Cours DQN – Section 4.1

4.1 – Architecture complète d'un DQN

Dans les sections précédentes, nous avons étudié les composants individuels qui rendent DQN efficace : le réseau de neurones principal, le réseau cible, la mémoire de répétition et l'échantillonnage aléatoire. Cette section les regroupe pour présenter l'architecture complète du DQN.

Composants clés de l'architecture DQN

Voici les éléments essentiels qui composent le système DQN :

1. **Environnement** : l'agent observe l'état et agit, comme dans tout problème de RL.
2. **Réseau principal (Q-network)** : prédit les valeurs Q pour chaque action à partir d'un état.
3. **Réseau cible (target network)** : fournit des cibles stables pour l'apprentissage.
4. **Mémoire de répétition (Replay Memory)** : stocke les expériences vécues.
5. **Stratégie d'exploration (ex. ε-greedy)** : équilibre entre exploration et exploitation.
6. **Algorithme d'apprentissage** : met à jour les poids du réseau principal en minimisant l'erreur.

Schéma logique du système

Boucle d'interaction agent-environnement :

1. L'agent observe l'état actuel s_t.
2. Il choisit une action a_t via la stratégie ε-greedy.
3. Il exécute l'action, reçoit une récompense r_t et observe l'état suivant s_{t+1}.
4. Il stocke l'expérience $(s_t, a_t, r_t, s_{t+1}, \text{done})$ dans la mémoire.

Boucle d'apprentissage périodique :

1. Tirer un mini-batch aléatoire d'expériences depuis la mémoire.
2. Calculer la cible pour chaque expérience avec le *target network* :

$$y = r + \gamma \cdot \max_{a'} Q(s', a'; \theta^-)$$

3. Mettre à jour les poids du réseau principal en minimisant :

$$\mathcal{L} = (y - Q(s, a; \theta))^2$$

4. Toutes les C étapes, copier les poids du réseau principal dans le réseau cible.

Cycle global de DQN

Observation → Action → Récompense → Stockage → Apprentissage → Mise à jour réseau cible

Ce cycle tourne pendant des milliers d'épisodes, permettant à l'agent d'apprendre progressivement la meilleure stratégie.

Remarque sur la stabilité

C'est grâce à la combinaison des trois techniques suivantes que DQN parvient à stabiliser l'apprentissage :

— **Replay Memory** : évite la corrélation temporelle.
— **Target Network** : empêche les mises à jour instables.
— **Échantillonnage aléatoire** : améliore la diversité des apprentissages.

Conclusion

Cette architecture a permis à DQN d'apprendre à jouer à des jeux Atari à partir des pixels bruts, surpassant parfois les humains. Elle constitue un modèle fondamental de l'apprentissage par renforcement profond, et une base pour les variantes plus avancées.

La prochaine section détaillera le processus d'apprentissage complet du DQN, étape par étape.