

Réseaux de neurones convolutionnels (CNN) et apprentissage profond par renforcement (DRL)

Arthur Aubret

Université Lyon 1

12/12/2019 & 19/12/2019

Sommaire

- 1 Réseaux de neurones convolutionnels
 - Introduction
 - Masque convolution 1D
 - Masque convolution 2D
- 2 Apprentissage profond par renforcement
 - Rappels
 - DQN
 - Policy-based methods
 - Actor-Critic
- 3 Motivation intrinsèque
 - Problèmes du RL
 - Récompenses intrinsèque
 - Différents types de récompense intrinsèque

Idée

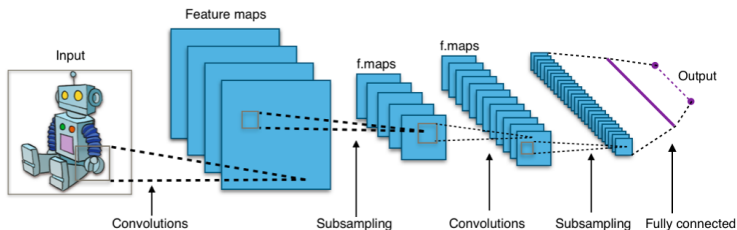


Figure: Réseau de neurones convolutionnel 2D

- Extraction automatique de caractéristiques.
- Partage de paramètres: localité de l'information.

Trois types de calculs

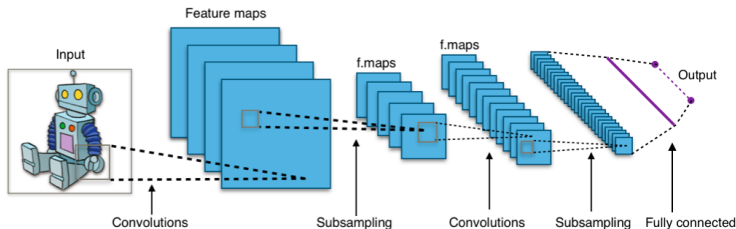


Figure: Réseau de neurones convolutionnel 2D

Convolutions : Extraction des caractéristiques locales.

Pooling : Invariance par translation et réduction de la taille des images.

Activation ReLU : Non-linéarité sans saturation.

Avantages/Limites



Figure: Invariances CNN.

Avantages:

- Invariance par translation.

Désavantages:

- Variance par rotation.
- Variance par réflexion.
- Non-encodage de la position.

Cross-corrélation

Appliquer d'une cross-corrélation entre g et f en u :

- 1D: $g_{filtre}(u) = \sum_{x=-\infty}^{\infty} f(x)g(x-u)$
- 2D: $g_{filtre}(u, v) = \sum_{x=-\infty}^{\infty} \sum_{y=-\infty}^{\infty} f(x, y)g(x-u, y-v)$
- 3D: $g_{filtre}(u, v, w) = \sum_{x=-\infty}^{\infty} \sum_{y=-\infty}^{\infty} \sum_{z=-\infty}^{\infty} f(x, y, z)g(x-u, y-v, z-w)$
- etc...

Souvent, $\forall x \notin [-k, k], f(x) = 0$, avec k réel.

