Reinforcement Learning

Thomas Ranvier

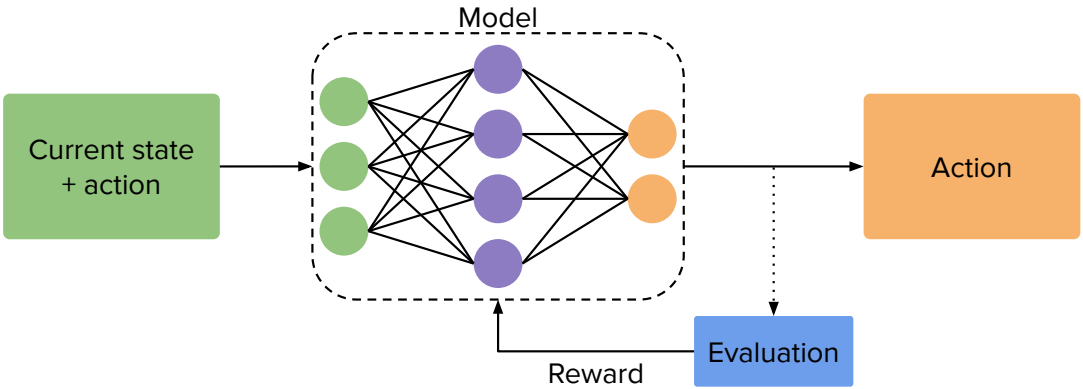
Université Claude Bernard, Lyon 1

Sommaire

Introduction

Reinforcement Learning

Un autre type d'apprentissage



Contexte de l'apprentissage par renforcement

Contexte

L'agent qui réalise des actions

Agent

Contexte de l'apprentissage par renforcement

Contexte

L'environnement au sein duquel l'agent interagit

Agent

Environment

Contexte de l'apprentissage par renforcement

Contexte

L'agent réalise des actions qui prennent toutes place au sein de l'environnement

Contexte de l'apprentissage par renforcement

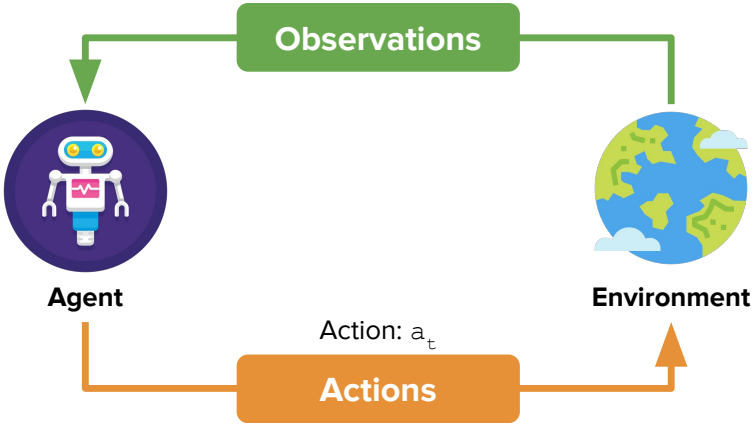
Contexte

L'action que l'agent réalise au temps t est notée a_t

Contexte de l'apprentissage par renforcement

Contexte

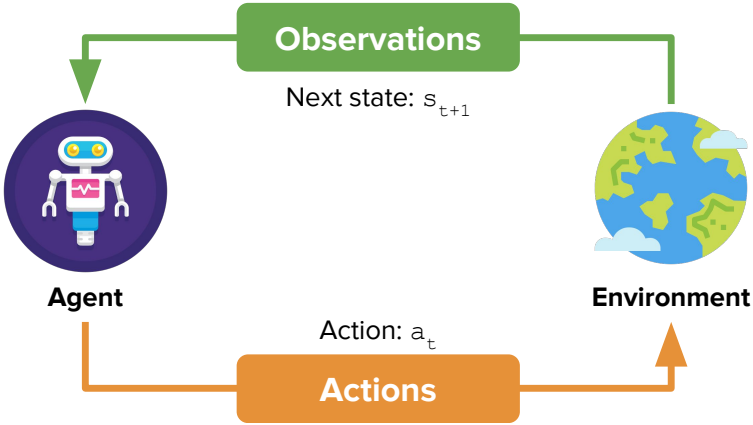
Suite à chaque action l'agent observe son environnement



