#### Part 1

使用DDQN（Double Deep Q-Network）算法搭配OpenAI訓練超級瑪利歐具有以下優勢：

解決過度估計（overestimation）問題：在Q-learning中，Q-values的估計容易受到過度估計的影響，導致策略不穩定。DDQN通過使用兩個獨立的神經網絡，一個用於選擇最佳動作，另一個用於估計該動作的Q-value，來降低過度估計的風險。

提升學習效率：DDQN通過使用目標網絡（target network）進行Q-value估計，可以降低時間差分目標（TD target）的波動性，提升學習的穩定性和效率。

增強探索能力：DDQN在選擇動作時，利用目標網絡進行動作的選擇，從而減少了在選擇最佳動作時的過度依賴，促進了探索和挖掘潛在最佳動作的能力。

支持高維狀態空間：使用神經網絡作為函數近似器，DDQN能夠處理較大的狀態空間，對於超級瑪利歐這樣的遊戲，可以更好地處理複雜的狀態表示。

可延伸性：DDQN算法是強化學習中的一種基礎算法，可以與其他技術和改進結合使用，例如優先經驗回放（prioritized experience replay）、雙重Q學習（double Q-learning）、Dueling架構等，進一步提升學習的性能和穩定性。

#### Part 2

我們參考了PyTorch Tutorials > Train a Mario-playing RL Agent

首先，我們使用gym-super-mario-bros==7.4.0的library

接著，我們將所需要的library匯入，如torch、gym、JoypadSpace from nes\_py.wrapper which is specifically designed for interacting with NES (Nintendo Entertainment System) games using OpenAI Gym environments.

接著，我們將分為以下幾點作為我們介紹的重點:

* **Environment**
* **Agent**
* **Result**

##### Environment

首先，我們使用GrayScaleObservation將原先RGB dimension轉成單通道，接著使用ResizeObservation將image轉成[1,84,84]的方形image。

SkipFrame 是一個自定義包裝器，它繼承自 gym.Wrapper 並實現了 step() 函數。 因為連續的幀變化不大，所以我們可以跳過 n 個中間幀而不會丟失太多信息。 第n幀聚合了在每個跳過的幀上累積的獎勵。在此code中，我們選用skip=4。

FrameStack 是一個包裝器，它允許我們將環境的連續幀壓縮成一個單一的觀察點，以提供給我們的學習模型。

##### Agent

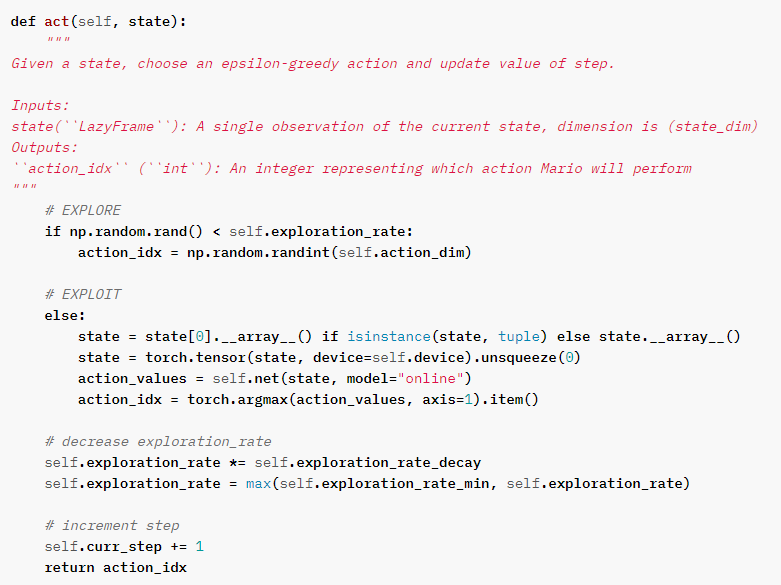
我們創造了一個class, named Mario, 其中包含了3個主要的部份

1. Act: according to the optimal action policy based on the current state
2. memory: experiences. Experience = (current state, current action, reward, next state)
3. Learn: find a better action policy over time

## Act

For any given state, an agent can choose to do the most optimal action (exploit) or a random action (explore).