Masque convolution 1D

Masque convolution 1D

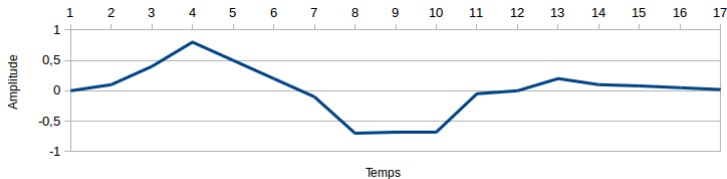


Figure: Serie de données F

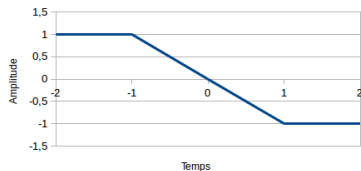


Figure: Masque de convolution G

Masque convolution 1D

0	0,1	0,4	0,8	0,5	0,2	-0,1	-0,7	-0,69	-0,68
-0,05	0	0,2	0,1	0,08	0,05	0,02	0	0	0

Table: Série temporelle F

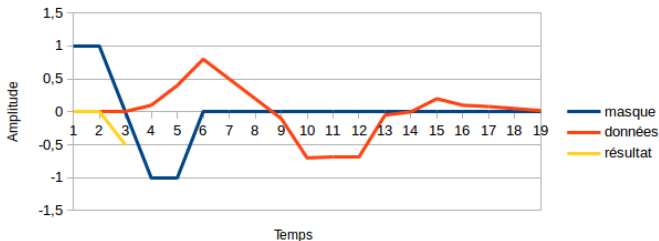
1	1	0	-1	-1
---	---	---	----	----

Table: Masque de convolution G

- Appliquer le filtre G à F.
- Masque=filtre=noyau de convolution

Masque convolution 1D

Appliquer masque convolution 1D



1	1	0	-1	-1
---	---	---	----	----

$u=3; x=5; f(x)=0,4; g=(x-u)=g(5-3)=-1$
 $u=3; x=4; f(x)=0,1; g=(x-u)=g(4-3)=-1$

0	0,1	0,4	0,8	0,5	0,2	-0,1	-0,7	-0,69	-0,68
-0,05	0	0,2	0,1	0,08	0,05	0,02	0	0	0

$$1*0+1*0+0*0-1*0,1-1*0,4=-0,5$$

-0,5									

Masque convolution 1D

Appliquer masque convolution 1D

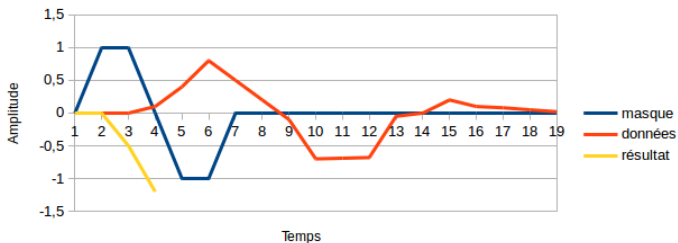


Diagram illustrating the calculation of the first row of the payoff matrix:

1	1	0	-1	-1
---	---	---	----	----

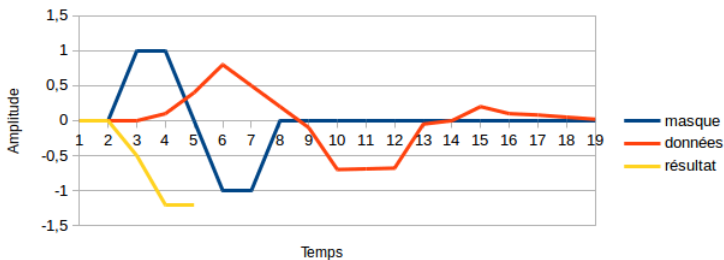
0	0,1	0,4	0,8	0,5	0,2	-0,1	-0,7	-0,69	-0,68
-0,05	0	0,2	0,1	0,08	0,05	0,02	0	0	0

