Apprentissage par renforcement (1a/3)

Bruno Bouzy

23 septembre 2014

Ce document est le chapitre « Apprentissage par renforcement » du cours d'apprentissage automatique donné aux étudiants de Master MI, parcours Informatique. Il est basé sur les parties I et II de (Sutton & Barto 1998). Les figures contenues dans ce document sont directement reprises de la version html du livre de Sutton & Barto en ligne sur le web.

Introduction

Généralités

L'Apprentissage par Renforcement (AR) consiste à apprendre quoi faire, comment associer des actions à des situations, afin de maximiser quantitativement une récompense. On ne dit pas à l'apprenant quelle action faire, mais au lieu de cela, il doit découvrir quelles actions donnent le plus de récompense en les essayant. Dans le cas le plus intéressant, des actions peuvent affecter non seulement les récompenses immédiates mais aussi la situation suivante, et par là, les récompenses à plus long terme. Ces deux propriétés – *recherche par essai-erreur* et *récompense à long terme* – sont les deux caractéristiques les plus importantes de l'apprentissage par renforcement.

(L'apprentissage par renforcement se définit plus par le type de problème auquel il s'applique et moins par le type de la méthode d'apprentissage.)

L'apprentissage par renforcement est différent de l'apprentissage supervisé. Ce dernier nécessite un superviseur qui dit à l'agent quelle action est correcte dans telle situation. Dans l'AR, l'agent n'a pas d'oracle à sa disposition, il *interagit avec l'environnement* qui lui donne un retour quantitatif sur les valeurs de ses actions.

L'un des défis de l'AR est le compromis à trouver entre *exploration* et *exploitation*. Pour obtenir beaucoup de récompense, un agent AR doit préférer des actions qu'il a essayées avant et trouvées être efficaces par les récompenses qu'elles ont entraîné. L'agent doit *exploiter* ce qu'il connaît déjà pour obtenir une récompense, mais il doit aussi *explorer* pour faire de meilleure actions dans le futur. Le dilemme est que ni l'exploration ni l'exploitation ne peuvent être poursuivie exclusivement sans faillir à la tâche.

Une autre clé de l'AR est de considérer le problème *dans son ensemble*, par contraste avec d'autres méthodes qui découpent le problème à résoudre en sous-problème et en perdant la vision globale. Dans l'AR, les agents ont des *buts* qu'ils cherchent à atteindre.

Exemples

- Un joueur d'Echecs.

- Un contrôleur adaptatif qui ajuste des paramètres en temps réel.
- Une gazelle nouveau-né qui galope à 30km/h une demi-heure après sa naissance.
- Lucas qui prépare son petit déjeuner.

Elements de base de l'AR

En plus de l'agent et de l'environnement, on peut identifier trois éléments principaux d'un système d'AR : la politique, la fonction de récompense, la fonction de valeur[1]. La *politique* définit la manière de se comporter de l'agent à un instant donné, c'est-à-dire quelle action effectuer à cet instant. La *fonction de récompense* correspond au but du problème considéré. Elle associe chaque état (ou paire état-action) de l'environnement à un nombre, la récompense. Elle définit quels sont les événements mauvais et les événements bons pour l'agent. La *fonction de valeur* spécifie ce qui est bon à long terme. Pour parler approximativement, la valeur d'un état est la quantité totale de récompense qu'un agent peut s'attendre à accumuler dans le futur en partant de cet état.

Feed-back quantitatif

La caractéristique distinguant le plus l'AR des autres méthodes d'apprentissage est le fait d'utiliser une évaluation des actions effectuées dans une situation plutôt que d'utiliser une instruction venant d'un oracle disant quelle action est correcte dans une situation. C'est ce qui oblige l'agent à explorer pour rechercher le bon comportement par essai-erreur.

Le problème du « N-armed bandit », la méthode gloutonne

Supposons que l'on soit en face d'un choix de n actions. Après un choix on reçoit une récompense sous la forme d'un nombre choisi dans un distribution de probabilité stationnaire qui dépend de l'action choisie. La récompense est non déterministe. Cela signifie que si l'on exécute deux fois l'action dans les mêmes conditions, la récompense peut ne pas être la même. On veut maximiser la récompense totale obtenue sur le temps, c'est-à-dire par exemple au bout de 1000 actions. Chaque sélection d'action s'appelle un jeu. Ceci est la forme originale du problème du « N-armed bandit[2] ».