Contexte de l'apprentissage par renforcement

Contexte

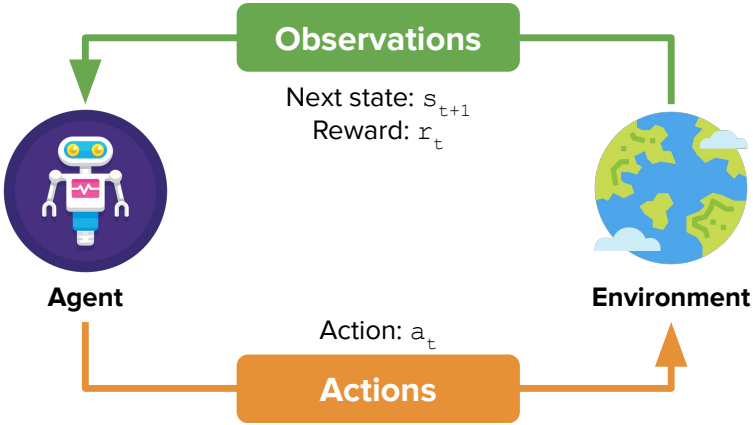
Cela lui permet de voir son état s_{t+1} suite à l'action a_t



Contexte de l'apprentissage par renforcement

Contexte

Il obtient également sa récompense r_t



Intuition derrière l'apprentissage par renforcement

Intuition

L'apprentissage par renforcement est basé sur le principe de "trial and error", à chaque étape t :

Intuition derrière l'apprentissage par renforcement

Intuition

L'apprentissage par renforcement est basé sur le principe de "trial and error", à chaque étape t :

1. L'agent observe son état actuel s_t

Intuition derrière l'apprentissage par renforcement

Intuition

L'apprentissage par renforcement est basé sur le principe de "trial and error", à chaque étape t :

1. L'agent observe son état actuel s_t
2. Basé sur cette observation l'agent décide de réaliser l'action a_t

Intuition derrière l'apprentissage par renforcement

Intuition

L'apprentissage par renforcement est basé sur le principe de "trial and error", à chaque étape t :

1. L'agent observe son état actuel s_t
2. Basé sur cette observation l'agent décide de réaliser l'action a_t
3. Suite à cette action l'agent reçoit sa récompense r_t

Intuition derrière l'apprentissage par renforcement

Intuition

L'apprentissage par renforcement est basé sur le principe de "trial and error", à chaque étape t :

1. L'agent observe son état actuel s_t
2. Basé sur cette observation l'agent décide de réaliser l'action a_t
3. Suite à cette action l'agent reçoit sa récompense r_t
4. Cette récompense sert de feedback pour orienter l'apprentissage

Intuition derrière l'apprentissage par renforcement

Intuition

L'apprentissage par renforcement est basé sur le principe de "trial and error", à chaque étape t :

1. L'agent observe son état actuel s_t
2. Basé sur cette observation l'agent décide de réaliser l'action a_t
3. Suite à cette action l'agent reçoit sa récompense r_t
4. Cette récompense sert de feedback pour orienter l'apprentissage

En suivant ce principe à chaque étape l'agent apprend à favoriser les actions menant à de hautes récompenses et à éviter les actions menant à des récompenses négatives.

Q-function

Qu'est ce que la *Q*-function ?