$$1 \cdot 0 + 1 \cdot 0 + 0 \cdot 0,1 - 1 \cdot 0,4 - 1 \cdot 0,8 = -1,2$$

-0,5	-1,2							

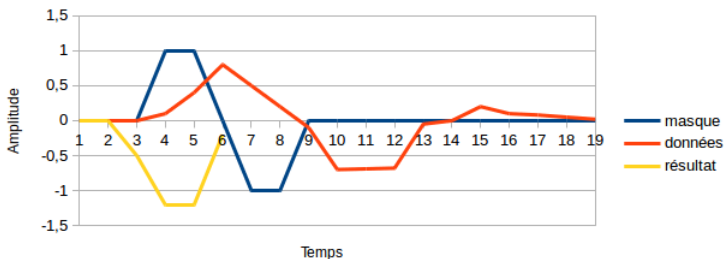
Masque convolution 1D

Appliquer masque convolution 1D



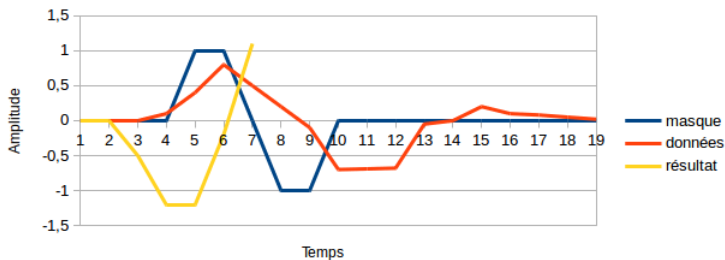
Masque convolution 1D

Appliquer masque convolution 1D



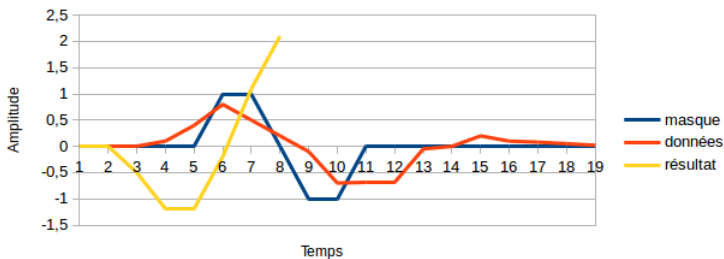
Masque convolution 1D

Appliquer masque convolution 1D



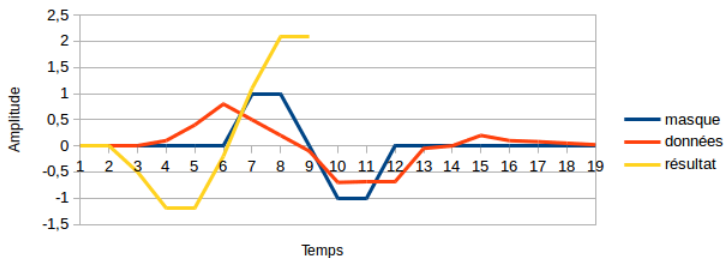
Masque convolution 1D

Appliquer masque convolution 1D



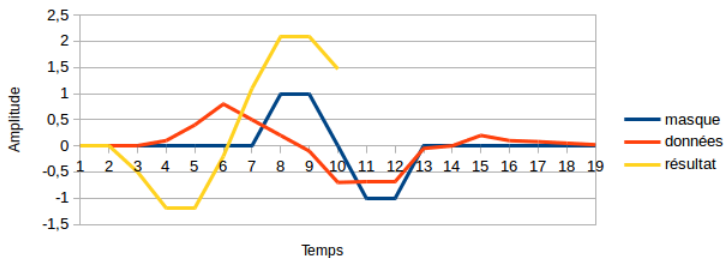
Masque convolution 1D

Appliquer masque convolution 1D



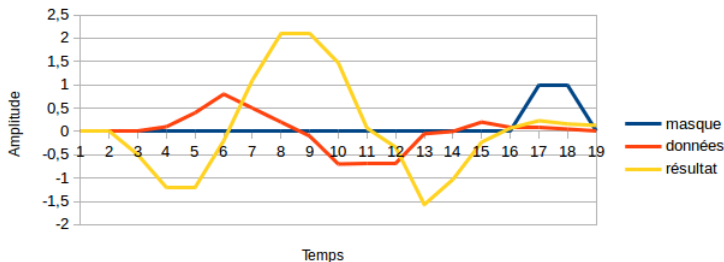
Masque convolution 1D

Appliquer masque convolution 1D



Masque convolution 1D

Appliquer masque convolution 1D



0	0,1	0,4	0,8	0,5	0,2	-0,1	-0,7	-0,69	-0,68
-0,05	0	0,2	0,1	0,08	0,05	0,02	0	0	0

-0,5	-1,2	-1,2	-0,2	1,1	2,1	2,09	1,47	0,07
-0,34	-1,57	-1,03	-0,23	0,07	0,23	0,16	0,13	0,02

Masque convolution 2D

Masque convolution 2D

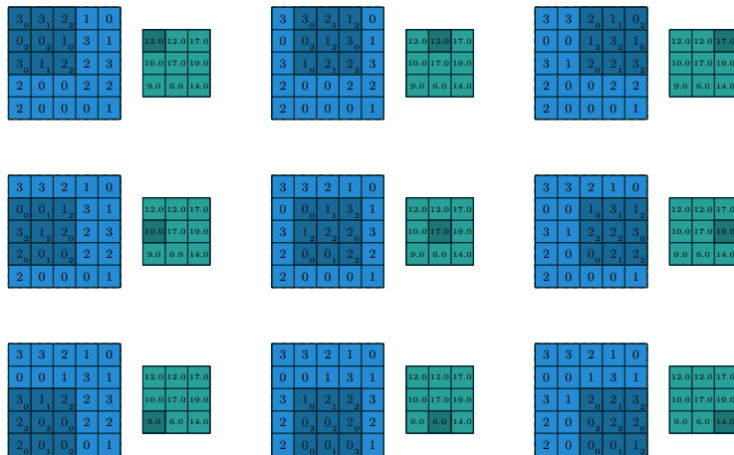


Figure: Dumoulin, V., & Visin, F. A guide to convolution arithmetic for deep learning.

Masque convolution 2D

Padding et striding

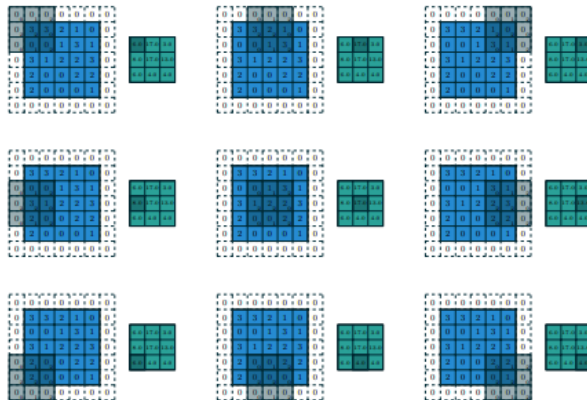
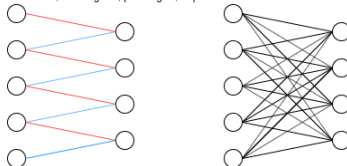


Figure: Dumoulin, V., & Visin, F. (2016). A guide to convolution arithmetic for deep learning.

Calculer la taille de la couche de sortie

kernel size = 2; striding = 1; padding=0, depth=1



F : taille du filtre de convolution.

W : taille des données d'entrées.

