Rapport du projet Reinforcement Learning

Raphaël Duprat
Philippe Gnansounou

Contents

1 Abstract

Ce projet vise à appliquer une méthode d'apprentissage par renforcement sur un jeu de poker simplifié afin d'apprendre le jeu à deux intelligence artificielles A et B. Notre objectif à terme et de faire en sorte que les IAs apprennent à jouer intelligemment et à bluffer pour maximiser leur gain.

Nous avons donc choisit d'implémenter en C++ la méthode Linear Reward Inaction qui a suffit à faire apparaître le bluff et le contre-bluff, montrant l'efficacité de la méthode. Cependant, nous avons également pu observer des failles de celle ci qui seront élaborées dans l'analyse des résultats en 4.3.

2 Le Renforcement Learning

2.1 Domaine d'utilisation

Le Renforcement Learning est une technique d'apprentissage automatique. Le principe consiste à laisser l'IA (ici le joueur) apprendre de son expérience, c'est-à-dire de ses victoires et défaites. Dans notre approche, nous partons d'une probabilité associée à chaque mouvement possible que l'on augmente en cas de victoire ou que l'on baisse en cas de défaite (récompense/pénalité).

Il est souvent utilisé lorsque l'on veut accomplir un but précis avec un large champ de possibilités, trop large pour être entièrement intégré à la machine, comme apprendre à un robot à marcher ou lui apprendre à jouer à un jeu.

2.2 Examples notable

Le Renforcement Learning a été popularisé ces dernières années par son utilisation en robotique ou encore dans des jeux à deux joueurs comme le jeu de Go, dominé par « Alpha Go » en 2018. Le succès de la méthode est tel que les joueurs professionnel sont tout simplement incapables de gagner contre de telles machines. L'approche permet également de tester et d'envisager de nouvelles stratégies, au Go comme aux échecs. Par exemple, aux échecs, on accorde traditionnellement une valeur à chaque pièce afin de guider les échanges, mais les IAs ont tendance à accepter la perte d'une pièce pour obtenir une position avantageuse qui paie par la suite, ce qui est fortement éloigné des tactiques usuelles. Au Go, des coups parfois considérés comme mauvais sont joués par les IAs et cela se révèle par la suite un coup décisif dans la partie. La méthode est donc tout à fait pertinente pour notre problème.

3 Les méthodes envisagées

3.1 LRI

Nous avons envisagé les méthodes du "Q-learning", du "Linear Reward Inaction" (LRI), ainsi que "State–action–reward–state–action" (SARSA), et avons décidé de nous concentrer dans un premier temps sur le LRI qui nous a semblé adapté à un jeu comme le poker, c'est-à-dire un jeu stochastique à deux personnes sans somme et à information incomplète. Cet algorithme consiste à récompenser une stratégie si elle a été sélectionnée précédemment et qu'elle a amené une victoire.
La récompense suit une fonction linéaire établie par la relation suivante :

$$p_i = p_i + \alpha(1 - p_i)$$

$$p_j = pj - \alpha(p_j)$$

avec pi la probabilité de choisir l'action i, pj la probabilité de choisir l'action opposée j, et α la vitesse de convergence (plus α est petit, plus l'algorithme converge lentement et plus la probabilité de converger vers de mauvais choix est faible, au prix d'itérations plus nombreuses) . Naturellement, nous avons du adapter notre algorithme pour que la probabilité soit soustraite de toutes les actions autres que celle récompensée, ceci afin que la somme des probabilités reste à 1.

Chaque action aura au départ la même probabilité, l'IA agira au début de façon aléatoire. Puis après un certain temps, lorsque l'IA aura exploré son champ de possibilités au travers de milliers de parties, les probabilités de choisir différentes actions convergent vers celles qui apportent une récompense optimale.
l'IA va alors agir logiquement en favorisant de miser sur les cartes à forte valeur. A noter que les joueurs ne connaissent pas les possibilités du joueur adverse.