Définition de la Q-function

Définition

R_t est la récompense totale que l'agent obtiendra à partir de l'étape t :

$$R_t = r_t + \gamma^1 r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \dots$$

Définition de la Q-function

Définition

R_t est la récompense totale que l'agent obtiendra à partir de l'étape t :

$$R_t = r_t + \gamma^1 r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \dots$$

Le facteur d'actualisation $0 < \gamma < 1$ détermine l'importance accordé aux récompenses futures :

Définition de la Q-function

Définition

R_t est la récompense totale que l'agent obtiendra à partir de l'étape t :

$$R_t = r_t + \gamma^1 r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \dots$$

Le facteur d'actualisation $0 < \gamma < 1$ détermine l'importance accordé aux récompenses futures :

- Proche de 0 il accordera peu d'importance aux récompenses futures : bénéfique dans les cas où une action entraîne une récompense immédiate sans qu'il n'y ait besoin d'anticipation

Définition de la Q-function

Définition

R_t est la récompense totale que l'agent obtiendra à partir de l'étape t :

$$R_t = r_t + \gamma^1 r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \dots$$

Le facteur d'actualisation $0 < \gamma < 1$ détermine l'importance accordé aux récompenses futures :

- Proche de 0 il accordera peu d'importance aux récompenses futures : bénéfique dans les cas où une action entraîne une récompense immédiate sans qu'il n'y ait besoin d'anticipation
- Proche de 1 il accordera beaucoup d'importance aux récompenses futures : bénéfique dans les cas où il faudra réaliser des actions par anticipation pour obtenir une récompense

Définition de la Q-function

Définition

R_t est la récompense totale que l'agent obtiendra à partir de l'étape t :

$$R_t = r_t + \gamma^1 r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \dots$$

Le facteur d'actualisation $0 < \gamma < 1$ détermine l'importance accordé aux récompenses futures :

- Proche de 0 il accordera peu d'importance aux récompenses futures : bénéfique dans les cas où une action entraîne une récompense immédiate sans qu'il n'y ait besoin d'anticipation
- Proche de 1 il accordera beaucoup d'importance aux récompenses futures : bénéfique dans les cas où il faudra réaliser des actions par anticipation pour obtenir une récompense

On ne peut pas connaître R_t en pratique, donc on utilise la Q-function pour l'estimer :

$$Q(s_t, a_t) = r_t + \gamma Q(s_{t+1}, a_{t+1})$$

Définition de la Q-function

Définition

R_t est la récompense totale que l'agent obtiendra à partir de l'étape t :

$$R_t = r_t + \gamma^1 r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \dots$$

Le facteur d'actualisation $0 < \gamma < 1$ détermine l'importance accordé aux récompenses futures :

- Proche de 0 il accordera peu d'importance aux récompenses futures : bénéfique dans les cas où une action entraîne une récompense immédiate sans qu'il n'y ait besoin d'anticipation
- Proche de 1 il accordera beaucoup d'importance aux récompenses futures : bénéfique dans les cas où il faudra réaliser des actions par anticipation pour obtenir une récompense

On ne peut pas connaître R_t en pratique, donc on utilise la Q-function pour l'estimer :

$$Q(s_t, a_t) = r_t + \gamma Q(s_{t+1}, a_{t+1})$$

Ainsi si la Q-function est correctement définie, on peut calculer la fonction pour toutes les actions possibles à l'état s_t et sélectionner l'action a_t donnant la valeur maximale pour $Q(s_t, a_t)$

Intérêt de la Q-function

Intérêt

L'agent doit apprendre une politique π qui guidera ses mouvements afin de toujours choisir la meilleure action a_t à réaliser donné son état courant s_t

On peut apprendre cette politique π en maximisant la future récompense que l'agent est susceptible d'obtenir :