Mario randomly explores with a chance of self.exploration\_rate; when he chooses to exploit, he relies on MarioNet (implemented in Learn section) to provide the most optimal action.



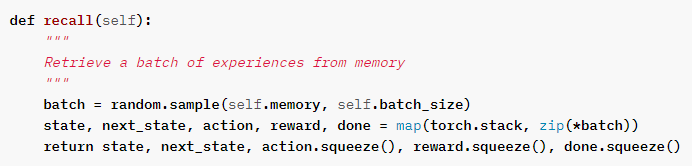
## Memory

Cache and Recall functions serve as Mario’s “memory” process.

cache(): Each time Mario performs an action, he stores the experience to his memory. His experience includes the current state, action performed, reward from the action, the next state, and whether the game is done.

recall(): Mario randomly samples a batch of experiences from his memory, and uses that to learn the game.





## Learn

Mario uses the DDQN algorithm(Double Deep Q-Network)，分別為Qonline跟Qtarget,

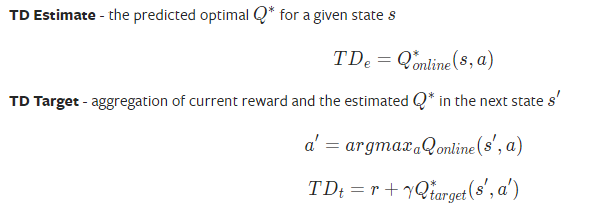
In our implementation, we share feature generator features across Qonline跟Qtarget.

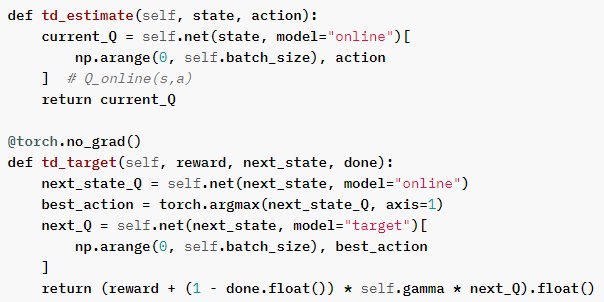
*"""mini CNN structure*

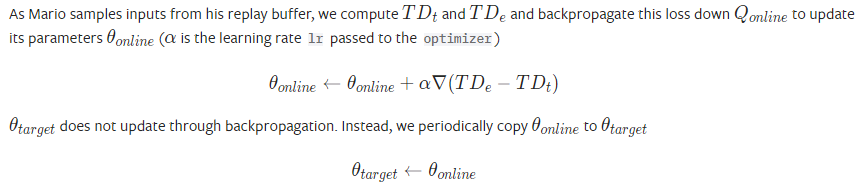
*input -> (conv2d + relu) x 3 -> flatten -> (dense + relu) x 2 -> output*

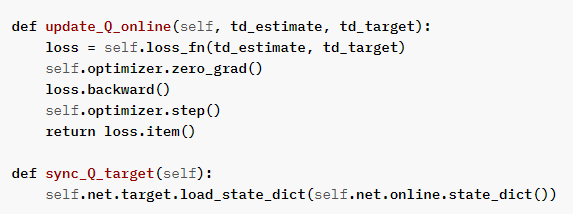
*"""*

使用上式中的mini CNN架構，再使用兩種不同的value function, 一種是TD Estimation, 另一種是TD Target。









總結來說，使用DDQN RL搭配OpenAI訓練超級瑪利歐可以獲得更穩定、高效的學習過程，並具有更好的探索能力和應對複雜環境的能力。這些優勢使得DDQN成為訓練超級瑪利歐等強化學習任務的一個強大工具。