

---

**Algorithm: Deep Q Learning algorithm**

---

Memory  $\leftarrow$  Initialize memory container with zeros

step  $\leftarrow$  0

Network  $\leftarrow$  Initialize weights with zeros

$\gamma \leftarrow$  Discount factor

$\epsilon \leftarrow$   $\epsilon$ -greedy algorithm factor

**function** GET RESULT(S as state)

**for**  $1 \leq i \leq 108$  **do**  $\triangleright$  There are 108 cards in a UNO deck

        Mask<sub>*i*</sub>  $\leftarrow$  [Card<sub>*i*</sub>  $\in$  S.Playable]  $\triangleright$  [ ] represents Iverson bracket

    Mask<sub>0</sub>  $\leftarrow$  1  $\triangleright$  Player may always take a card from the deck

    Result  $\leftarrow$  Sigmoid(Network.Run(S)) + 1

**return** (Mask  $\odot$  Result) - 1  $\triangleright$   $\odot$  represents Hadamard product

**function** GET ACTION(S as state)

**if** Random.Uniform(0, 1)  $> \epsilon$  **then**

**return** Dummy Algorithm.Get Action(S)

**else**

**return** Argmax(Get Result(S))

**function** OBSERVE(S as state, A as action, R as reward, S' as state)

    Memory[step mod N]  $\leftarrow$  S, A, R, S'  $\triangleright$  N is an arbitrary number

    Values  $\leftarrow$  Network.Run(S)[A] + R +  $\gamma \cdot \max_{A'} \text{Get Result}(S')[A']$

    Network.Run(S)[A]  $\leftarrow$  Perform gradient descent on Values

    step  $\leftarrow$  step + 1

---