Dans notre cas, en fonction de la carte tiré par le joueur, si l'action qu'il a choisie était la bonne l'on va augmenter la probabilité de cette action. Un exemple simple : si A tire un 2 et B tire un 7, si A décide de se coucher, l'on va considérer que c'était la meilleure action possible dans ce cas et favoriser cette action.

Mais comment « favoriser » ? Chaque joueur peut tirer une carte de 1 à 10 et le nombre d'action possible pour les joueurs est définie. On commence par donner la même probabilité à chaque action possible aux deux joueurs pour chaque carte tirée. Par la suite, à chaque fin de partie, en fonction de la carte tiré et du résultat obtenu, on va augmenter une de ces probabilités et en baisser une autre -en suivant l'équation de la méthode LRI- pour que la somme reste à 100.

3.2 LRP

La méthode LRP est une méthode ressemblent à la LRI. Là où la LRI favorise seulement les actions du joueur ayant conduit à un gain, la LRP défavorise également les actions ayant mené à une perte.

4 Application au poker simplifié

4.1 Présentation

Le projet consiste à programmer un jeu de poker simplifié et d'y appliquer une méthode d'apprentissage par renforcement. Les actions possibles pour le joueur A sont : se coucher, miser 1 euro, miser 2 euros, miser 4 euros. Les actions possibles pour le joueur B sont : se coucher ou suivre. Le joueur B ne peut pas miser au delà de ce que mise le joueur A car la partie ne se fait qu'en un seul tour.

C'est donc un jeu à information incomplète, les joueurs devant se baser sur les actions des adversaires pour déduire les informations manquante, créant ainsi la possibilité de tromper en jouant de manière opposée à ce que nos probabilités de victoire suggèrent.

4.2 Organisation du code

Une itération se déroule de la façon suivante : les IAs reçoivent chacune une carte aléatoire. Elles misent une somme initiale obligatoire puis un nombre est tiré au sort et celui ci détermine le choix de l'IA. Cela fait, l'IA ayant la plus forte carte emporte la mise. C'est la que se lance la fonction d'apprentissage, les probabilités de l'IA gagnante varient suivant la méthode LRI, et celles de l'IA perdante varient suivant la méthode LRP. Il faut environ 300000 itérations pour obtenir un système stable.

4.3 Cerveau et fonctionnement des IAs

On remarque du bluff lorsqu'un joueur mise alors que les probabilités de l'emporter lui sont défavorables. Les probabilités liées aux différentes actions sont notées dans les fichiers "BrainX.txt" dans un format CSV. Chaque ligne correspond à une carte différente, et chaque colonne correspond dans l'ordre à la probabilité de se coucher, celle de miser, de doubler, ou de tripler.

Cependant, l'IA B possède 3 cerveaux. "Brain B1, "Brain B2" et "Brain B3" s'activent respectivement à lorsque A choisit l'action "miser" "doubler" et "tripler". C'est ce choix qui nous a permit de faire en sorte que l'IA B s'adapte suivant les décisions de A.

Il est également utile de savoir que les ligne de probabilités de l'IA B ne sont pas interprétées de la même façon que celles de l'IA A. L'IA B ayant le choix de "suivre" ou "se coucher", nous avons considéré qu'elle devait prendre une décision égale ou supérieure à A pour "suivre" et inférieure à A pour "se coucher". Exemple : si A choisi de doubler la mise, et que B choisi de miser, c'est équivalent à "se coucher" pour B car ce choix est inférieur au fait de doubler. Si B avait cependant choisi de tripler, ce serait considéré comme l'équivalent de "suivre" A. Cette approche nous a permit de rendre les cerveaux complètement interchangeable, ainsi le joueur B peut devenir le joueur A n'importe quand. Cela nous permettrait d'adapter rapidement le code pour un apprentissage en self-learning avec une IA jouant contre elle même, et nous a permit également d'obtenir des résultats en inversant le rôle des cerveaux.

Pour éviter que des choix soient totalement écartés lors de l'apprentissage, nous avons fixé une valeur limite de 5% en dessous de laquelle la probabilité ne peut descendre.

