TME6 RLD

Advanced Policy Gradient

Edwyn Brient, Léo Hein - Décembre 2021

1 Proximal Policy Optimization

L'algorithme Proximal Policy Optimization (PPO) est une extension de l'algorithme actor critic. Ils sont basés sur le concept d'importance sampling ainsi que l'idée selon laquelle une bonne convergence est plus stable en empêchant la politique d'être trop différente d'un apprentissage au suivant.

La loss du critic ne change pas tandis que la loss d'actor devient :

$$loss_{std} = -\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}\hat{A}^{\pi_{\theta_k}}(s,a)$$

Sur cette base, nous allons comparer la version standard ainsi que deux versions de l'algorithme PPO : la version KL adaptative et la version clipped.

— La version KL adaptative consiste en l'ajout d'un terme mesurant la divergence KL entre la politique de l'apprentissage précédent et la politique actuelle. Ce terme sera alors pondéré par un poids variant en fonction de la divergence et d'une valeur de divergence visée (à fixer au préalable). Ce poids β est mis à jour à chaque apprentissage (pas à chaque pas d'optimisation).

$$loss = loss_{std} + \beta_k D_{KL}(\pi_{\theta_k}(.|s)|\pi_\theta(.|s))$$

— La version clipped borne supérieurement la $loss_{std}$ par $(1-\epsilon)A^{\pi_k}$ si l'avantage est négatif et par $(1+\epsilon)A^{\pi_k}$ si l'avantage est négatif. Cet ajout d'une borne permet à l'algorithme de ne faire de grosses modifications de politique que si l'algorithme a donné une plus forte probabilité de faire une mauvaise action ou une plus faible probabilité de faire une bonne action. Dans le cas où l'algorithme aurait une bonne direction de gradient pour un exemple, on ne veut pas que la modification soit trop importante et que l'algorithme perte de sa pertinence sur l'exemple.

$$loss = min(loss_{std}, -clip(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}, 1-\epsilon, 1+\epsilon)\hat{A}^{\pi_{\theta_k}}(s,a)) \text{ avec}$$

$$clip(a, 1-\epsilon, 1+\epsilon) = \begin{cases} 1-\epsilon & \text{si } a < 1-\epsilon \\ a & \text{si } a \in [1-\epsilon, 1+\epsilon] \\ 1+\epsilon & \text{sinon} \end{cases}$$

Dans toutes nos expériences un terme d'entropie a été ajouté pour aider à l'exploration. En particulier, ce terme pourrait avoir un effet intéressant sur l'environnement lunarlander où l'algorithme actor critic ne parvenait pas à sortir du minimum local le poussant à ne pas atterir. Le facteur de l'entropie a été fixé à 0.01. Voici la forme de l'entropie :

$$\mathcal{H}(s) = -\sum_{a \in \mathcal{A}} log(\pi_\theta(a|s))\pi_\theta(a|s)$$

2 Expériences

2.1 Paramètres et implémenation

Dans le cadres des expériences que nous avons menés dans ce TP, nous avons implémenté les différentes versions de l'algorithme PPO. Ces versions ont été testées sur les environnements Cartpole et LunarLander. Dans toutes les expériences qui vont suivre, le calcul de l'avantage est le même que pour le TP5, à l'aide d'un réseau target, les paramètres λ et γ varient selon l'environnement. A chaque étape d'apprentissage, K optimisations sont effectuées sur le batch d'apprentissage. Une fois toutes les optimisations effectuées, le réseau target est mis à jour après avoir en copiant les poids du critic et le poids de la divergence KL est mis à jour.

Les résultats qui suivent ont été produits avec l'algorithme PPO et ses différentes versions sur l'environnement Cartpole. Les étapes d'apprentissages sont effectuées toutes les 1000 itérations en attendant la fin de l'épisode en cours. Le learning rate pour tous les réseaux a été fixé à 10^{-4}, la fonction utilisée pour la loss du critic est la huber loss. Enfin, comme dans le TP précédent, les réseaux ont tous 2 couches de 64 neurones à activation ReLU.

Les paramètres utilisés sur les différentes versions sont les suivants :
— $K = 100$ pour la version clippé, $K = 10$ pour les deux autres versions
— le ϵ du clipping est fixé à 0.2 pour lunarlander et 0.02 pour cartpole (avec 100 optimisations par apprentissage, même un clipping faible permet de modifier correctement π)
— le target du KL (la valeur autour de laquelle β n'est pas modifié) est de 0.003

2.2 Résultats

Nos premières expériences ont été réalisées sur l'environnement cartpole d'openAI. A la suite des expériences avec l'algorithme actor critic, nous espérons obtenir des résultats plus stables grâce à la forme des algorithmes PPO qui sont supposés moins modifier la politique entre deux itérations d'apprentissage.

