Clément PROVENCHER

Technologies Émergentes

420-5A3-VI gr.01

INTELLIGENCE DÉMONTRÉE

Rapport de projet

Travail présenté à

Étienne RIVARD

Département des Techniques de l’informatique

Cégep de Victoriaville

Date de remise (Le 30 novembre 2023)

Table des matières

[Buts du projet 3](#_Toc151909929)

[Concepts importants 3](#_Toc151909930)

[Cartographie 4](#_Toc151909931)

[Tableau comparatif des technologies 5](#_Toc151909932)

[Journal des enjeux et solutions 6](#_Toc151909933)

[Chapitre 1 – L’installation 6](#_Toc151909934)

[Chapitre 2 – Apprendre à guider la machine 6](#_Toc151909935)

[Chapitre 3 – Le saut du roi 7](#_Toc151909936)

[Chapitre 4 – La chute d’un roi 8](#_Toc151909937)

[Conclusion 10](#_Toc151909938)

[Médiagraphie 11](#_Toc151909939)

# Buts du projet

Le but principal de mon projet était, pour moi, d’apprendre une technologie nouvelle qui me fascine particulièrement. Je voulais apprendre, et être capable d’utiliser cette technologie. Le choix à été plutôt facile, j’ai toujours adoré l’intelligence artificielle et de pouvoir en forger une moi-même, ça semblait tout à fait magique. J’étais aussi intéressé par le fonctionnement derrière cette magie, comprendre comment ça fonctionne, pour pouvoir encore mieux l’apprécier. De plus, étant donné que je souhaite aller dans le domaine du jeu vidéo, il est probable que j’aille affaire avec l’intelligence artificielle, même sous une autre forme.

# Concepts importants

**Intelligence artificielle (IA) –** Ensemble de théories et de techniques mises en œuvre en vue de réaliser des machines capables de simuler l'intelligence humaine.

**Apprentissage automatique (*machine learning*) -** Une forme d’intelligence artificielle qui est axé sur la création de systèmes qui apprennent ou améliorent leurs performances en fonction des données qu’ils traitent.

**Apprentissage par renforcement (*renforcement learning*) -** l'apprentissage par renforcement consiste, pour un agent autonome, à apprendre les actions à prendre, à partir d'expériences, de façon à optimiser une récompense quantitative au cours du temps. L'agent est plongé au sein d'un environnement et prend ses décisions en fonction de son état courant. En retour, l'environnement procure à l'agent une récompense, qui peut être positive ou négative. L'agent cherche, au travers d'expériences itérées, un comportement décisionnel optimal, en ce sens qu'il maximise la somme des récompenses au cours du temps. C’est une façon d’effectuer de l’apprentissage automatique.

**Réseaux de neurones artificiels (*neural network,* ANN) -** Les réseaux de neurones artificiels sont constitués de couches nodales, contenant une couche d'entrée, une ou plusieurs couches cachées et une couche de sortie. Chaque nœud, ou neurone artificiel, se connecte à un autre et possède un poids et un seuil associés. Si la sortie d'un nœud est supérieure à la valeur de seuil spécifiée, ce nœud est activé et envoie des données à la couche suivante du réseau. Sinon, aucune donnée n'est transmise à la couche suivante du réseau. C’est ce qui est créé et peaufiné avec l’apprentissage automatique, c’est ce qui guide les décisions de l’intelligence artificielle résultante

# Cartographie

Une image contenant diagramme, texte, cercle

Description générée automatiquement

# Tableau comparatif des technologies

L’évaluation de technologies d’apprentissage automatique selon certains critères pour mieux effectuer un choix. Chaque critère est noté de 1 à 10

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Outil | Familiarité du langage | Documentation disponible | Conçu pour quel type de machine learning | Cout monétaire ($) | Difficulté d’implémentation sur Unity | Total |
| mlagents  (PyTorch) | 9 (C#) | 9 | Apprentissage par renforcement -> 10 | 10 | 10 | 48 |
| Tensorflow | 5 (api) | 6 | Apprentissage de fond -> 3 | 10 | 1 | 25 |
| Scikit-learn | 7 (python) | 7 | Apprentissage supervisé -> 5 | 10 | 1 | 30 |

Mon choix était très facile, puisque mlagents est conçu particulièrement pour ce que je voulais faire. Il est donc normal qu’il soit le langage qui correspondait le mieux à ce que j’avais envie de faire. C’est l’outil qui ressort en premier quand on cherche comment faire de l’apprentissage automatique sur Unity.

