Syllabus projet

Année : **2023-2024**

Enseignant(s)	Email(s)
VIDAL Nicolas	nvidal@myges.fr

2024-4A-IABD - (Deep) Reinforcement Learning P1

1 Matières, formations et groupes

Matière liée au projet : **S2 - deep reinforcement learning**

Formations : -

Nombre d'étudiant par groupe : **3 à 4**

Règles de constitution des groupes: **Imposé**

Charge de travail estimée par étudiant : **15,00 h**

2 Sujet(s) du projet

Type de sujet : **Imposé**

3 Détails du projet

Objectif du projet (à la fin du projet les étudiants sauront réaliser un...)

> Implémenter et comprendre quand utiliser les algorithmes classiques de l'apprentissage par renforcement, qu'ils soient issus de la programmation dynamique ou de techniques à base de simulations Monte Carlo, ou de TD Learning.

Descriptif détaillé

Dans ce projet, les étudiants devront écrire :
- une bibliothèque dans laquelle un certain nombre d'algorithmes d'apprentissage par renforcement sont développés
- une bibliothèque dans laquelle un certain nombre d'environnements devront être développés

A l'aide de ces deux bibliothèques, les étudiants devront mener des expérimentations (test de tous les algorithmes sur tous les environnements et études de l'impact des hyper paramètres) afin de déterminer quel algorithme parait le plus judicieux sur quel environnement et pourquoi.

Les étudiants devront également mener des expérimentations sur des environnements 'mystères' fournis afin d'essayer de trouver la meilleure stratégie possible sur ces derniers.

Algorithmes à implémenter :
- Dynamic Programming
  - Policy Iteration
  - Value Iteration

- Méthodes Monte Carlo
  - Monte Carlo ES
  - On-policy first visit Monte Carlo Control
  - Off-policy Monte Carlo Control

- Temporal Difference Learning
  - Sarsa
  - Q-Learning
  - Optionnel : Expected Sarsa

- Planning
  - Dyna-Q
  - Optionnel : Implémentation de l'algorithme "Dyna-Q+"

Environnements à développer :

- Line World

- Grid World

- Two round Rock Paper Scissors (description ci-dessous) :
  > L'agent fait une partie constituée de 2 rounds de Pierre Feuille Ciseaux face à un adversaire qui joue de manière particulière, ce dernier joue aléatoirement au premier round, mais au deuxième round, il joue FORCEMENT le choix de l'agent au premier round. Chaque round peut être gagné (+1 reward) perdu (-1 reward) ou sans vainqueur (0 reward).

- Monty Hall "paradox" level 1 (description ci-dessous) :
  > L'agent est un candidat au jeu Monty Hall, Il doit prendre 2 décisions successives. Dans cet environnement, il y a 3 portes A, B et C. Au démarrage de l'environnement, une porte est tirée au hasard de manière cachée pour l'agent, il s'agit de la porte gagnante. La première action de l'agent est de choisir une porte parmi les trois portes. Ensuite, une porte parmi les 2 restantes non choisies par l'agent est retirée du jeu, il s'agit forcément d'une porte non gagnante. L'agent ensuite doit effectuer une nouvelle action : choisir de conserver la porte choisie au départ ou changer pour la porte restante. Une fois le choix fait, la porte choisie est 'ouverte' et l'on découvre si elle était gagnante (reward de 1.0) ou non (reward de 0.0).

- Monty Hall "paradox" level 2 (description ci-dessous)
  > Il s'agit du même environnement que précédemment, seulement maintenant 5 portes sont disponibles, et l'agent doit effectuer 4 actions successives avant l'ouverture d'une des deux portes restantes.

Environnements sur lesquels essayer de trouver la meilleure stratégie possible avec chaque algorithme (quand possible) :

- Secret Env 0

- Secret Env 1
- Secret Env 2

Pour chacun des environnements, il devra être possible de visualiser de manière 'agréable' l'état de l'environnement résultant de chaque action réalisée (interface graphique, pretty print sur command line, etc...).

Après apprentissage et convergence vers une stratégie de jeu, il devra être possible de la dérouler 'pas à pas' sans avoir à relancer l'apprentissage (notamment pour la soutenance).

Il devra aussi être possible de pouvoir agir 'manuellement' (agent humain) sur chaque environnement pour pouvoir s'assurer du bon respect des règles de ces derniers.

L'interface graphique des environnements *secrets* ne sera fournie qu'en fin de cours.

Pour chaque environnement, les étudiants devront étudier les performances des algorithmes et retranscrire leurs résultats dans un rapport.

Les étudiants devront fournir l'intégralité du code leur ayant permis d'obtenir leurs résultats ainsi que les policy, value functions et action value functions entraînées et sauvegardées prêtes à être exécutées pour confirmer les résultats présentés.

Les étudiants devront présenter ces résultats lors d'une soutenance. Dans cette dernière, les étudiants devront faire valoir leur méthodologie de choix d'hyperparamètres, et proposer leur interprétation des résultats obtenus

Ouvrages de référence (livres, articles, revues, sites web...)

Reinforcement Learning: An Introduction :
http://incompleteideas.net/book/the-book.html
Richard S. Sutton
and Andrew G. Barto

Outils informatiques à installer

Pour la visualisation Unity / Unreal Engine / PyGame / command line / ... ?

4 Livrables et étapes de suivi

1	Rendu final	Présentation des résultats et du protocole suivi pour les environnements de (Deep) Reinforcement Learning Livrables : - rapport sous la forme de Notebook Jupyter + pdf - présentation (slides) de 10 minutes - sources complètes du projet - démonstration	**mercredi 24/07/2024 23h59**

5 Soutenance

Durée de présentation par groupe : **15 min** Audience : **A huis clos**

Type de présentation : **Présentation / PowerPoint - Démonstration**

Précisions :