4.4 Variables d'apprentissage

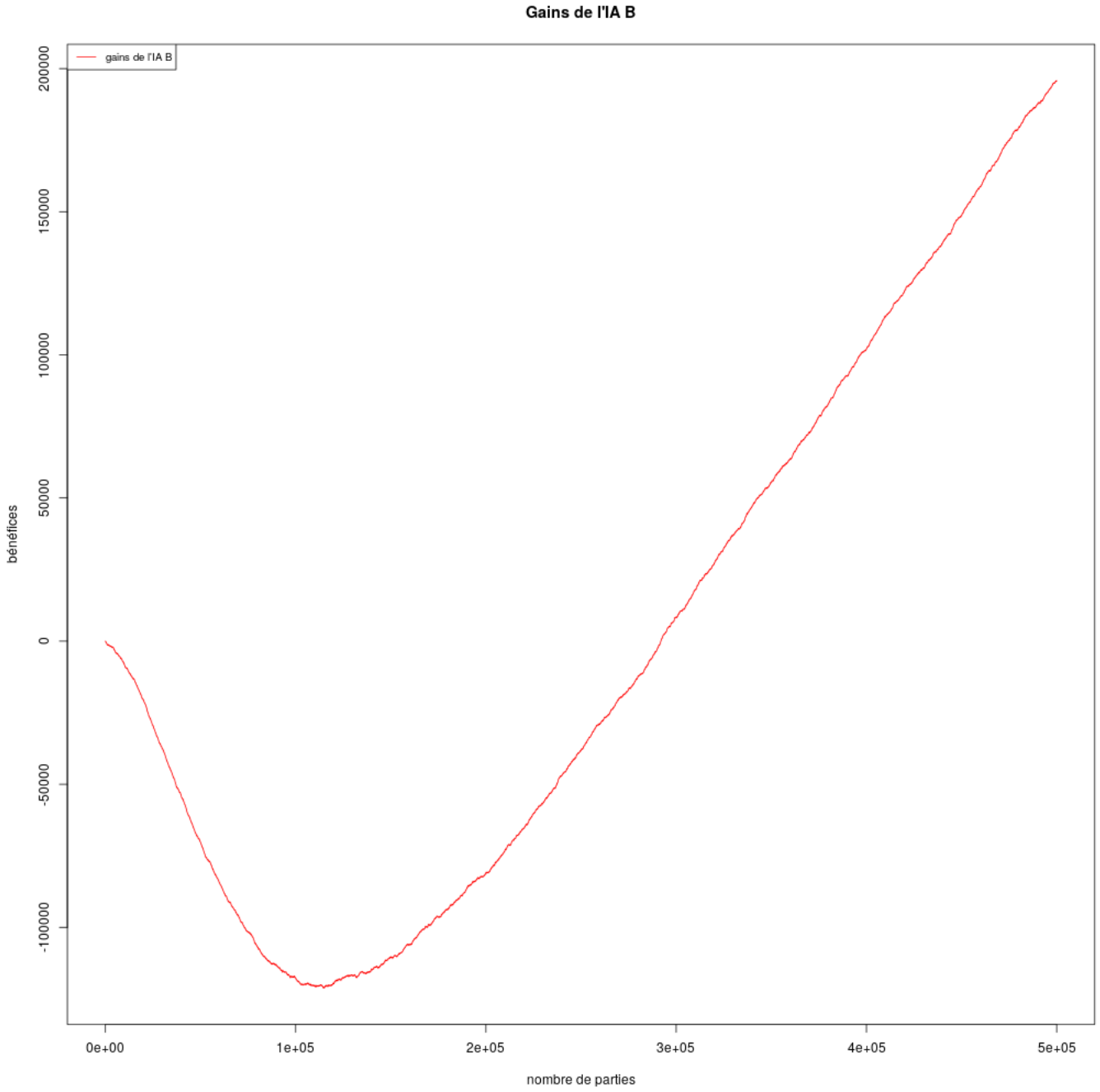
Afin d'assurer une convergence pertinente -tout en maintenant un apprentissage pouvant être accomplit en temps raisonnable- nous avons réduit au centième de pourcentage l'influence d'une récompense sur les probabilités des choix. Ainsi, un bon choix ne fait varier que de 0.03% à 0.09%.