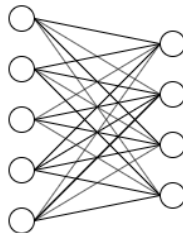
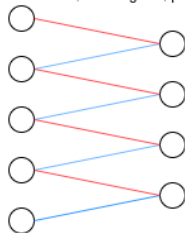
P : taille du padding.

S : taille du striding.

Taille de la couche de sortie: $(W - F + 2P)/S + 1$

Linéarité

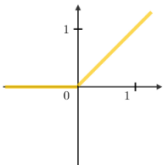
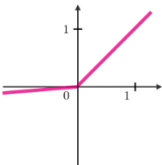
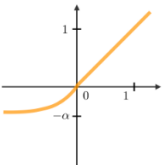
kernel size = 2; striding = 1; padding=0, depth=1



- Opération linéaire.
- Poids partagés.
- Poids à 0.

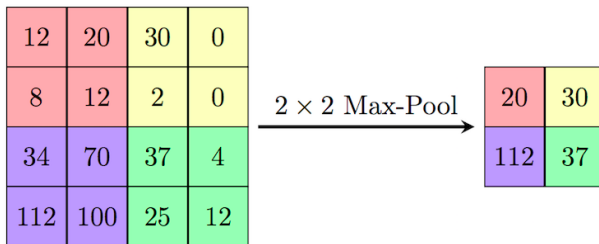
Masque convolution 2D

Fonction d'activation

ReLU	Leaky ReLU	ELU
$g(z) = \max(0, z)$	$g(z) = \max(\epsilon z, z)$ with $\epsilon \ll 1$	$g(z) = \max(\alpha(e^z - 1), z)$ with $\alpha \ll 1$
		

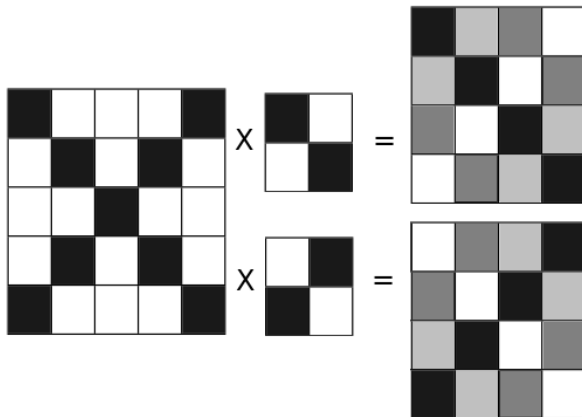
Max pooling

- Permet de réduire le nombre de neurones.
- Généraliser par translation.



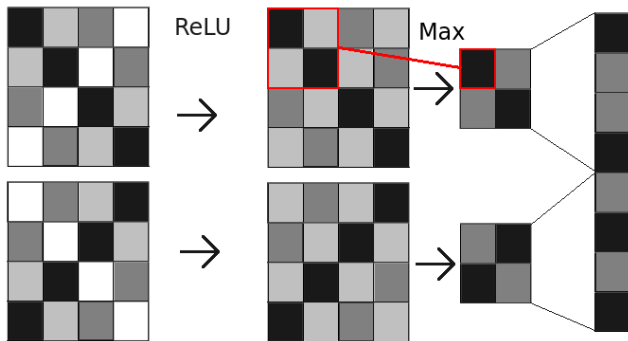
Masque convolution 2D

Exemple - étape 1 : convolution



Masque convolution 2D

Exemple - étape 2 : ReLU + Max pooling



Apprentissage des filtres

Avant : Définition manuelles des caractéristiques.

Deep learning : Apprentissage via back-propagation.



Rappels - MDP

Processus de décision markovien:

- S l'ensemble d'états;
- A l'ensemble d'actions;
- P la fonction de transition d'un état à l'autre; **inconnue**
- R la fonction de récompense; **inconnue**
- γ le facteur d'atténuation;
- ρ_0 la distribution initiale d'état. **inconnue**

Objectif RL : Trouver la politique π^* :

$$\pi^* = \arg \max_{\pi} \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, s_{t+1}, \pi(s_t)) \right]. \quad (1)$$

Rappels - Équation de Bellman

On utilise l'espérance cumulée de récompense suivant un couple (état, action):

$$Q_{\pi}(s, a) = \mathbb{E}_{a_t \sim \pi(s_t)} \left(\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) | s_0 = s, a_0 = a \right). \quad (2)$$

$$V_{\pi}(s) = \mathbb{E}_{a_t \sim \pi(s_t)} \left(\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t)) | s_0 = s \right). \quad (3)$$

On applique l'équation de Bellman:

$$Q_{\pi}(s_t, a_t) = R(s_t, a_t) + \gamma \max_a Q_{\pi}(P(s_t, a_t), a). \quad (4)$$

Rappels

Approche tabulaire

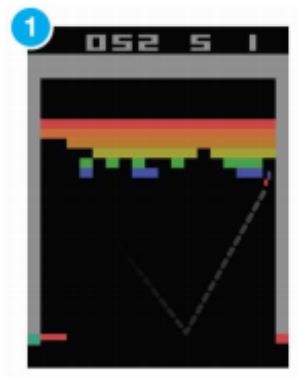
	s1	s2	s3	s4
a0	1	0	2	1.5
a1	0.5	0	0.1	4
a2	1	1	1	1

Idée : Table contenant les valeurs de **tous** les couples (état,action).