$$\pi^*(s) = \underset{a}{argmax}(Q(s, a))$$

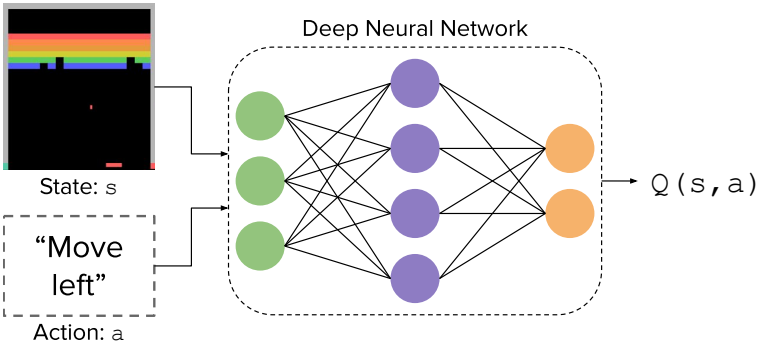
Deep Q Networks

DQN, un algorithme de deep learning pour apprendre une Q-function

Utiliser un réseau de neurone pour modéliser la Q-function

Idée initiale

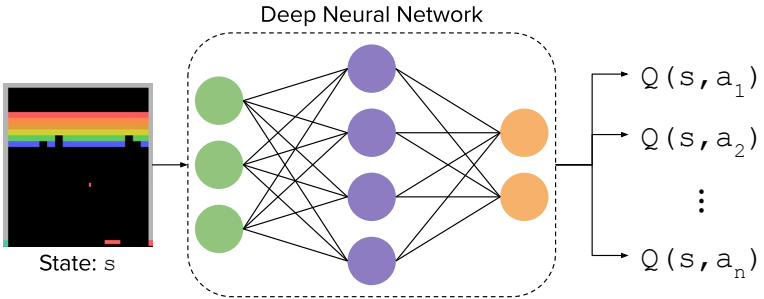
Pour un environnement complexe il est impossible de définir la Q-function à la main, on peut donc utiliser un réseau de neurones pour qu'il l'apprenne et la calcule à notre place



Utiliser un réseau de neurone pour modéliser la *Q*-function

Idée plus efficiente

Pour un temps de calcul plus court on peut apprendre au modèle à estimer la *Q*-function pour toutes les actions possibles d'un seul coup plutôt que de les passer une à une



Entraîner un DQN

Définir une fonction de loss

On cherche à trouver la Q-function définie comme suit (équation de Bellman) :

$$Q(s_t, a_t) = r_t + \gamma Q(s_{t+1}, a_{t+1})$$

Entraîner un DQN

Définir une fonction de loss

On cherche à trouver la Q-function définie comme suit (équation de Bellman) :

$$Q(s_t, a_t) = r_t + \gamma Q(s_{t+1}, a_{t+1})$$

Ce qui revient à résoudre l'équation suivante :

$$r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) = 0$$

Entraîner un DQN

Définir une fonction de loss

On cherche à trouver la Q-function définie comme suit (équation de Bellman) :

$$Q(s_t, a_t) = r_t + \gamma Q(s_{t+1}, a_{t+1})$$

Ce qui revient à résoudre l'équation suivante :

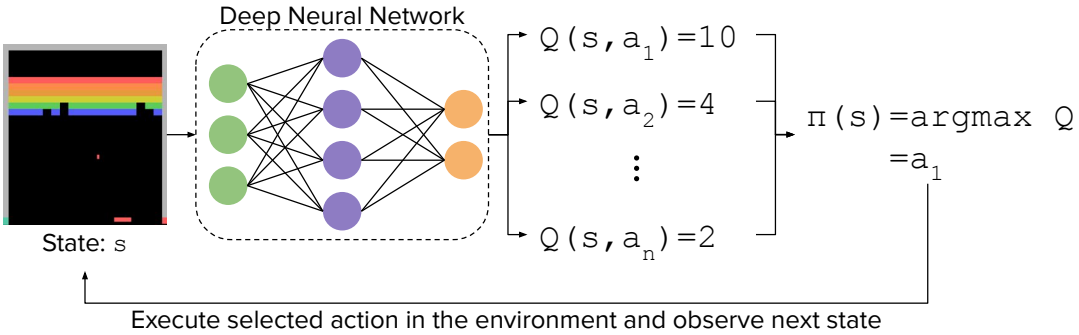
$$r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) = 0$$

On ne peut pas la calculer mais en peut l'approximer en minimisant l'équation suivante :

$$\mathcal{L} = || \overbrace{(r_t + \gamma Q(s_{t+1}, a_{t+1}))}^{target} - \overbrace{Q(s_t, a_t)}^{prediction} ||_2^2$$