De plus, nous avons rencontré un problème : lorsqu'un choix mène trop souvent à la défaite, sa probabilité chute tellement qu'il n'est plus jamais emprunté. Sauf que ce choix pourrait devenir bon par la suite si la situation du jeu évolue, nous avons donc fixé une valeur minimale de 5% en deçà de laquelle la probabilité d'un choix ne peut descendre. Sans cette valeur plafond, le problème est qu'un choix souvent prit menant au final plus souvent à la défaite qu'à la victoire (cette situation apparaissant grâce à l'apprentissage de l'adversaire) maintenait malgré tout sa probabilité et réduisait trop fortement l'exploration des possibilités.

5 Résultats attendu

5.1 Apprentissage avec LRI

Etant donné que le progrès de l'IA B est celui qui nous intéresse le plus, nous allons suivre sa progression en priorité. Le jeu étant à deux joueurs, un gain de l'IA B implique une perte équivalente de l'IA A, ainsi il n'est pas nécessaire de représenter les deux courbes.



La courbe précédente représente le bénéfice de l'IA B au fil d'une session d'apprentissage complète. Durant cette session, les IAs sont initialisés comme définit en 3.3. Nous allons détailler chaque étape de l'apprentissage et suivre l'évolution des cerveaux.

Durant les 100000 premières parties, l'IA A jouant en premier, elle découvre rapidement un plan de jeu optimal et gagne donc une grande somme d'argent,

Voici le cerveau de l'IA A après 100000 parties, on peut y constater que l'IA bluff bel et bien en doublant et en triplant sa mise (les deux colonnes les plus à droites) sur des cartes faibles.

```
#
0,79103,1433,4954,14510
1,69930,1095,4891,24084
2,52144,4382,5026,38448
3,44938,4970,5030,45062
4,23343,5022,5017,66618
5,5046,5069,29796,60089
6,5028,5033,30881,59058
7,5028,5022,34876,55074
8,5015,5021,35104,54860
9,5025,40290,5030,49655
```

Une fois ce plan de jeu stabilisé, l'IA B joue alors contre un adversaire répétitif et commence à ajuster correctement ses propres variables. Doté d'un cerveau par décision possible de l'IA A, elle est plus apte à réagir et fini par prendre le dessus. Elle a donc compris le bluff de l'IA A et le contre efficacement

Voici le cerveau de l'IA A après 300000 parties

```
#
0,75746,5011,5010,14233
1,58873,5049,5016,31062
2,44744,5028,5018,45210
3,5115,5045,24629,65211
4,5016,5034,27196,62754
5,5017,5019,27677,62287
6,5018,5014,29316,60652
7,5016,5014,29581,60389
8,5016,5019,35634,54331
9,5019,5016,30136,59829
```