Problème : On retourne rarement dans le même état si l'espace d'état est continu ou trop grand.

Rappels

Exemple d'espace d'état trop grand



Taille de l'espace d'état:

$$|S| = 255^{84 \times 84 \times 4} = 255^{28224}$$

Approximation linéaire

Idée:

- $Q(s, a) = \theta^T \phi(s, a)$
- $\theta = \theta + \alpha \times \phi(s, a) \times \delta Q$
- Choisir des caractéristiques $\phi(s, a)$ manuellement.
- Apprendre les poids θ de l'approximation linéaire.

Problèmes:

- Difficile de choisir les bonnes caractéristiques.
- Limité par la linéarité.

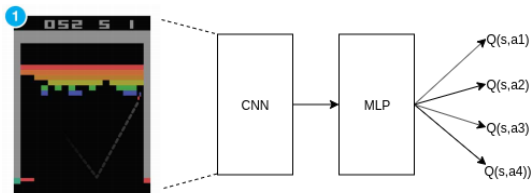
Solution:

- Apprendre des caractéristiques non-linéaires.

Approches

- ➊ Méthodes "Value-based" (DQN): Choix des actions en fonction des valeurs $Q(s, a)$
- ➋ Méthodes "Policy-based" (REINFORCE): Paramétrage direct de la politique $\pi_{\theta'}$.
- ➌ Méthodes "Actor-critic" (A2C): Modification de π_{θ} en fonction de $Q(s, a)$.

Deep Q-network (DQN)



- <https://www.youtube.com/watch?v=V1eYniJ0Rnk>

- $$G_\theta = \left[Q_\theta(s_t, a_t) - (R(s_t, a_t) + \gamma \max_a \hat{Q}_\theta(P(s_t, a_t), a)) \right]^2$$

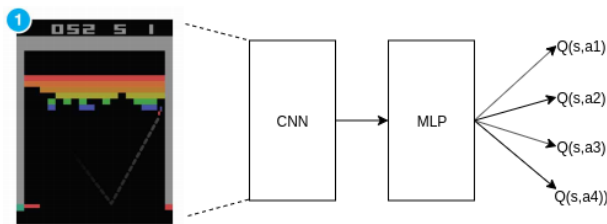
- Minimisation de G_θ via backpropagation du gradient.

Problèmes:

- 1 \hat{Q}_θ dépend aussi de θ , pouvant faire diverger l'algorithme.
- 2 Les exemples sont corrélés, rendant l'apprentissage instable.

DQN

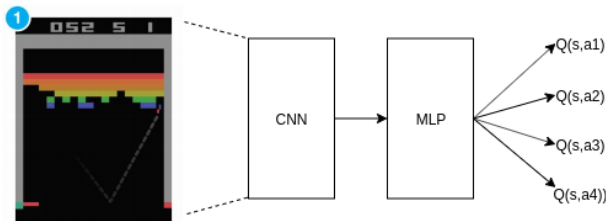
Deep Q-network (DQN) - Astuces



- $G_{\theta} = Q_{\theta}(s_t, a_t) - R(s_t, a_t) - \gamma \max_a \hat{Q}_{\theta}(P(s_t, a_t), a)$
- Utilisation de l'experience replay : stockages des 100 000 (par exemple) dernières interactions.
- Utilisation du target network: $\hat{Q}_{\theta'}$ est modifié graduellement vers Q_{θ} .

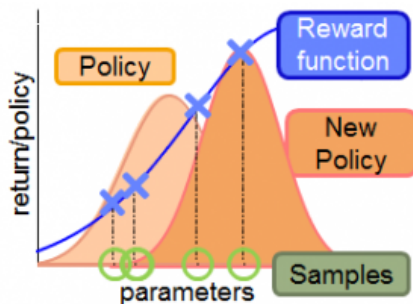
DQN

Deep Q-network (DQN) - Problème



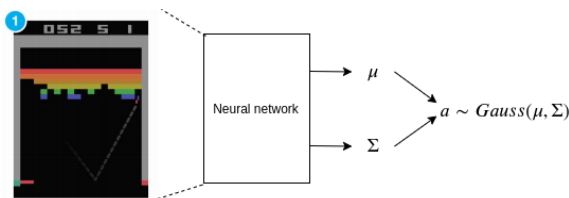
- Le réseau calcule $Q(s, a)$ pour chaque action.
- Mais comment faire si les actions sont continues ?

