PPO vs. DQL : Une Comparaison

Question

PPO (Proximal Policy Optimization) et DQL (Deep Q-Learning) sont-ils pareils?

Réponse

Non, **PPO (Proximal Policy Optimization)** et **DQL (Deep Q-Learning)** ne sont pas la même chose. Bien qu'ils soient tous deux des algorithmes d'**apprentissage par renforcement** utilisant des réseaux de neurones, ils diffèrent dans leurs objectifs, méthodes et applications. Voici une comparaison détaillée pour comprendre leurs différences, forces, et faiblesses.

1 1. PPO vs. DQL : Objectifs et méthode de base

Deep Q-Learning (DQL)

- **But principal :** Approximer la "table Q" traditionnelle (de Q-Learning) avec un réseau de neurones. DQL essaie d'apprendre la **valeur d'une action dans un état donné** $Q(s, a)$.
- **Comment il fonctionne :**
  1. Dans chaque état s, l'agent teste toutes les actions possibles a.
  2. Le réseau de neurones prédit la valeur $Q(s, a)$ (récompenses futures attendues) pour chaque action.
  3. L'agent choisit l'action avec la plus grande valeur $Q(s, a)$.
- **Approche axée sur les valeurs (Value-Based) :** DQL se concentre sur **apprendre la qualité de chaque action possible**, sans se baser sur des probabilités d'actions comme PPO.

Proximal Policy Optimization (PPO)

- **But principal :** PPO optimise directement la **politique probabiliste** $\pi(a|s)$, qui donne une probabilité pour chaque action dans un état donné.

- **Comment il fonctionne :**
  1. PPO ajuste les probabilités des actions pour maximiser les récompenses futures.
  2. Une fonction objectif avec "clipping" limite les changements brusques dans la politique.
- **Approche axée sur les politiques (Policy-Based) :** PPO optimise directement une politique probabiliste, plutôt que d'estimer les valeurs des actions.

2 2. Différences fondamentales

Aspect	**DQL (Deep Q-Learning)**	**PPO (Proximal Policy Optimization)**
Type d'approche	Basé sur les valeurs (**Value-Based**)	Basé sur les politiques (**Policy-Based**)
But principal	Estimer $Q(s, a)$, la valeur des actions	Optimiser directement $\pi(a\|s)$
Décision	Choisit l'action avec la valeur Q maximale	Prend des actions selon une politique probabiliste
Réseaux de neurones	Un réseau pour approximer $Q(s, a)$	Deux réseaux : politique (π_θ) et valeur (V_ϕ)
Stabilité	Moins stable, nécessite des techniques comme Replay	Plus stable grâce au clipping et aux garde-fous
Adaptation	Moins efficace pour des espaces d'actions continus	Fonctionne bien dans des environnements continus
Exploration	Exploite et explore via ϵ-greedy	Encourage l'exploration via l'entropie

3 3. Quand utiliser DQL?

DQL est adapté pour des problèmes où:

1. **L'espace d'action est discret et limité :** Par exemple, un jeu où l'agent a des choix simples comme bouger à gauche, à droite, ou sauter.
2. **Les états peuvent être approximés efficacement avec un réseau unique :** Les environnements comme les jeux Atari (Pac-Man, Space Invaders) fonctionnent bien avec DQL.
3. **La politique optimale est déterministe :** Si le meilleur choix est toujours évident, DQL fonctionne bien (il choisit l'action avec la meilleure valeur $Q(s, a)$).

4 4. Quand utiliser PPO?

PPO est mieux adapté pour des problèmes où:

1. **L'espace d'action est continu :** Par exemple, contrôler un robot ou des véhicules autonomes avec des actions infiniment variables (direction, vitesse, etc.).

2. **L'environnement est complexe :** PPO gère bien les environnements nécessitant une exploration prudente et des changements progressifs de politique.

3. **La politique optimale est probabiliste :** Dans des situations où il faut diversifier les actions pour explorer (par exemple, plusieurs trajectoires possibles).

5 5. Forces et faiblesses

Critère	**DQL (Deep Q-Learning)**	**PPO (Proximal Policy Optimization)**
Facilité de mise en œuvre	Plus simple à implémenter	Plus complexe (deux réseaux à gérer)
Convergence	Plus difficile à stabiliser	Converge de manière plus stable
Exploration	Moins efficace dans des environnements complexes	Encourage l'exploration
Échelle	Ne s'adapte pas bien aux actions continues	Fonctionne bien dans des espaces continus
Performance générale	Efficace dans des environnements simples	Performant dans des environnements complexes

6 6. Exemple concret : Conduire une voiture autonome

- **DQL :** Si l'espace d'action est discret (par exemple : tourner à gauche, tourner à droite, accélérer, freiner), DQL peut prédire la meilleure action et maximiser les récompenses.

- **PPO :** Si l'espace d'action est continu (par exemple : tourner de 15 degrés, accélérer à 30 %), PPO est mieux adapté pour gérer les probabilités d'actions continues.

7 7. Pourquoi PPO est préféré?

PPO est devenu un standard pour les environnements complexes comme la robotique et les simulations grâce à:

1. **Stabilité accrue :** Les techniques de clipping et de régularisation rendent l'apprentissage robuste.

2. **Flexibilité :** Fonctionne bien dans des espaces d'action discrets et continus.

3. **Exploration efficace :** Encourage le modèle à explorer grâce à une pénalité d'entropie.

Résumé : Quand choisir quoi?

- **DQL :** Si l'environnement est simple, discret, et nécessitant moins de flexibilité.
- **PPO :** Si l'environnement est complexe, continu, ou nécessite une exploration prudente.