Voici les cerveaux de l'IA B après 300000 parties

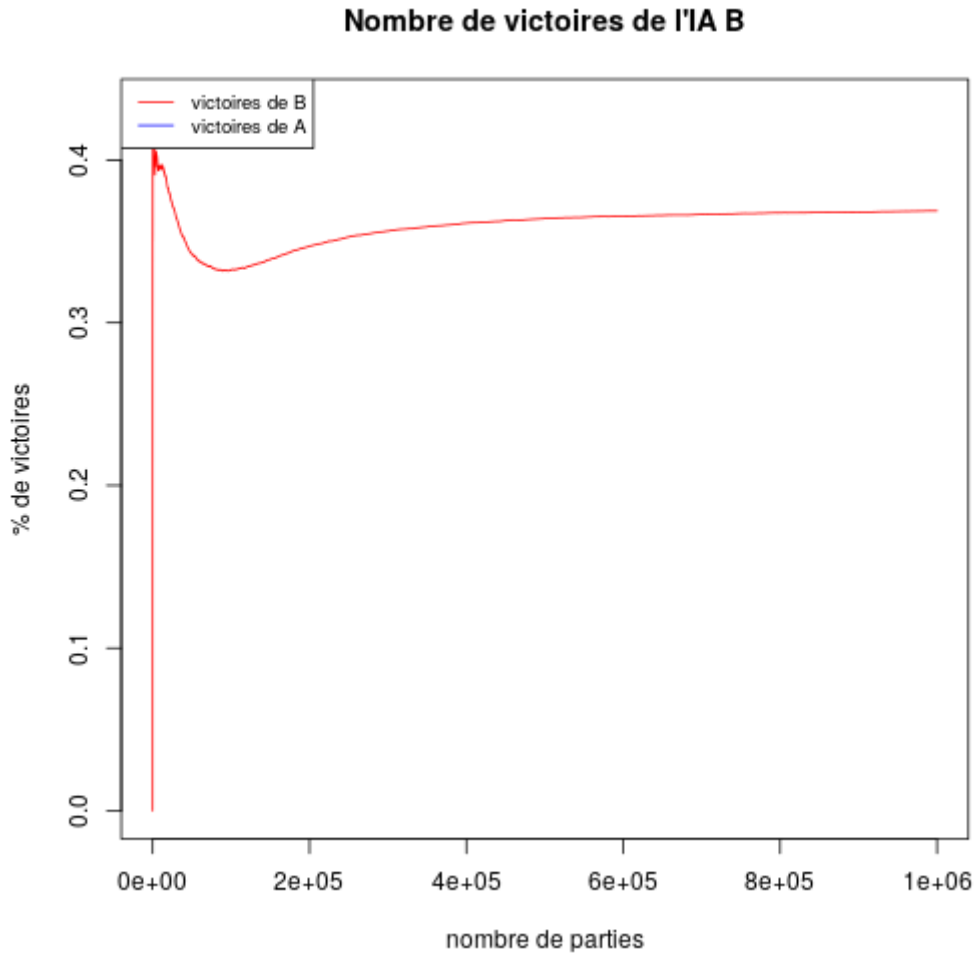
```
#
0,55024,14992,14992,14992
1,50293,20961,14373,14373
2,46113,26693,13597,13597
3,41019,32071,13455,13455
4,34802,39786,12706,12706
5,30618,46034,11674,11674
6,24165,53465,11185,11185
7,18646,60382,10486,10486
8,12884,67852,9632,9632
9,9034,72898,9034,9034
```

```
#
0,84973,5009,5009,5009
1,84980,5007,5006,5007
2,84974,5009,5008,5009
3,83264,5009,6718,5009
4,75320,5012,14656,5012
5,55077,5024,34875,5024
6,40067,5022,49889,5022
7,29964,5021,59994,5021
8,8491,5009,81491,5009
9,5009,5009,84973,5009
```

```
#
0,84973,5009,5009,5009
1,84973,5009,5009,5009
2,84973,5009,5009,5009
3,84975,5009,5009,5007
4,82220,5009,5009,7762
5,75014,5012,5012,14962
6,27957,5020,5020,62003
7,24873,5018,5018,65091
8,5022,5007,5007,84964
9,5009,5009,5009,84973
```

Chaque cerveau de B correspond à une action de A, le cerveau le plus à gauche correspond à "lorsque A mise" le cerveau du milieu correspond à "lorsque A double" et celui le plus à droite correspond à "lorsque A triple". On peut constater que B a su adapter son comportement, elle suivra A plus régulièrement lorsque celui ci mise simplement, et ne le suivra que si elle a une forte carte en main lorsque A double ou triple la mise.

Cependant un résultat présenté dans la courbe suivante survient, l'IA B perd plus souvent que l'IA A et malgré tout engrange plus de bénéfice. L'IA B sait donc jouer intelligemment en prenant des risques calculés quand cela en vaut la peine, ce qui lui permet de l'emporter.



