What is evolution?

Evolution is the small subtle changes in genes and DNA, that has, over countless of generations, evolved life on earth from simple, uniform and singular cells to a massively diverse and complex tree of life. More specifically, it is the small, random changes in behavior and composition, that reward life with a sudden prosperity, allowing the change to propagate through generations. It learns from experience and quantity.

Reinforcement learning is a method of Machine learning that takes this idea in order to solve complex tasks. It is an algorithm that has no knowledge about the environment it is in (ModelFree), and is expected to be able to adapt and solve a problem specified by a “Reward”. This reward can be everything, from learning by burning to learning by pampering. The algorithm will seek out as much pamper as it can get, while avoiding the fires of death.

However, as it has no knowledge about what either death or pamper is, it has to explore, and die. Each time it dies, a small change in the algorithm is made, that emphasizes the dangers of death, until it starts to avoid it. With the advantage of modern computer simulations, we are able to train algorithms in simulated environments, where death is an acceptable outcome. The Alfago chess machine has trained, and lost millions of games, before it actually learned how to play the game, before it could be deployed in real life. This method of learning is called deep learning, and relies on a unknown neural network attempting to learn the envirovment.

But real life is unpredictable. Simulations cannot depict real life accurately, and thousands of deaths or losses for an algorithm that is deployed is not acceptable. A robot that is sent to mars, can only do one mistake before it fails. But if it is going to be autonomous, then it has to be able to learn. So how can we design an algorithm that has the learning properties of neural networks, but not the unpredictability of them.

The network can be replaced by an other optimal function approcimator