Méthodes policy-based



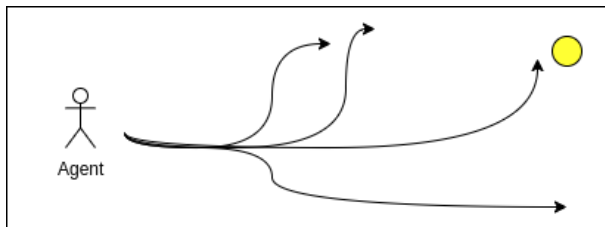
- Policy π génère une probabilité d'action.
- On augmente la probabilité des actions générant de fortes récompenses.

REINFORCE



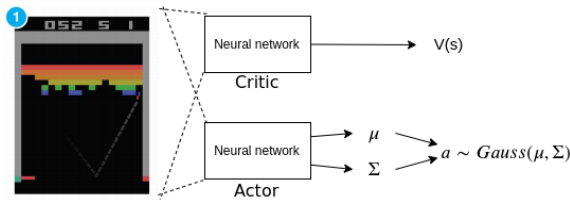
- $J(\theta) = \mathbb{E}_{a, s_t \sim \pi_\theta} \left[\sum_{t'=t}^T R(s_{t'}, a_{t'}) \right]$.
- $\nabla_{\theta} J(\theta) = \mathbb{E}_{a, s_t \sim \pi_\theta} \left[\nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \sum_{t'=t}^T R(s_{t'}, a_{t'}) \right]$.
- $\sum_{t'=t}^T R(s_{t'}, a_{t'})$ "pondère" la probabilité de l'action.
- Mais pour de longs épisodes, la variance est très importante.

REINFORCE - Variance importante



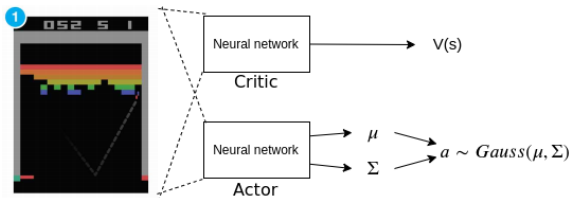
- $\nabla_{\theta} J(\theta) = \mathbb{E}_{a, s_t \sim \pi_{\theta}} [\nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \sum_{t'=t}^T R(s_{t'}, a_{t'})]$
- Méthode Monte-carlo.
- Variance importante.

Architectures Actor-Critic



- $\sum_{t'=t}^T R(s_{t'}, a_{t'}) = R(s_t, a_t) + \gamma V_{\theta'}(s_{t+1}) = Q(s_t, a_t).$
- $\nabla_{\theta} J(\theta) = \mathbb{E}_{a, s \sim \pi_{\theta}} [\nabla_{\theta} \log \pi_{\theta}(a|s) Q(s, a)].$
- Actor : Calculer $\log \pi_{\theta}(a|s).$
- Critic : Calculer $V_{\theta'}(s')$ ou $Q_{\theta'}(s, a).$

Réduction de la variance



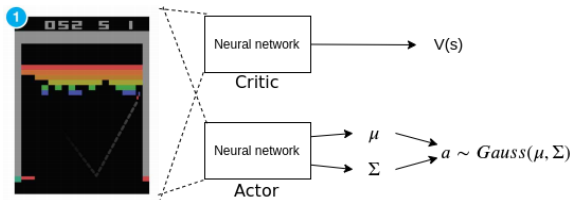
$$\nabla_{\theta} J(\theta) = \mathbb{E}_{a,s,s' \sim \pi_{\theta}} [\nabla_{\theta} \log \pi_{\theta}(a|s) (Q(s,a) - b(s))]$$

- Baseline $b(s)$ indépendante de l'action pour ne pas biaiser le gradient.

Dérivation

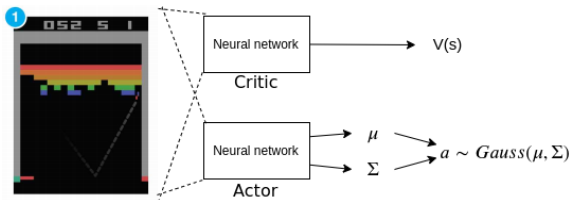
$$\begin{aligned}
 \nabla_{\theta} J(\theta) &= \mathbb{E}_{a,s \sim \pi_{\theta}} [\nabla_{\theta} \log \pi_{\theta}(a|s)(Q(s, a) - b(s))] \\
 &= \nabla_{\theta} J_{prev}(\theta) - \mathbb{E}_{a,s \sim \pi_{\theta}} \nabla_{\theta} \log \pi_{\theta}(a|s) b(s) \\
 &= \nabla_{\theta} J_{prev}(\theta) - \mathbb{E}_{a,s \sim \pi_{\theta}} b(s) \nabla_{\theta} \log \pi_{\theta}(a|s) \\
 &= \nabla_{\theta} J_{prev}(\theta) - \sum_s \mu(s) b(s) \sum_a \pi_{\theta}(a|s) \nabla_{\theta} \log \pi_{\theta}(a|s) \\
 &= \nabla_{\theta} J_{prev}(\theta) - \sum_s \mu(s) b(s) \sum_a \pi_{\theta}(a|s) \frac{\nabla_{\theta} \pi_{\theta}(a|s)}{\pi_{\theta}(a|s)} \\
 &= \nabla_{\theta} J_{prev}(\theta) - \sum_s \mu(s) b(s) \nabla_{\theta} 1 \\
 &= \nabla_{\theta} J_{prev}(\theta)
 \end{aligned}$$