# Journal des enjeux et solutions

## Chapitre 1 – L’installation

La première chose que j’ai dû faire pour travailler sur mon projet, était d’installer mlagents sur mon poste et de faire un premier test, en suivant un tutoriel basique, pour comprendre les concepts clés.

Cependant, je n’arrivais pas à trouver comment télécharger pytorch, qui était au cœur même du fonctionnement de mlagents, sans ce module, ça ne fonctionnerait pas. J’ai passé des heures à essayer de le télécharger, mais surtout de télécharger la bonne version. Il fallait que la version de pytorch soit compatible avec la version de mlagents (côté python), mais soit aussi compatible avec le plugin mlagents de Unity…

Le lien d’installation sur la documentation d’installation de mlagents ne fonctionnait pas, j’ai dû désinstaller python au complet pour tout le réinstaller sur la bonne version, pour avoir une version de python qui fonctionnait avec l’installation de pytorch. Et encore, une fois télécharger, il n’arrivait pas à connecter le mlagents côté python avec le mlagents de Unity.

Après de longue heure, et un peu d’aide de mon super professeur de technologie émergentes, j’ai trouvé le problème; j’avais une version de python qui roulait sur 32 bit, alors qu’il me fallait python qui roule sur 64 bit.

Après avoir pratiquement tout réinstaller depuis le départ, j’ai enfin eu une installation qui fonctionnait et j’ai pu commencer mon travail.

## Chapitre 2 – Apprendre à guider la machine

La partie difficile du projet était de trouver la meilleure façon de faire apprendre à l’agent ce qu’on veut qu’il fasse. Dépendamment des données à l’entrée que je lui donne et des actions possibles qu’il peut effectuer, comment balancer toutes ces données pour qu’il apprenne le mieux possible. En fait, si ce travail est mal effectué, il n’apprendra même pas la bonne chose…

En premier lieu, je voulais faire apprendre à l’agent à aller chercher une boule. Après quelques itérations, l’agent était rendu bon pour se rendre à la position de la boule, qui ne changeait pas. Cependant, la seconde où j’ai déplacé la boule, ils n’étaient plus capables de s’y rendre. C’est parce qu’ils n’avaient pas nécessairement associé le fait que la position de la boule correspondait à certains paramètres que je lui envoyais. Il a fini par comprendre, mais ce n’Est qu’une simple démonstration de ce qu’il était possible d’apprendre.

Un véritable problème que j’ai eu, qui empêchait complètement mes agents de se rendre à la boule est le suivant : Les agents ne faisaient que tourner en rond sur eux-mêmes, éternellement. Ils n’essayaient même plus de bouger à partir d’un certain temps. C’est parce que j’avais mis une pénalité de -20 points lorsqu’un agent percute un mûr et j’ai mis une récompense de 20 points s’ils attrapaient la boule. C’était donc désavantageux pour eux de bouger, car ils prenaient le risque de perdre 20 points à répétition. En bougeant au hasard, ils avaient beaucoup plus de chance de percuter un mûr que de percuter la boule. Donc le comportement qui a été avantagé était celui de ne pas bouger, parce que les agents qui ne bougeaient pas perdaient moins de point que ceux qui prenaient des risques.

Pour corriger cela, j’ai baissé considérablement la pénalité pour le contact d’un mûr, pour qu’il y ait une bonne raison de prendre des risques. Le bénéfice potentiel dépassait la pénalité potentielle. C’est ainsi que j’ai réussi à les dompter.

## Chapitre 3 – Le saut du roi

La dernière situation pour laquelle je voulais entrainer des agents était celle du roi de la colline (*king of the hill*). Je voulais mettre quatre agents sur un seul et même environnement et les faire se déplacer vers un endroit précis qui change au fil du temps. Je voulais me donner un certain défi, alors j’ai surélevé un coin de l’environnement pour obliger les agents à sauter pour accéder au sommet.

Le problème c’est que vu qu’ils devaient sauter, j’ai rajouté l’option pour eux de décider de sauter. Cependant, le problème avec l’action de sauter, c’est que contrairement à avancer ou tourner, c’est un choix qui affecte plusieurs cycles de décisions après (le temps ou l’agent est dans les airs). De plus, la probabilité de départ pour qu’un agent choisisse une certaine option est la même pour chaque choix d’un même genre (par exemple, sauter ou ne pas sauter ont chacun 50% de se faire choisir). Ceci évolue au fur et à mesure que l’agent apprends, mais mon point est que l’agent ne faisait que souter, constamment. Dès qu’il était au sol, ce n’était pas long avant qu’il saute de nouveau dans les airs.