Dans ce problème, chaque action a une récompense moyenne qui est reçue lorsque l'action est choisie. Nous l'appelons la valeur de l'action. Si on connaissait les valeurs des actions, il serait facile de jouer au N-Armed Bandit Problem (NABP). On sélectionnerait toujours l'action avec la plus forte valeur. On suppose que l'on ne connaît pas les valeurs des actions avec certitude mais que l'on a des estimations. Si l'on maintient des estimations des valeurs d'actions, on peut toujours sélectionner l'action dont l'estimation est la plus grande, que nous appelons l'action « gloutonne[3] ». Si on sélection l'action gloutonne, on dit que l'on exploite nos connaissances actuelles. Si l'on ne sélectionne pas l'action gloutonne, on dit que l'on explore car cela permet d'améliorer nos estimations de valeur d'action. Il existe beaucoup de théories sur la balance exploration/exploitation. Dans la suite, nous allons montrer des balances simples entre exploration et exploitation qui marchent mieux que des méthodes basées sur l'exploitation seule.

---

[1] « value fonction » en anglais.
[2] Machine à sous à N leviers.
[3] « greedy » en anglais.

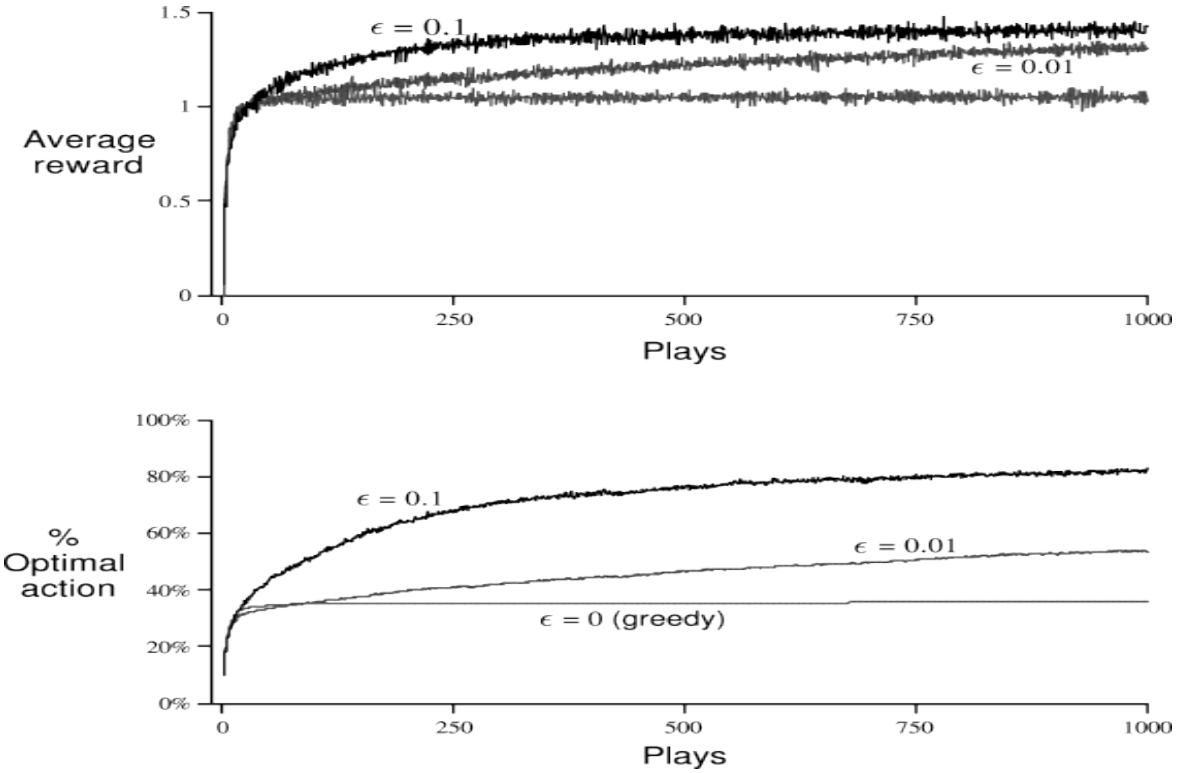

Figure 1 : La courbe du haut montre la récompense moyenne obtenue par les méthodes gloutonnes et ε-gloutonnes. La courbe du bas montre le pourcentage de fois pour lesquelles la méthode a sélectionné la meilleure action.

