Deep Q Network : Maitrise d'Atari par le Deep Reinforcement Learning

Pierre COUY

24 novembre 2021

Contexte

Réseaux de neurones profonds

- ~1990 : Yann LeCun développe les réseaux de neurones convolutionnels (CNN)
- 2005 : Article *Using GPUs for Machine Learning Algorithms*
- Depuis ~2012 : Plein essor du *Deep Learning* suite au succès des *CNN*
- 2018 : LeCun reçoit conjointement le prix Turing avec Yoshua Bengio et Geoffrey Hinton

Apprentissage par renforcement

- 1992 : Joueur surhumain de backgammon (TD-Gammon, qui contient un petit réseau de neurones)
- Échec de la plupart des autres tentatives d'allier réseaux de neurones et renforcement.
- Quelques maigres succès, mais beaucoup de contraintes

Contributions

Découverte et maitrise de plusieurs jeux Atari par un agent dont les observations sont constituées des pixels de l'écran.

Plusieurs innovations rendent possible cette avancée.

Replay d'expériences

Problèmes résolus

- Forte correlation entre les transitions successives
- Opportunités manquées d'apprentissage

$$Q^{new}(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\Big(\underbrace{\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}_{\text{new value (temporal difference target)}} - \underbrace{Q(s_t, a_t)}_{\text{old value}}\Big)}^{\text{temporal difference}}$$

Replay d'expériences

Problèmes résolus

- Forte correlation entre les transitions successives
- Opportunités manquées d'apprentissage

$$Q^{new}(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\Big(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}}\Big)}^{\text{temporal difference}}$$
$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{\text{new value (temporal difference target)}}$$

Principe

- Stocker les transitions observées dans une mémoire
- Échantilloner une *minibatch* depuis la mémoire pour faire la MàJ du Q-Network

Valeur cible

Problème résolu

- La valeur cible (membre de droite de la MàJ de Q) varie à chaque MàJ

Valeur cible

Problème résolu

- La valeur cible (membre de droite de la MàJ de Q) varie à chaque MàJ

Principe

- On utilise pour la valeur cible un clone du réseau de neurones (poids w^-)
- On met périodiquement à jour les poids w^- en copiant les poids w de la fonction de valeur.
- Les poids w^- sont mis à jour moins souvent que les poids w

$$r + \gamma \max_{a'} Q(s', a', \mathbf{w}^-)$$

Architecture du *Q-Network*

Problème résolu

- Nécessité d'effectuer un passage par le *Q-network* pour chaque action afin de déterminer l'action de valeur max.

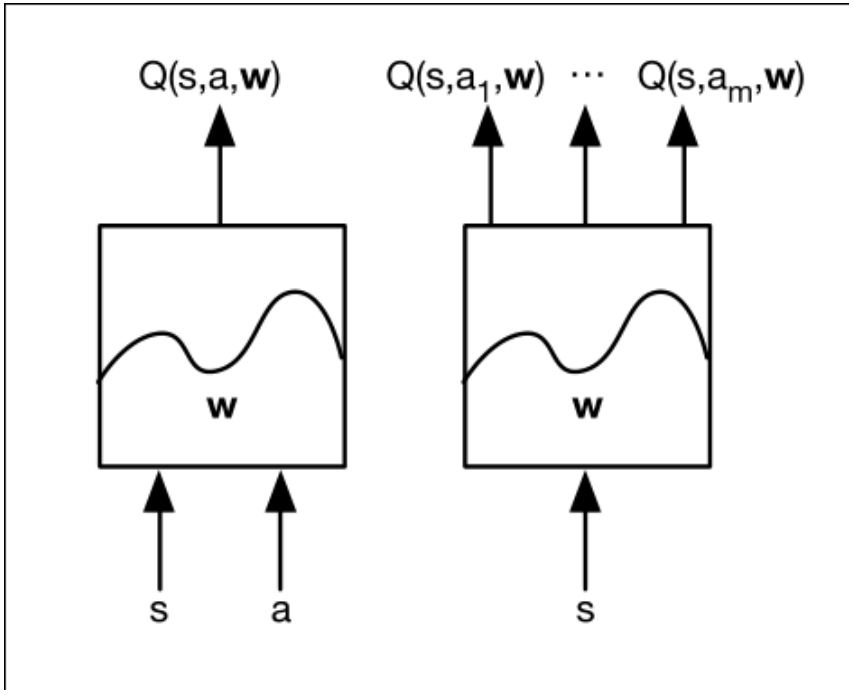
Architecture du *Q-Network*

Problème résolu

- Nécessité d'effectuer un passage par le *Q-network* pour chaque action afin de déterminer l'action de valeur max.

Principe

- Calculer les valeurs de toutes les actions à la fois en ne prenant que l'état en entrée ; et en ayant une sortie par action.



Résultats

- Bat l'état de l'art de l'époque sur plusieurs jeux avec un seul algorithme
- Ne nécessite aucun pré-traitement expert pour extraire des informations : travaille avec les pixels bruts de l'écran

	B. Rider	Breakout	Enduro	Pong	Q*bert	Seaquest	S. Invaders
Random	354	1.2	0	−20.4	157	110	179
Sarsa [3]	996	5.2	129	−19	614	665	271
Contingency [4]	1743	6	159	−17	960	723	268
DQN	**4092**	**168**	**470**	**20**	**1952**	**1705**	**581**
Human	7456	31	368	−3	18900	28010	3690
HNeat Best [8]	3616	52	106	19	1800	920	**1720**
HNeat Pixel [8]	1332	4	91	−16	1325	800	1145
DQN Best	**5184**	**225**	**661**	**21**	**4500**	**1740**	1075

Table 1: The upper table compares average total reward for various learning methods by running an ϵ-greedy policy with $\epsilon = 0.05$ for a fixed number of steps. The lower table reports results of the single best performing episode for HNeat and DQN. HNeat produces deterministic policies that always get the same score while DQN used an ϵ-greedy policy with $\epsilon = 0.05$.