Reinforcement Learning

Deep Q-Learning (DQN)

Or how to combine RL and DL

Akka Zemmari

Introduction

What is Deep Q-Learning?

What is Deep Q-Learning?

- Combination of Reinforcement Learning and Deep Learning.

What is Deep Q-Learning?

- Combination of Reinforcement Learning and Deep Learning.
- Extension of Q-Learning:

What is Deep Q-Learning?

- Combination of Reinforcement Learning and Deep Learning.
- Extension of Q-Learning:
    - Traditional Q-learning uses a tabular representation.

What is Deep Q-Learning?

- Combination of Reinforcement Learning and Deep Learning.
- Extension of Q-Learning:
  - Traditional Q-learning uses a tabular representation.
  - Deep Q-Learning approximates the Q-function using a Neural Network.

What is Deep Q-Learning?

- Combination of Reinforcement Learning and Deep Learning.
- Extension of Q-Learning:
  - Traditional Q-learning uses a tabular representation.
  - Deep Q-Learning approximates the Q-function using a Neural Network.
- Key goal: Solve high-dimensional state-action spaces.

What is Deep Q-Learning?

- Combination of Reinforcement Learning and Deep Learning.
- Extension of Q-Learning:
  - Traditional Q-learning uses a tabular representation.
  - Deep Q-Learning approximates the Q-function using a Neural Network.
- Key goal: Solve high-dimensional state-action spaces.

  To get an idea of the size of the state space:

  - Backgammon: 1020 states

  - Go: 10170 states

  - Robotics: continuous state space, real-life

Mathematical Formulation

Q-Learning Recap

- Q-function: $Q(s, a)$ estimates the expected cumulative reward for taking action a in state s.

Q-Learning Recap

- Q-function: $Q(s, a)$ estimates the expected cumulative reward for taking action a in state s.
- Update rule:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left(r_t + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right)$$

Q-Learning Recap

- Q-function: $Q(s, a)$ estimates the expected cumulative reward for taking action a in state s.
- Update rule:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left(r_t + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right)$$

- Challenges:
  - High-dimensional state spaces.
  - Q-table size grows exponentially.

Deep Q-Learning

- Replace Q-table with a Neural Network:

$$Q(s, a; \theta) \approx Q(s, a)$$

where θ represents the network parameters.

Deep Q-Learning

- Replace Q-table with a Neural Network:

$$Q(s, a; \theta) \approx Q(s, a)$$

  where θ represents the network parameters.

- Loss function:

$$L(\theta) = \mathbb{E}_{(s,a,r,s')}\left[(y - Q(s, a; \theta))^2\right]$$

  with target y:

$$y = r + \gamma \max_a Q(s', a; \theta^-)$$

- θ^-: Parameters of a target network.

Algorithm

Deep Q-Learning Algorithm

1. Initialize:
   - Replay memory $\mathcal{D}$
   - Action-value network $Q(s, a; \theta)$
   - Target network $Q(s, a; \theta^{-})$

Deep Q-Learning Algorithm

1. Initialize:
    - Replay memory $\mathcal{D}$
    - Action-value network $Q(s, a; \theta)$
    - Target network $Q(s, a; \theta^-)$
2. For each episode:
    - Observe state s.
    - Select action a using ϵ-greedy policy.
    - Execute a, observe r and s'.
    - Store (s, a, r, s') in $\mathcal{D}$.
    - Sample minibatch from $\mathcal{D}$.
    - Update θ to minimize $L(\theta)$.
    - Periodically update θ^-.

Practical Considerations

Challenges and Solutions

- Instability in training:
  - Use of experience replay.
  - Target networks for stable updates.
- Exploration vs Exploitation:
  - ϵ-greedy strategy.
  - Decaying ϵ over time.

Applications

Applications of Deep Q-Learning

- Game playing (e.g., Atari, Go)[1]:

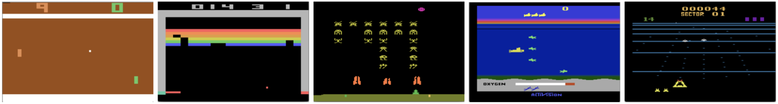

Figure 1: Screen shots from five Atari 2600 Games: (*Left-to-right*) Pong, Breakout, Space Invaders, Seaquest, Beam Rider

- Robotics: Control and navigation.
- Autonomous vehicles: Decision-making in dynamic environments.

[1] Playing Atari with Deep Reinforcement Learning, Mnih et al. (2013)

Conclusion

Conclusion

- Deep Q-Learning extends Q-Learning to complex problems.
- Key techniques:
    - Neural Networks.
    - Experience Replay.
    - Target Networks.
- Opens doors to applications in AI and Robotics.

Hands-on Session

DQN for Cart Pole problem.

(https://gymnasium.farama.org/environments/classic_control/