Project Report

Introduction

In recent years, an increasing number of works have been published with the aim of combining the disciplines of quantum information processing and Machine Learning. Theories to merge these two fields have been put forward regularly since quantum information became an independent discipline.
And as we know almost all the games nowadays are based on machine learning, the question that can be asked is why not use this new technique quantum machine learning to develop the games and also in order to be usable in quantum computers because of their efficiency compared to classical computers.

RL Part

We don't want to hard-code knowledge about the environment and the best action to take in each case; that would be too much work and would become useless at the slightest change in the environment. We need methods that allow the agent to learn by itself to find the best moves to achieve the best cumulative reward.
Reinforcement learning offers a solution here that differs from supervised and unsupervised learning methods. There are no predefined labels as in supervised learning. However, the reward system also leaves us not completely blind as in unsupervised learning. Rewards can be positive, negative or neutral. While the agent observes the rewards and relates them to the action performed, it can learn how to perform actions better.

Quantum Part

Hybrid Quantum Classical Neural Network

Aim of the Project:

- Show feasibility of using Deep Q-Reinforcement Learning powered by Quantum Circuits using a Game setting
- Provide modular Framework for future Quantum ML Tests in an environment with fixed rules for the Quantum Community

RL Part

Deep Q- Networks is a popular method for computer games. The paper https://www.nature.com/articles/nature14236 was an important step in reinforcement learning, because it demonstrated that it is possible to use non-linear approximations with this method. This evidence led to a great deal of interest in the field of Q-learning in particular and reinforcement learning in general Deep-Q-Learning. Deep reinforcement learning reached a milestone in 2015 when AlphaGo, a computer program trained with deep Reinforcement Learning became the first computer program to beat a human professional Go player on a full-sized 19×19 board.
For the game of Othello two Deep-Q-Learning methods are applicable:

- Application of Q-Learning to Grid-World Environments (Tabular Q-Learning),
- Q-learning in conjunction with neural networks, a combination referred to as Deep-Q-Network (DQN).

The Tabular Q-Learning approach is not practical for large state spaces or a large number of actions. In the case of Othello, there are too many states for this method, which must be recorded and for which approximate values must be calculated.

Deep-Q-Network addresses these difficulties and is therefore used in this approach?

Q-Learning Theory:

Is a technique that determines, based on a state-action-value-(reward) function, the value of performing an action or the quality of a state-action combination. The so-called Q-function takes as input the current state and a possible action and calculates the potential reward for this action and all following actions. More precisely, this is called *Temporal Difference Learning*, which adjusts the estimate of the future total reward per step.

The Quantum Part with RL?

Implementation

Key Classes and Data Structure
CNN

We have decided to build on the shoulders of giants and therefore looked for a neural network approach that we can suitably extend with a quantum layer in order to create a hybrid Quantum Classical Neural Network. After a thorough research we decided to use the approach of [1], because it used exactly the structure that is suitable for Othello and could be used in Deep Q Reinforcement Learning.
The architectural structure is shown in figure X.

We wanted to extend this neural network with a quantum layer in order to use the computational power of the quantum computers to speed up the computation on the one hand and to investigate the results that this hybrid quantum network should deliver on the other hand.

Parameters:
- Input layer = current state + action + reward + next state ?
- Convolutional Layer = ?
- ? Hidden layer = same dimension as input layer?
- Fulled Connected Layer = -> Output 16 outputs?
- Quantum Layer? = VQC -> Input 16 Outputs from Fulled Connected Layer mapped to 16 Qubits?
- Output layer = optimisation/update of parameters, (either a Q-value per action or a final Q-value?) -> returning value to the game
- Learning algorithm?

Controller: controls the interaction between the agent and the environment
Agent: performs actions and has a "Q table" that represents his memory/experiences
Environment: provides the arena for the agent and determines what rewards an agent will receive for a given action

The CNN needs a certain amount of data to learn, so the agent will perform random actions at the beginning. Once the array is filled, we start learning.

Results
The approach showed that is is possible to start on classical machine learning techniques and combine them with Qiskit. In our approach, we mainly faced the problem of dimension mismatching. The backpropagation of the neural network did not cope with the answers of the quantum layer. Here, a different vector space was returned than the classical network would expect. This problem persists and could not be solved by the end of the project.

Further Work
The general applicability and flexibility of reinforcement learning comes at a price. The biggest obstacle to reinforcement learning in this game setting is that rewards for an action may be delayed considerably. In Othello, as well as in chess, a single strong move in the middle of the game can be game-changing. In learning, we need to recognize such things, which can be difficult when we focus purely on the game play and actions. In future Work the sum of future discounted rewards should be taken more into consideration.
Also, the debugging in novel hybrid approaches, as we describe here, has to be done thoroughly and is therefore time-consuming. At this point, it is definitivly worthwhile to go troubleshooting again to bring the Quantum Opponent to life.

Bibliography

[1] https://github.com/colinmsnow/othelloAI