A2C: réduire la variance



- $\nabla_{\theta} J(\theta) = \mathbb{E}_{a, s \sim \pi_{\theta}} [\nabla_{\theta} \log \pi_{\theta}(a|s) (Q_{\theta'}(s, a) - V_{\theta'}(s))]$.
- $A(s, a) = Q_{\theta'}(s, a) - V_{\theta'}(s)$ fonction avantage.
- $A(s, a) > 0$ si a est meilleure que la politique moyenne.
- $A(s, a) < 0$ si a est moins bonne que la politique moyenne.

A2C: Explorer



- Convergence prématurée.
- Ajout d'un terme d'entropie forçant une forte variance.
- $$\nabla_{\theta} J(\theta) = \mathbb{E}_{a, s \sim \pi_{\theta}} [\nabla_{\theta} \log \pi_{\theta}(a|s) A(s, a) + \nabla_{\theta} H(\pi_{\theta}(a|s))].$$

Autres modèles...

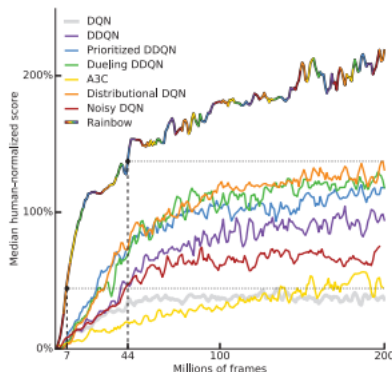
Améliorations du DQN:

- Double DQN [Van Hasselt et al., 2016].
- Prioritized Experience replay [Schaul et al., 2015].
- Dueling DQN [Wang et al., 2015].
- Distributional DQN [Dabney et al., 2018].
- Rainbow DQN [Hessel et al., 2018].

Modèles Actor-Critic:

- Trust region policy optimization [Schulman et al., 2015].
- Proximal policy optimization [Schulman et al., 2017].
- Deep deterministic policy gradient [Lillicrap et al., 2015].
- Soft actor-critic [Haarnoja et al., 2018].

Efficacité computationnelle



- En réalité : pas de simulateurs.
- 200k frames = 1h humaine; 44M = 220h humaines.

Exploration

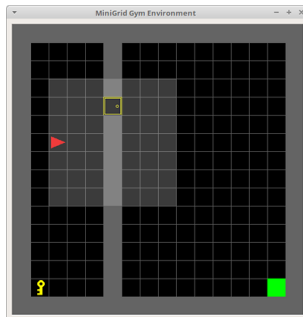
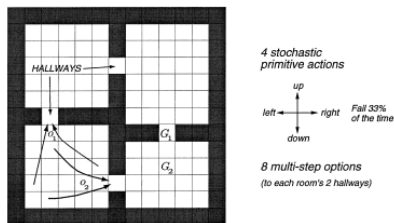


Figure: Environnement très simple avec des récompenses éparses.

- L'agent n'apprend rien car il ne trouve jamais la récompense.

Abstraction des décisions



- Nous sommes capable de prendre des décisions de haut niveau.
- Plus facile d'apprendre sur 10 actions haut-niveau que sur 1000 actions bas-niveau [Sutton et al., 1999].

Solution

- Mixer apprentissage développemental et apprentissage par renforcement.
- Utiliser une récompense **intrinsèque** plutôt qu'extrinsèque.

Extrinsèque vs intrinsèque

- ① J'ai envie de jouer avec mes jouets !
- ② Je dois arrêter de jouer car c'est enfantin.
- ③ Je dois arrêter de jouer car les autres se moquent de moi.
- ④ Je suis excité à l'idée d'appuyer sur ce bouton magique que je ne connais pas.
- ⑤ Je travaille pour avoir de bonnes notes à l'école.
- ⑥ Je veux devenir plus fort.
- ⑦ J'ai faim et je vais chercher de la nourriture.
- ⑧ J'aime découvrir et comprendre les mathématiques.

