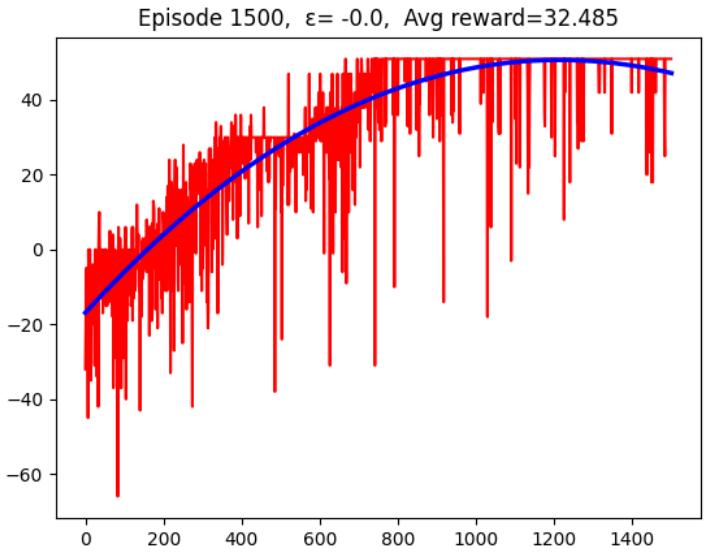

Figure 5: The total reward at each episode and the trend line along all the episodes.

References

[1] Masakazu Sato, Kaori Koga, Tomoyuki Fujii, and Yutaka Osuga. Can reinforcement learning be applied to surgery? In *Artificial Intelligence - Emerging Trends and Applications*, 2018.

[2] Samy Zafrany. Deep reinforcement learning for maze solving. `https://www.samyzaf.com/ML/rl/qmaze.html#Deep-Reinforcement-Learning-for-Maze-Solving`.