La fonction de loss ci-dessus est appelé la Q-loss

Récapitulatif de l'exécution d'un DQN



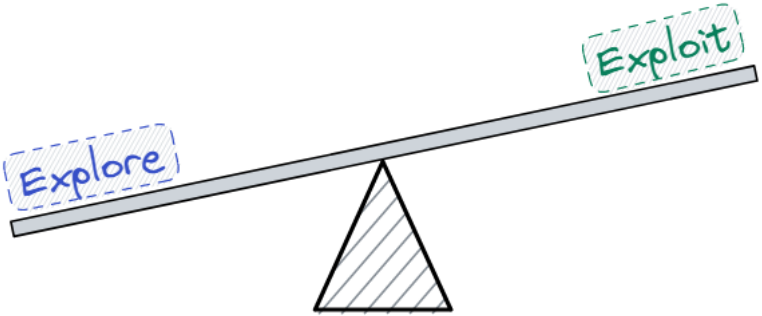
DQN en pratique

DQN, détails techniques pour l'implémentation

Exploration vs exploitation

Équilibre entre exploration et exploitation de l'environnement

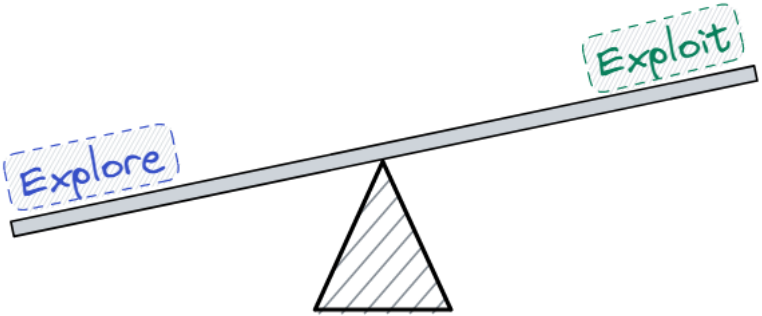
- Une action d'exploration est une action aléatoire



Exploration vs exploitation

Équilibre entre exploration et exploitation de l'environnement

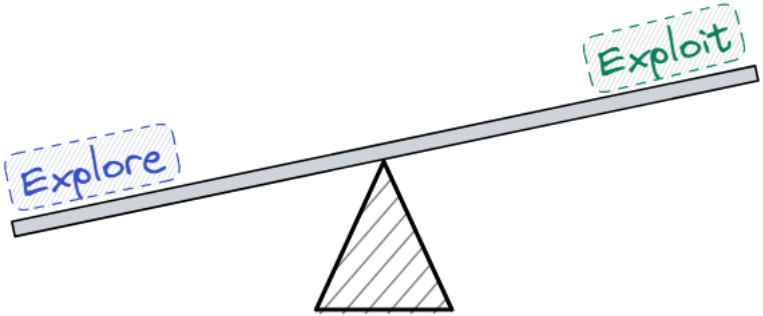
- Une action d'exploration est une action aléatoire
- Une action d'exploitation est l'action que l'agent souhaite effectuer en réponse à son observation



Exploration vs exploitation

Équilibre entre exploration et exploitation de l'environnement

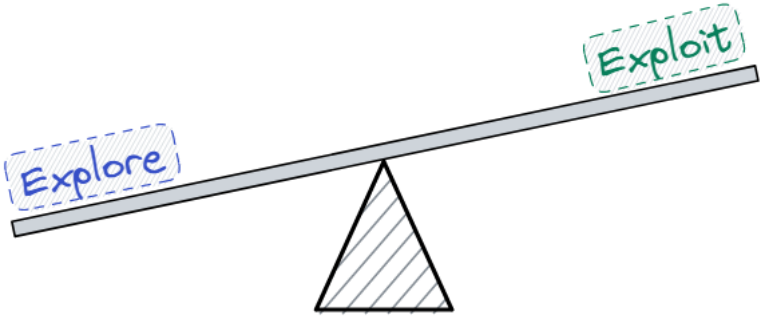
- Une action d'exploration est une action aléatoire
- Une action d'exploitation est l'action que l'agent souhaite effectuer en réponse à son observation
- Un agent ne pourra pas apprendre en réalisant uniquement de l'exploration ou de l'exploitation



