Projet informatique: entraîner un agent à jouer au jeu de Snake avec l'apprentissage par renforcement

Yannick Terme

May 7, 2018

Introduction

Un domaine de l'intelligence artificielle a fait beaucoup parler récemment: l'apprentissage par renforcement. Cet ensemble de méthodes est notamment derrière des projets tels que AlphaGo, le programme de Google Deepmind qui a battu les meilleurs joueurs de jeu de Go au monde. Les méthodes qui été utilisés par les équipes de Deepmind présentent l'avantage d'être réutilisables d'un jeu à l'autre. En effet, ils les réutilisent pour plusieurs jeux Atari [1]. Pour ma part, j'ai décidé de m'attaquer au jeu du Snake, ce jeu où l'utilisateur dirige un serpent. L'utilisateur gagne des points lorsque le serpent mange une pomme et perd la partie lorsque le serpent se mord la queue ou rentre dans un mur. A chaque pomme mangée, le serpent grandit d'une case. Le jeu, popularisé en 1998 par Nokia lorsqu'il fut intégré dans un de ses téléphones, est très simple. Il est assez facile de créer un algorithme déterministe qui obtienne de bons scores et même qui gagne le jeu, c'est-à-dire qui réussisse à remplir l'écran avec la queue du serpent. Le but de ce projet n'est donc pas d'élargir le domaine des possibles de l'IA mais plutôt de mieux maîtriser les concepts de l'apprentissage par renforcement, de les implémenter et de les faire fonctionner sur cet exemple.

Les étapes du projet

Documentation Dans un premier temps, j'ai dû me documenter sur l'apprentissage par renforcement et plus particulièrement ses applications à des jeux. L'article de Deepmind explique la procédure d'apprentissage d'un algorithme par renforcement pour les jeux; je me suis donc appuyé. Je me suis aussi aidé d'un blog qui explique en détail certaines parties de ce papier [2] ainsi que d'un étudiant qui a également essayé de résoudre le jeu de Snake [3]. Lors de mes recherches sur Internet, je me suis rendu compte qu'il fallait lire les articles de recherche lorsqu'on veut vraiment comprendre le fond d'un problème.

Implémentation du Snake Ensuite, j'ai implémenté le jeu du Snake. Cette partie était relativement rapide. J'ai repris une structure proche de celle du package Python Gym AI qui propose des interfaces Python pour plusieurs jeux et dont le but est d'entraîner des algorithmes de Machine Learning sur ces jeux. Le jeu du Snake y était déjà implémenté. Cependant j'ai préféré développer le mien pour pouvoir modifier la taille de l'écran de jeu.

Implémentation de l'algorithme J'ai ensuite implémenté l'algorithme qui dirige le serpent. Cette partie est bien entendu celle qui comporte le plus de difficultés. J'ai commencé par développer un simple algorithme déterministe qui fera office de benchmark. A chaque étape, on communique à cet algorithme les coordonnées de la pomme, la tête du serpent et du corps du serpent. L'algorithme prend la direction du plus court chemin entre la tête et la pomme. Cet algorithme fonctionne bien tant que le serpent n'est pas très grand et il ressemble à la stratégie qu'un humain ferait. Ensuite, j'ai implémenté l'algorithme qui utilise du machine Learning. Dans un premier temps, j'ai voulu tout de suite utiliser le réseau de neurone convolutif (CNN) à 5 couches proposé par Deepmind. Cependant, n'ayant pas d'expérience en Deep Reinforcement Learning, mon algorithme

[1] https://deepmind.com/research/publications/playing-atari-deep-reinforcement-learning/
[2] http://neuro.cs.ut.ee/demystifying-deep-reinforcement-learning/
[3] https://danielegrattarola.github.io/files/projects/2016_grattarola_snake.pdf

ne réussissait pas à converger. J'ai alors revu ma stratégie et décidé de m'y prendre étape par étape. J'ai donc diminué la résolution du jeu et suis passé à une carte de taille 5x5 pixels et ai utilisé un réseau de neurones plus simple à 3 couches. De plus, je me suis replongé plus rigoureusement dans la lecture des articles de recherche ce qui m'a permis de rendre compte que j'avais mal implémenté certaines méthodes.
A la suite de cela, j'ai réussi à entraîner un agent qui obtienne des bons scores sur un jeu de taille 5x5 pixels. L'agent réussissait à manger en moyenne 10 à 11 pommes par partie. Le but ultime serait d'avoir un agent qui puisse résoudre totalement le jeu c'est-à-dire qui mange 24 pommes pour un jeu de taille 5x5.

Remarques générales sur le déroulement du projet

- Au fur et à mesure que le projet avance et que notre quantité de code augmente, il m'a semblé important d'être organisé dans la manière de passer à l'échelle. J'ai commencé par un simple Jupyter Notebook qui a l'avantage d'être pratique à manipuler. Mais il faut ensuite vite passer à une organisation du code en fichier python. Ensuite lorsque j'ai commencé à avoir plusieurs modèles et plusieurs paramètres, j'ai commencé à perdre des résultats car je n'avais pas automatisé le stockage des résultats? J'ai donc créé une classe Python qui stocke les résultats en mémoire ainsi qu'un parseur qui permette d'appeler les fonctions depuis la ligne de commande. A terme, on a donc un système où toutes les commandes sont effectuées depuis la ligne de commande.

- Lorsque j'ai voulu essayer plusieurs modèles et plusieurs valeurs d'hyper-paramètres, j'ai eu besoin de plusieurs machines. J'ai ainsi profité de mon accès aux serveurs de Télécom ParisTech. Pour cela, j'ai donc mis le code sur GitHub.