Quantum Reinforcement Learning Team Report

Also known as: Birth and Life of Elliot, the Qiskit Quantum Agent

Paolo, Stefano, Jani and Laura

1 Motivation

Reinforcement Learning (RL) is a Machine Learning paradigm where intelligent agents interact with an external environment, and are rewarded for performing *good actions*. The goal of the agents is tho determine the ideal strategy (called *policy*) to maximize the received rewards, thus *solving* the environment (i.e. a task). As the field of artificial intelligence advances, the demand for algorithms that can learn quickly and efficiently increases, and speed-ups are more welcome than ever.
Driven by the promising idea of a *quantum speed-up*, the development of Quantum Reinforcement Learning (QRL) algorithms has begun in the last few years [1, 2, 3]. The main idea of our project is to create and train a Quantum agent, from now on named *Elliot*, developed with `Qiskit`, capable of using quantum computing to speed up its own learning process and possibly outperform its classical counterpart [3]. Since it's an almost completely unexplored field, the first questions we addressed are the fundamentals: how can we translate the building blocks of Classical RL efficiently in a quantum setting? How do quantum agents perceive? What does it mean to take a quantum action?
Last, but not least, the most important questions: how does our quantum agent perform in a fully quantum environment? Can it learn to play quantum games better than a classical agent and human minds?

2 Implementation & Results

Our work can be divided into two macro phases: in the first one, our aim was to give birth to Elliot, a RL algorithm that uses quantum computation as an internal resource, and train it on classical environments; while in the second phase we let Elliot interface with a quantum game. In the first phase, we all collaborated to the algorithm design and theory, although Paolo is the major contributor of the Grover-like algorithm while Stefano is the major contributor to the Parametric Quantum Circuit (PQC) code. In the second phase again we all collaborated to the theory discussion, while Jani wrote the code for the Quantum environment we train Elliot against.

2.1 A Quantum Agent in a Classical World

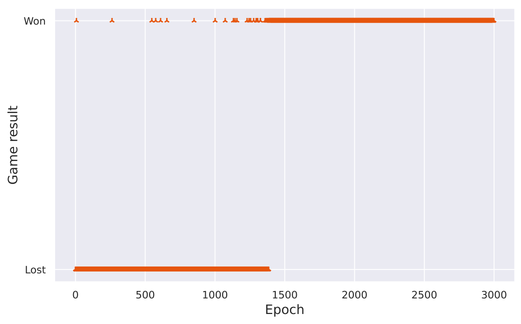
When Elliot interacts with a classical environment, both the *percepts* s, also called states, and the *actions* a it can perform, are classical, meaning that no quantum magic can be cast upon the environment. However, this does not stop Elliot from *thinking quantumly*, storing information about its actions outcomes in a quantum computer. In fact, as first proposed in [1], the quantum agent can pair each percept with a quantum state $|a\rangle_s = \sum_i \alpha_i^s |a_i\rangle_s$, where th *observable* basis elements correspond to the actions it can take, and the amplitudes will encode its *policy*. Thus, upon perceiving s, Elliot will measure $|a\rangle_s$ observing $|a_i\rangle_s$, and hence taking the corresponding action, with probability $|\alpha_i^s|^2$. In this sense, the stochastic nature of quantum measurements outcomes is perfect to guarantee a sort of flexibility between *exploration* (trying previously unexplored strategies) and *exploitation* (taking the most advantage of previous experiences). Indeed, unless only one amplitude is non vanishing, the measurement outcome may not be the most probable one, allowing Elliot to try new strategies. What is left is to cleverly tune the amplitudes α_i^s in order to follow the best, i.e. most rewarded, trajectory in the environment. The point now is: what is the best strategy to enhance the amplitude associated to the right action?

Grover-like amplitude enhancing

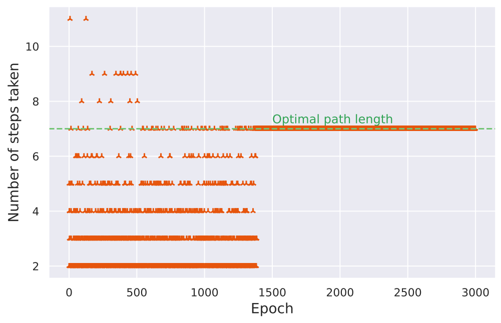
Among the most famous quantum algorithms, Grover's fast search routine is just what Elliot needs. Indeed, Grover's algorithm works by amplifying the amplitude associated to the quantum state we have to find, this way making it more likely to retrieve it upon measurement. Thus, the learning strategy of Elliot is that of resorting to classical Q-Learning to tell the difference between good and bad actions, and then performing a step of Grover's amplitude amplification to reinforce the good ones.