Ce bondissement constent à très négativement affecté leur capacité à se déplacer. Tellement que, même après des centaines de milliers d’itérations, ils avaient de la misère à rester sur un point, ce qu’ils sont supposés apprendre très vite considérant qu’ils commencent dessus.

J’ai donc pris des mesures drastiques, j’ai ajouté une pénalité pour les agents qui tentaient de sauter alors qu’ils étaient déjà dans les airs (l’agent ne sautait pas s’il n’était pas à terre, mais il pouvait continuer d’envoyer l’ordre de sauter, j’ai donc pénalisé les agents qui ordonnaient de sauter alors qu’ils étaient déjà en saut).

Les agents ont arrêté de sauter en un temps record. En quelques générations, plus aucuns agents ne sautaient. Mais, je n’étais pas plus avancé, il fallait maintenant qu’ils apprennent à sauter pour se rendre sur la plateforme…

Cependant, maintenant qu’ils avaient compris que sauter tout le temps n’était pas bien, et qu’ils savaient raisonnablement bien se déplacer, j’ai retiré la pénalité sur les sauts pour encourager les prises de risques. J’ai aussi fait en sorte que le seul endroit où le but pouvait être était sur la colline.

Ce fut long, mais ils finirent par apprendre.

## Chapitre 4 – La chute d’un roi

Bon, ils savaient sauter, ils savaient se rendre sur la colline. Le seul problème c’est qu’ils étaient particulièrement inconsistants Ce que j’ai donc fait c’est que je suis aller modifier les paramètres d’entrainement pour pratiquement retirer le nombre d’itérations.

Je me suis dit que beaucoup de pratique les aiderait. Mais, après beaucoup de pratique, ce qu’ils ont commencé à faire c’est à se jeter dans le mur pour recommencer au début et plus facilement se rendre là où ils devaient aller. Soit ils se jetaient dans le mur, soit ils tournaient sur eux-mêmes pour attendre que le point retombe sur eux.

Il y avait clairement un problème avec ma pédagogie, ils n’apprenaient vraiment pas la bonne chose. En fait, ils étaient trop dépendants sur le fait qu’ils se faisaient déplacer au milieu de l’arène au début de tous les épisodes (J’ai fait ceci pour faciliter l’apprentissage au début, mais ça m’est vraiment retombé dessus). Dès que je retirais ce déplacement, ils tombaient complètement perdus (C’est pour ça qu’ils rentraient dans les murs pour se déplacer…)

La solution, me suis-je dit c’était d’augmenter la pénalité pour les murs, pour que ça ne soit plus rentable comme tactique. Ils se sont mis à arrêter de bouger, pour ne pas prendre le risque de toucher un mur…

Ok, donc tout ce que j’ai à faire, c’est de retirer le déplacement lors d’un contact avec le mur. Après avoir fait ça, il y a eu des problèmes de chèvres qui sorte de l’arène, mais c’est hors sujet. Ils ont arrêtés d’utiliser les murs comme téléporteur, maintenant ils faisaient des câlins au murs pour aucune raison (Je n’ai pas continuer le même cerveau, J’ai recommencé à un point un peu avant, donc ce n’est pas à cause de l’expérience ancienne, ils veulent juste des câlins…)

Après tout ça… Ils sont encore super terribles. Je ne comprends pas, il y en a un, des fois, qui va sur le point, reste sur le point et attend que ça se termine. Parfait, il est super bon, c’est ce pourquoi je l’entraine. Mais pour chaque chèvre qui va sur le point, il y en a deux qui tournent en rond, pis une qui va dans la direction inverse, ou qui est pris dans un mur…

Sans mentionner que, à cause d’une erreur d’index de tableau, le point en haut à droite n’apparaissait jamais au début (et il est à l’opposé du premier point qu’ils ont appris) Ils ne vont donc jamais, jamais sur ce point-là. J’ai bien essayé mais ils n’ont pas voulu apprendre…

J’ai passé vraiment beaucoup de temps à essayer de régler ça, mais je pense que rendu là ce n’est pas la peine. Voici quelques réflexions que j’ai eu sur des raisons possibles de cet échec pédagogique :

