Reinforcement Learning Training 2025

Deep Q-Learning (DQN)

Update Equations

- Q-learning (tabular)

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \cdot \left[R_{t+1} + \gamma \cdot \max_{A_{t+1}} Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t) \right]$$

- Deep Q-learning

$$w_{t+1} = w_t + \alpha \,.\, \frac{1}{N} \sum_{\mathrm{i}=1}^{N} \left[r_{\mathrm{i}} + \gamma \max_{a_i'} \tilde{q}(s_i', a_i'; w_t^-) - \hat{q}(s_i, a_i; w_t) \right] . \nabla_{w_t} \hat{q}(s_i, a_i; w_t)$$

DQN

- No theoretical guarantee of convergence under non-linear function.

Components

- Policies
  - *Behavior policy*: ϵ-greedy policy to explore and generate samples.
  - *Target policy*: deterministic greedy policy (no exploration)
- Networks
  - Target network:
  - Primary network:
- Replay buffer

Improvements to DQN

Prioritized Replay

- Using Prioritized Experience Replay (PER)
  - samples experiences from the replay buffer based on their temporal-difference (TD) error.
  - Focusing on more important or surprising transitions rather than uniform sampling.

$$\delta_i = r_i + \left((1 - \text{done}_i) \,.\, \gamma \,.\, \max_{a'} \hat{q}(s_i', a'; w_t^-)\right) - \hat{q}(s_i, a_i; w_t)$$

Prioritized Replay

- This improves learning efficiency
  - Allowing the agent to learn more effectively from impactful experiences
    - Typically those with large TD errors
  - Leading to better and faster performance in many reinforcement learning tasks.

Double Deep Q-Learning

- Variation of the deep Q-learning algorithm
- Decomposing the `max` operation in the target value into separate action selection and action evaluation processes.
  - Reduce the overestimation of action values calculated in deep Q-learning.

Double Deep Q-Learning

- DQN

$$r + \gamma \,.\, \hat{q}\left(s', \arg\max_{a'} \hat{q}\left(s', a'; w_t^-\right); w_t^-\right)$$

- DDQN

$$r + \gamma \,.\, \hat{q}\left(s', \arg\max_{a'} \hat{q}\left(s', a'; w_t\right); w_t^-\right)$$

w_t^- is the target network, and w_t is the online network.

Dueling Double Deep Q Learning

- Separating the neural network's output into two parallel streams
  - One for estimating the overall state-value
  - Another one for estimating the **advantage** of each action within that state.

$$A_\pi(s,a) = Q_\pi(s,a) - V_\pi(s)$$

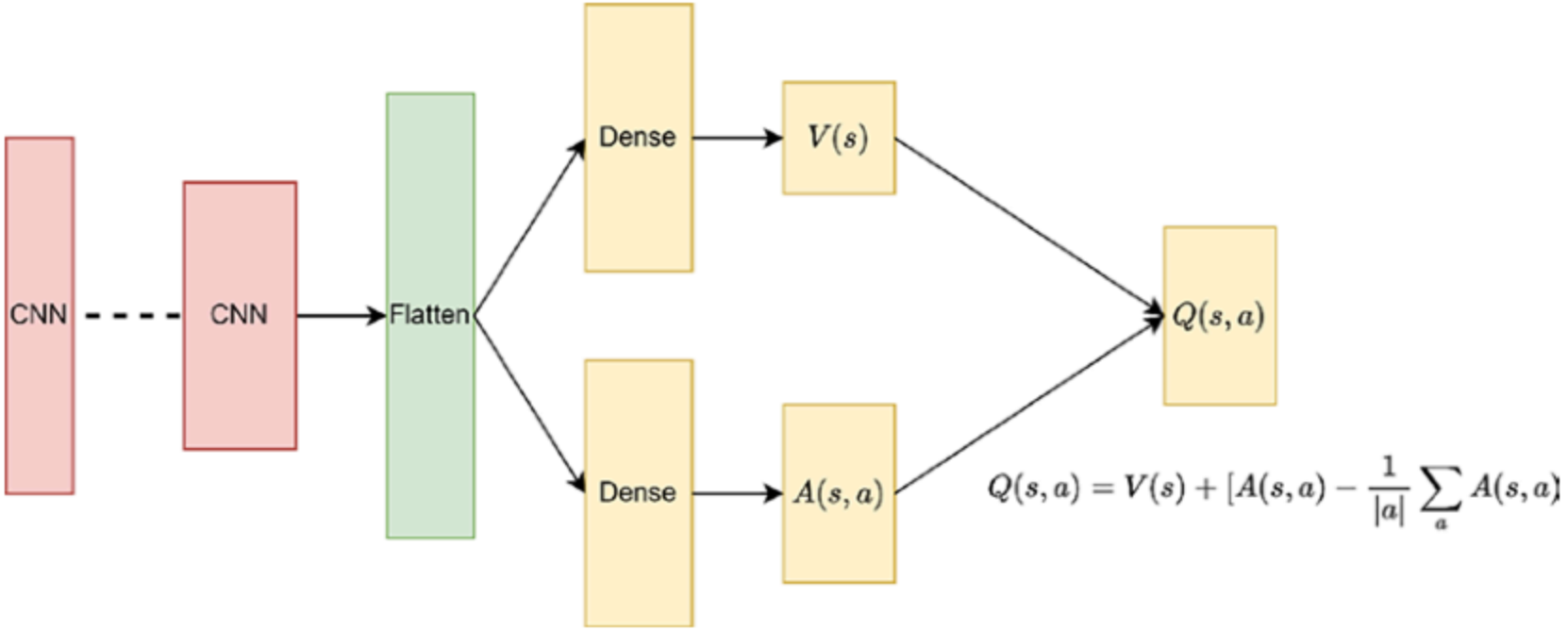

Figure 7-3. *Dueling network. The network has a common set of weights in the initial layers, denoted as w_1, and then it branches off to have one set of weights w_2, producing value V and another set of weights w_3, producing advantage A*

Dueling Double Deep Q Learning

- Allows the network to learn
  - Which states are valuable independently of the actions available
  - How much better one action is than another
- This leads to more stable and accurate Q-value estimations and improved agent performance.