How Using `Qiskit` standard libraries we set up Elliot in such a way that it can accept an OpenAI gym [4] *discrete* environment, that is one with a finite dimension of both the percept and action spaces, and initialize proper size quantum circuits for each state. The latter qubits registers are initialized in a completely superposed state, providing totally random first action choices. Exploring the environment step by step, Elliot uses Bellman equation [1] to update the *Quality Values* associated to each state-action pair, and upon exceeding a certain threshold performs one Grover step on the state-action circuit. As information is lost upon measurement, the number of Grover steps taken on each circuit is saved in a classical memory, so that Elliot can dispatch the needed circuits any time it wants.

Results We tested Elliot against the Frozen Lake environment [4]. Frozen lake is a 4x4 board game representing a frozen lake in which some tiles are holes (if you step there you lose) and some are frozen (safe to walk on). The goal is to go from the top left of the grid (start tile) to the bottom right (goal tile), without walking on any holes. The possible actions in every state of the board are four: go left, down, right and up, so the action Hilbert space must be 4-dimensional and the usual computational basis states $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ are chosen as the action markers. Figure 1 shows that Elliot is able to learn to win consistently, see Fig. 1(a), and to do so in optimal manner, Fig. 1(b).



(a) Elliot learns to play Frozen Lake



(b) Lengths of Elliot's trajectories

Figure 1: **(a)** Results of Elliot's games during training. Sporadic wins guide Elliot to understanding the best set of moves to reach its target.**(b)** Lengths of Elliot's wanderings over the frozen lake. Early wins do not necessarily correspond to optimal strategies, as opposed to late ones.

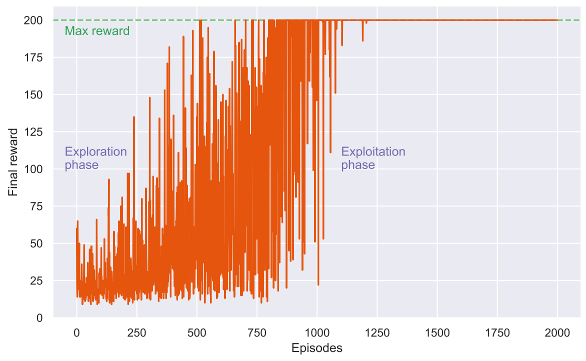
Quantum Approximate Q-learning with Parametric Quantum Circuits

Why While very efficient for simple environments as FrozenLake, Q-Learning is unviable for complex ones with large percept and action spaces. Learning all of the quality values is impossible and one needs to resort to *approximate* strategies. DeepMind [5] showed that Deep Neural Networks (DNN) are the answer.

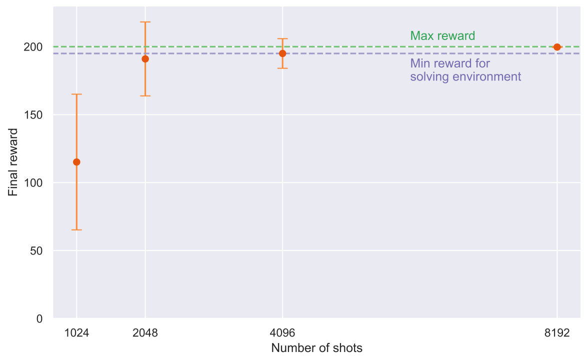
Where (is the quantum)? DNNs can be replaced by parametric quantum circuits, also known as *Quantum Neural Networks* (QNN). In our project, we implement a slightly simplified –yet efficient– version of the algorithm proposed in ref. [2], and train Elliot against the CartPole environment provided by OpenAI gym. In CartPole, Elliot has to balance a pole attached to a cart moving along a track. The pendulum starts upright, and, based on the cart velocity, the angle of the pole, its instantaneous velocity given by gravity, and the last movement of the cart (*state* of the system), the goal is to prevent it from falling over by pushing the cart "left" or "right" (the two possible *actions*).

How The QNN consists of repeated layers of parametrized gates (R_y and R_z rotations) and entangling operations (CZ), while the classical state of the environment is loaded using angle encoding, that is using R_x rotations on a quantum register initialized in $|0\rangle^{\otimes 4}$. The two possible actions "left" and "right" are associated to the expectation value of operators $O_l = \langle Z_0 Z_1 \rangle$ and $O_r = \langle Z_2 Z_3 \rangle$, measured at the circuit output. In addition, the quantum circuit is preceded and followed by two classical pre- and post-processing layers, each one coming with its own trainable parameters.

