Thank you Alexa. Now I will introduce the Q-Learning -> DQN path.

We all have our own code of conduct when we do things. For example, our parents used to say, "Don't watch TV until you finish your homework." So in the state of doing homework, good behavior is to continue to write homework, until it is finished, then we can be rewarded; bad behavior is going to watch TV before finishing, if it was found by parents, the consequence is very serious. When we do this kind of things more, it becomes our indelible memory. This can be represented by a Q-table in the Q-Learning methods. This Q-table is like the code of conduct we mentioned. So, Q-Learning is a value-based RL method. “Value-based” means that it outputs the value of all the actions, and we choose the actions which can get the highest value. In this Q-table, Q is the action-utility function, which is used to evaluate the goodness of taking an action in a specific state. It is the memory of the agent.

At each step, the table is updated according to this formula until it converges. Here let me explain this formula. First is the actual Q, Rt+1 denotes the reward acquired when taking the next step, max Q(St+1, a) denotes the maximum Q in the next state with different actions. Evaluated Q is the current value. Then the loss is the difference between the actual Q and the evaluated Q. γ is the decay factor (like a pair of glasses, to foresee the following steps), and α is the learning rate. But such a choice may cause Q to fall into local optimum. The improved strategy is ε-greedy method: each state is explored with the probability of ε.

However actually, in general Q-learning, when the state and action space become more, Q-table will have too much space and too many states and is difficult to use. We just said that Q is the action-utility function. Therefore, Q-table updating can be transformed into a function fitting problem here. We can integrate deep neural network with Q-Learning. This is the DQN algorithm.

This is the structure of DQN. You can have a look.

So to convert a Q-Learning algorithm to a DQN, we will encounter these problems.

Problem 1: deep learning is the supervised learning that requires training set. The solution is: Labels are constructed using rewards through Q-learning.

We return to the picture, you can see that there is an reward arrow pointing to the neural network.

Recall that in Q-learning above, we used the formula to represent the actual Q, here we can use it as the label Q value.

Problem 2: In supervised learning, data is independent of each other. For machine learning models based on the maximum likelihood method, we have a very important assumption: the training samples are independent and come from the same distribution. Once this assumption is not valid, the effectiveness of the model will be greatly reduced.

Solution: Experience replay, for each updating, a portion of the data is extracted from Memory for the update. The records in the experience pool of DQN is used to learn from previous experiences. The random training from previous experience in the learning process will make the neural network more efficient. It solves the problem of correlation and non-static distribution.

Problem 3: Every time the neural network is updated, the target will also be updated, which will easily lead to parameter non-convergence. Due to the instability of data itself, some fluctuations may be generated in each iteration. If we follow the above calculation formula, these fluctuations will be immediately reflected in the calculation of the next iteration, so it is difficult for us to get a stable model. To mitigate the impact of related problems, we try to decouple the two parts as much as possible.

Solution: Q-targets is introduced. The Eval Q network is updated every iteration, whereas the Target Q network is updated every once in a while. It is actually a mechanism to disrupt the correlation, which reduces the correlation between the current Q value and the target Q value to a certain extent and improves the algorithm stability.

That is the part of DQN. Then we will proceed to another path.