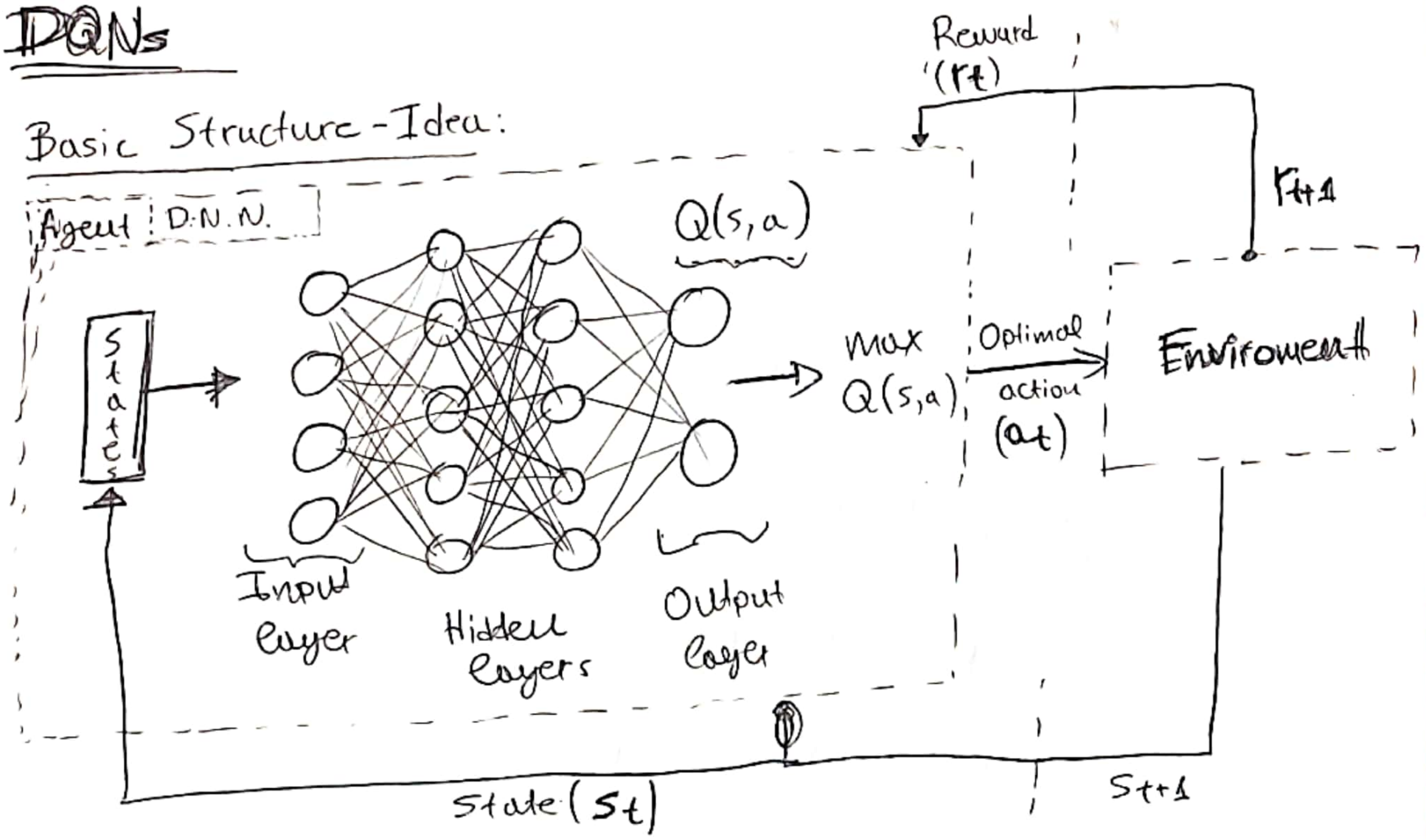
# DQNs

## Basic Structure-Idea:



Q-learning function: $Q(s,a) = r + \gamma \max_{a'} Q(s',a')$

Προβλημα! Πρέπει να ξέρω $Q(s',a')$, $\forall (s',a') \rightarrow$ inefficiency.

Solution: Deep Q-Network.

Updating Q-learning function - Bellman Equation:

$$Q(s,a) \leftarrow Q(s,a) + \underbrace{\ell}_{\text{learning rate}} \Big[ \underbrace{\overset{\text{immediate reward}}{r} + \underbrace{\gamma}_{\text{Discount factor (How much we value the future")}} \max_{a'} Q(s',a')}_{\text{target } Q(s,a) \text{ (what it should be}} - \underbrace{Q(s,a)}_{\text{current } Q(s,a)} \Big]$$

Καταλαβαίνουμε οτι νευρωνικό μας πρέπει να ελαχιστοποιήσει την

$$Q(s,a) - \left[ r + \gamma \max_{a'} Q(s',a') \right] \leftarrow \underline{\text{loss function}}$$

<u>Target network</u>

- Τα $Q(s,a)$, $Q(s',a')$ δεν πρέπει να υπολογίζονται με τον ίδιο τρόπο
  - Δεν έχει νόημα!

Φτιάχνουμε δυο όμοια νευρωνικά: $\begin{cases} \text{policy-net} \leftarrow \{ \text{απόφαση, training} \} \\ \text{target-net} \leftarrow \{ Q(s',a') \} \end{cases}$

Τα βάρη του policy-net ανανεώνονται συνεχώς, ενώ του target_net περιοδικά.

⚠ Notation: Βάρη policy: $\theta$, Βάρη target: $\theta'$.

<u>DQN Bellman Equation</u>: $Q(s,a) \leftarrow Q(s,a) + \ell \left[ \underbrace{r + \gamma \max_{a'} Q(s',a';\theta') - Q(s,a;\theta)}_{\text{loss function}} \right]$

*

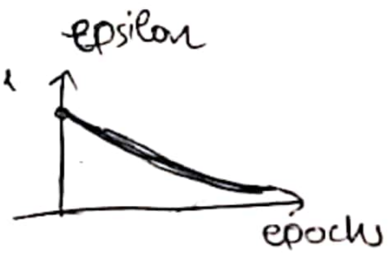
Μεχρι τώρα: $\begin{cases} \text{Αρχιτεκτονική ✓} \\ \text{Loss function για υπολογισμό των βαρών} \end{cases}$

* Το target_net θα γίνεται smoothly, (soft update) ως εξής:

$$\theta' \leftarrow \tau\theta + (1-\tau)\theta', \text{ όπου } \tau \ll 1.$$

<u>Πως κάνουμε train:</u>

- <u>Selecting action</u>: έχουμε ένα epsilon



οποτε παιρνουμε τυχαίο αριθμό x στο $[0,1]$

Αν $x <$ epsilon τοτε: random-action ← Exploration

Αλλιώς: χρησιμοποιούμε το policy-net ← Exploitation

↳ This is called epsilon greedy policy.

- Calculating the reward

Depends on the enviroment,

General Rules
- small ~~[illegible]~~ and dense rewards > large and sparse ones
- give penalties to control learning.
- be creative, there is no single best solution

- Experience Replay

Αποθηκεύουμε στην μνήμη το (state, action, next state, reward) για κάθε action που κάνουμε. Έτσι μπορούμε να υπολογίσουμε το loss. σε batches των experiences και έτσι μαζέψει την

τελικά να ανανεώνουμε τα βάρη.