Results The whole hybrid quantum-classical architecture has been implemented in Qiskit using the brand new `Qiskit Machine Learning` module. In particular, thanks to the `TorchConnector`, it was possible to create an end-to-end quantum-classical `PyTorch` model, and use seamless and fast gradient evaluations for training. The training algorithm used is standard Deep-Q Learning, with the substitution of the classical neural network with the quantum one. The implementation proved successful and in Figure 2 we summarize the results obtained: in Fig. 2(b) we show the score obtained by Elliot during training which increases until reaching the maximum value of 200 imposed by the CartPole environment; in Fig. 2(a) we test the trained quantum agent in the presence of noise coming from stochastic measurement outcomes, again with remarkably good results!



(a) Training the Parametrized Quantum Elliot



(b) Trained Elliot performances using `qasm_simulator`

Figure 2: **(a)** Optimization process of the parametrized quantum circuit approximating the optimal Q-values $Q^*(s,a)$. After an initial *exploration* strategy producing an erratic behaviour, Elliot starts *exploiting* the past experiences, learning how to balance the pole, thus solving the environment. **(b)** Performances of the optimized quantum agent in the presence of noise coming from stochastic measurement outcomes implemented using `qasm_simulator`.

2.2 A Quantum Agent in a Quantum World

The second phase of our work aimed at training Elliot in a Quantum environment. We have chosen a Quantum TicTacToe environment that we wrote ourselves in Qiskit. The classic 3×3 board is now a set of 9 qubits, initialized in the $|+ + \cdots +\rangle$ state, and each player is associated to a computational basis state of the qubits, $|0\rangle$ or $|1\rangle$. At each turn, a player can act on the board, that is a quantum circuit, with a quantum gate from the set $\{X, H, CNOT\}$ on whichever qubit. After a fixed number of turns, the board is measured in the computational basis, collapsing the state: if there are three zeros or three ones aligned a player wins, if there are none or more than one, then it's a draw. Elliot has to learn how to maximize his probability of winning, by transforming the state of the board qubits in such a way that it is more likely to collapse in its "symbol" (0 or 1), after the measurement. We chose to use the Grover-like algorithm for the enhancement of amplitudes of the quantum state of actions associated with every state of the board and let Elliot play against itself to train. This part of the project is still in its early stages, and we plan to work on it in the future.

3 Impact and outlook

Our project is, as far as we know, the first Quantum Reinforcement Learning agent written entirely using Qiskit. It represents a first step towards a complete Quantum Reinforcement Learning module in Qiskit, and precisely because it's a novel feature, the possibilities of different outlooks are endless: from generalizing to other quantum and classical environments to implementing variations on the learning algorithm itself. Specifically on the learning algorithm, in the future one may want to implement a variation on Deep Q-learning (PQC) algorithm, in order to optimize the variational ansatz, the learning time and the performances in general. Also, in the future we'd like to play, for example, Quantum TicTacToe against Elliot, because we're genuinely curious about the possibility of playing a quantum game against a quantum agent as classical human beings.
Finally we remark that even if in the second phase of our project we interface Elliot, that decides what to do following the output of a quantum circuit, with a quantum environment, constructed with a quantum circuit too, some work in this direction is on the to-do list. In particular, we would like to find a way in which two quantum circuits mentioned above interact directly as subroutines of a single quantum circuit, so that Elliot and the environment interact fully quantumly and we, as classical humans, don't enter in this truly quantum dialogue.

References

[1] Daoyi Dong et al. "Quantum Reinforcement Learning". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 38.5 (2008), pp. 1207–1220. DOI: `10.1109/TSMCB.2008.925743`.

[2] A. Skolik, Sofiène Jerbi, and V. Dunjko. "Quantum agents in the Gym: a variational quantum algorithm for deep Q-learning". In: 2021. URL: `https://arxiv.org/pdf/2103.15084`.

[3] V. Saggio et al. "Experimental quantum speed-up in reinforcement learning agents". In: *Nature* 591 (2021), pp. 229–233. DOI: `10.1038/s41586-021-03242-7`.

[4] Greg Brockman et al. “Openai gym”. In: *arXiv preprint arXiv:1606.01540* (2016).

[5] Volodymyr Mnih et al. *Playing Atari with Deep Reinforcement Learning.* 2013. arXiv: `1312.5602 [cs.LG]`.