Les méthodes par valeur d'actions, les méthodes ε–gloutonnes

Dans cette partie nous appelons $Q^*(a)$ la valeur théorique de l'action a, et $Q_t(a)$ la valeur estimée au $t^{ième}$ jeu. Rappelons que la vraie valeur d'une action est la valeur moyenne des récompenses reçues lorsque cette action est sélectionnée. On peut donc estimer une valeur d'action avec la moyenne des récompenses réelles effectivement reçues quand l'action a été sélectionnée. Autrement dit, si au $t^{ième}$ jeu, l'action a été choisie k_a fois donnant les récompenses $r_1, r_2, \ldots, r_{ka}$, alors on a :

$$Q_{ka} = Q_t(a) = (r_1 + r_2 + \ldots + r_{ka})/k_a \qquad (1)$$

Si ka=0, on définit $Q_t(a)$ avec une valeur par défaut $Q_0(a) = 0$. Quand ka -> ∞, la loi des grands nombres fait que $Q_t(a)$ -> $Q^*(a)$. Cela s'appelle la technique de l'échantillonnage moyen pour estimer les valeurs d'action. C'est juste une méthode, mais pas nécessairement la meilleure.

La règle de sélection d'action la plus simple est de sélectionner l'action avec la plus forte valeur, c'est-à-dire une action gloutonne a^* pour laquelle $Q_t(a^*) = \max_a Q_t(a)$. Cette méthode exploite la connaissance courante pour maximiser les récompenses. Une alternative simple est de se comporter de manière gloutonne la plupart du temps et de temps en temps mais régulièrement, avec une probabilité ε, sélectionner une action au hasard, indépendamment des estimations. Nous appelons ce type de méthode une méthode ε–gloutonne. Un avantage de ces

méthodes est que toutes la actions seront sélectionnées un nombre très grand de fois, garantissant que pour toute action a, $Q_t(a) \rightarrow Q^*(a)$. Cela entraîne que la probabilité de sélection de l'action optimale converge vers un nombre plus grand que 1-ε, c'est-à-dire pas loin de la certitude.

Pour estimer approximativement l'efficacité relative de ces méthodes, gloutonne et ε–gloutonne, nous les avons comparé sur des problèmes de test. C'est un ensemble de 2,000 problèmes nAB avec n=10. Pour chaque action a, les récompenses étaient sélectionnées à partir d'une distribution de probabilité gaussienne de moyenne $Q^*(a)$ et de variance 1. Les 2,000 nAB tâches ont été engendrées en re-sélectionnant les $Q^*(a)$ 2,000 fois, à chaque fois avec une distribution de probabilité gaussienne de moyenne 0 et de variance 1. En moyennant sur les tâches, on peut construire la courbe de performance et de comportement des méthodes au fur et à mesure de leur amélioration sur 1000 jeux, comme sur la figure 1. On appelle cet ensemble de tâches de test, le jeu de tests à 10 leviers.

La figure 1 compare une méthode gloutonne avec deux méthodes ε-gloutonnes ($\varepsilon = 0.1$ et $\varepsilon = 0.01$). Toutes les méthodes ont estimé les valeurs d'action avec la technique de l'échantillonnage moyen. Au tout début, la méthode gloutonne s'améliore légèrement plus vite que les méthodes ε-gloutonnes, mais reste ensuite bloquée à un niveau plus bas que celui des méthodes ε-gloutonnes. La méthode gloutonne obtient un niveau de récompense moyen par jeu qui vaut 1, à comparer à 1.55 le niveau maximal que l'on peut obtenir sur ce jeu de tests. La courbe du bas montre que la méthode gloutonne ne sélectionne l'action optimale que un tiers de fois. Dans les deux autres tiers, les échantillonnages initiaux étaient décevants, la méthode ne s'en relève pas. Les deux méthodes ε-gloutonnes marchent éventuellement mieux car elles continuent d'explorer, et augmentent leurs chances de trouver l'action optimale. La méthode avec $\varepsilon = 0.1$ explore plus mais ne peut mieux faire que 91%. . La méthode avec $\varepsilon = 0.01$ s'améliore plus lentement, mais peut éventuellement marcher mieux et atteindre 99%. Il serait possible de faire une méthode avec ε grand au début et diminuant ensuite.

L'avantage des méthodes ε-gloutonnes sur la méthode gloutonne dépend de la tâche. Si la variance avait été plus grande, disons 10, cela aurait pris plus de temps pour trouver l'action optimale, et les méthodes ε-gloutonnes se seraient encore mieux comportées. A l'inverse, si la variance étaient nulle, la méthode gloutonne serait la meilleure. Avec une variance nulle mais avec des valeurs d'action non-stationnaires, les méthodes ε-gloutonnes seraient meilleures que la méthode gloutonne.