* Trop d’information fournie -> J’ai ajouté la position en y de l’agent et la position en y du point à atteindre à l’information que j’envoie au cerveau. C’est donc probablement plus difficile pour lui de faire des liens pertinents entre ces informations-là.
* Trop d’options sur quoi faire -> J’ai ajouté l’option de sauter ce qui, non seulement, ajoute un choix de plus à prendre, mais qui aussi nuit beaucoup au déplacement de l’agent
* Mauvais ordre d’apprentissage -> J’ai conditionné les agents à commencer en bas en droite, puis je les ai placés au centre. J’ai ensuite mis beaucoup de temps sur le point sur la colline, mais les deux autres points ont été négligés, c’est donc pourquoi leur capacité à aller à chacun des points varie autant. (Aussi l’erreur qui ne fait jamais apparaitre le point en haut à gauche)

# Conclusion

Je crois que le but de mon projet à été un franc succès. J’ai pu apprendre cette technologie fascinante que je trouve tout aussi intéressante qu’avant de m’embarquer dedans. J’ai pu mieux comprendre comment utiliser cette technologie, je crois tout à fait être capable de bien utiliser cet outil maintenant. J’ai pu voir les défis uniques que l’apprentissage automatique avait à me proposer. J’ai eu un exemple de ce que cette magie pouvait donner lorsque bien maniée, et ce qu’elle donne quand on la guide mal. J’ai eu beaucoup de plaisir à apprendre et à utiliser cette technologie.

Selon moi, l’apprentissage automatique, bien que parfois difficile à bien préparer, est très bon pour optimiser. Je sais que l’apprentissage automatique à d’autre utilités super pratiques comme le *pattern recognition,* mais je veux me concentrer sur l’idée d’utiliser cette technologie dans le contexte d’une simulation. Elle excelle le mieux dans une situation où il s’agit d’optimiser une tâche. Il ne s’agit pas de faire ce que l’homme n’est pas capable, c’est de le rendre meilleur que l’homme à certaines tâches. Je pense par exemple à l’intelligence artificielle qui à été entrainée pour jouer aux échecs. Plus aucun humain n’est capable de le battre maintenant. L’apprentissage automatique sert à devenir, de plus en plus bon sur une tache extrêmement précise. Si on s’attends à trop de chose d’une simple intelligence entrainée, on sera déçu. Mais si on veut qu’il fasse UNE chose, une chose seulement. Il deviendra, avec le bon encadrement, le meilleur outil pour faire cette chose unique.

C’est donc parfait pour développer des outils pour l’homme. On peut utiliser cette technologie pour voir des liens entre des données que personne n’aurait penser chercher. Mon projet simpliste n’offre pas le meilleur exemple, mais cela reste qu’il m’a bien montré que j’avais un biais humain sur comment effectuer une certaine tâche. Basé sur ce que j’ai vécu, mes décisions sont teintées par la vie que j’ai vécue, mais une machine, elle, ne prends en compte que ce que tu lui donne. On peut donc se faire surprendre, parfois la solution la plus optimale n’est pas celle qu’on croit. Il faut être prudent, parce que, ce que la machine optimise, ce sont les récompenses que tu lui donne, donc il y a peut-être une façon d’obtenir beaucoup de points sans remplir le but attendu. Donc, encore une fois, bien encadrer cette magie.

# Médiagraphie

TECHNOLOGIES, Unity. *Build More Engaging Games with ML Agents | Unity*, <https://unity.com/products/machine-learning-agents> (Page consultée le 17 novembre 2023).

« Apprentissage par renforcement », dans *Wikipédia*, [s.l.], 2023, <https://fr.wikipedia.org/w/index.php?title=Apprentissage_par_renforcement&oldid=208915714> (Page consultée le 17 novembre 2023).

*How to use Machine Learning AI in Unity! (ML-Agents)*, 2020, 44:50, <https://www.youtube.com/watch?v=zPFU30tbyKs> (Page consultée le 17 novembre 2023).

*Intelligence artificielle - LAROUSSE*, <https://www.larousse.fr/encyclopedie/divers/intelligence_artificielle/187257> (Page consultée le 17 novembre 2023).

« Ml-agents/docs/Installation.md at develop · Unity-Technologies/ml-agents », dans *GitHub*, <https://github.com/Unity-Technologies/ml-agents/blob/develop/docs/Installation.md> (Page consultée le 17 novembre 2023).

*Qu’est-ce qu’un réseau de neurones ? | IBM*, <https://www.ibm.com/fr-fr/topics/neural-networks> (Page consultée le 17 novembre 2023).

*Qu’est-ce que l’intelligence artificielle (IA) ? | Oracle Canada*, <https://www.oracle.com/ca-fr/artificial-intelligence/what-is-ai/> (Page consultée le 17 novembre 2023).