5.2 Variations de α

Nous avons remarqué qu'avec différentes valeurs de α la convergence ne change pas, à moins d'augmenter la valeur au delà de 100000, auquel cas les résultats deviennent totalement hasardeux (ce qui est normal car α rend alors les récompenses bien trop grandes). La valeur de 5000 nous donne un bon équilibre entre vitesse de convergence et résultats cohérents.

5.3 Analyse des résultats

L'IA favorise le gain ponctuel de partie plus que le fait de gagner de l'argent dans notre modèle, ainsi il faudrait intégrer uniquement le fait de gagner de l'argent et supprimer le fait de gagner une partie. Pour cela, on noterait les décisions prises sur un nombre arbitraire de parties, et si en résultat l'IA a eu des gains au total, on récompense tous les choix y compris ceux qui ont ponctuellement mené à une défaite. Si l'IA a eu des pertes au total, on pénalise tous les choix de la même façon. A long terme, cela favorisera donc tous les choix qui permettent de gagner de l'argent, même si localement certains choix mauvais seront récompensés, ils ne le seront pas à chaque fois, ainsi les probabilités devraient s'affiner correctement sur un grand nombre de parties. Cependant, cette approche sort du cadre du LRI.

Nous aurions aimé que l'IA A soit capable d'apprendre à son tour, et observer les deux IAs s'adapter continuellement l'une à l'autre sans qu'aucune ne puisse définitivement prendre le dessus. C'était le résultat attendu, cependant l'IA A n'apprend pas suffisamment de ses défaites car ses choix ne la font pas suffisamment perdre pour que ses probabilités convergent définitivement vers un nouvel état. Ainsi, nous avons intégré la méthode de Linear Reward Penalty pour combler ce manque.

Cependant, nous avons également observé un autre point intéressant : lorsque l'IA B joue contre un adversaire A réinitialisé, celle ci va de nouveau avoir tendance à perdre de l'argent au début, puis une fois adaptée, la situation se stabilisera à nouveau dans un état similaire à celui obtenu par l'apprentissage parti de zéro.

5.4 Résultats obtenus avec LRP

En appliquant la méthode LRP lorsque A perd, nous supposions qu'elle serait capable d'apprendre de ses erreurs. Après implémentation les tests n'ont cependant pas été concluants, la convergence reste la même et nous n'avons pas eu l'opportunité d'étudier en profondeur les résultats. Cela reste une ouverture possible pour une évolution potentielle du programme.

6 Conclusion

Nous pouvons donc affirmer que la méthode du LRI est fonctionnelle dans ce cas très simple du poker, cependant les limites de l'algorithme se montrent assez rapidement lorsque celui ci tente de sortir d'une situation considérée "optimale" pendant un temps. Il est difficilement capable de "désapprendre" ce qu'il a déjà apprit lorsque l'environnement dans lequel il se situe change, en partie car il ne fonctionne que d'itérations en itérations et ne regarde pas la situation sur le long terme. Malgré le succès, nous gardons donc un regard critique sur la méthode.

Nous avons montré qu'il est suffisant pour apprendre à un joueur à bluffer, et apprendre

à son adversaire à jouer autour du bluff en minimisant les risques et en maximisant ses gains. Cependant nous pouvons difficilement envisager une application de cette méthode dans un véritable jeu de poker dans lequel il serait non seulement nécessaire d'analyser une situation à long terme, mais également de réfléchir en terme de gains totaux et non de victoires ponctuelles. Une autre approche comme SARSA semble plus pertinente car celle ci prend en compte le gain potentiel d'un état futur dans son apprentissage, le facteur appelé "learning rate" de cette méthode servant précisément à définir à quelle vitesse une nouvelle information remplace ce que l'IA a déjà apprit, elle permettrait sans doute au joueur de mieux s'adapter à un environnement dynamique.

Le véritable jeu de poker étant particulièrement complexe, des algorithmes efficaces consistent à combiner une méthode de Q-learning avec un réseau de neurone, ainsi pour une évolution potentielle du projet il serait également intéressant de se diriger vers cette méthode.