Résumé de l’approche théorique formant la base du sujet de notre projet.

Sujet : *DDPG -* *Deep Deterministic Policy Gradient*

-----

L’algorithme *Deep Deterministic Policy Gradient* ou DDPG est un algorithme d’apprentissage profond par renforcement dans des espaces d’action continus. L’algorithme peut être vu comme une implémentation *du Deep Q learning* pour des espaces d’action continus. DDPG utilise une architecture Acteur & Critique, où l’acteur vise à apprendre la politique optimale, et le Critique la *Q-Function*.

# Exemple

Imaginez un individu sans aucune expérience voulant apprendre à jouer au golf. Cet individu est l’acteur. Il compte apprendre la bonne technique en suivant les commentaires d’un second individu, lui aussi sans aucune expérience. Ce second individu est le critique.

L’acteur joue et tire des balles à répétition sur le même terrain pour s’entraîner. Après un moment, le critique examine des reprises vidéo d’une sélection de tirs, et donne des conseils au joueur. Par exemple, pour un coup en particulier, le critique pourrait dire: « La balle n’est pas aller en ligne droite, c’est bien. Dans cette situation, essaie de continuer à sauter avant de frapper la balle. » Ce n’est pas un bon conseil, et l’acteur ne le sait pas, le critique non plus. L’acteur suit le conseil, et son jeu ne s’améliore pas.

Il est ainsi essentiel que le critique apprenne à bien évaluer les coups faits par l’acteur dans leur contexte. L’acteur apprend quels coups faire selon la situation en suivant les conseils du critique, qui lui-même apprend en temps réel à bien évaluer les coups de l’acteur. Le critique doit apprendre ce que constitue un « bon coup », étant donné que l’acteur ne sera aussi bon au golf que les conseils qui lui sont donnés.

Dans l’algorithme DDPG, L’Acteur et le Critique sont modélisés par des réseaux de neurones. L’apprentissage se fait par descente du gradient sur des expériences antérieures, comme des reprises vidéos des tirs.

# Replay Buffer

*Deep Deterministic Policy Gradient* est un algorithme hors politique, c’est-à-dire que l’apprentissage est fait avec des échantillons d’expériences antérieures. Ces expériences sont contenues dans une mémoire d’expériences ou *Replay Buffer*. À chaque itération, on ajoute à la mémoire une entrée contenant l’état actuel, l’action choisie, la récompense et le prochain état.

< Représentation du Replay Buffer >

Les réseaux Acteur et Critique sont entraînés sur des mini-batch d’expérience. Dans le cadre de l’algorithme, on ne commence l’apprentissage qu’au moment où la mémoire d’expérience contient autant ou plus d’observation que la taille du mini-batch.

# Acteur

L’acteur est un paramétrage de la politique par un réseau de neurones. Dans le modèle DDPG, la politique apprise est déterministe, c’est-à-dire que l’acteur associe directement une action pour chaque état, et non une distribution de probabilité des actions possibles.

Dans un espace d’action discret fini, il est possible d’identifier l’action optimale en évaluant la Q-value de chaque action, et choisir la meilleure. Cette technique est inefficace dans un espace continue, alors qu’il y a une infinité d’actions possibles. DDPG approche ce problème de la manière suivante : Comme l’espace d’action est continue, la *Q-function* est dérivable en fonction de l’action choisie. L’action optimale est ainsi identifiée par la résolution d’un problème de maximisation par descente du gradient et l’acteur utilise l’approximation de la fonction Q modélisée par le critique pour apprendre la politique optimale.

Comme mentionné plus haut, l’apprentissage est hors politique car il est fait avec des expériences passées. L’acteur met à jour la distribution de probabilité des actions possibles pour privilégier les actions menant à une plus grande récompense attendue selon l’état actuel. On fait une descente du gradient pour minimiser -Q, maximisant ainsi la qualité des actions choisies.

# Critique

Le critique est un paramétrage de la fonction Q par un réseau de neurones. Il utilise l’algorithme hors politique (l’acteur) et une estimation de la valeur Q avec l’équation de Bellman pour apprendre la fonction Q.

Comme dans l’approche classique du *Q-learning*, la valeur Q mise à jour est obtenue par l’équation de Bellman. Le calcul nécessite la valeur Q du prochain état ou Q\_next.

< Équation Bellman >

La valeur Q\_next est calculée avec les réseaux de neurones Target\_Acteur et Target\_Critique. Les réseaux de neurones *Target* sont des copies des réseaux Acteur et Critique. Ils diffèrent cependant dans leur technique de mise à jour. On peut voir ces réseaux comme des versions antérieures. À chaque itération, leurs paramètres sont mis à jour avec une portion des apprentissages de l’Acteur et du Critique.

< Mise à jour des réseaux cibles >

Étant donné que l’équation (XX) utilisée pour entraîner le Critique nécessite des valeurs calculées par le réseau lui-même, les réseaux cibles servent à stabiliser l’apprentissage et éviter la divergence de l’algorithme.

Une fois les valeurs connues, on fait une descente du gradient pour minimiser l’erreur entre la valeur Q trouvée par le critique et la valeur Q mise à jour. Cette optimisation apprend au critique à bien évaluer les action prises par l’acteur.

# Exploration

Le problème d’exploration vs exploitation dans un espace d’action continue est adressé par l’ajout de bruit aux actions prises par l’acteur. L’implémentation de l’algorithme DDPG suggère l’utilisation d’un bruit provenant du processus *Ornstein-Uhlenbeck*. Le processus dépend du temps et a originalement été introduit comme une modélisation de la vélocité d’une particule sous un mouvement Brownian.

< Formule du noise sur l’action >

L’ajout du bruit aux action permet d’explorer l’environnement, de familiariser l’agent à un échantillon de bonne et mauvaise action, et ultimement d’apprendre une meilleure politique optimale.

# Sources

http://stat.math.uregina.ca/~kozdron/Teaching/Regina/441Fall14/Notes/L31-32-Nov19.pdf

https://towardsdatascience.com/understanding-actor-critic-methods-931b97b6df3f

https://towardsdatascience.com/deep-deterministic-policy-gradients-explained-2d94655a9b7b

<https://medium.com/@thechrisyoon/deriving-policy-gradients-and-implementing-reinforce-f887949bd63>

https://spinningup.openai.com/en/latest/algorithms/ddpg.html