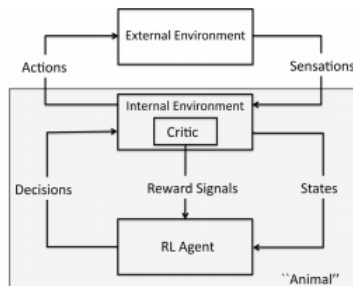
Extrinsèque vs intrinsèque

- ① J'ai envie de jouer avec mes jouets ! **Intrinsèque**
- ② Je dois arrêter de jouer car c'est enfantin. **Extrinsèque**
- ③ Je dois arrêter de jouer car les autres se moquent de moi.
Extrinsèque
- ④ Je suis excité à l'idée d'appuyer sur ce bouton magique que je ne connais pas. **Intrinsèque**
- ⑤ Je travaille pour avoir de bonnes notes à l'école.
Extrinsèque
- ⑥ Je veux devenir plus fort. **Ca dépend**
- ⑦ J'ai faim et je vais chercher de la nourriture. **Extrinsèque**
- ⑧ J'aime découvrir et comprendre les mathématiques.
Intrinsèque

Récompenses intrinsèque

Combiner motivation intrinsèque et apprentissage par renforcement

Les sources des signaux de récompense sont internes à l'agent



Curiosité

- Récompenser l'erreur de prédiction des états suivants [Pathak et al., 2017].
<https://pathak22.github.io/noreward-rl/>
- Récompenser des états loins de ceux en mémoire [Savinov et al., 2018] <https://ai.googleblog.com/2018/10/curiosity-and-procrastination-in.html>.
- Récompenser une erreur de prédiction [Burda et al., 2018].
- Récompenser l'agent selon la nouveauté des états [Bellemare et al., 2016][Ostrovski et al., 2017].

Différents types de récompense intrinsèque

Génération d'objectifs

Récompenser sa capacité de distinction des compétences.
[Gregor et al., 2016]:

Utiliser la théorie de l'information
<https://sites.google.com/view/diayn/>
[Eysenbach et al., 2018]

Ressources et références I

Réseaux de neurones convolutionnels :

- <http://cs231n.github.io/convolutional-networks/>
- <https://towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1>
- <https://stanford.edu/~shervine/l/fr/teaching/cs-230/pense-bete-reseaux-neurones-convolutionnels>
- https://computersciencewiki.org/index.php/Max-pooling/_/_Pooling

Apprentissage profond par renforcement:

- Excellent livre de Sutton gratuit :
<http://incompleteideas.net/book/the-book.html>
- Cours en ligne de David Silver
- Cours de Berkeley

Ressources et références II



Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. (2016).
Unifying count-based exploration and intrinsic motivation.
In Advances in Neural Information Processing Systems,
pages 1471–1479.



Burda, Y., Edwards, H., Storkey, A., and Klimov, O. (2018).
Exploration by random network distillation.
arXiv preprint arXiv:1810.12894.



Dabney, W., Ostrovski, G., Silver, D., and Munos, R. (2018).
Implicit quantile networks for distributional reinforcement learning.
arXiv preprint arXiv:1806.06923.

Ressources et références III



Eysenbach, B., Gupta, A., Ibarz, J., and Levine, S. (2018).
Diversity is all you need: Learning skills without a reward
function.

arXiv preprint arXiv:1802.06070.



Gregor, K., Rezende, D. J., and Wierstra, D. (2016).
Variational intrinsic control.

arXiv preprint arXiv:1611.07507.



Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. (2018).
Soft actor-critic: Off-policy maximum entropy deep
reinforcement learning with a stochastic actor.

arXiv preprint arXiv:1801.01290.

Ressources et références IV



Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., and Silver, D. (2018).

Rainbow: Combining improvements in deep reinforcement learning.

In *Thirty-Second AAAI Conference on Artificial Intelligence*.



Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2015).

Continuous control with deep reinforcement learning.

arXiv preprint arXiv:1509.02971.

Ressources et références V



Ostrovski, G., Bellemare, M. G., Oord, A. v. d., and Munos, R. (2017).

Count-based exploration with neural density models.

arXiv preprint arXiv:1703.01310.



Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. (2017).

Curiosity-driven exploration by self-supervised prediction.

In *International Conference on Machine Learning (ICML)*, volume 2017.



Savinov, N., Raichuk, A., Marinier, R., Vincent, D., Pollefeys, M., Lillicrap, T., and Gelly, S. (2018).

Episodic curiosity through reachability.

arXiv preprint arXiv:1810.02274.

Ressources et références VI



Schaul, T., Quan, J., Antonoglou, I., and Silver, D. (2015).
Prioritized experience replay.
arXiv preprint arXiv:1511.05952.



Schulman, J., Levine, S., Abbeel, P., Jordan, M., and
Moritz, P. (2015).
Trust region policy optimization.
In International conference on machine learning, pages
1889–1897.



Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and
Klimov, O. (2017).
Proximal policy optimization algorithms.
arXiv preprint arXiv:1707.06347.

Ressources et références VII



Sutton, R. S., Precup, D., and Singh, S. (1999).
Between mdps and semi-mdps: A framework for temporal
abstraction in reinforcement learning.
Artificial intelligence, 112(1-2):181–211.



Van Hasselt, H., Guez, A., and Silver, D. (2016).
Deep reinforcement learning with double q-learning.
In *Thirtieth AAAI conference on artificial intelligence*.



Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot,
M., and De Freitas, N. (2015).
Dueling network architectures for deep reinforcement
learning.
arXiv preprint arXiv:1511.06581.