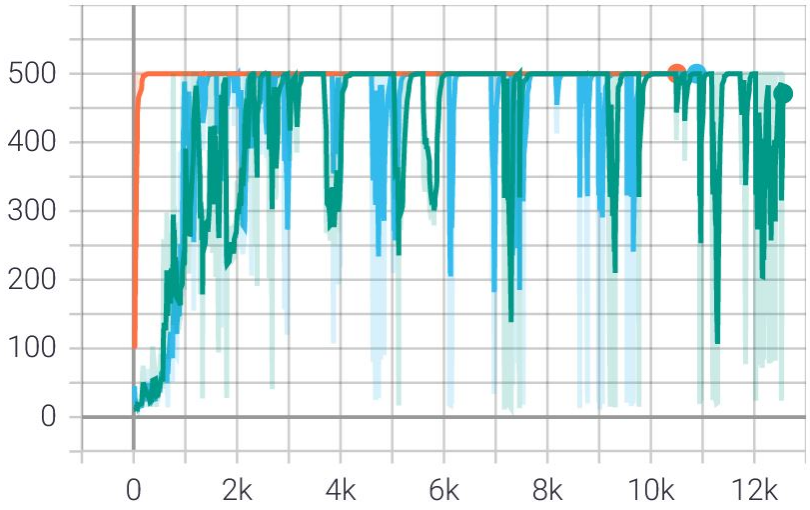

FIGURE 1 – Récompenses des différentes version de l'algorithme PPO sur Cartpole en fonction du nombre d'épisodes. En orange, la version clippé, en vert la version KL et en bleu la version standard.

Les résultats sont à la hauteur de nos espérances. Si l'unique version à rester à 500 reward à partir de l'épisode 200 est la version clippée, les deux autres versions sont légèrement moins stables mais ne redescendent jamais en dessous de 300 plus de 10 épisodes d'affilés et restent en moyenne au maximum.

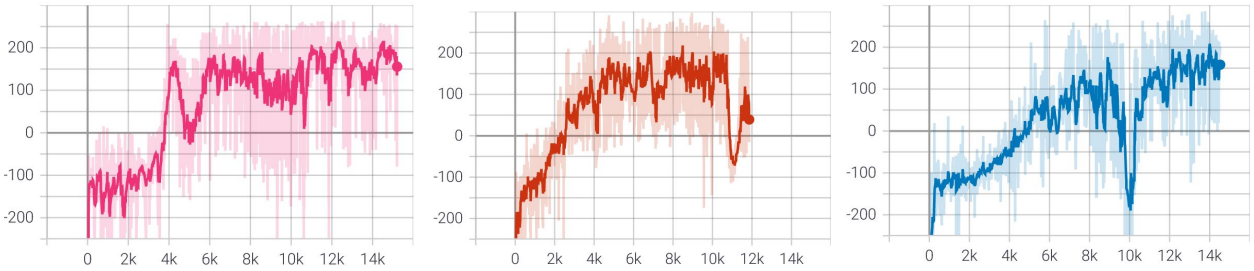

FIGURE 2 – Récompenses des différentes version de l'algorithme PPO sur lunarlander en fonction du nombre d'épisodes. En rose, la version clippé, en marron la version standard et en bleu la version KL.

Ci-dessus, les expériences des versions sur l'environnement lunarlander. Si toutes les versions montrent ici une légère instabilité, celles-ci atteignent un reward positif sur cet environnement difficile et hormis le pic de reward descendant présent pour chaque version, les rewards restent stables autour de 150. La version clippé semble cependant préférable car son instabilité s'est produite au début de son apprentissage à la suite de quoi celle ci reste assez stable contrairement par exemple à la version standard qui a vu son instabilité arriver à la fin de son apprentissage.

3 Interprétations

La version clippé est assez différente de la version KL mais nous pouvons quand même les comparer en terme de stabilité ajoutée. Si la version KL apporte un terme diminuant les grosses modifications de politique peu importe l'avantage, la version clippé n'empêche les grosses modifications de politique qu'en cas d'avantage négatifs. En ce sens, la version clippé semble plus précise que KL et ceci peut expliquer les performances sur cartpole ainsi que sur lunarlander.

L'algorithme standard semble plutôt performant alors qu'il n'empêche pas les gros changements de politique, il est un peu moins stable mais obtient des performances très bonnes comparé à l'algorithme actor critic. Ceci peut s'expliquer par la présence de l'entropie qui aurait pu être rajouté dans la loss de l'actor critic aussi.

4 Conclusion

Cet algorithme et ses différentes versions avec l'ajout de l'entropie ont montré des résultats très bons sur les différents environnements testés. Il aurait pu être intéressant de le tester sur l'environnement MountainCar qui pose des difficultés différentes des environnements vus jusqu'ici et qui pourrait être la limite de cet algorithme.