# Quantum Neural Network for Continuous Reinforcement Learning

Erik Sorensen

Data Science Honors Project Proposal

## Introduction

Quantum computing and machine learning are two exciting fields. Quantum computing brings a new perspective on computers by using properties of quantum mechanics to do computations, which can be much faster than the classical computers we know today. Not only are they faster, but some tasks that we thought were impossible with the computers we know today, can be accomplished using quantum computers. (CITATION) Machine learning learns complex problems from large amounts of data that are too difficult for humans to manually solve. It has been used in facial recognition (CITATION) and identifying weeds in precision agriculture (CITATION). A more recent sub-field of machine learning is *reinforcement learning*. Reinforcement learning uses a positive or negative reinforcement signal from an environment, known as the *reward*, to provide feedback to the learning system so that it may learn and improve to maximize this reward. Some areas that use reinforcement learning are in economics and artificial intelligence in games. (CITATION) Reinforcement learning is an attractive tool to use. While other machine learning techniques often times require a well tuned, large dataset to learn, reinforcement learning only requires a reward from an environment and time to train. Even further, using a quantum computer to do reinforcement learning has the potential to solve problems that would be too computationally complex for classical computers to solve. This increases the scope of problems we can solve exponentially.

## Reinforcement learning

Reinforcement learning may be applied to many fields. As an example, I will use a vacuuming robot to show a specific use case of how reinforcement learning can be applied in the world. Lets imagine if this robot used reinforcement learning techniques. The robots main goal is to vacuum an *environment*, while avoiding furniture and other obstacles.

With reinforcement learning, the robot can self-teach itself by exploring the room. The only feedback the robot gets is *observations* about its surroundings via sensors, and a *reward* signal that indicates how well the robot is doing at its job. Every action the robot takes it will receive either a positive reward signal for good actions or a negative reward signal for bad actions. The robots goal then is to maximize this positive reward signal so that it knows it is doing a good job. The part of the robot that decides the next *action* it should take, whether it be going left or right or going back to charge, we will call the *agent*.

How then can the robot learn to efficiently do this task with these tools it has at its disposal? This problem is called the reinforcement problem [1], which gives the instructions for most reinforcement learning algorithms. It is stated here:

> For an agent in a certain state at a certain time, what action should be taken that will maximize the overall reward, now and in the future?

To summarize, the goal of reinforcement learning is to find a *policy* which maximizes the *reward*. A *policy* (pi) is a probability distribution that describes the probability the agent will take a certain action given it is in a certain state. The *optimal policy* is a policy that on average receives the greatest total reward. This is useful, especially in terms of the vacuuming robot who does not know anything about the environment it has been placed in. By defining the rewards the robot will search toward, it will learn itself how to best maximize this reward in the environment. Since we defined the robots reward as avoiding furniture in the environment, it

will learn the best paths it can take to avoid these obstacles. All methods of reinforcement learning have this same goal, which is to find this optimal policy to maximize reward.

1. Example to understand it
2. Value Function and Action-Value Function

## References

[1]: R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction* (MIT Press, 1998).