La méthode softmax

Bien que très populaire en AR, le désavantage des méthodes ε-gloutonnes est que, lorsque elles explorent, elles sélectionnent l'action au hasard. Dans des tâches où effectuer une action mauvaise est très pénalisant, les méthodes ε-gloutonnes ne peuvent pas obtenir de bons résultats. Pour améliorer l'exploration, on peut donc utiliser une méthode qui sélectionne les actions proches de la meilleure. Par exemple, une méthode de sélection simple qui sélectionne les actions proches de la meilleure est celle qui choisit l'action a avec une probabilité :

$$Q_t(a)/\Sigma_b Q_t(b) \qquad (2)$$

La méthode « softmax » est une amélioration de cette méthode ; elle choisit l'action a avec une probabilité :

$$\exp(Q_t(a)/\tau)/\sum_b \exp(Q_t(b)/\tau) \qquad (3)$$

Le paramètre τ s'appelle la température et permet de jouer sur le déterminisme de la méthode et sur la balance entre exploitation et exploration. Avec une température très grande, toutes les exponentielles valent 1, la méthode de sélection suit une distribution de probabilité uniforme, et la méthode est uniquement exploratoire. Avec une température proche de 0, l'action la meilleure est sélectionnée très souvent et la méthode favorise l'exploitation. A la limite, quand τ->0, la méthode est totalement déterministe avec aucune exploration. Savoir laquelle de softmax ou de ε-gloutonne est la meilleure dépend de la tâche. Les deux méthodes dépendent d'un paramètre, τ ou ε.

La méthode UCB

La méthode UCB sélectionne le bras b_{ucb} avec la formule :

$$b_{ucb} = \text{argmax}_{\ b \text{ in } B}\ \ m\hat{}(b) + C \text{ sqrt } (\log(T)/N(b)) \qquad (4)$$

m^(b) est la moyenne empirique du bras b. T est le nombre total d'essais sur tous les bras. N(b) est le nombre d'essais du bras b. Le premier terme de (4) correspond à *l'exploitation*: utiliser la moyenne empirique. Le second terme de (4) correspond à l'*exploration*. Plus faible est N(b), plus grande est l'exploration. Au temps t, si un bras b n'est pas assez exploré, son terme d'exploration est grand, et b est choisi. Si b est un bon bras, sa moyenne augmente et b sera sélectionné à nouveau. Tant que b est sélectionné, son terme d'exploration diminue. Pendant ce temps, la moyenne empirique de ce bras converge vers sa valeur théorique, et le terme d'exploration des autres bras augmente - avec le terme log(T). Donc, à un certain moment, un bras différent de b sera sélectionné. Un bras ne peut donc rester inexploré. Tous les bras seront visités une infinité de fois et leur moyenne empirique convergera vers leur moyenne théorique.

UCB signifie « Upper Confidence Bound ». Cela traduit que la valeur UCB est la borne supérieure d'un intervalle de confiance autour de la moyenne empirique.

Evaluation vs Instruction ; Renforcement vs Apprentissage Supervisé.

Dans le problème de nAB, le retour de l'approche par AR est purement « quantitatif ». Il donne une « évaluation » de l'action. Il ne dit rien sur le fait que l'action est « correcte » ou pas dans la situation considérée. Cela contraste avec l'apprentissage supervisé qui donne un retour « instructif » en disant si l'action est correcte ou pas. Pour expliquer la nature de la différence entre « évaluation » (propre à l'AR) et « instruction » (propre à l'apprentissage supervisé), considérons l'exemple suivant. Supposons qu'il existe 100 actions possibles. On sélectionne l'action 32 et dans de cas de l'approche AR, l'environnement nous informe qu'elle vaut 7.2. Dans le cas de l'apprentissage supervisé, un oracle nous dit que l'action 32 n'est pas correcte et que l'action 67 aurait été correcte.

Incrémentalité

Les méthodes de valeurs d'action définies avant estiment des valeurs d'action avec la technique de l'échantillonnage moyen. Pour cela on peut utiliser la formule 1, mais on peut aussi utiliser une formule incrémentale :

$$Q_{k+1} = Q_k + (r_{k+1} - Q_k)/(k+1) \qquad (5)$$

Les formules 4 et 5 sont caractéristiques de la mise à jour utilisée en AR. En général la mise à jour en AR utilise une formule de la forme suivante :

$$NelleEstim = AncEstim + Pas \text{ x } (Cible - AncEstim) \quad (6)$$

(*Cible – AncEstim*) correspond à une erreur de l'estimation. Pas est le pas de l'apprentissage. Dans la formule 5, il valait 1/k. On le note souvent α.