Exploration vs exploitation

Équilibre entre exploration et exploitation de l'environnement

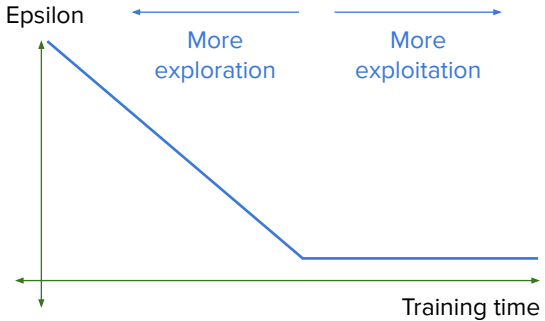
- Une action d'exploration est une action aléatoire
- Une action d'exploitation est l'action que l'agent souhaite effectuer en réponse à son observation
- Un agent ne pourra pas apprendre en réalisant uniquement de l'exploration ou de l'exploitation
- Il est donc important de définir un équilibre entre les deux évoluant au cours de l'apprentissage



Exploration vs exploitation

Équilibre entre exploration et exploitation de l'environnement

- Une action d'exploration est une action aléatoire
- Une action d'exploitation est l'action que l'agent souhaite effectuer en réponse à son observation
- Un agent ne pourra pas apprendre en réalisant uniquement de l'exploration ou de l'exploitation
- Il est donc important de définir un équilibre entre les deux évoluant au cours de l'apprentissage



Replay memory

Replay memory

- Il s'agit simplement d'une mémoire des expériences passées entre l'agent et l'environnement

Replay memory

Replay memory

- Il s'agit simplement d'une mémoire des expériences passées entre l'agent et l'environnement
- L'intérêt de cette mémoire est de permettre l'entraînement de l'agent sur de nombreuses expériences passées dont on connaît le résultat

Replay memory

Replay memory

- Il s'agit simplement d'une mémoire des expériences passées entre l'agent et l'environnement
- L'intérêt de cette mémoire est de permettre l'entraînement de l'agent sur de nombreuses expériences passées dont on connaît le résultat
- Elle permet d'accélérer l'apprentissage tout en améliorant les performances de l'agent

Pseudocode DQN

Pseudocode DQN train for one episode

1. Initialize environement and get first state

Pseudocode DQN

Pseudocode DQN train for one episode

1. Initialize environement and get first state
2. Loop until the end of the episode :

Pseudocode DQN

Pseudocode DQN train for one episode

1. Initialize environement and get first state
2. Loop until the end of the episode :
3. With probability ϵ select a random action,
   Otherwise select an action depending on model output

Pseudocode DQN

Pseudocode DQN train for one episode

1. Initialize environement and get first state
2. Loop until the end of the episode :
3. With probability ϵ select a random action,
   Otherwise select an action depending on model output
4. Execute selected action in environment and get next state and reward

Pseudocode DQN

Pseudocode DQN train for one episode

1. Initialize environement and get first state
2. Loop until the end of the episode :
3. With probability ϵ select a random action,
   Otherwise select an action depending on model output
4. Execute selected action in environment and get next state and reward
5. Store current transition (state, action, next state and reward) in replay memory

Pseudocode DQN

Pseudocode DQN train for one episode

1. Initialize environement and get first state
2. Loop until the end of the episode :
3. With probability ϵ select a random action,
   Otherwise select an action depending on model output
4. Execute selected action in environment and get next state and reward
5. Store current transition (state, action, next state and reward) in replay memory
6. Extract 1 minibatch from the replay memory

Pseudocode DQN

Pseudocode DQN train for one episode

1. Initialize environement and get first state
2. Loop until the end of the episode :
3. With probability ϵ select a random action,
   Otherwise select an action depending on model output
4. Execute selected action in environment and get next state and reward
5. Store current transition (state, action, next state and reward) in replay memory
6. Extract 1 minibatch from the replay memory
7. Perform optimization on the model on this minibatch using Q-loss