**Deep Deterministic Policy Gradient (DDPG)**

DDPG is a model-free policy based learning algorithm in which the agent will learn directly from the un-processed observation spaces without knowing the domain dynamic information. That means the same algorithm can be applied across domains which is a huge step forward comparing with the traditional planning algorithm.

In contrast with DQN that learn indirectly through Q-values tables, DDPG learns directly from the observation spaces through policy gradient method which estimates the weights of an optimal policy through gradient ascent which is similar to gradient descent used in neural network. Also, policy based method is better suited in solving continuous action space environment.

DDPG also employs Actor-Critic model in which the Critic model learns the value function like DQN and uses it to determine how the Actor’s policy based model should change. The Actor brings the advantage of learning in continuous actions space without the need for extra layer of optimization procedures required in a value based function while the Critic supplies the Actor with knowledge of the performance.

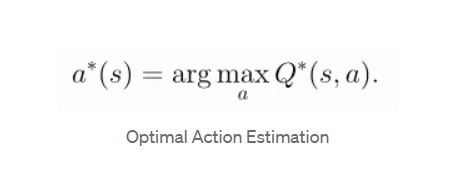
To mitigate the challenge of unstable learning, a number of techniques are applied like Gradient Clipping, Soft Target Update through twin local / target network and Replay Buffer. The most important one is Replay Buffer where it allows the DDPG agent to learn offline by gathering experiences collected from environment agents and sampling experiences from large Replay Memory Buffer across a set of unrelated experiences. This enables a very effective and quicker training process. Also, Batch Normalization plays an important role to ensure training can happen in mini batch and is GPU hardware optimization friendly.

Deep Deterministic Policy Gradient (DDPG) is a reinforcement learning technique that combines both Q-learning and Policy gradients. DDPG being an **actor-critic technique** consists of two models: Actor and Critic. The actor is a policy network that takes the state as input and outputs the exact action (continuous), instead of a probability distribution over actions. The critic is a Q-value network that takes in state and action as input and outputs the Q-value. **DDPG is an “off”-policy method**. DDPG is used in the continuous action setting and the “deterministic” in DDPG refers to the fact that the actor computes the action directly instead of a probability distribution over actions.  
**DDPG is used in a continuous action setting and is an improvement over the vanilla actor-critic.**

Deep Deterministic Policy Gradient or commonly known as DDPG is basically an off-policy method that learns a Q-function and a policy to iterate over actions. It employs the use of off-policy data and the Bellman equation to learn the Q function which is in turn used to derive and learn the policy.

**Learning Process:**

The learning process is very closely related to Q-learning where if you know the optimal-action-value function **Q\*(s, a)**, the best and optimal action to taken in that state can be found out using a\*(s) which is :

****

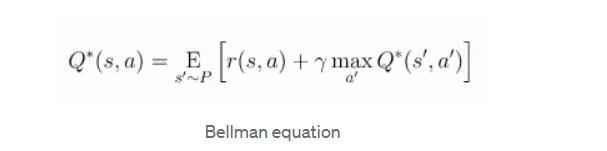
# ontinuous Action Space Derivation:

DDPG was developed specifically for dealing with environments with continuous action spaces and in essence that is to estimate the max over actions in **max Q\*(s, a).**

1. In the case of Discrete action spaces, Q-values for each action can be estimated separately and hence compared easily for the best max value.
2. In the case of Continuous action spaces, computation and individual comparison for each Q-value becomes very exhaustive leading to non-stationary target values and unstable learning. Not to mention, the process for such is quite exhaustive and computationally expensive.

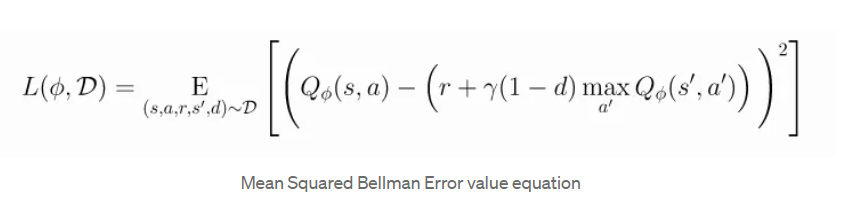
Q-learning based algorithms, specifically DDPG employs the use of the following to deal with a continuous action space:

* Make use of the **Bellman equation** to obtain the optimal action for a given state using its state-action/Q-value.

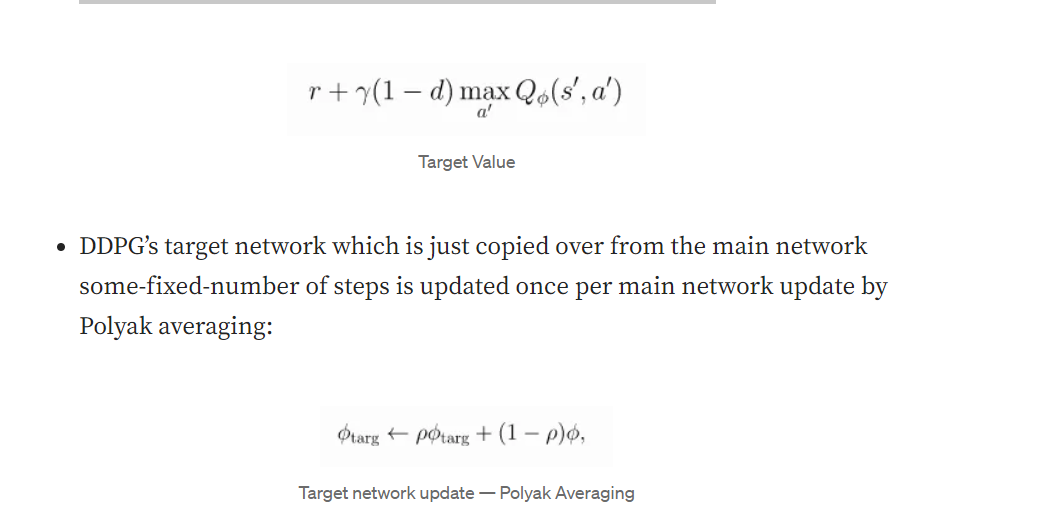
****

*In the equation*s’~P*refers to the next state s’ being obtained from the environment from a probability distribution of*P(.|s, a).

* DDPG employs the use of **mean-squared Bellman error (MSBE)** function which estimates how close Q\* comes close to satisfying the Bellman equation as shown in the equation:

****

* Making use of **Experience Replay Buffer** which is a set of previous experiences which helps in providing Q-learning based approximators a stable learning behavior
* DDPG also deploys the use of a T**arget network** to deal with non-stationary target values and make the learning more stable. Following describes what a **Target**is because when we minimize the MSBE loss, we are trying to make the Q-function be more like this target.

****

Thus DDPG deals with this humongous continuous action space challenge and expensive computation by using a **target policy network** to compute an action that approximately maximizes **Q\*(Target).**

# ****Key Takeaways :****

* Q-learning in DDPG is performed by minimizing the following MSBE loss with stochastic gradient descent.
* DDPG is essentially and Off-policy learning method
* It is basically Q-learning for continuous action spaces.
* It uses Target network with Experience replay for stable and efficient learning.